SESSION

THEORY AND ALGORITHMS

Chair(s)

TBA

Support Algorithms for Eye-Hand Coordination Robotic Therapy

N. Pernalete[1], F. Tang[1], S.M. Chang[1], F.Y. Cheng[1], P. Vetter[1], M. Stegemann[1], and J. Grantner[2]
[1]California State Polytechnic University, Pomona, CA, USA
[2] Western Michigan University, Kalamazoo, MI, USA

Abstract - *This paper presents the support algorithms used for developing a system intended to improve eye-hand coordination in children diagnosed with this problem, using a robotic mapping from a haptic user interface to a virtual environment. A detailed analysis of patters (e.g., labyrinths, letters and angles) created in a virtual environment, as well as the support algorithms (position, force, velocity, inertia and viscosity) developed is presented in this paper. These algorithms are incorporated into the tasks in order to introduce general computer assistance to the mapping of the user's movements to the computer screen. Users' results are evaluated in Accuracy and Time by an edge-detection based evaluation function described in section 3. A fuzzy-logic based Intelligent Decision System takes the data produced by the evaluation function and suggests the most appropriate test to be executed next by the user.*

1 Introduction

HAPTIC devices provide a proprioceptive input allowing the user to perceive movement and location of 3-dimensional space through force feedback. They record and report 3D spatial positions, velocity vectors which features make them highly useful for training and learning purposes. In [1, 3, 6] several different approaches for teaching handwriting have been proposed. The application will help users with disabilities acquiring the writing and drawing skills by using a haptic device. It uses a robotic mapping from a haptic user interface to a virtual environment. For the tasks, the application consists of a force reflecting haptic interface drive, PHANToM Omni with OpenHaptics Toolkit (version 3.0) [7] and OpenGL (Open Graphics Library). The OpenHaptics is patterned after the OpenGL API, making it familiar to graphics programmers and facilitating integration with existing OpenGL applications. OpenHaptics is composed of two layers: a haptic device API (HDAPI) and a haptic library API (HLAPI). HDAPI provides a low-level access to the haptic device. HLAPI provides an advanced haptic rendering capability. HDAPI and HLAPI are built on top of the PHANTOM device drivers (PDD). Eye-hand coordination is the ability of our vision system to coordinate the information received through the eyes to control, guide, and direct the hands in the accomplishment of a given task. Eye-hand coordination uses the eyes to direct attention and the hands to execute a particular task. Most hand movements we perform require visual input to be carried out effectively. The results of a well-designed study [8] reported that using a PHANTOM robot to study eye-hand coordination with force feedback improves the subject's accurate perspective performance by approximately 9% and contact by another 12%. Grip strength measurements are an important method for occupational therapists to learn about the degree of disability of the hand [9]. Intervention for eye-hand coordination and grip strength becomes necessary to successfully perform activities of daily living such as dressing, feeding, drawing and writing.

Project Funded by The National Science Foundation Research to Aid People with Disabilities (NSF-RAPD) AwardID#0828492

2 Background

Extensive research is being done in the field of haptics to improve hand and arm movements. One approach taken was to expose a subject to a perturbing field to develop an internal model of the field as a relation between experienced limb states and forces [10]. In this study [10], they concluded that after-effects persisted after many trials using a force field. Former studies have shown that haptic technologies can be very instrumental in teaching motor skills and manual crafts based on a principle of rehabilitation in which simple movements can be improved by constant practice. Other work by Bardorfer et al. [11] shows some labyrinths or mazes created in the virtual environment, in which the user has to move the pointer (ball) through it and could feel the reactive forces of the walls. In this paper, we discuss the possibility of improving eye-hand coordination in children diagnosed with this problem, using a robotic mapping from a haptic user interface to a virtual environment. The goal is that by improving their coordination and strength, they will increase their participation in handwriting as well as in activities of daily living. The evaluation function was designed and tested first using college-age subjects with no defined eye-hand coordination problems before it was implemented with children. The current results of this function (% accuracy and time) are being used as inputs to the Intelligent Decision Support System (IDSS), which in turn suggests the next task to be performed by the user. This system has been designed and implemented for testing with a group of children at the Motor Development Clinic at Cal Poly Pomona.

3 Design of Support Algorithms

At first, a test pattern is chosen by the therapist or suggested by the IDSS (Intelligent Decision Support System), and then the application loads it as an image into the haptic workspace. The image properties are shown in Table 1. There are three patterns shown in Fig. 1: Maze, LeLe hand writing pattern, and Complex Labyrinth. The subject is required to move the stylus along and follow the specified trajectory while holding the stylus on the X-Y plane (air). A mapping was also designed for the children to follow the trajectories on a piece of paper (X-Z plane).

TABLE 1. THE IMAGE PROPERTIES

Property	**Value**
Type	Bitmap Image
Width	512 pixels
Height	327 pixels
Horizontal Resolution	71 dpi
Vertical Resolution	71 dpi
Bit Depth	24

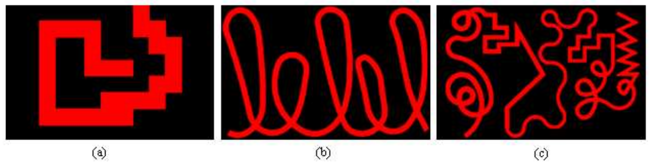


Figure 1. (a) Maze; (b) LeLe hand Writing Pattern; (c) Complex Labyrinth

3.1 Support Algorithms Developed for the Tasks

Various effects are implemented [4, 5, 7] in the form of assistant functions; these are plane constraints, force feedback effect, inertia effect and viscosity effect. Some resemble the handwriting assessments test by using regular pencil and paper.

3.1.1 Regulation of Positions

It is important to know how to haptically render contact with the plane constraints. The plane does not allow passing through such that if the user applies a force against the plane, the plane will give a reverse force and will not allow the user to go through it. The application is built with an X-Y plane workspace and set the Z-coordinate position to zero throughout the task execution in order to keep the slave position (virtual representation of the stylus position) within the workspace. The application created a virtual wall model (see Fig. 2) which is applied to the Z-plane in order to prevent the user from penetrating through the surface. A large coefficient K is used to simulate the force F from the virtual wall, where Zwall represents the point position on the surface corresponding to the real slave position Zp.

$$F = \begin{cases} 0 & \text{->} \quad if\, Z_p > Z_{wall} \\ K(Z_{wall} - Z_p) & \text{->} \quad if\, Z_p \le Z_{wall} \end{cases} \qquad (1)$$

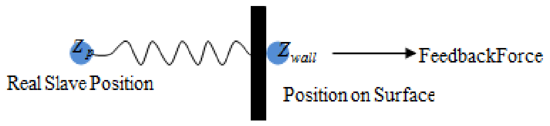


Figure 2. Virtual Wall Model

3.1.2 Force Assistance Functions

This application assists the user by applying the force in the direction of the task to be completed, or attracting the user to the desired path if he/she diverts from it. Force feedback can be used to guide the subjects hand in a predetermined trajectory when he/she is unable to move in response to visual feedback. The force (see Fig. 3) is applied either in the direction of the master (stylus) or opposite to the direction of the master given by:

$$\begin{aligned} &Fx = [\,\pm Fx,\ 0,\ 0\,] \quad \text{->} \quad \textit{For assisting motion in X-plane} \\ &Fy = [\,0,\ \pm Fy, 0\,] \quad \text{->} \quad \textit{For assisting motion in Y-plane} \end{aligned} \qquad (2)$$

Where Fx and Fy correspond to the constraint forces exerted in the Master side on X and Y directions. The motion in the Z-plane is constrained at zero, making the force effect always zero in that plane. The force is varied from 0.5 N to 1 N depending on the level of difficulty to be implemented in the execution of the task.

$$\begin{aligned} &Fx = [\,+Fx,\ 0,\ 0\,] \quad \text{->} \quad if\, X_c < X_1 \\ &Fx = [\,-Fx,\ 0,\ 0\,] \quad \text{->} \quad if\, X_c > X_2 \\ &Fy = [\,0,\ +Fy,\ 0\,] \quad \text{->} \quad if\, Y_c < Y_1 \\ &Fy = [\,0,\ -Fy,\ 0\,] \quad \text{->} \quad if\, Y_c > Y_2 \end{aligned} \qquad (3)$$

For the force effect, the system triggers an impulse by commanding a force with a direction and magnitude for a small duration.

3.1.3 Inertia Effect Assistance Function

The inertia effect is mainly used to reduce tremor in the hands of the user having problems with eye-hand coordination and, at the same time, improve their grip strength as this test is conducted both with power and precision grasps. The effect is implemented by increasing the mass, damping coefficient and spring stiffness. The implementation of this function is based on pulling a point mass around by a spring.

3.1.4 Viscosity Effect

The Viscosity effect is developed to provide a smooth execution of the task for the user by eliminating the fast and irregular movements of the master (stylus) that is added by the user. It provides the effect of moving the drive in a denser medium by a small force to the user's hand at the same time, which in turn, activates the hand muscles while executing the task. The application used OpenGL to draw the ideal path with a force to guide the user. A good approximation of the snap distance can be solved by the following force formula:

$$\begin{aligned} &F = K*X, \text{ where} \\ &\quad F: \text{Force in Newtons (N).} \\ &\quad K: \text{Stiffness control coefficient (N/mm).} \\ &\quad X: \text{Displacement (i.e. snap distance).} \end{aligned} \tag{4}$$

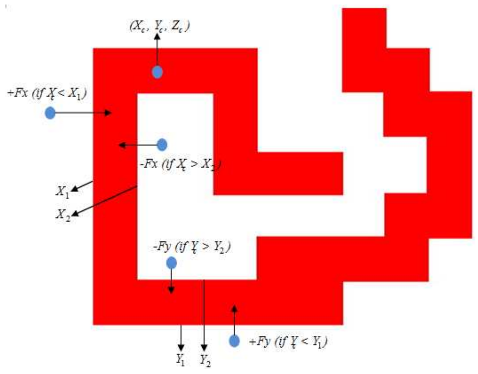
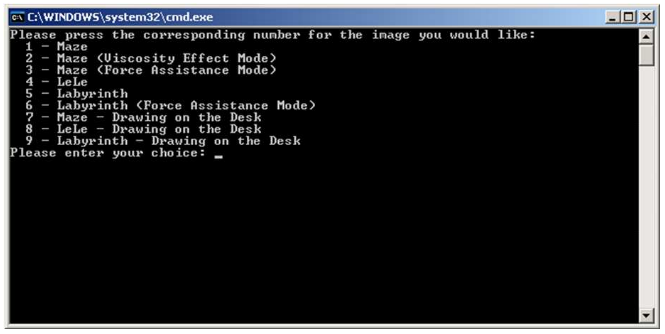


Figure 3. Representation of the Force Assistance Function

The system defines the snap distances to use based on approximating how much force the user needs to exert to pull off from the ideal path. The project is a Win32 console application (see Fig. 4). The data flow of the application is shown in Fig. 5. When the user enters the application, he/she will be asked what test mode he/she would like to run. After loading the test environment, he/she can choose a level for inertia effect, start recording, or quit the application by using the right-click menu.



```
C:\WINDOWS\system32\cmd.exe
Please press the corresponding number for the image you would like:
  1 - Maze
  2 - Maze (Viscosity Effect Mode)
  3 - Maze (Force Assistance Mode)
  4 - LeLe
  5 - Labyrinth
  6 - Labyrinth (Force Assistance Mode)
  7 - Maze - Drawing on the Desk
  8 - LeLe - Drawing on the Desk
  9 - Labyrinth - Drawing on the Desk
Please enter your choice: _
```

Figure 4. The Menu of the Application

3.2 Description of the Ratio Adjustment

Due to the fact that the units for the evaluation function and haptic device's coordinate system are different from each other, a ratio adjustment is required to get a more accurate evaluating score. Fig. 8 shows the different start point (0, 0) between the evaluation function and the haptic device's coordinate system. Since the evaluation function stores the image pixels in an integer array, there are no negative values for either coordinate. That is not the case with the haptic device's coordinate system, which starts at the center of the image. Since the two coordinate systems cannot be mapped to each other without a unit definition or a ratio, one or several base points are introduced to solve this problem. This point is demonstrated in Fig. 6, which shows a blue base point on the top left hand corner of the image. The red and green points represent different start points (0, 0) for two different coordinate systems (See Fig. 6). The red point is located on the top left hand corner of the image while the green point is located on the center of the image. For demonstration purposes, the ratio between the green and red points' coordinate systems was set to two as shown. Then, the location of the blue point is mapped from one coordinate system (red) to the other (green). In order to map a point from a coordinate system to another, the start point (0, 0) must be identified first. Fig. 6 shows the difference between coordinate systems of red and green points. A blue point is used to demonstrate the mapping process. The first step of the mapping process is to make the start point (0, 0) of the two coordinate systems coincide, as shown in Fig. 7.

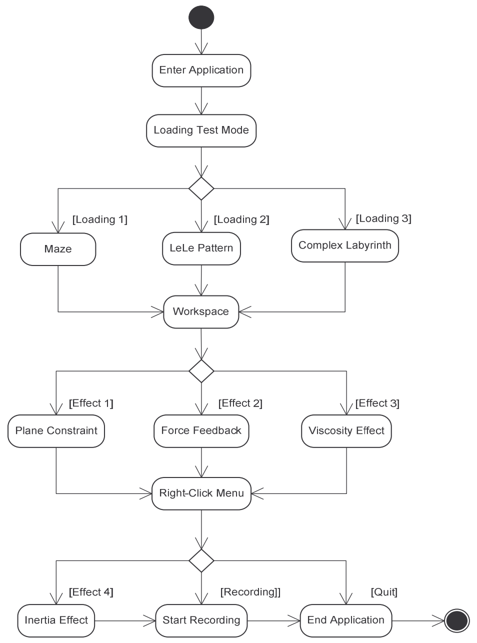


Figure 5. The Data Flow of the Application

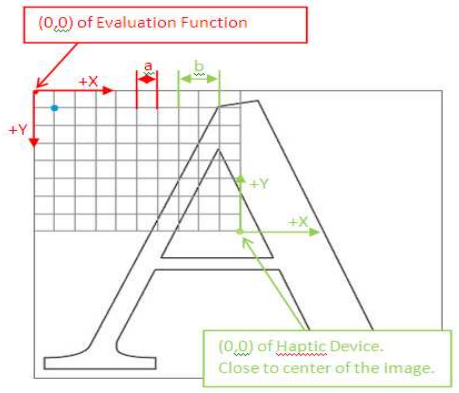


Figure 6. Two Coordinate Systems as Red and Green, Indicating the Evaluation Function and the Haptic Device Respectively. This Figure Shows the Conversion Between them.

Next, the ratio is obtained by given the location of a point in two different coordinate systems. Fig. 7 shows an example, given the locations (-4.5, 3.5), (-9, -7) of the same point in two different coordinate systems. As shown, the ratio is then set to two. The x and y coordinate ratios coincide in this case, but it is possible to set them differently. By using this mapping procedure, the coordinates recorded by the haptic device can be easily mapped into coordinates used in the evaluation function (pixel based) to get a more accurate evaluating score. The following formula is used to calculate the ratio x and y:

$$\text{Ratio } X = |A_{x1}-A_{x2}|/|B_{x1}-B_{x2}| \quad (5)$$
$$\text{Ratio } Y = |A_{y1}-A_{y2}|/|B_{y1}-B_{y2}|$$

Where
A and B represent pixel and haptic device's coordination respectively.
($_{x1}$, $_{y1}$) represents the First Point in two coordination system.
($_{x2}$, $_{y2}$) represents the Second Point in two coordination system.

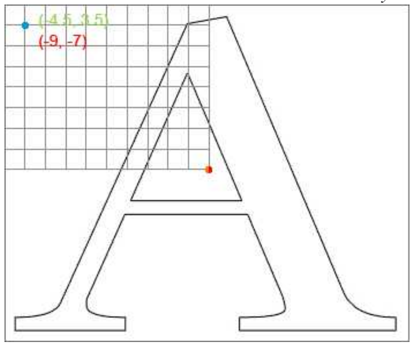


Figure 7. The Start Point (0, 0) of Red Point Is the Same as the Blue One; The Coordinate of the Blue Point in the Coordinate System of Red Point Now is (-9, -7); The Coordinate of the Blue Point in the Coordinate System of the Green Point Now is Still (-4.5, 3.5)

3.3 Description of the Evaluation Function

In order to implement the Intelligent Decision Support System [4], an appropriate evaluation function is designed and tested to determine the accuracy (% of valid points) and time (seconds) of the tasks executed by users. There are three steps taken in the design of this function: First: is to count the percentage of valid points, Second: is to find out how many points are located on the ideal path, and Third: is to measure the completion time based on the average. Appropriate values are carefully chosen for penalties.

3.3.1 Percentage of Valid Points

Since the result is captured by time intervals, the path itself is comprised of several points. Edge detection [2] is used to seek out the valid boundary. Results show in Fig. 8 (Left) and (Right-Red Line). After the boundary is obtained, the percentage of valid points is then calculated.

3.3.2 Number of Points Located on the Ideal Path

The ideal path is determined manually as the input images are complex and irregular in shapes shown in Fig. 8 (Right-Green Line) and (Right-Black Line).

3.3.3 Completion Time

It is calculated based on the sampling time and the number of samples obtained after execution. Penalties for both accuracy and time are carefully chosen after careful testing and discussion. Fig. 8 (Left) shows the original image for which users are to follow the black line. After getting the results from users after executing the haptic tasks, they are simply loaded into the evaluation function. In order to evaluate the results correctly, the image must be of a fixed size. Different image sizes may cause incorrect mapping from the result to the values in the evaluation function. There are two required steps before applying the evaluation function. First, a black-white based image is required in this step. It is used for the program to easily identify the edge. In the user interface, the image can be displayed as colorful as possible.

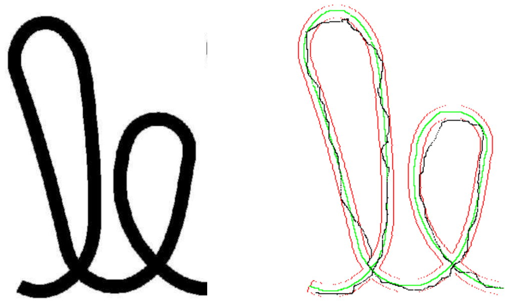


Figure 8. (Left) Original Image; (Right) Image Combinations of Edge (Red Line), Ideal Path (Green Line) and Result (Black Line)

In the "PGM.h", it contains a class "PGM", which is relative to any possible function that used in this program for portable gray map image (refer to PGM image), such as loading the PGM image, detecting the edge, or writing the result to a PGM image. By using this class to load the original image displayed in Fig. 8 (Left) and employing its function "boundaryDetect()", it is able to generate the edge image shown in Fig. 8 (Right-Red Line). This function also outputs a file whose extension is ".bdy". Notice that the edge file generated by the above function is an edge-like image instead of a real edge image. The reason for not employing the actual edge detection algorithms is the decreasing number of the calculation to know the exactly edge. This function simply eliminates the middle part of a line and leaves the head and tail pixels (see Fig. 9).

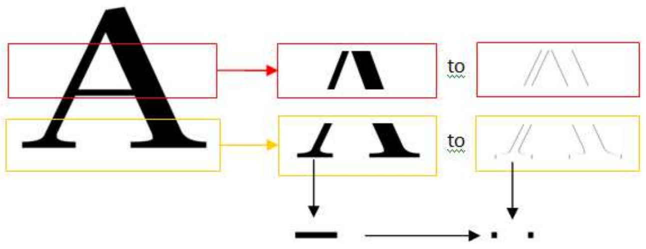
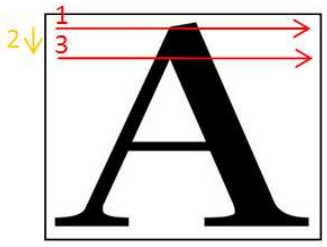


Figure 9. Showing the difference from actual edge

Fig. 10 shows how the function "boundaryDetect()" works. First, it scans the image left to right. Once it reaches the right-most pixel, it will go to the second step. The second step will start at the next line from the left-most pixel and repeat the first step. It continues to do so until the function hits the bottom line's right-most pixel. The output of this function will provide a set of pairs of points to describe a valid area.



1. Do step 1 until hit the right-most pixel for line 1.
2. Go to next line (line 2), and repeat step 1.
3. Do step 1 until hit the right-most pixel for line 1.

Continue to do the above 3 cycles, until hit the bottom line's right-most pixel.

Figure 10. Two steps of "boundaryDetect()"

Second, since the image has an irregular shape, some manual modification is required to generate the ideal path of the image shown in Fig. 8 (Right-Green Line). The image of the ideal path can be imported to the PGM class to create a file whose extension is ".pts". After these two steps are completed, the evaluation function can be applied to the result. The class, "evaluation", is a collection of all necessary evaluation functions. After the edge (bdy file) and ideal path images (pts file) are produced by the class "PGM", the class "evaluation" is ready to be employed. There are four steps needed to be done before evaluating the result in this class. The first three steps are:

1) Loading the boundary file, by using the "loadBdyFile()" function
2) Loading the points file of ideal path, by using the "loadPtsFile()"function, and
3) Loading the results, by using the "loadIdealFile()" function.

After loading these files, the first and second evaluating rules will be applied by the function "startEval()", by using two separate function calls namely "step1()" and "step2()". The function, "step1()", simply seeks these valid points by comparing all points from the results to a valid area described by a ".bdy" file and records these valid points to an array for the step2 function to use. The accuracy percentage is calculated from the number of total valid points divided by the number of total points. A sample result from this function is as follows: *Total Number of Points is 4132; Number of Valid Points is 3648; Accuracy Percent is 88.2865%.* Next, the function, "step2()", uses the valid points from the previous step to determine if a particular point is considered to be close to the ideal path. The tolerance pixel is introduced in this function. Fig. 11 (Left), shows the tolerance pixel, which can be set to 5, 7, or 10 pixels. Fig. 11 (Right) shows how the evaluation function determines if a point is considered to be close to the ideal path. With careful consideration, selecting this tolerance pixel will help determine the user's accuracy in executing the haptic tests. A sample result from this function is as follows (with the tolerance pixel set to 5px).

Total Ideal Valid Points: 2724; Total Valid Points: 3648; Percentage of Points which are considered close to the Ideal Path: 74.6711%

After some testing and analysis, this function will help determine both the user's accuracy and the elapsed time, which will serve as the basis for the Intelligent Decision Support System planned later in this project.

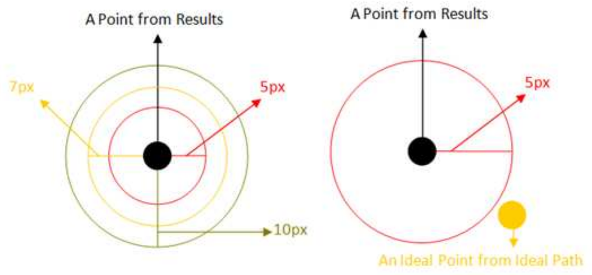


Figure 11. (Left) The tolerance pixels; (Right) The tolerance pixels is 5pixels. If there is an ideal point close to a point from results within 5 pixels, then it consider not close to ideal path.

3.4 Verification of the Evaluation Function

After applying the evaluation function to the users' results, it is necessary to check that the results are not made to cheat on the evaluation function, producing an inaccurate result. For example, in Fig. 12, the evaluation function will come out with a high score that shows the user successfully completed the task and is qualified to proceed to the next level. However, it should not be categorized as high performance, as the user only finished part of the task. The red thick line indicates the valid area which we assume a user will always try to follow in order to get a high score. The white thin line indicates an example of a user's result that is made to cheat on the evaluation function.

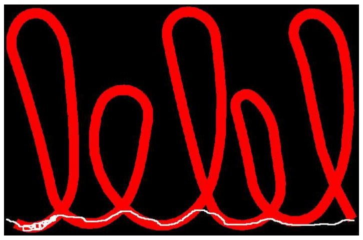


Figure 12. Violation Result

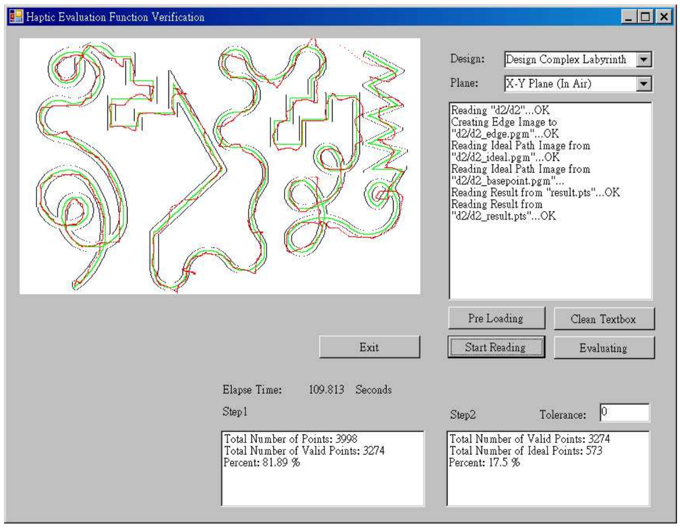
To make sure the ratio adjustment is appropriate for this project; a graphical user interface was developed to measure the transformed results (original coordinates to pixel coordinates). As Fig. 13 shows, the GUI provides four main functions, "Pre Loading", "Start Reading", "Evaluating", and "Exit". The "Pre Loading" function, reads the original image, then the edge file is loaded through the boundary detect algorithm. It also imports the ideal path and stores this data into files for further usage. The "Start Reading" function converts the haptic coordinate to the pixel coordinate, performs the evaluation function, and displays an image for results with the edge file. The "Evaluating" function provides a way to re-evaluate the results while the tolerance range is selected. Each function can run separately, however, the "Start Reading" function will require the data (edge file) from the "Pre Loading" function. The data (edge file) is usually fixed while the program is being tested by the subjects. Therefore, the program can use previous calculated data in the "Start Reading" function. Fig. 14 shows the flow graph of the main program for the evaluation function.

4 Experiments and Analysis of Results

The subjects were instructed to start the tasks at the workspace and simply guide the PHANToM to perform the various tasks to be executed. Position data in X, Y and Z coordinates, and the time and trajectories taken to execute the tasks were recorded for further analysis done by the evaluation function. A group of children (11 in experimental group and 10 in control group) participated in a collaborative research project between the Motor Development Clinic, the Department of Kinesiology and Health Promotion, and the College of Engineering at Cal Poly Pomona. They were given three tests during the first and last week of the Motor Development Clinic session that included: a handwriting test, Developmental Test of Visual Perception and the Motor Free Visual Perceptual Test. Testing and practice using the Robotic Haptic Device occurred before or after the child's motor development session for 30 minutes once a week. The clinic's environment for the pre and post testing was different although it was the same location and test administrators. For the pretest, children were alone in the testing rooms, and for the post testing, the Motor Development Clinic was in session. It is speculated that the additional noise and distractions could account for the lack of increase in scores from pre to post test. However, for the experimental group there was a slight increase in handwriting and over all Visual Motor Integration (see Table 3). Fig. 15 and 16 show the Evaluation Function's results (%Accuracy, and Time in seconds) for LeLe with and without Inertia effect employed. Children executed the tasks manipulating the PHANToM's stylus in the X-Y plane (air) while observing the computer's monitor. The overall observation is that children did improve in both Accuracy and time.

5 Current Work

The simulation of the IDSS described in [5] with the results obtained from experiments is being successfully implemented. The accuracy and time for each trial run was determined using a defuzzification process. Using these values and the current state of the subject, the program decides the next state for the subject's test.



Design: Design Complex Labyrinth
Plane: X-Y Plane (In Air)

```
Reading "d2/d2"...OK
Creating Edge Image to
"d2/d2_edge.pgm"...OK
Reading Ideal Path Image from
"d2/d2_ideal.pgm"...OK
Reading Ideal Path Image from
"d2/d2_basepoint.pgm"...
Reading Result from "result.pts"...OK
Reading Result from
"d2/d2_result.pts"...OK
```

Elapse Time: 109.813 Seconds

Step1

```
Total Number of Points: 3998
Total Number of Valid Points: 3274
Percent: 81.89 %
```

Step2 Tolerance: 0

```
Total Number of Valid Points: 3274
Total Number of Ideal Points: 573
Percent: 17.5 %
```

Figure 13. Evaluation Function Result for Complex Labyrinth

6 Conclusions

The project has shown that the use of a robotic haptic interface with assistance functions can help reduce the execution time and increase accuracy for the tests chosen to assess eye-hand coordination problems. The edge-detection based evaluation function proved to be successful in determining these data for the IDSS to make the suggestion on the next task. While the change shown in Table 2 is not significant, it begins to establish the need for more testing and work using the Robotic Haptic Device for children with a disability. The study presented in [4] used only children with identified eye-hand coordination disability, while this study's children had a number of disabilities. The Cal Poly Motor Development is non-categorical, which means the identification of the disability is not important and all children have some type of motor skills delay. It may be worthwhile in future studies to break the children into categories to determine if there is a difference in the groups.

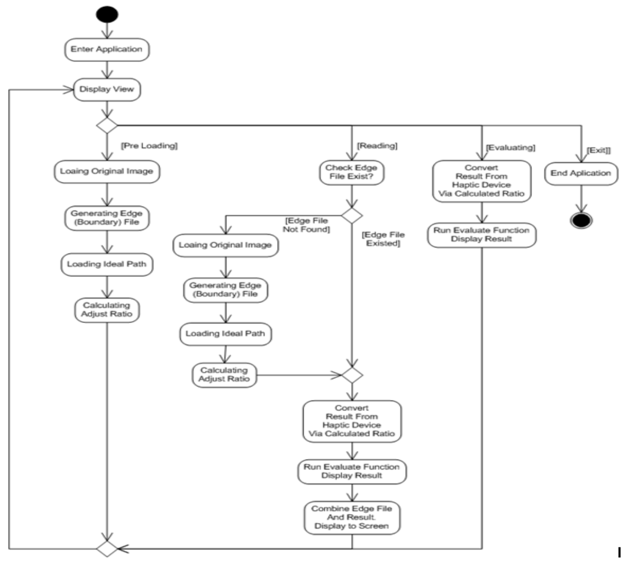

Enter Application
Display View
[Pre Loading]
[Reading]
[Evaluating]
[Exit]]
Loaing Original Image
Generating Edge (Boundary) File
Loading Ideal Path
Calculating Adjust Ratio
Check Edge File Exist?
[Edge File Not Found]
[Edge File Existed]
Loaing Original Image
Generating Edge (Boundary) File
Loading Ideal Path
Calculating Adjust Ratio
Convert Result From Haptic Device Via Calculated Ratio
Run Evaluate Function Display Result
Combine Edge File And Result Display to Screen
Convert Result From Haptic Device Via Calculated Ratio
Run Evaluate Function Display Result
End Aplication

Figure 14. The Flow Graph of the Evaluation Function's Main Program

TABLE 2. THE RESULTS COMPARING THE TWO GROUPS OVER-ALL AVERAGES

Group	Handwriting	Total score of Visual Motor Integration Test	Motor Reduced Performance - subtest	Visual Motor Integration - subtest
Control	No change	-1.7 decrease in performance	No change	-2.4 decrease in performance
Experimental	+.4 increase in performance	+1.7 increase in performance	-2.7 decrease in performance	No change

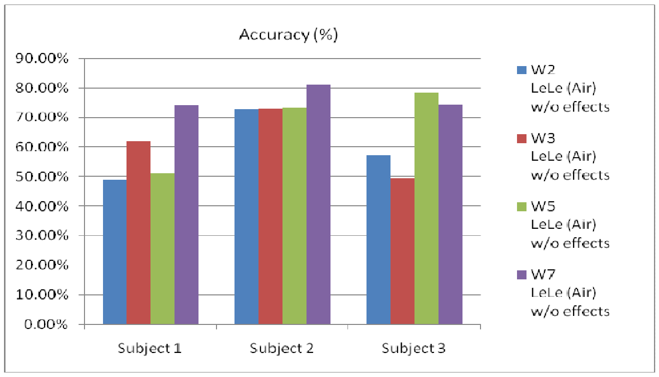


Figure 15. Result for LeLe Pattern without Effects (Accuracy)

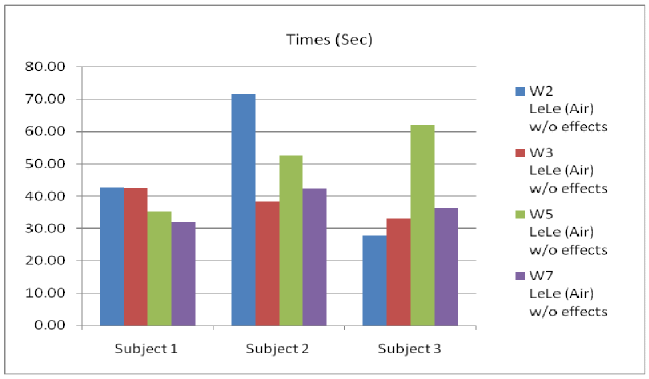


Figure 16. Result for LeLe Pattern without Effects (Time)

7 References

[1] B. Plimmer, A. Crossan, S. A. Brewster, R. Blagojevic, "Multimodal collaborative Handwriting Training for Visually-Impaired People", Chi 2008, Florence, Italy, ACM, pp. 393–402.

[2] J. Canny, "A Computational Approach To Edge Detection", IEEE Trans. Pattern Analysis and Machine Intelligence, 1986, pp. 679–698.

[3] M. A. Eid, M. Mansour, A. El-Saddik, R. Iglesias, "A Haptic Multimedia Handwriting Learning System", ACM Multimedia EMME Workshop 2007, pp.103–108.

[4] N. Pernalete, R.Gottipati, V.Kolipakam, J.Tipple, S.Edwards,R.Dubey, "Eye-Hand Coordination Assessment/Therapy Using a Robotic Haptic Device", The 9th IEEE International Conference on Rehabilitation Robotics ICORR, Chicago, June 2005.

[5] N. Pernalete, R. Gottipati, S. Edwards, D. Janiak, J. Haskin, R.V. Dubey, "Integration of an Intelligent Decision Support System and a Robotic Haptic Device for Eye-Hand Coordination Therapy", The 10th IEEE International Conference on Rehabilitation Robotics ICORR, Netherlands, June 2007.

[6] N. Vishnoi, C. Narber, Z. Duric, N. L. Gerber, "Guiding hand: a teaching tool for handwriting", ICMI 2009, pp. 221–222.

[7] OpenHaptics Toolkit Programmer's Guide.

[8] R. Arsenault, C. Ware. "Eye-hand co-ordination with force feedback", ACM CHI 2000 Proceedings.

[9] S. Edwards, D. Buckland, J. McCoy-Powlen, "Developmental and Functional Hand Grasps", New Jersey, Slack, 2002.

[10] F. Mussa-Ivaldi A., J. Patton L., "Robots can teach people how to move their arm," International Conference on Robotics and Automation. pp. 300-305, 2000.

[11] A. Bardorfer, M. Munih, A. Zupan, A. Primožič, "Upper Limb Motion Analysis Using Haptic Interface," IEEE/ASME Transactions on Mechatronics, vol. 6, no. 3, pp. 3721-3726, September 2007.

A Real-time Algorithm for Search-based Motion Synthesis

Chao Peng, Yong Cao
Department of Computer Science, Virginia Tech, Blacksburg, VA, USA

Abstract—*Search-based approach, such as motion graph, is one of the widely used motion synthesis approaches. However, the performance of the most of existing motion search algorithms do not scale well with respect to the size of the input constraints. We present a novel search algorithm, Single Stepping, that can provide a real-time solution when searching a large motion database. The algorithm starts with a greedy search result and can enumerate all possible optimal solutions in linear time with respect to the input constraints. A smoothness metric is also introduced to find the best resulting motion from these optimal solutions. By comparing with standard branch and bound algorithm, we show that our approach can generate resulting motions with the same quality within a fraction of search time.*

Keywords: motion capture, motion synthesis, full-body animation

1. Introduction

Creating realistic character animation is one of the main challenges in the area of Computer Graphics. Data-driven approaches with motion capture have been proven to be able to generate highest quality character motion to date. Search-based algorithms, one of the commonly used data-driven approach, however, suffer from their poor execution performance when searching on a large motion database with thousands of motion clips.

Most of these algorithms take a set of input constraints and try to synthesize the matching result. A sequence of motion segments are picked by the searching algorithm and then blend together over the transition points between them. The research for search-based motion synthesis primarily aims at the smooth connection over the transition points, or simply tries to eliminate unnecessary transitions by pulling out the already connected motion segments in the motion database. The challenge here is to design efficient algorithms that scale well with respect to the complexity of the database and the size of input constraints.

Cao et al. [1] has presented a greedy and also optimal search algorithm, which provides a linear time search for motion capture-based speech motion synthesis. This work demonstrates a great research direction on search-based motion synthesis. However, it can not find all possible optimal results. In this paper, we present a *Single Stepping* algorithm and a novel data structure to allow an efficient search of all optimal results. In our framework, we represent the recorded motions as a sequence of annotations, such as “stretch arms, twist and stoop", where every annotation indicates the frames appearing sequentially. Then our novel data structure encapsulates the motion database into a *Double-pointer Trie* structure. Given an input, a sequence of annotations describing a desired motion, we search *Double-pointer Trie* to find all optimal matching results. Using *Single Stepping* algorithm, the system performs in linear time for each matching result.

[1]This paper is submitted to the *2011 International Conference on Computer Graphics and Virtual Reality*, CGVR’11.
[2]Chao Peng is the contact author of this paper.

2. Related Work

Researchers have studied on a number of algorithms for motion editing to generate a novel motions. By warping motion parameter curves, Andrew Witkin et al. [2] introduced motion warping technique to edit captured motions. Michael Gleicher [3] presents a spacetime constraints solver, where the constraints are described as character positions at a specific time. The solver derives from an objective function that considers the entire motion simultaneously and computes optimal physical motions. Gleicher [4] and Lee et al. [5] describe the optimization methods of retargeting motions onto a new character. Hecker et al. [6] extends these ideas by studying on varied morphologies.

To create more complex motions, researches have studied on how to direct a character perform multiple actions by synthesizing a corpus of captured motions while preserving the quality of the original data. Motions can be represented as a graph structure that contains possible transitions between poses. A novel motion is built from walking on the graph: following a path, going to a particular position and playing a particular action at a specific time. Kovar et al. [7] construct a directed graph storing transition points and present branch and bound search algorithm for building the roadmap of the graph. Arikan et al. [8] represent motion as a hierarchy of graph and execute randomized search algorithm to synthesize motions interactively. Lee et al. [9] extend the graph approach with a two-layer structure: low-layer keeps the details of original motions; high-layer generalizes the data by clustering. Gleicher et al. [10] address that the process of searching can be more efficient by constructing a cyclic graph. Lau et al. [11] also present that a method for pre-computing search trees to improve graph search. In contrast to motion graph techniques, Kovar et al. [12], [13] introduce Parametric Motion Blending, where parameterized motions are built from a set of identified blendable motions in data. Blending methods are improved by registration curves, involving timing, local coordinate frames and input constraints. Mukai et al. [14] use an interpolation method to blend similar motion samples by determining optimal values from control parameters. Arikan et al. [15] introduce a dynamic programming algorithm to synthesize motions while satisfying a set of annotations, where every pose is annotated a proper motion type then constructed into a graph structure. The time complexity of their approach is polynomial. In data processing, unlike the method in Arikan et al.[15], we annotated motions by marking motion segments.

Our research benefits a lot from the data-driven approach of facial motion synthesis in Cao et al. [1], where a desired facial motion can be generated by using greedy search algorithm. Although Cao et al. [1] provides a real-time approach, it can not explore complete database to find all optimal results. In our system, we present an efficient algorithm to find all optimal synthesized motions based on a novel data structure that we call *Double-pointer Trie*.

3. Problem Statement

Search-based data driven approaches, such as motion graph, produced some of the high fidelity character animation results to date. The goal is to synthesize new motions that satisfy certain input constraints by searching an example motion database. One of the most important criteria for the search is to minimize the amount of blending/interpolation over transition points between example motions, where unrealistic animations, such as foot-sliding or robotic actions, can be introduced.

To give a formal definition of the search-based algorithms, let us denote the database Ω as a collection of n example motion *clips* $M_1, M_2, \ldots, M_n$. Each motion clip, M_i, is naturally segmented into m_i motion *segments*, $M_i = S_1^i S_2^i \ldots S_{m_i}^i$, if there are $m_i - 1$ possible transition points in the clip. Each motion segment, S_j^i is annotated with a *tag*, a_j^i, which comes from a vocabulary of labels, $L = a_1, a_2, \ldots, a_p$, indicating user-defined “meanings” for a motion segment, such as “walk”, “pick”, or “bend”. Therefore, each motion clip can be represented as an ordered list of segment-tag pairs:

$$M_i =< S_1^i, a_1^i >< S_2^i, a_2^i > \cdots < S_{m_i}^i, a_{m_i}^i > . \quad (1)$$

Problem 1. Given a motion database $\Omega = \{M_i | i \in [1, n]\}$, where M_i is a clip of motion which consists of a sequence of tagged segments as defined in Equation 1, and a user defined tag list $Q_{input} = a_1 a_2 \ldots a_l$, find a list of tagged motion segments $< S_1, S_2 \ldots S_l >$ from Ω such that the tag for segment $S_i (i \in [1, l])$ matches the input tag a_i.‘ For example, if the input tag list is “run, walk, bend, pick up, walk, run”, a sequence of motion segments will be generated that follow the order of the actions specified by the input.

A simply solution to Problem 1 is to create a list of motion segments for each annotated tag and pick one of them as the resulting segment for each tag in the input tag list. However, the problem is that the transition between two segments can be discontinuous and unnatural. We need to search for motion segments that are adjacent to each other in the motion database so that the transitions between them are smooth. Specifically, in result motion segments $< S_1, S_2 \ldots S_l >$, the two adjacent segments S_i and S_{i+1} should appear to be adjacent to each other in one of the motions M_j is database Ω. If there are not two adjacent motion segments for two adjacent input tags, a discontinuous *jump* is introduced. The problem addressed in this paper, then, can be formulated as follows:

Problem 2. Find a solution to Problem 1 with minimal number of *jumps*.

4. Overview

The solution to Problem 2 requires a constrained search algorithm. The existing search algorithms, such as branch and bound [7] or dynamic programming [8], do not scale well with respect to the size and complexity of the example motion database, and the number of annotations/tags from the user input.

Cao et.al [1] provides an efficient search algorithm for speech motion synthesis. In this work, the tags are phoneme informations from a speech motion. A greedy search algorithm is proposed which can generate one optimal solution in linear time (with respect to the length of input tag list). However, there exists multiple optimal solutions with the minimal number of *jumps*. We shall use addition criteria to find the smoothest resulting motion from this set of optimal solutions.

In this paper, we present an efficient search algorithm that can find all optimal solutions for Problem 2. Each of them is a list of matching motion segments. We then use a defined smoothness metric to synthesis the best resulting motion from these optimal solutions.

Given a user defined input tag list as input, our algorithm takes the following four steps to generate a list of motion segment that matches the input annotation list, and has the smoothest motion transitions between these segments. The overview of these steps are also shown in Figure 1.

1) Organize the example motion database using our *Trie* data structure, as shown in Figure 2.
2) Use Greedy search algorithm introduced by Cao et.al [1] in Trie structure to find one optimal solution to Problem 2.
3) Apply our proposed *Single Stepping* algorithm to find all possible optimal solutions in a modified Trie structure, *Double-pointer Trie* based on the solution from the previous step.
4) Use our defined metric to find the smoothest resulting motion from all optimal solutions.

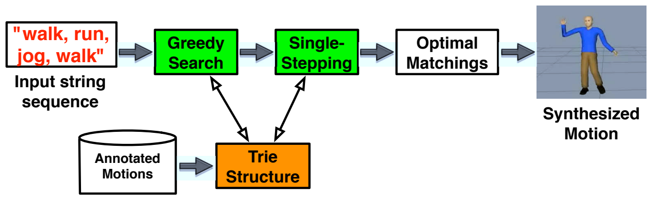


Fig. 1: Overview of our system.

5. Trie Data Structure

We organize the example database Ω in a novel data structure, called *Trie*, which is designed to represent transition information between different motion segments in the database. First, we create a graph node $P^i_{b,e}$, called motion *piece*, for any sequentially appeared sub-list of motion segments in motion M_i, as defined in Equation 1,

$$P^i_{b,e} =< S^i_b, a^i_b >, < S^i_{b+1}, a^i_{b+1} >, \dots, < S^i_e, a^i_e >,$$

where b is the start index of the sub-list, and e is the end index of the sub-list. An edge is then built between two pieces $P^i_{b,k} \rightarrow P^i_{b,k+1}$, since the two segments S^i_k and S^i_{k+1} appear sequentially in motion M_i. In general, the Trie structure for each motion M_i is similar to a suffix tree [16].

The number of nodes in Trie affects the efficiency of our search algorithm. To eliminate unnecessary nodes from the structure, we consider two pieces $P^k_{b,e}$ and $P^{k'}_{b',e'}$ to be the same if they contain the same tag list, despite the difference between their embedded motions. These two pieces are merged in Trie as the same node, and defined as a *Fragment*:

$$F =< \boldsymbol{A}, \{P_1, P_2, \dots P_c\} >, \qquad (2)$$

where A is a list of annotated tags, and $P_i, i \in [1, c]$, is a piece that has the same tag list as A.

In Figure 2, an example of Trie is illustrated. To simplify terminology, we present tags using English letters. A database with two motions is used in this example, one has a tag list “abca" and another has “cac".

6. Greedy Search Algorithm

To provide an efficient and optimal solution to Problem 2, Cao et al. [1] provided an approach based on a greedy search algorithm. Because our proposed *Single Stepping* algorithm takes the result of this greedy algorithm as the starting point. we will give an introduction of the algorithm and demonstrate the execution of the algorithm on the introduced Trie structure.

The greedy algorithm takes an input tag list $Q = a_1 a_2 \dots a_l$, and starts at the fragment node with the tag a_1. It then follows the edges and trying to reach fragment a_1, a_2. If the edge does not exist, it will go back to the beginning of the Trie and start over with a_2. The algorithm ends with all tags in the input has been followed. The motion segments can be generated based the path it goes through the Trie.

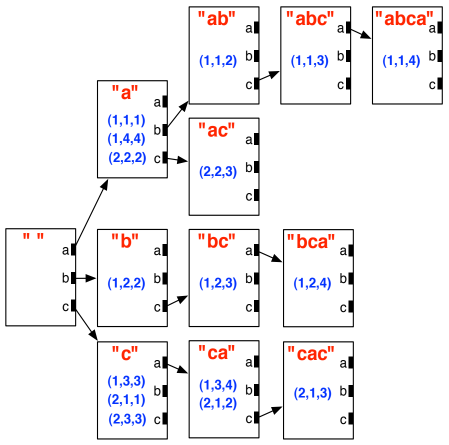


Fig. 2: An example of Trie structure built from two annotated motions with tag lists “abca" and “cac".

Basically, the algorithm tries to follow the longest sub-list in the database until it reaches a jump. Obviously, this is greedy approach. But Cao et al. [1] has proven it provides an optimal solution with the minimum number of jumps. We review the algorithm as follows: for each tag a_i in Q, the algorithm matches it by walking the trie until it can not walk any more. Then we output the content of that fragment, go back to the root fragment, and continue. This generates a *greedy-matching* result which has minimized jumps. The details of the algorithm is listed in Algorithm 1.

Algorithm 1 Greedy-Searching (Q)

Input: search sequence $Q = a_1 a_2 \cdots a_l$
Output: fragment sequence $S_F = F_1 F_2 \cdots F_n$

```
1: $n \leftarrow 0$
2: for each $a \in Q$ do
3:   if $n = 0$ or$F_n.R[a]$ does not exsit then
4:     $n \leftarrow n + 1$; $F_n \leftarrow$ root
5:   end if
6:   $F_n \leftarrow F_n.R[a]$
7: end for
```

For example, given the search sequence Q = “aca", the algorithm walks the Trie of Figure 2 from “ "(root) to “a" to “ac". Since there is no “aca", it would output the content of fragment “ac" and move back to the root. To continue, it starts from “" to “a" and output the content of fragment “a". At the end, a greedy-matching result is found.

As we mentioned before, the problem of the greedy search algorithm is that it can only find one solution with minimal number of jumps.

7. Single Stepping Algorithm

Although greedy search minimize the number of jumps, it can not find all possible solutions. In the previous example in Section6, greedy search gives the result that contains two fragments “ac" and “a". However, in Figure 2, we can find another result, “a” and “ca", which has the same number of jumps. However, the greedy algorithm can only find the first one. In this section, we want to provide a novel approach, which can find all optimal solutions with the minimal number of jumps.

7.1 Double-pointer Trie structure

In Section 5, we build a Trie by growing a piece of an example motion from left to right. To explore all possible solutions, we extend the Trie structure by adding another pointer, which reflects a piece growing from right to left.

The new Trie has two sets of pointers for each fragment. As shown in Figure 3, the R pointers are the same the pointers as the original *Trie* structure: they present pieces growth to the right. In Figure 3, following the $R[c]$ of the fragment “ab", the Trie leads to fragment “abc". The other set of pointers are L pointers, which represent pieces growth to the left. That is, following the $L[a]$ pointer of the fragment “bc", the Trie leads to fragment “abc". For the root fragment, its L and R pointers are identical.

If N_F is the total number of fragments in a Trie, we can formally defined *Double-pointer Trie* as follows:

$$\begin{aligned} Trie :&< Fragments, Rpointers, Lpointers >, \\ Fragments &= \{F_i\}, (1 \le i \le N_F), \\ Rpointers &= \{R_i : F_s \to F_t\}, (1 \le s, t \le N_F), \\ Lpointers &= \{L_i : F_s \leftarrow F_t\}, (1 \le s, t \le N_F). \end{aligned} \tag{3}$$

Moreover, each fragment but root maintains LP and RP parent pointers, which are the inverse of L and R pointers, respectively. For example, given a fragment labeled with “aσb", where σ is some string of letters(annotations), we have “aσ".$R[b]$ = “aσb"; “σb".L[a] = “aσb"; “aσb" $.LP =$ “σb"; “aσb" $.RP =$ “aσ".

7.2 Single Stepping

In this section, we present our *Single Stepping* algorithm which can enumerate all fragment sequences that match an input with the minimal number of jumps.

Definition 1 (Single Stepping) Take two fragment sequences that match the same input and have the same number of fragments, call them $S_F = [F_1, \cdots, F_n]$ and $S_F^{'} = [F_1^{'}, \cdots, F_n^{'}]$. Then $S_F^{'}$ is a single step from S_F, written $S_F \mapsto S_F^{'}$, if the following two properties hold.

1) There exists some $I < n$ so that $F_i = F_i^{'}$ for all $i < I$ and all $i > I + 1$.
2) $|len(F_I) - len(F_I^{'})| = 1$. That is, F_I is either one annotation longer than $F_I^{'}$ or one annotation shorter.

From these properties we can get $|len(F_{I+1}) - len(F_{I+1}^{'})| = 1$ as well. Clearly single stepping is symmetric: if $S_F \mapsto S_F^{'}$, then $S_F^{'} \mapsto S_F$.

Theorem 1 (Sufficiency of Single Stepping) If S_F and $S_F^{'}$ are two fragment sequences that match the same of input and use as few fragments as possible (minimized jumps), then there exists a single-stepped chain transforming S_F to $S_F^{'}$ (See Appendix A for the proof):

$$S_F = S_F^1 \mapsto S_F^2 \mapsto S_F^3 \mapsto \cdots S_F^m = S_F^{'}.$$

Using the single-stepping property, we introduce an algorithm that can enumerate all optimal matchings with the minimal number of jumps. First, We need to find a greedy-matching using Algorithm 1, defined in Section 6. We also maintain a pointer to the Trie for each searched fragment.

Once we have the greedy-matching, we alter it step by step with a recursive procedure, working from right to left. Every optimal-matching is generated exactly once, and the proposed Single Stepping algorithm is illustrated in Algorithm 2.

Algorithm 2 Single-Stepping (S_F, i)

```
1:  if $i = 1$ then
2:      output $S_F$ as an optimal-matching
3:  else
4:      Single-Stepping $(S_F, i - 1)$
5:      $T_i \leftarrow F_i$
6:      $T_{i-1} \leftarrow F_{i-1}$
7:      while $F_i.L[rightmost(F_{i-1})]$ exists do
8:          $F_i \leftarrow F_i.L[rightmost(F_{i-1})]$
9:          $F_{i-1} \leftarrow F_{i-1}.RP$
10:         Enumerate-Matching-Step $(S_F, i - 1)$
11:     end while
12:     $F_i \leftarrow T_{i-1}$, $F_i \leftarrow T_{i-1}$
13: end if
```

To start the algorithm, we call *Single-Stepping*(S_F, n), where S_F is the output of *Greedy-Searching*(S_Q). In this algorithm, T_i is a temporary pointer to save fragment F_i, and the $rightmost(F_i)$ function returns the rightmost tag in F_i. Since it takes $O(n)$ time to generate one optimal-matching, the time complexity for each matching is $O(n)$.The first output is the greedy-matching; the last is a reversed greedy-matching, where F_n, the fragment at the end, matches the last subsequence of S_Q as long as possible.

7.3 Post processing

After searching the Double-pointer Trie structure, we have found all optimal solutions (optimal-matchings). For an optimal-matching $S_F = [F_1, \cdots, F_n]$, a motion is assembled by choosing one piece $P_{b,e}^k$ from each F. Since each piece, $P_{b,e}^k$, points to a set of continuous segments, it represents a snippet of a original motion. If R_i is the number of pieces in F_i, the number of motions generated from an optimal-matching is the product of all $R_i (1 \le i \le n)$.

In order to produce continuous motions and evaluate their qualities, we preform the following two steps: Animation Weighting and Motion Interpolation.

Animation Weighting. Each $P^k_{b,e}$ reflects a snippet of original motion. By connecting the chosen pieces together, they yields $n-1$ jumps, and the synthesized motion may not be smooth at these transition points/jumps. In order to find the best synthesized motion, we evaluate their qualities by giving each of them a weight. We calculate each weight with a sum of the differences of poses presenting at transition points/jumps.

In our system, motion data is specified as the root joint positions and joint rotations of each joint in a skeleton at every frame. If two pieces, $P^k_{b,e}$ and $P^{k'}_{b',e'}$, are chosen from F_i and $F_{i+1}(1 \le i \le n-1)$, the difference between two poses is measured as joint angle difference at the last frame of S^k_e and at the first frame of $S^{k'}_{b'}$.

Our weighting metric incorporates a consideration that the pose is essentially unchanged if it is applied a 2D aligning transformation(motion alignment). Thus, the calculation of a weight, W, is formalized as:

$$W = \sum_{i}^{n-1} D_i, \qquad (4)$$

$$D_i = \|S^k_e[last] - T_{\theta,x,z} \times S^{k'}_{b'}[first]\|, \qquad (5)$$

where $S^k_e[last]$ denotes the last frame of S^k_e; $S^{k'}_{b'}[first]$ denotes the first frame of $S^{k'}_{b'}$; D_i is the difference between the poses; $T_{\theta,x,z}$ denotes the 2D aligning transformation, which rotates a pose about y(vertical) axis by θ degree, then translates it by(x, z) on 2D floor.

After we calculated all animation weights, we sort the synthesized motions in ascending order of weights. Therefore, we can find which motion has the smoothest transition among all optimal-matchings.

Motion Interpolation. In order to deal with the discontinuous motion introduced by jumps, motion interpolation can make animation more continuous by generating intermediate frames. Interpolation takes place over a predefined number of frames, h (the default is 15). Before blending the frames,we first apply Dynamic Time Warping(DTW) to determine the similarity of a pair of poses in the duration of the blending window. Then those h interpolated poses are generated using Equation 6.

$$\begin{aligned} G[i] = \quad & (1-\tfrac{i-1}{h-1}) \times S^k_e[last-h+i] + \\ & (\tfrac{i-1}{h-1}) \times S^{k'}_{b'}[first+i-1], (1 \le i \le h), \end{aligned} \qquad (6)$$

where $G[i]$ is the ith interpolated pose.

8. Experience and Results

In our experiment, we choose a set of motions from CMU motion capture database. In order to evaluate Single

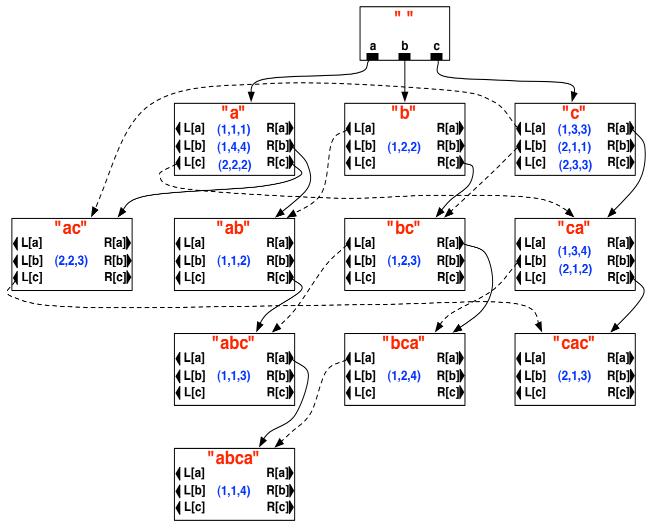


Fig. 3: *Double-pointer Trie* is an extension of Fig. 2, where the dash edges are L pointers created by growing substring on the left.

Stepping algorithm, we exam both performance in searching Trie structure and quality of the output results by comparing with Branch and Bound(BB) and Greedy algorithms.

In the experiments, BB algorithm traverses the dataset with all possible jumps ($[0, l-1]$), and guarantees finding all optimal-matchings. This approach can be described as:

1) We start searching with the assumption that optimal-matchings do not have jumps.
2) If there is not any optimal-matching existing with current assumption, we increase the number of jumps by one, and find if any exists.
3) The search ends until we find that, with certain number of jumps, there is at least one optimal-matching extracted from dataset.

Three of our experiments are summarized in Tables 1-3. In each table, given the number of annotations, we randomly select 5 different search sequences. Based on the approaches described in Section 6 and 7, “Min. Jumps" shows the minimal number of Jumps for each input search sequence; "Search time" indicates the execution time of finding optimal-matchings in Trie Structure; and “Matchings" shows the number of optimal-matchings/fragment sequences. Referring to Section 7.3, “Synth. motions" shows the number of synthesized motions from optimal-matchings; and “Lowest weight" indicates the best-quality synthesized motion (The lower the weight, the better the quality of a synthesized motion). We also show the synthesis times in “Synth. Time" column, which include both the “Search time" and the post-processing time.

Performance. To measure performance of the algorithms, we compare their “Search time". Although Greedy is the fastest algorithm, it cannot find all optimal-matchings. As showing in “Matchings" column of tables, Greedy always find only one optimal-matching. In contrast, the other two algorithms find the complete set of the optimal-matchings. The tables also show the exponential complexity of BB with respect to the length of the input sequences. Single Stepping has a very similar search time as Greedy, proven to be an efficient algorithm to find all optimal matchings.

Quality. As shown in “Lowest weight" column, Greedy does not guarantee to find the best-quality synthesized motion; however, the other two algorithms do. For example, the minimal number of jumps of the third search sequence in Table 3 is 9. For this search sequence, Single Stepping and BB generate the best motion with the lowest weight of 2832.7; but the lowest weight based on Greedy is 2960.4, which indicates it does not synthesize the best quality motion.

We show several frames of a synthesized motion in Figure 4 and a screen capture of our animation system in Figure 5.

Table 1: The number of the input annotations is 10.

Algorithms	Min. Jumps	Search time (sec)	Matc-hings	Synth. mo-tions	Synth. Time (sec)	Lowest weight
	7	0.001	1	128	0.797	2364.8
	5	0.001	1	24	0.109	2004.0
Greedy	5	0.001	1	8	0.031	2106.4
	4	0.001	1	4	0.015	1760.5
	6	0.001	1	80	0.422	1842.8
	7	0.001	1	128	0.781	2364.8
	5	0.001	2	48	0.219	1898.0
Single-	5	0.001	1	8	0.031	2106.4
Stepping	4	0.001	1	4	0.016	1760.5
	6	0.001	2	160	0.859	1842.8
	7	0.016	1	128	0.813	2364.8
	5	0.016	2	48	0.219	1898.0
BB	5	0.016	1	8	0.062	2106.4
	4	0.016	1	4	0.032	1760.5
	6	0.016	2	160	0.875	1842.8

Table 2: The number of the input annotations is 15.

Algorithms	Min. Jumps	Search time (sec)	Matc-hings	Synth. mo-tions	Synth. Time (sec)	Lowest weight
	8	0.001	1	256	1.828	3029.4
	9	0.001	1	960	8.094	3081.3
Greedy	7	0.001	1	8	0.047	2578.7
	8	0.001	1	384	2.766	2764.8
	10	0.001	1	1152	10.859	3960.8
	8	0.001	2	512	3.766	3028.3
	9	0.002	6	5760	65.453	3081.3
Single-	7	0.001	3	64	0.391	2393.8
Stepping	8	0.001	4	768	5.734	2669.2
	10	0.002	2	2304	23.500	3924.4
	8	0.344	2	512	4.079	3028.3
	9	0.422	6	5760	66.047	3081.3
BB	7	0.281	3	64	0.672	2393.8
	8	0.375	4	768	6.062	2669.2
	10	0.453	2	2304	23.797	3924.4

Table 3: The number of the input annotations is 20.

Algorithms	Min. Jumps	Search time (sec)	Matc-hings	Synth. mo-tions	Synth. Time (sec)	Lowest weight
	12	0.001	1	768	8.406	4371.9
	12	0.001	1	720	7.875	4657.6
Greedy	9	0.001	1	40	0.313	2960.4
	11	0.001	1	96	0.937	4031.5
	11	0.001	1	960	9.782	3939.6
	12	0.002	1	768	8.408	4371.9
	12	0.002	2	1200	13.484	4657.6
Single-	9	0.001	2	80	0.641	2832.7
Stepping	11	0.002	2	336	3.312	3919.7
	11	0.002	1	960	9.788	3939.6
	12	13.203	1	768	21.610	4371.9
	12	13.203	2	1200	26.671	4657.6
BB	9	7.016	2	80	7.656	2832.7
	11	11.688	2	336	15.000	3919.7
	11	11.812	1	960	21.593	3939.6

9. Discussion and Future works

We have presented an efficient, search-based approach for realistic full-body motion synthesis. We explain that *Single-Stepping* algorithm performs a linear time per matching step with respect to the number of fragments in the greedy-matching.

Like most data-driven approaches, our approach is unable to synthesize poses that do not exist in the database. In order to generate diverse synthesized motions, hundreds of more motions might be required in database. Our Single-Stepping method focuses on finding sequences of fragments with the minimal jumps. By preprocessing the database, each motion is tagged with a sequence of annotations. The less of number of annotation used for a motion, the more jumps would be introduced into the optimal-matchings. In an extreme case of tagging each motion with a single annotation, Trie structure would contain only one depth-level fragments extending from the root, and we could find a single result of S_F, whose length n would be identical the length l of search sequence S_Q. Thus, in such extreme case, since each Q in S_Q is associated with a single F in S_F, the minimal jumps are meaningless.

In our research, preprocessing database is time consuming. it requires experienced animators to tag each motion with appropriate annotations. Arikan et.al [8] provide a machine learning approach requiring less manual works to annotate motions. In the future, we plan to investigate such a way to

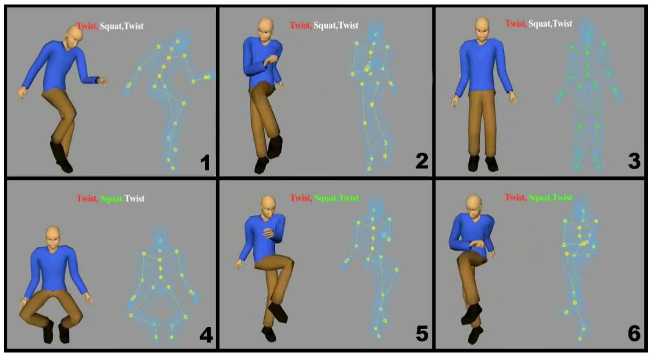
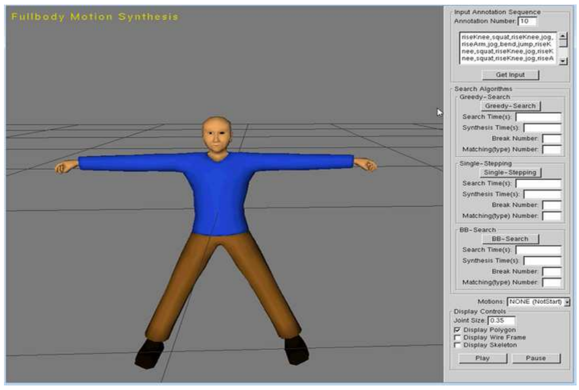


Fig. 4: One of the synthesized motions generated by our system associating with the annotations of "Twist, Squat, Twist". In each image, the left side shows the pose of character model; the right side shows the underneath skeleton.



Annotation Number: 10

riseKnee,squat,riseKnee,jog,riseArm,jog,bend,jump,riseKnee,squat,riseKnee,jog,riseKnee,squat,riseKnee,jog,riseA

Motions: NONE (NotStart)

Joint Size: 0.35

[✓] Display Polygon

Fig. 5: The screen shot of our system. The panel on the right side consists of the input string sequences, Greedy, Single-Stepping and BB searching and the outputs of synthesized motions.

tag the database automatically.

Acknowledgements

This work is partially funded by National Science Foundation, IIS 0940723, titled “EAGER: Drummer Game: A Massive-Interactive Socially-Enabled Strategy Game”.

Appendix A: Proof of Theorem 1

We prove it by induction on n, the number of fragments in each matching result.

Base Case: $n = 1$. There is exactly one fragment, so $S_F = S_F^{'}$ and the single-stepped chain is degenerate.

Inductive Step: Consider the first fragment in each matching result, F_1 and $F_1^{'}$. If $F_1 = F_1^{'}$, then we immediately apply the inductive hypothesis to the rest of the fragments.

Otherwise, assume without loss of generality that $F_1^{'}$ is longer than F_1. Note that $F_1^{'}$ must be shorter than F_1F_2; for if it were as long or longer, then we could replace the two fragments F_1 and F_2 with a subsequence of fragments of $F_1^{'}$ to obtain a fragment matching with fewer fragments than S_F.

It’s easy to single-step S_F into a version S_F^* where $F_1^* = F_1^{'}$, by removing a single character from the beginning of $F_2^{(i)}$ and tacking it to the end of $F_1^{(i)}$ at each step. (The intermediate fragments are legal since they are subsequences of the legal fragments $F_1^{'}$ and F_2.) Once we have obtain S_F^*, we can apply the inductive hypothesis to S_F^* and $S_F^{'}$, since they have the same first fragment. (PROVED)

References

[1] Y. Cao, P. Faloutsos, E. Kohler, and F. Pighin, “Real-time speech motion synthesis from recorded motions,” in *SCA ’04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2004, pp. 345–353.

[2] A. Witkin and Z. Popovic, “Motion warping,” in *SIGGRAPH ’95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1995, pp. 105–108.

[3] M. Gleicher, “Motion editing with spacetime constraints,” in *SI3D ’97: Proceedings of the 1997 symposium on Interactive 3D graphics*. New York, NY, USA: ACM, 1997, pp. 139–ff.

[4] ——, “Retargetting motion to new characters,” in *SIGGRAPH ’98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1998, pp. 33–42.

[5] J. Lee and S. Y. Shin, “A hierarchical approach to interactive motion editing for human-like figures,” in *SIGGRAPH ’99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999, pp. 39–48.

[6] C. Hecker, B. Raabe, R. W. Enslow, J. DeWeese, J. Maynard, and K. van Prooijen, “Real-time motion retargeting to highly varied user-created morphologies,” in *SIGGRAPH ’08: ACM SIGGRAPH 2008 papers*. New York, NY, USA: ACM, 2008, pp. 1–11.

[7] L. Kovar, M. Gleicher, and F. Pighin, “Motion graphs,” *ACM Trans. Graph.*, vol. 21, no. 3, pp. 473–482, 2002.

[8] O. Arikan and D. A. Forsyth, “Interactive motion generation from examples,” *ACM Trans. Graph.*, vol. 21, no. 3, pp. 483–490, 2002.

[9] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard, “Interactive control of avatars animated with human motion data,” *ACM Trans. Graph.*, vol. 21, no. 3, pp. 491–500, 2002.

[10] M. Gleicher, H. J. Shin, L. Kovar, and A. Jepsen, “Snap-together motion: assembling run-time animations,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 702–702, 2003.

[11] M. Lau and J. J. Kuffner, “Precomputed search trees: planning for interactive goal-driven animation,” in *SCA ’06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 299–308.

[12] L. Kovar and M. Gleicher, “Flexible automatic motion blending with registration curves,” in *SCA ’03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2003, pp. 214–224.

[13] ——, “Automated extraction and parameterization of motions in large data sets,” *ACM Trans. Graph.*, vol. 23, no. 3, pp. 559–568, 2004.

[14] T. Mukai and S. Kuriyama, “Geostatistical motion interpolation,” *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1062–1070, 2005.

[15] O. Arikan, D. A. Forsyth, and J. F. O’Brien, “Motion synthesis from annotations,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 402–408, 2003.

[16] P. Weiner, “Linear pattern matching algorithms,” in *SWAT ’73: Proceedings of the 14th Annual Symposium on Switching and Automata Theory (swat 1973)*. Washington, DC, USA: IEEE Computer Society, 1973, pp. 1–11.

An Improve to Adaptive Triangulation Algorithm Based On Shepard Interpolation

JIANG Heng-heng [1], **LI Qi-min** [2], **ZHAO Shuang-ling** [2], **and TANG Bao-ping** [3]
[1,2]The State Key Laboratory of Mechanical Transmission, Chongqing University, Chongqing, China

Abstract - *This is a An adaptive triangulation algorithm for three dimensional unorganized point clouds is proposed in this paper, since the most existing algorithms are not very adaptable and are difficult to express the detail characters of the real surface well. In the proposed method，we combine 4D Shepard surface with multi-resolution analysis and implement the modified octree algorithm, which the curvature of each point in the point cloud is calculated. Then a hierarchical grid with adaptive resolution is constructed for generating a triangular mesh from point clouds. Experimental results show that the original algorithm can preserve more characters, but inefficient and not flexible; the improved algorithm is greatly advanced and generally applicable, and forms high quality triangle grid surface and reproduces initial 3D object's detail characters, which is suited to popularize in CAGD and surface modeling.*

Keywords: point clouds; triangulation; Shepard interpolation; octree searching; subdivision.

1 Introduction

Surface reconstruction from a sampling in the form of an unorganized cloud of points is widely applied in the CAD model reconstruction of reverse engineering. As one of the important methods of surface reconstruction, mesh reconstruction is always the research hotspot. Typically the point clouds are acquired by a laser scanner or by the photometric stereo methods, thus are often unorganized, noisy, and lack the differential quantities information. However, the differential quantities at the points such as oriented normals and curvatures play a crucial role in reconstructing the surface, which augmenting the points with orientation is a difficult task[1]. Various algorithms for triangular-mesh reconstruction have been developed in recent years, which are generally classified into three categories: approaches based on Delaunay triangulation, implicit functions, and region-growing approaches[2-7]. Though all of them can run well in some fields, the surface reconstruction is finished only on a single scale. So the acquired mesh can't give well expression to the local details.

Zhouwang Yang and Yeong-Hwa Seo[8] presented an adaptive triangular-mesh reconstruction algorithm for reconstructing a triangular mesh to approximate the underlying surface of a point cloud from the closed objects. This method constructs a hierarchical grid with an adaptive resolution and estimates the differential quantities of the underlying surface using a moving parabolic approximation (MPA). However, there are two obvious deficiencies in this algorithm: firstly, it is limited in many practical applications, for example, the point clouds of the thin plate; secondly, not only the efficiency of the mesh reconstruction is not actually high, but it takes much more time than the region-growing approaches, especially for the sparse point clouds.

Based on the above analysis, an improved adaptive algorithm for triangulation is presented in this paper. By the combination of the Shepard interpolation and the multiscale method, the hierarchical grids are created while the geometric information of the data points are computed, which not only reduces the complexity of the triangulation model and speeds up the process, but the quality of the mesh is right high.

2 Adaptive triangular-mesh reconstruction algorithm

With the traditional sampling methods of the grid, the whole point cloud is uniformly partitioned, which are easy to be programmed and simple in methodology. However, for the non-uniform point clouds, there are many disadvantages, for example, it is generally difficult to choose an appropriate resolution in such a manner that the holes in sparsely sampled areas can be efficiently filled without losing the genus of a closed surface and the details in the dense sampled regions, and, we cannot get obvious effect with larger resolution for the smooth region. When the datasets are larger in point cloud, more grids not only slower the processing speed of the model, but also expend a great deal of memory. The above factors can affect the sequent processing so that it is difficult to control the model.

Hence, Zhouwang Yang and Yeong-Hwa Seo employed a method to construct a hierarchical grid with adaptive resolution, which is based on repeated and selective subdivision of the region that is occupied by the input set of data points. The approaches are as follows: starting with the level-0 grid with resolution m generated from a uniform partition of the bounding cube, then applying an adaptive octree-based subdivision to the data cells, simultaneously estimating the differential quantities of the underlying surface of the unorganized points by using the moving parabolic approximation. The algorithm produces an approximating mesh with a resolution that is adapted to the local details of the target shape.

One example of an initial point cloud of the turbine blade is given in Fig. 1, including 81424 points. Fig. 2 and Fig. 3 show the comparison of mesh reconstruction by using the region-growing approach and the adaptive algorithm, respectively (Fig. 2 and Fig. 3 (a) show the final triangular meshes; (b), (c) and (d) are all rendering 3D graphics, in which the normals of the blue regions are pointing outwards, and the white ones are quite the opposite).

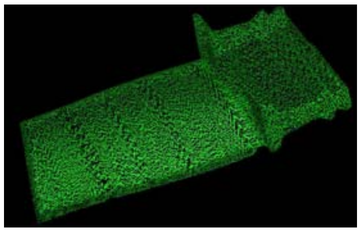
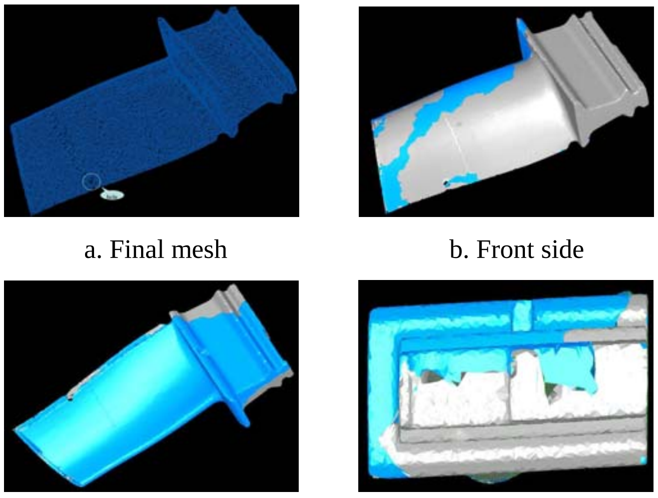
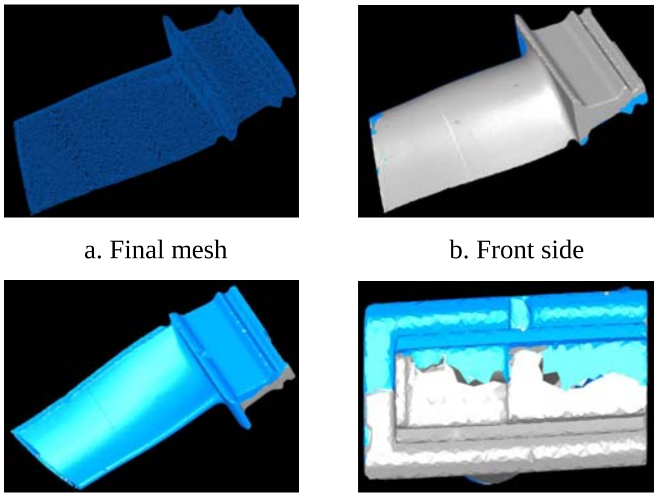


Figure 1. Point cloud (turbine blade)


a. Final mesh b. Front side

c. Back side d. Handle

Figure 2. Final mesh of region-growing approach (turbine blade)


a. Final mesh b. Front side

c. Back side d. Handle

Figure 3. Final mesh of adaptive algorithm (turbine blade)

The turbine blade belongs to the point cloud of the thin long plate, including more additional features, of which the curvatures change a great deal. From Fig. 2, because the region-growing approaches cannot distinguish the smaller features in the larger scale, the obvious holes are created in the edges and handles of the mesh. Furthermore, the normals of the reconstruction surface are non-uniform so that the normal consistency of the model need to be processed separately. These drawbacks cann't ensure the following steps.

From Fig. 3, adopting the adaptive algorithm, though the holes of the sides are efficiently filled, and the edges become smoother, which guarantees the meshing match in the reconstruction surface to a certain extent, the holes of the handle still remain. It is because the moving parabolic approximation is to locally fit the given point clouds using an osculating paraboloid for estimating the normals and curvatures that it reduces the accuracy of the differential quantities at the sharp features.

Fig. 4(a) shows the compressed point cloud of a cartoon. There are 276479 points in the initial point cloud, which leaving 37788 points after compressed. Fig. 4(b) and 4(c) show the comparison of the mesh reconstruction by using the region-growing approach and the adaptive algorithm, respectively. The experimental datum are showed in Tab. 1 and Tab. 2.

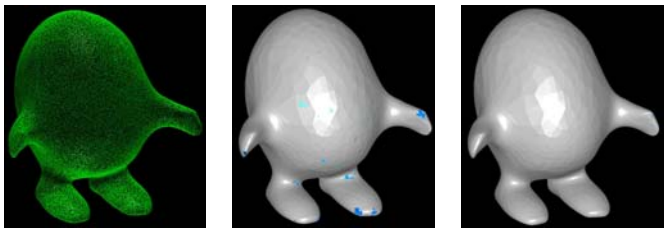

(a) Compressed point cloud (b) Region-growing (c) Adaptive algorithm

Figure 4. Final mesh (Cartoon model)

From Fig. 4 and Tab. 2, we outline that the region-growing approach can process much faster than the adaptive algorithm as the lower sampling density in the point cloud, although the latter is much smoother in detail than the former.

It is clear that the adaptive algorithm has a great advantage for the mesh reconstruction of the scattered-points, but there are also some deficiencies in many actual applications. For the above-mentioned analysis, this paper presents an improved adaptive triangulation algorithm based on the shepard interpolation, which can solve the insufficiencies of the original algorithm effectively.

Table 1
Experimental data of the region-growing approach (turbine blade and Cartoon)

Point cloud	Number of cells	Computation time (s)	Number of triangles	CPU use	PF use
Turbine blade	90×45×183 =741150	9.01s	237606	50%	646MB
Cartoon	59×80×77 =363440	3.04s	113349	50%	494MB

Table 2
Experimental data of the adaptive algorithm (turbine blade and Cartoon)

Point cloud	Number of cells	Computation time (s)	Number of triangles	CPU use	PF use
Turbine blade	110×110×110 =1331000	10.59s	237825	56%	835MB
Cartoon	82×82×82 =551368	5.29s	113316	54%	652MB

3 Adaptive triangulation algorithm based on the Shepard interpolation

We wish keep the geometric shape features of the complicated models and reconstruct the sharp features of the meshes. The original adaptive algorithm takes two following steps: first of constructing a hierarchical grid and then of estimating the differential quantities using moving parabolic approximation. Although the moving parabolic approximation has an ability of fast fitting, it is only used to estimate the curvature and normal of a single point in the surface. This requires a great lot of fitting calculations for estimating the whole point cloud. In this paper, we propose an improved adaptive triangulation algorithm based on the Shepard interpolation, which estimates the differential quantities while constructing a hierarchical grid. The shepard interpolation estimates the differential quantities based on the global surface model rather than on the local paraboloid, which substitute simple linear operation for abundant calculation of surface approximation. The experiments show the method is desirable in effectiveness and robustness.

The method for the Shepard interpolation-based mesh reconstruction from a point cloud involves the following stages:

1) A tight bounding box is constructed that is required to enclose the whole point cloud, and uniformly partitioned to generate the level-0 grid with resolution m.

2) If the number of sampling points is less than a user-specified threshold ε_1, performing the step 5; otherwise approximating sampling points for constructing a local surface by using the least square method.

3) Estimating the curvatures and normals of each point and of the cell center of grids in each level by using the adaptive Shepard interpolation-based method.

4) If the mean curvature of the data points and the cell center in one cell is greater than a specified threshold ε_2, the cell will be subdivided into eight subcells by using the improved Octree; otherwise performing the step 5.

5) In accordance with the maximum criterion of normal vectorial angle, output the resulting mesh as the final approximation to the point cloud.

Some key steps are explained in details as follows. In Section 3.1, we provide an overview of our initial mesh reconstruction scheme. In Section 3.2, we suggest an efficient procedure for constructing the hierarchical grid with the adaptive Shepard interpolation. In Section 3.3, we present the improved Octree solver.

3.1 Initial mesh for the point clouds

The original adaptive algorithm starts with the level-0 grid generated from a uniform partition of the bounding cube, applying an octree-based subdivision to the data cells. So when subdividing a long and narrow point cloud, it takes much more time or memory, as shown in Fig. 1-4.

In the paper, the bounding box is created to enclose the whole point cloud by a smallest hexahedron. We denote a point cloud as $P=\left\{p_j=\left(x_{p_j},y_{p_j},z_{p_j}\right)^T\right\}_{j=1}^{n}$, and let

$$\begin{aligned} x_{min} &= \min_{1\le j\le n} x_{p_j},\ x_{max} = \max_{1\le j\le n} x_{p_j}, \\ y_{min} &= \min_{1\le j\le n} y_{p_j},\ y_{max} = \max_{1\le j\le n} y_{p_j}, \\ z_{min} &= \min_{1\le j\le n} z_{p_j},\ z_{max} = \max_{1\le j\le n} z_{p_j}. \end{aligned} \tag{1}$$

The length of its sides are set to the following values:

$$\begin{cases} size_x = \dfrac{x_{max}-x_{min}+2\varepsilon}{x_{num}-2} \\ size_y = \dfrac{y_{max}-y_{min}+2\varepsilon}{y_{num}-2} \\ size_z = \dfrac{z_{max}-z_{min}+2\varepsilon}{z_{num}-2} \end{cases} \tag{2}$$

where ε is the given difference, x_{num}, , y_{num} and Z_{num} are the number of the cells along x,y and z axes, respectively. Then, we identify the set of the data cells, by calculating the index

$$(i_x = \left\lfloor \frac{x_i - x_{min}}{size_x} \right\rfloor, i_y = \left\lfloor \frac{y_i - y_{min}}{size_y} \right\rfloor, i_z = \left\lfloor \frac{z_i - z_{min}}{size_z} \right\rfloor) \tag{3}$$

of the cell that contains a data point $p \in P$.

With the method, the number of the cells is obviously fewer than the original algorithm. As shown in Tab. 1-2, the number of the cells in turbine blade has decreased by 4%, and the one in the sparse model has almost decreased by 85%. Our method reduces the searching time and the storage space during the process, thus the efficiency of the process is improved effectively.

3.2 Adaptive Shepard interpolation

To construct a hierarchical grid, each data cell generated during the subdivision process is marked as belonging to one of two groups: {data cell for further subdivision} and {data cell for non-further subdivision}. According to the curvature estimation principle, if the local mean curvature is less than the given threshold, the points in the small area are distributed more homogeneous and the cell needn’t be subdivided; or else, the distribution of the points is steep and the cell should be further subdivided. In this way, the number of the cells can be reduced as well as the geometric features of models are preserved well.

KE Ying-lin[9] proposed a method of the curvature estimation based on the 4D Shepard surface for estimating the differential quantities of the data points $\left\{p_j = \left(x_{p_j}, y_{p_j}, z_{p_j}\right)^T\right\}_{j=1}^{n}$ in point cloud, in which the 3D Shepard function was changed as follows:

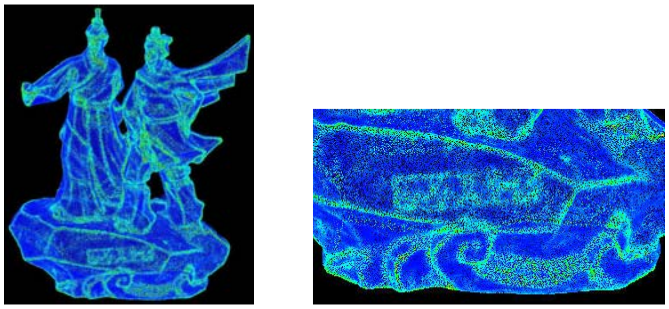
$$F(x,y,z) = \sum_{j=1}^{N} G_j(x,y,z)\delta_j \Big/ \sum_{j=1}^{N} G_j(x,y,z) \tag{4}$$

where

$$G_j(x,y,z) = d_j^{\mu}\ , \quad \mu = -2$$

$$d_j = [(x-x_j)^2 + (y-y_j)^2 + (z-z_j)^2]^{1/2}$$

δ_j is the differential quantities. Fig. 5 shows the mean curvature distribution map based on the 4D Shepard surface. This model is the statuary of Libing and his son that including 685606 points.



a. Whole model b. Base

Figure 5. Mean curvature distribution map based on the 4D Shepard surface

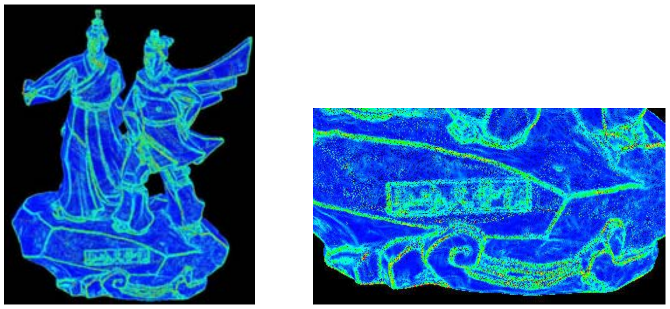
From Fig. 5, the 3-D contour information of the model can be presented primely by the mean curvature distribution map based on the 4D Shepard surface. However, the features in details, such as the pucker, spray and engraving, are not clear enough yet. This is because the Shepard basic function possesses the local smoothing itself, which can effectively suppress the noise, but at the same time, it brings bluntness of the sharp transitional features. In order to overcome the imperfect, in this paper, the Shepard interpolation is combined with the multi-scale analysis. The main means is that first the grid is partitioned based on the improved Octree, then the near data points are searched base on the grid structure, finally the differential quantities of all data points are acquired. The proposed method is processed by the following steps:

1) Constructing the level-0 grid from a uniform partition of the bounding box;

2) If the point density of one of the cells in the grid is less than a given threshold, taking step 3; or else the cell is partitioned by the improved Octree ;

3) For any point $p_j(x_j, y_j, z_j)$ in the point cloud, its near point $p(x,y,z)$ based on the grid structure are searched, then the geometric quantities δ of p and the value of the basis functions are taken into (4), the geometric quantities of p_j are obtained.

From Fig. 6, we can see the mean curvature distribution map based on the adaptive Shepard interpolation (the model of Libing and his son), in which the features in details, such as the engraving content on the base and the spray, are appeared much clearer than Figs. 5, and the more distinct contour is fetched out. This shows that combining the Shepard Interpolation and the multiresolution analysis, our method can select the optimal level of details based on the drawing capacity and error of the hardware through constructing much approximation of the original model. Thus the method has high accuracy in estimating the differential quantities of point cloud, and makes a good foundation for the subsequent step.



a. Whole model b. Base

Figure 6. Mean curvature distribution map based on the adaptive Shepard interpolation

3.3 Improved Octree solver

To improve the efficiency of the searching, this paper presents an improved Octree partitioning method that confined the operation on data points in their neighborhood.

The subdivision in the original algorithm is based on the traditional Octree model[10], in which each grid is divided into 8 cubes, thus the cube box is adopted. But in this paper, the tight box is used to save time and storage. We present our modified Octree algorithm: a hexahedron grid is divided into 8 small hexahedrons whose length, width and height aren't all same, and each sub-hexahedron is regarded as the child nodes of the root node; continue to do so until the process of the adaptive subdivision has been finished. Let the centre of a hexahedron is $V(v_x, v_y, v_z)$, and its length, width and height is $length, width, height$ respectively. The centre of sub-hexahedron are set to the following value:

$$\begin{cases} A0: (v_x - length/4, v_y + width/4, v_z + height/4) \\ A1: (v_x - length/4, v_y + width/4, v_z - height/4) \\ A2: (v_x + length/4, v_y + width/4, v_z - height/4) \\ A3: (v_x + length/4, v_y + width/4, v_z + height/4) \\ A4: (v_x - length/4, v_y - width/4, v_z + height/4) \\ A5: (v_x - length/4, v_y - width/4, v_z - height/4) \\ A6: (v_x + length/4, v_y - width/4, v_z - height/4) \\ A7: (v_x + length/4, v_y - width/4, v_z + height/4) \end{cases} \tag{5}$$

4 Implementation and experimental results

We implemented our triangulation algorithm with Visual C++ 2003 and OpenGL on Microsoft Windows, running on a PC with an Intel P4, and 2.60 GHz processor and 2.0 GB memory. We tested our method on the above models and show the experimental results in Figs. 7~11 and Table 3~4, respectively.

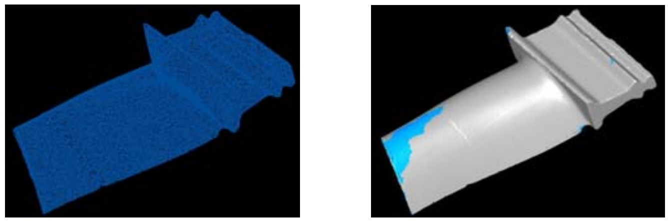
In the paper, as tight box is used, the number of the cells is greatly reduced in 0-level, especially for sparse model. As shown in Table 1, 3~4, the CPU usage and PF usage in our mothed are really close to the region-growing algorithm. Furthermore, the improved Octree is adopted to record the process of subdivision, by which the cells in the steep regions(for example, boundary, handle, and so on) are subdivided, only the nearest neighbor in model is searched for building the topology relation between point and point. The experimental result shows our method is handy, rapid and accurate during the process of subdivision.

From Figs. 7((a) show the final triangular meshes; (b), (c) and (d) are all rendering 3D graphics, in which the normals of the blue regions are pointing outwards, and the white ones are quite the opposite), as the curvature is useful to enhance the shape description, the mesh of blade generated by the proposed method is not only smooth and accuracy, and the cavity in the handle is filled well, but also the normals of the reconstruction surface are more uniform than the first two methods. From Figs. 8, for sparse point cloud, the geometric and shape features can be preserved by both the proposed method and Zhouwang Yang and Yeong-Hwa Seo's method, however, the triangulation process is speeded up obviously by the former method, approximating that of the region-growing algorithm.

Figs. 9~11 show the meshes reconstructed from the model of Libing and his son by three algorithms. We can see that the spray and the engraving region on the base are much more fine and smooth fluent from Figs. 10~11 than that from Figs. 9. It is showed that the adaptive algorithms can preserve the geometric and shape features of complex models as well as reconstruct the sharp features; but it expends much more time with the original algorithm than our. Therefore, the improved algorithm has obvious advantage for the complex large-scale point cloud, too.

Table 3
Experimental data of our algorithm (turbine blade and Cartoon)

Point cloud	Number of cells	Computation time (s)	Number of triangles	CPU use	PF use
Turbine blade	94×47×190 =839420	11.28s	238959	50%	672MB
Cartoon	62×83×81 =416826	4.00s	113280	50%	496MB


a. Final mesh b. Front side

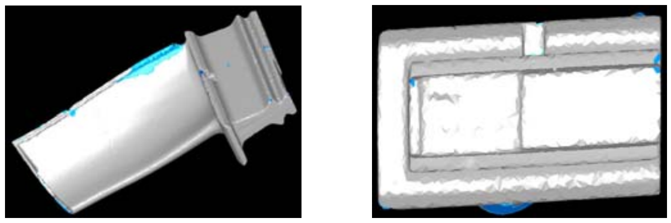

c. Back Side d. Handle

Figure 7. Final mesh of our approach (turbine blade)

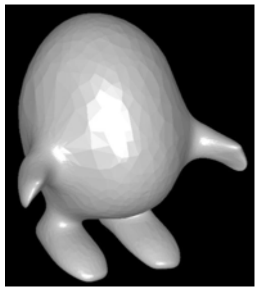


Figure 8. Final mesh of our approach (Cartoon)

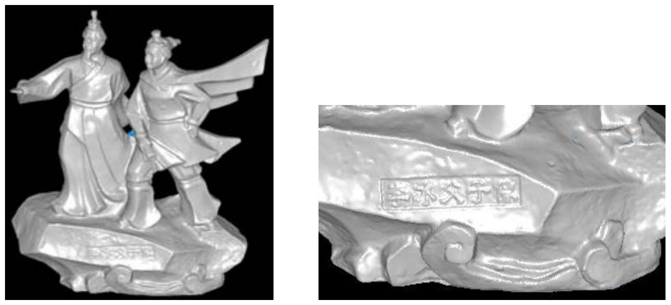

a. Whole model b. Base

Figure 9. Final mesh of Region-growing approach (Libing and his son)

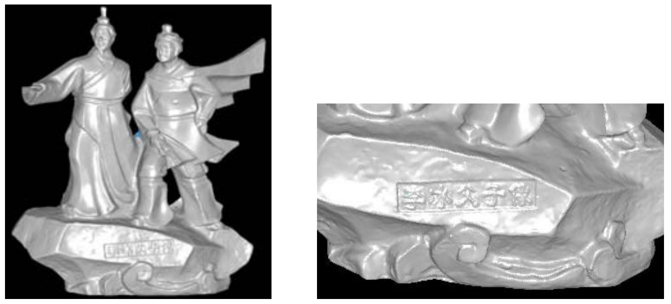
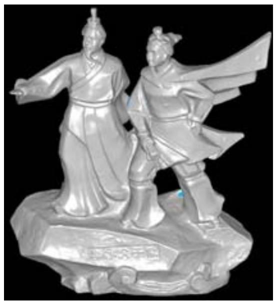

a. Whole model b. Base

Figure 10. Final mesh of original adaptive algorithm (Libing and his son)


a. Whole model

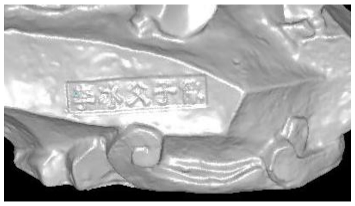

b. Base

Figure 11. Final mesh of our approach (Libing and his son)

Table 4
Experimental data (Libing and his son)

Arithmetic	Number of cells	Computation time (s)	Number of triangles	CPU use	PF use
Region-growing approach	362×266×436 =41983312	76.515s	4086669	51%	1. 18GB
Original arithmetic	360×360×360 =46656000	92.243s	4096470	54%	1. 45GB
Our method	368×270×443 =44016480	86.141s	4095813	51%	1.23GB

To sum up the above arguments，the main advantage of our method as compared with the original adaptive algorithm is as follows:

1) Combining the Shepard interpolation with multi-scale analysis, our method is effective to solve the bluntness of the sharp features, and the visualization is shown to be satisfactory;

2) During the process of the grid subdivision, we use the Shepard interpolation to estimate the differential quantities of the data points, which the redundant computation is reduced as well as the accuracy in the area immediately adjacent is improved;

3) The improved Octree is employed in our method, by which the cells in the mesh are greatly reduced in numbers. The research indicates that it is of better convergence and higher retrieval efficiency.

5 Conclusions

We have shown how to construct a triangular mesh from a point cloud, as computing the normals and curvatures of the underlying shape. The effectiveness of the algorithm has been demonstrated in the reconstruction of point clouds obtained by sampling several different models, including a non-uniform one, a sparse one, and a mass one. In these examples, the accuracy of the curvature estimates is shown increased at the boundaries of sharp shapes so that the details in the original surface are kept better and the quality of the triangular meshes are made further improvement.

We would like to make the method of LOD Models better, which would allow the Multi-resolution Shepard interpolation to be less difficult, and find the applications in more operations, such as the feature extraction, smoothing, simplification and segmentation and so on.

Acknowledgement(s)

This work was supported by the National Natural Science Foundation of China under Grant No. 50905190 and Doctor Subject Foundation of the Ministry of Education of China under Grant No. 20090191120007.

References

[1] G. Eason, Lei Zhang, LigangLiu, & Craig Gotsman. “Mesh reconstruction by meshless denoising and parameterization”; Computers & Graphics, Vol. 34, Issue 3, 198—208, Jun. 2010.

[2] Lin H W, Tai C L, & Wang G J. “A mesh reconstruction algorithm driven by an intrinsic property of a point cloud”; Computer Aided Design, Vol. 36, Issue 1, 1—9, Jan. 2004.

[3] Maarten Loffler, & Jack Snoeyink. “Delaunay triangulartion of imprecise points in linear time after preprocessing”; Computational Geometry, Vol. 43, Issue 3, 234—242, Apr. 2010.

[4] LIU Bin, & SHANGGUAN Ning. “Triangular Mesh Model Reconstruction from Scan Point Clouds Based on Template”. Tsinghua Science & Technology, Vol. 14, Issue 1, 56—61, Jun. 2009.

[5] Alliez P, Cohen-Steiner D, Tong Y, & Desbrun M. “Voronoi-based variational reconstruction of unoriented point sets”//Proceedings of the fifth Eurographics symposium on geometry processing. Eurographics Association, 39—48, 2007.

[6] DONG Hongwei. “A Triangular Mesh Reconstruction Algorithm for Points Cloud”. Computer Engineering, Vol. 31, Issue 15, 30—32, 2005.

[7] Kazhdan M, Bolitho M, Hoppe H. “Poisson surface reconstruction”//Proceedings of the fourth Eurographics symposium on geometry processing, 61—70, 2006.

[8] Zhouwang Yang, Yeong-Hwa Seo, and Tae-wan Kim. “Adaptive triangular-mesh reconstruction by mean-curvature-based refinement from point clouds using a moving parabolic approximation”; Computer-Aided Design, Vol. 42, Issue 1, 2—17, Jan. 2010.

[9] KE Ying-lin， CHEN Xi. “Efficient curvature estimation based on 4D Shepard surface”; Journa1 of Zhejiang University (Engineering Science), Vol. 39, Issue 6, 761—764, Jun. 2005.

[10] S. Ding, M.A. Mannan, & A.N. Poo. “Oriented bounding box and octree based global interference detection in 5-axis machining of free-form surfaces”; Computer-Aided Design, Vol. 36, Issue , 1281—1294, 2004.

A prototype of a sweeping surface reconstruction algorithm

V. Domiter, G. Smogavec
Faculty of Electrical Engineering and Computer Science, University of Maribor, Maribor, Slovenia

Abstract – *In this article we present a new approach for surface reconstruction using a sweeping paradigm. The algorithm uses advancing fronts to localize triangle formation and fast 2D searching to locate closest fronts for point insertion, resulting in low execution times.*

Keywords: Surface reconstruction, sweeping, computational geometry

1 Introduction

In recent years, the area of surface reconstruction has been widely popularized and has attracted many researches. Most popular subarea of surface reconstruction deals with unorganized point sets, namely point-clouds, and builds a triangular mesh to represent the surface due to its simplicity. Many solutions to solve the problem have been proposed, among which the CRUST algorithm [1] was the first to provide a strong theoretical background for the problem. An exhaustive survey of surface reconstruction algorithms has been described in [10], where the algorithms are classified in four main categories: spatial subdivision methods [4], distance function methods [8], deformation methods [12] and incremental methods [7].

Advancing front methods have been mostly used within incremental strategy [3]. Common to this group of algorithms is progressive growth of the surface by moving its boundary until the whole surface is built. A topological framework for analysis of advancing front triangulation based on handle body theory has been proposed in [9].

Advancing front algorithms differ in the way the fronts are advanced. Our algorithm will focus on moving the advancing fronts to follow a sweeping paradigm. The is based on the fast sweep-line Delaunay triangulations [6][13]. Those algorithms resulted in high performance and minimal memory costs. If we could manage to modify those algorithms in manner to handle a 3D case, the algorithm could result in high performances as well.

Let us observe a plane traversal through a solid watertight body from two different directions (Fig. 1a, b). Intersections between the plane and the object results in closed curves. The sum of all curves describes a surface of the body (a sequence of intersections 1-2-3-4-3-2-1 on Fig. 1a and sequence 1-2-3-4-5-6-7-8-7-6-5-4-3-2-1 on Fig. 1b). Both examples build a surface of a torus, but the curves differ depending on the sweeping direction. On Fig. 1a the surface is built with inner and outer curves, but on Fig. 1b one curve is split in two and those two are later joined. Observation gives us a conclusion, that there are rules between curves - the inner surface curve always lies inside the outer surface curve (curves are nested), curves are modified during sweeping, they can split, join, and open or close respectively. When applying the idea in practice, we are limited by a finite set of surface points.

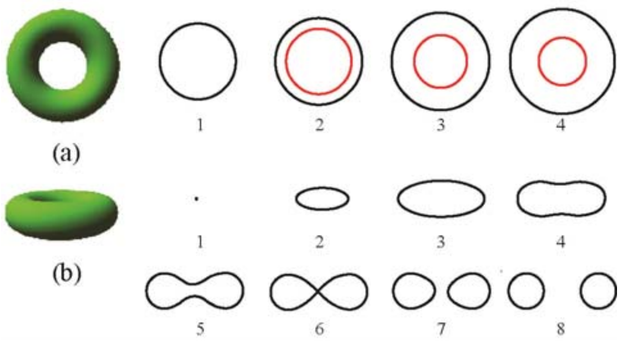


Fig. 1 Intersections between the sweeping plane and the object in two different sweeping directions.

2 Algorithm

Common to the sweeping strategy is sweeping the elements in an arbitrary direction from the first to the last element. Direction of sweeping and the sweeping body must be defined as well as status structure and events. In our case, the sweeping body is a sweeping plane traversing the points in a sorted order from the lower *z*-coordinate towards its highest value. For explanation purpose, let assume that the selected sweeping plane is the *xy*-plane. We will often refer to sweeping plane as *xy*-plane but keeping in mind that the position of the plane changes with sweeping (the *z*-coordinate). Each stop of the sweeping plane changes a status structure, represented by a set of advancing fronts. Point insertion launches one of the events of the advancing front, which updates the corresponding front(s) and results in surface mesh modification near the sweeping plane. Let us first define the advancing front.

2.1 Advancing fronts and searching

Advancing fronts (AF) present the upper boundary of the growing surface during sweeping, therefore new triangles are added to the mesh at advancing fronts only. An advancing front is defined as an oriented set of points/vertices $F=\{v_1,v_2,..,v_n\}$. Each edge v_iv_j ($j=i+1$ if $j<n$, and v_nv_0) is considered as a frontal segment. We define three forms of AF:

- *V-front* describes a front which contains only one point (Fig. 2a) – its topological representation is a vertex;
- *E-front* consists of 2 vertices (Fig. 2b) – its topological representation is an edge,
- *P-front* consists of 3 or more vertices (Fig. 2c) – its topological representation is a polygon. A segment of *P-front* is an edge of a mesh triangle.

V-front presents the initial form of the front after point insertion. When another point is inserted in its close surrounding, two *V-fronts* are combined to build an *E-front*. *P-front* appears when three *V-fronts* or one *E-* and one *V-* or *E-fronts* satisfy the conditions for building a triangle.

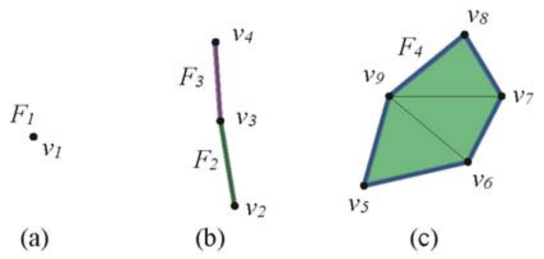


Fig. 2 Advancing fronts: V-front $F_1=\{v_1\}$ (a), E-fronts $F_2=\{v_2,v_3\}$, $F_3=\{v_3,v_4\}$ (b), and P-front $F_4=\{v_5,v_6,v_7,v_8,v_9\}$ (c).

Two different fronts can touch in a vertex (i.e. one point can belong to different fronts) (Fig. 3a). This information is stored in the point's segment list *seg(p)*, so the fronts and their segments are easily accessible from a point/vertex. Fig. 3b illustrates a possible situation. For example, $seg(p_5)=\{v_5(F_5),v_5(F_4)\}$ (p_5 is a corresponding point of the vertex v_5), whereas $v_5(F_5)$ refers to segment v_5v_8 of F_5 and $v_5(F_4)$ to segment v_5v_6 of F_4. Situations where *Fi* equals *Fj* are not allowed.

Fronts exists only near the sweeping plane (a basic property of sweeping strategy is localization of the problem, therefore no spatial 3D search is necessary. The search concerns only frontal vertices. A point enters a searching structure when visited by the sweeping plane, and leaves when it no longer belongs to any of the fronts (i.e. becomes an inner vertex of the triangular mesh).

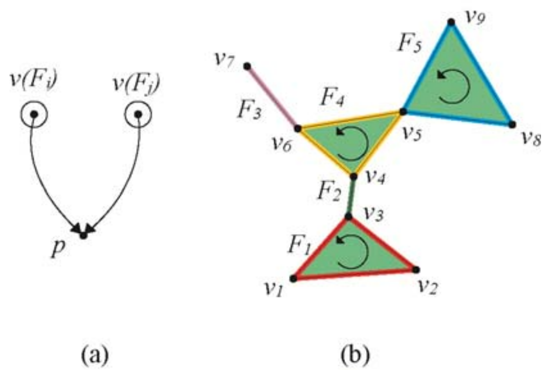


Fig. 3 Fronts sharing a point (a) and example of allowed situation (b).

To achieve fast searching, we chose a combination of a plane subdivision in stripes according to the point's y-coordinate, accessible through a hash table, and organization of points in each strip as a deterministic skip-list [7] according to the x-coordinate of the point, as shown on Fig. 4.

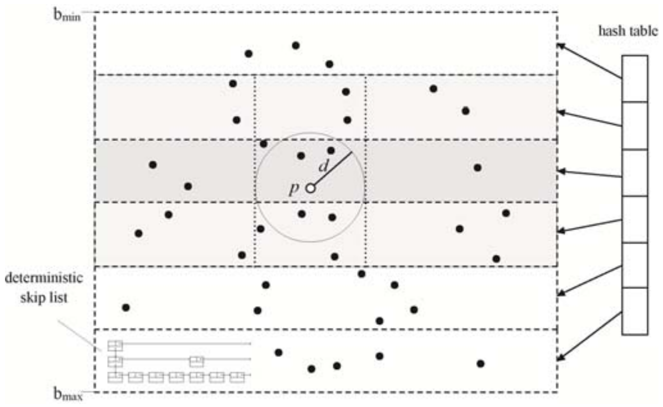


Fig. 4 Searching structure: bounding box (b_{min}, b_{max}) over point-cloud is split and hashed in stripes according to y-coordinate and further inserted in skip-lists according to x-coordinate.

The search for the closest points is executed in three steps:

1. according to the distance criterion *d*, intersected stripes are found by first determining the location of the inserted point *p* through hashing and by selecting the appropriate set of adjacent stripes;
2. points within the circle in each found stripe are located by searching the stripe's skip-list;
3. only the points within the distance sphere centered in *p* of radius *d* are kept to eliminate points lying far away from the sweeping plane regarding the *z*-distance.

2.2 Point insertion

Insertion of point *p* is performed in the following steps:

1. a list of close vertices *close(p)={v_1,v_2,..,v_n}* is determined by searching according to the distance criterion *d*,
2. *if close(p)* is empty, the point is either isolated (lying away from any frontal vertices by more than *d)* and a new *V-front* is created, or a *mesh-case* occurs;
3. else *E-fronts* with *p* and close vertices (*pv_1,pv_2,..,pv_n*) are formed and filtered,
4. triples of *pv_i-pv_j-v_iv_j* are examined to build triangles (v_i and v_j are neighbor vertices in angular order around *p*), and
5. if possible, additional triangles are attached to the mesh by observing *close(p).*

A simple analysis of close vertices is performed to eliminate formation of non-manifold triangles. Intersection tests are made between *pv_i* and frontal edges, observed on the *xy*-plane. For example, edge *pv_2* is tested with *v_0v_1, v_1v_2, v_2v_3, v_3v_4, v_7v_8* and *$v_{10}v_{11}$* (Fig. 5a). Intersection is found with edge *v_2v_3*, therefore *v_{11}* is removed from *close(p)*. The result of this test is a filtered set of close points *close(p)={v_1,v_2,v_3,v_4}* (Fig. 5b).

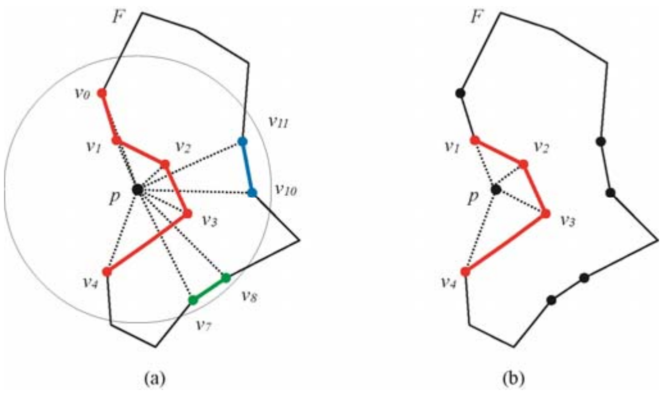


Fig. 5 Point filtering excludes v0, v7, v8, v10 and v11.

After filtering, we use edges *pv_i* to build *E-fronts*. Generated fronts are sorted in angular order around *p* to obtain pairs of adjacent edges. Now the process of triangle formation can start. For this reason, we define a *triangle operator* which can handle all possible situations between three possible frontal segments according to their types.

2.2.1 Triangle operator

Triangle operator is defined as a triple X-X-Y over vertices *v_0*, *v_1* and *v_2*, where X and Y denote the type of fronts of segments *v_0v_1-v_0v_2-v_1v_2*. X can be an *E-front* or a *P-front* (X=*E*|*P*), where Y can be the same except when a segment *v_1v_2* does not exists (Y=*E*|*P*|*0* – zero stands for no segment) (Fig. 6). By using this notation, all possible situations as shown on Fig. 7 can be properly handled.

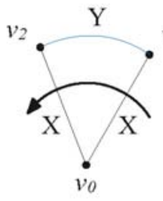


Fig. 6 Triangle operator as a triple X-X-Y. Edges *v_0v_1* and *v_0v_2* are segments of an X-front, and the third edge *v_1v_2* a segment of a Y-front.

The purpose of using the triangle operator is easy detection of frontal events (see paragraph 2.3). A basic criterion for triangle formation is a triangle criterion, which states that a sum of two triangle edges must have a length greater than the length of the third one. If fulfilled, the angle between *v_0v_1* and *v_0v_2* edges is checked, which is rather difficult to calculate. Using the scalar product between vectors *vec_1=v_1-v_0* and *vec_2=v_2-v_0* gives as a starting angle, and usually, if working on the *xy*-plane, the cross product of *vec_1× vec_2* would finally define the correct angle value. But because triangles can lie on an arbitrary plane, the use of the cross product is not straight forward. Therefore, if at least one of the vertices belongs to a *P-front*, the average surface normal of the close triangles can serve as triangle normal estimation. The cross product of *vec_1× vec_2* is compared to the estimated normal and the angle is determined. Otherwise, we have to rely on the relation tree, which defines front orientation regarding its depth (see paragraph 2.4). If the angle is smaller than 2π/3, a triangle is built.

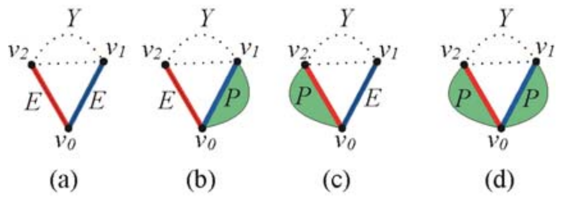


Fig. 7 All possible situations for segments *v_0v_1-v_0v_2-v_1v_2*: E-E-Y (a), E-P-Y (b), P-E-Y (c) and P-P-Y (d).

Fig. 8 illustrates insertion of point *p* with the triangle operator. First, a list of close vertices of *p* is found and the result after filtering is a list *close(p)={v_1,v_2,v_3,v_4,v_5}* (Fig. 8a). Then five *E-fronts F_4={p,v_2}*, *F_5={p,v_1}*, *F_6={p,v_5}*, *F_7={p,v_4}* and *F_8={p,v_3}* are built (Fig. 8b) and sorted. The triple to be examined contains edges *pv_2* (*F_4*) and *pv_1* (*F_5*). The third connecting segment *v_1v_2* (*F_1*) exists, which leads to operator E-E-P (*F_4* and *F_5* are *E-fronts*, *F_1* is a *P-front*). New triangle *T(v_1,p,v_2)* is formed and added to *P-front F_1*, *E-fronts F_4* and *F_5* are removed (Fig. 8c). We move to the next triple of *pv_1* (*F_1*), *pv_5* (*F_6*) and connecting segment *v_5v_1* (*F_4*): operator P-E-E adds a triangle *T(v_5,p,v_1)* to *F_1* where *F_3* and *F_6* are deleted (Fig. 8d). The same situation occurs with the next triple except there is no connecting segment: P-E-0 forms a triangle *T(v_4,p,v_5)* and *E-front F_7* is deleted (Fig. 8e). When examining a triple of edges pv_4 (F_1), pv_3 (F_2) and pv_3 (F_8), operator P-E-P launches a joining event between fronts F_1 and F_2, which is handled as follows. Triangle $T(v_3,p,v_4)$ is added to bigger *P-front* (F_1). Because F_1 and F_2 have overlapping but differently oriented segments v_4v_3 (F_1) and v_3v_4 (F_1) (Fig. 8f), fronts F_1 and F_2 are combined into F_1 and front F_2 is deleted (Fig. 8g). Last triple adds a triangle $T(v_2,p,v_3)$ to front F_1 (Fig. 8h).

After all triples have been examined, vertices of *close(p)* are checked if additional triangles can be formed. For each v_i from *close(p)*, its previous and next frontal vertex neighbors are taken to apply the triangle operator again (Fig. 8i). In this manner, triangles that could not be formed by previous insertions are built (the conditions for triangle formation could not be fulfilled at that time). When insertion is complete, local optimization is performed to improve the quality of the surface mesh.

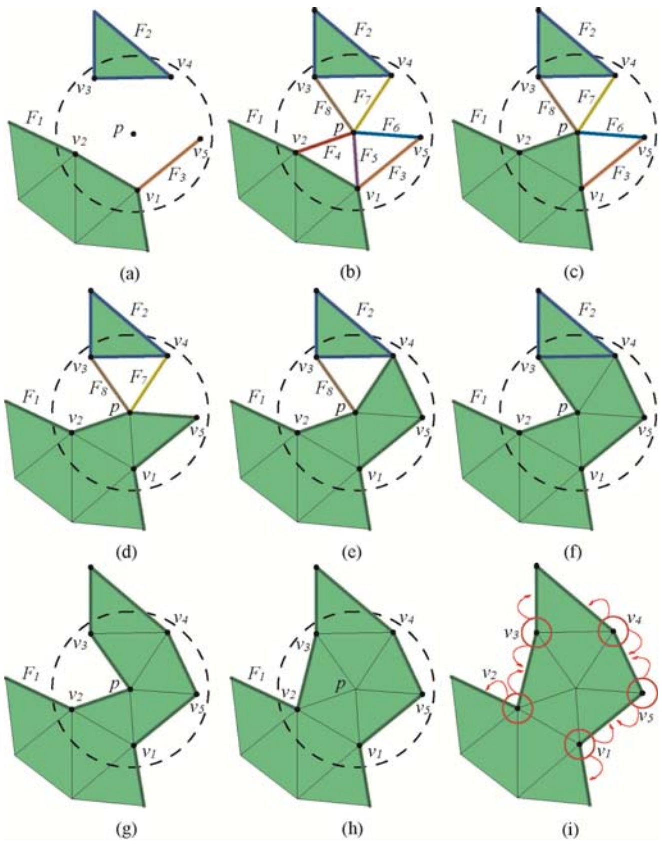


Fig. 8 Steps of insertion of point p.

2.2.2 Mesh-case

Mesh-case occurs when a point is located very close above the surface (i.e. its projection on the *xy*-plane lies within the surface mesh). Closeness is defined as a *z*-distance between *p* and the center of a triangle below *p*, which is << *d*. The usual case is when a front has recently been closed and another point that should lie on the same surface area was inserted. Another case is when the conditions for triangle formation have been met and a triangle was created before inserting the point.

Mesh case does not build any new fronts but rather modifies the surface mesh. First, a triangle below the point is located by a traversal through the surface triangles using a variation of the walking algorithm [5]. The traversal starts in one of the boundary triangles of the surrounding front if such a front exists, otherwise a list of mesh triangles is traversed in the reversed order (triangles inserted later are more likely to contain the point because they are closer to the sweeping plane). Traversal is done when a point-in-triangle containment test confirms the triangle. Three scenarios are possible:

- if the projection of *p* falls within the triangle, the triangle is divided in three;
- if the projection hits an edge, triangles sharing the edge are split in two;
- if the projection of *p* hits a triangle vertex, the vertex is replaced by *p*.

Afterwards, all new triangles are optimized.

2.3 Frontal events

Opening event occurs when the result of applying triangle operators results in a *P-front* for the first time (E-E-0, E-E-E).

Closing event occurs with the triangle operator P-P-P, if all three segments belong to the same *P-front* and triangle $T(p,v_1,v_2)$ does not exist. Triangle $T(p,v_1,v_2)$ is built and front *F* is removed (Fig. 9).

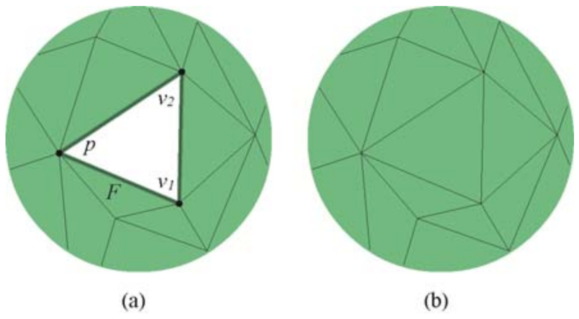


Fig. 9 Front closing event.

Front splitting event occurs when a front touches itself. This happens when at least one point appears twice as a frontal vertex of the same front: $seg(p)=\{v_i(F_k),v_i(F_l)\}$, $F_k=F_l$ (see Fig. 9).

Front joining event occurs when a partial mesh between fronts is built (result of applying the triangle operator P-E-P, E-P-P, P-P-E, and P-P-P if not closing). Joining event is performed by inserting the segments of a smaller front to the bigger in order to lower the number of insertions. Smaller front is removed.

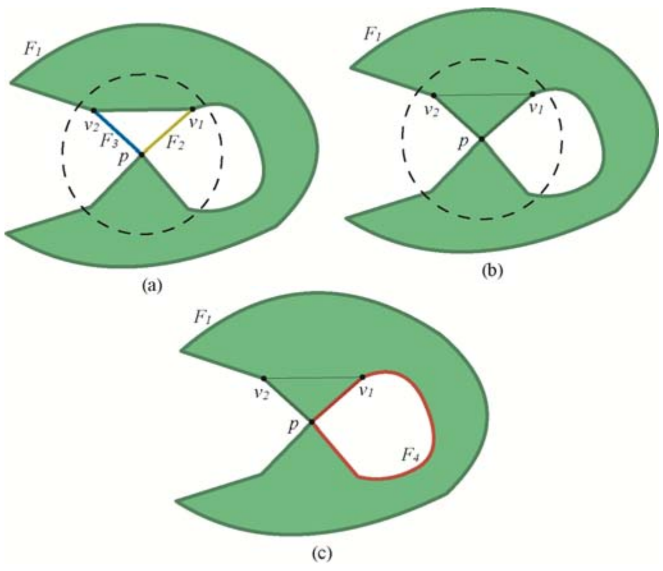


Fig. 10 Insertion of point p builds a triangle $T(v_2,p,v_1)$ using operator E-E-P (a), resulting in $seg(p)=\{p\ (F_1),p\ (F_1)\}$ (b) which is not allowed, therefore F_4 is subtracted from F_1 (c).

2.4 Relation tree

Relation tree (RT) is built to establish nesting relations between fronts. Nodes of RT are represented by *P-fronts*. Depth of the node in RT defines the orientation of the front. The outermost fronts have the depth of one. All fronts in odd depth nodes have a counter-clock-wise orientation whereas the fronts in even depth nodes are clock-wisely oriented. Child RT-nodes represent fronts lying within the front of the parent RT-node.

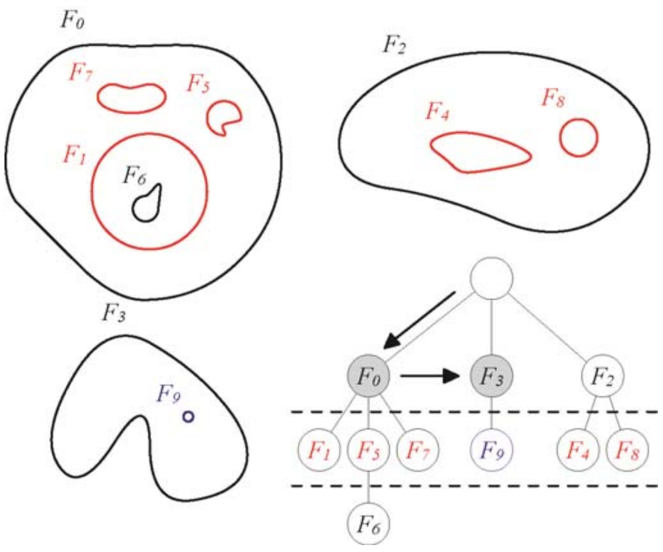


Fig. 11 Nested advancing fronts and corresponding relation tree. The arrows indicate the insertion of front F9, whereas grayed nodes present containment tests.

A *P-front* is inserted in RT when the opening event occurs. Fig. 11 illustrates the opening event for front F_9. Arrows indicate the path of insertion, grayed RT-nodes mark the fronts in which a containment tests point-in-polygon were performed to select the parent of the inserted front F_9. A *P-front* is removed from RT in case of closing event or joining event respectively.

RT is inspected in following situations:

- orientation of new front is determined by a depth of the RT-node if normal estimation was not possible,
- frontal vertices of one front are properly inserted in another front at joining event by knowing the orientation of each, and
- surrounding front of the inserted point is found to determine a starting triangle for triangle traversal in case of the mesh-case.

2.5 Mesh optimization

Each time a new triangle is formed it is immediately checked with adjacent triangles. We use a simple criterion to optimize a pair of adjacent triangles using a comparison of triangle normals and triangle areas. If a diagonal swap improves the angle between triangle normals or the ratio between triangle areas, we apply the swapping. The procedure is recursive, therefore the surface is optimized locally.

3 Results

We tested our algorithm on simple and complex point clouds of equal point distribution (Fig. 12). Tests were performed on dual-core 2.00 processor with 4 GB of memory. We measured the running times of our algorithm and compared them with the CRUST algorithm (Table 1). First impression was very promising because the running times of our algorithm compared to CRUST were extremely low. The reason for that is that we exploit the localization near the sweeping plane for searching the points and no hard-processing is involved.

When testing complex shapes, most of the surfaces were not built properly or not built at all. The reason for that is that we have a static distance criterion d, which is rather difficult to define. If d is to small, the majority of fronts are of type *V-front* or *E-front* and almost no surface is built (Fig. 13). If d is too big, too many mesh-cases appear, then the surface is incorrect or algorithm fails. Problems occured also in the filtering phase. Excluding vertices using only a 2D approach is not enough – some vertices still form non-manifold partial surfaces which eventually leads to an error.

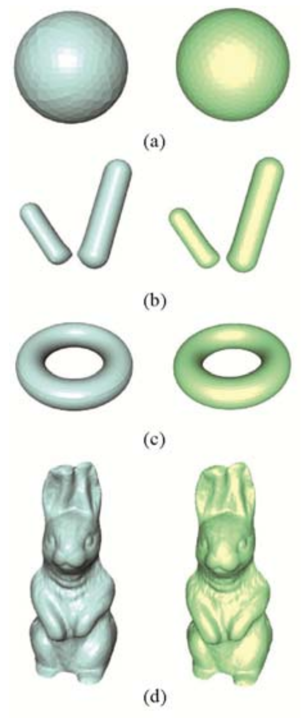


Fig. 12 Surfaces built with CRUST algorithm (left column) and the results of our sweep-plane algorithm (right column).

Table 1: Running times of CRUST and our sweep-plane algorithm

Sample	No. of points	CRUST [s]	Our algorithm [s]
(a)	902	1.120	0.032
(b)	1748	2.110	0.109
(c)	2592	10.050	0.320
(d)	56100	484.100	6.141

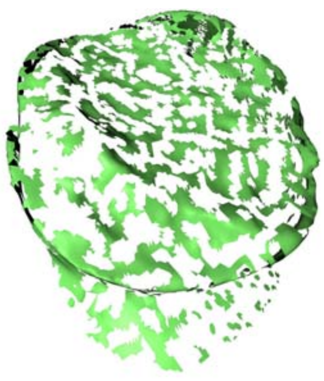


Fig. 13 Non-uniform point distribution leaves too many holes.

4 Conclusion

We have developed a prototype of surface reconstruction algorithm using a sweeping plane to advance the fronts. A 2D searching structure was used to obtain speed. We encountered problems with point distribution and with the selection of the static distance criterion *d*. For effective reconstruction, a dynamic criterion should be implemented. One of the solutions could be spatial partitioning of the point-cloud, for example employment of an oct-tree. Smaller tree-cells would represent higher densities and thus define a smaller *d*. And vice versa, bigger tree-cells would result in smaller *d*. By that and additional adjustments, a point distribution could probably be solved. Although the idea of surface reconstruction by using a sweeping paradigm looks simple, too many mechanisms need to be considered.

5 References

[1] N. Amenta, S. Choi, R. Kolluri, “The power crust, union of balls, and the medial axis transform”, Computational Geometry: Theory and Applications, vol. 19, no. 2-3, pp. 127–153, 2001.

[2] F. Bernardini, H. Rushmeier, “The 3D Model Acquisition Pipeline”, *Computer Graphics Forum*, vol. 21, no. 2, pp. 149–172, 2002.

[3] F. Bernardini, M. Mittleman, H. Rushmeier, C. Silva, G. Taubin, “The ball-pivoting algorithm for surface reconstruction”, *IEEE Transactions on Visualization and Computer Graphics*, vol. 5, no. 4, 1999, pp. 349–359.

[4] J. D. Boissonnat, “Geometric structures for threedimensional shape representation”, *ACM Transactions on Graphics*, vol. 3, no. 4, pp. 266-286, 1984.

[5] O. Devillers, S. Pion, and M. Teillaud, “Walking in a triangulation“, Proceedings of the 17th Annual Symposium on Computational Geometry, pp. 106–114, 2001.

[6] V. Domiter, B. Žalik, “Sweep-line algorithm for constrained Delaunay triangulation”, *International Journal of Geographical Information Science*, vol. 22, no. 4, pp. 449–462, 2008.

[7] E. Hartmann, “A marching method for the triangulation of surfaces”, *The Visual Computer*, vol. 14,issue 3, pp. 95-108, 1998.

[8] H. Hoppe, T. DeRose, T. Duchamp, J. Mc-Donald and W. Stuetzle, “Surface reconstruction from unorganized points”, ACM Computer Graphics, vol. 26, pp. 71-78, 1992.

[9] E. Medeiros, L. Velho, H. Lopes, “A Topological Framework for Advancing Front Triangulation”, *Proceedings of the 16th Brazilian Symposium on Computer Graphics and Image Processing*, pp. 45–51, 2003.

[10] R. Mencl, H. Müller, “Interpolation and approximation of surfaces from scattered data points”, *Proceedings of Eurographics ’98*, pp. 223–233, 1998.

[11] M. Zadravec, A. Brodnik, M. Mannila, M. Wanne, B. Žalik, “A practical approach to the 2D incremental nearest-point problem suitable for different point distributions”, *Pattern Recognition*, vol. 41, no. 2, pp. 646–653, 2008.

[12] H. K. Zhao, S. Osher, B. Merriman and M. Kang, “Implicit, non-parametric shape reconstruction from unorganized points using variational levelset method”, *Computer Vision and Image Processing*, 2002.

[13] B. Žalik, “An efficient sweep-line Delaunay triangulation algorithm”, *Computer-Aided Design*, vol. 37, no. 10, pp. 1027–1038, 2005.

Optimization of Collision Handling based on Differential Thresholds of Human Perception

Shaila Abraham
Cachematrix
Denver, CO 80206

Min-Hyung Choi
Department of Computer Science and Engineering
University of Colorado Denver
Denver, CO 80217

Abstract *- An efficient collision handling mechanism is essential for most computer animation and to achieve realistic yet efficient animations we need to consider human perception in these animation systems. Current states of art collision handling algorithms do not consider human perception in depth, particularly in differentiating thresholds of various collision parameters. Yet the human perception plays a significant role in the quality of perceived animation when multiple collisions are presented in an animated scene. The responsiveness of the objects at the right time and the right behavior makes the animation system look natural. In this paper we describe an approach, which approximates objects by using an interruptible detection algorithm to proximately test for collisions between different objects. Based on the inputs from the user study made we are able to adjust the various parameters in our collision detection algorithm and were able to get considerable speed up.*

Keywords: *Collision handling, human perception, collision detection, computer animation, real-time animation*

1 Introduction

With the high demand for highly interactive systems all over the world, the developers of animation systems are forced to produce more realistic, real-time animations. By exploiting user interactions and human perception of animation we can make the animation system realistic. Collision handling places high computational load on graphics workstations. From the human perception we find that any dynamic or casual event is dependant on the way in which objects are touching or colliding and on the way that they behave post collision. In computer graphics, we see that handling the various events of collision is a core part of any animation system and consumes a large proportion of the computation time for each step of a simulation. Thus we see several researches are made in collision handling in computer animation, both from the computational and perceptual perspectives. The solutions to address these collision handling issues in computer graphics include but not limited to adding more computational power, add hardware accelerators, and develop algorithms that may be implemented on multiple processors. However, these solutions make the issues to be postponed rather than eliminating them. People are highly sensitive to any physical event that occurs in their area of concern. There are some basic events that any human would not accept say for example one solid object cannot merge into another; on seeing and sensing a solid object we assume the properties of the solid based on the way in which they interact with each other. People judge whether objects are animate or inanimate based on the perception that the objects are moving of their own volition, or being moved by another object (referred to as the perception of causality). Much recent research, listed in reference [1], has shown that attention is one of the most important and interesting features of human perception, which can also be exploited for improving the efficiency and realism of graphical systems. Time critical approaches trade accuracy for speed, simplifying where necessary, to meet the demands of real-time rates throughout the simulation [3].

With the increase in the number of independently moving objects in the scene, the computational load also increases. We can reduce perceived inaccuracy in an animation by taking perceptual factors into account, these are done by estimating where on the screen a viewer is looking, either using analysis from user survey, or by attaching more importance to certain objects or regions in a scene. Collision handling plays a major role in any physically based animation where we can save computation time by optimizing the speed or accuracy. One of the major bottlenecks in any physically based animation includes collision handling.

We present a method in which we trade various parameters from the viewer's perspective. Section 2 presents the motivation to this work. In Section 3, we describe an architecture which performs collision detection based inputs from human's perception. We see some test scenarios in Section 4. In Section 5 we see about human threshold. In Section 6 we see the details of our experiments and we conclude in Section 7 with the approach to handle collision in an optimized manner based on human perception.

2 Motivation

Designing any computer animation, especially with collision handling and detection is a challenging task. Human character animation draped with cloth is a good example that requires intricate and precise collision handling. When objects move collisions that occur must be detected and processed robustly in a timely manner. If the object is a virtual human being collision between its clothes and hair, and self-collisions between limbs and digits must be considered. We should also be handling any physical interaction between the virtual human and the objects surrounding it such as touching, hitting and throwing. These actions are usually triggered by collision. If there are more complex objects in the environment, the burden on the computational machine which powers the animation will be higher. This leads to a greater need for extremely rapid collision detection. Increasing performance of any computer animation is often driven by hardware developments, but significant opportunities exist to increase performance via algorithmic improvement. Yet, human does not have keen ability to detect small physical incorrectness or to discern if any micro action must be handled in a particular way to preserve suspension of disbelief.

The main motivation for this work comes from the success of similar approaches used in stating that human perception is important for collision detection, where knowledge of the focus of humans that allows resolution and computational power to be concentrated on areas or objects in a display that are most important. We can make some savings in the computationally expensive area of collision detection and physical response determination. For example, considering low level of detail, proxy or fake objects could be used for collisions between objects that are not currently being attended or focused on by the viewer, or we can ignore response to the objects that are less likely to be noticed. We also want to determine if there is something about such dynamical events themselves i.e., their velocity or orientation or collision accuracy that attracts attention in some way. Based on the above assumptions we can say that we should be able to develop a model that does prediction for collision based on human perception.

3 Software Architecture

In this paper we explain briefly how collision detection for objects in 3D Space was implemented. Our approach started with having many objects colliding continuously. Then we did an in depth study of various parameters used in collision detection followed by the user study to gain knowledge about the user's perspective of various collision parameters. Based on the study we adjusted the collision handling algorithm to attain good speed up. We are using standard algorithms to compute intersection between ray and plane [1].

3.1 Model Geometry

Spheres were chosen as model objects due to their no orientation, no view point changes, and no deformation. The purpose of using a very simple geometry is to isolate the effect of individual human perception category of gap/penetration, velocity, and angle. For sphere, it's easy to predict the direction of movement after collision. While for other general rigid body objects, especially when they are occluded or spins with high angular velocity, the human ability to predict the behavior is drastically diminished and therefore human test subjects are having hard time to testify if a presented animation is right or wrong. This will lower the purity of perceptual threshold test on each category, resulting in general degradation. It would produce vague, unidentifiable source or perceptual errors. For example, if objects are sharply edged, human typically don't have ability to discern the consequential motion, so even very unlikely behaviors could be considered acceptable. To prevent this and to get direct relation between human perception and probable object behavior, a sphere was chosen as a model object. Also everyone has pretty good estimate of bounce off behavior of spheres.

3.2 Sphere-Sphere Collision Detection

A sphere is represented using its center and its radius. Determining if two spheres collide is easy. By finding the distance between the two centers we can determine if they intersect, if the distance is less than the sum of their two radiuses. The problem lies in determining if 2 moving spheres collide where 2 sphere move during a time step from one point to another. Their paths cross in-between but this is not enough to prove that an intersection occurred (they could not pass at a different time) nor can the collision point be determined. When complex shapes are used or when these equations are not available or can not be solved, a different method has to be used. The start points, endpoints, time step, velocity (direction of the sphere + speed) of the sphere and a method of how to compute intersections of static spheres is pretty straight forward. To compute the intersection, the time step has to be sliced up into smaller pieces. Then we move the spheres according to that sliced time step using its velocity, and check for collisions. If at any point collision is found (which means the spheres have already penetrated each other) then we take the previous position as the intersection point (we could start interpolating between these points to find the exact intersection position, but that is mostly not required).

If the time steps are smaller, the slices will increase with the accuracy of collision detection. Consider an example with time step 1 and with 3 slices In this case we would be checking the two balls for collision at the following time intervals 0, 0.33, 0.66, 1 and so on.

3.3 Collision Response

To determine how to respond after hitting static objects like bounding planes is as important as finding the collision point itself. Using the algorithms and functions described, the exact collision point, the normal at the collision point and the time within a time step in which the collision occurs can be found. To determine how to respond to a collision, laws of physics have to be applied. When an object collides with the surface its direction changes that is it bounces off. The angle of the new direction (or reflection vector) with the normal at the collision point is the same as the original direction vector. When a sphere hits another sphere getting the collision response is much more difficult. Ordinary differential equations of particle dynamics have to be solved and the final velocities are determined.

3.4 Euler's Equation & Explosions

To simulate realistic movement with collisions, determining the collision point and computing the response is not enough. Movement based upon physical laws also has to be simulated. The most widely used method for doing this is using a simple first order Euler's method. As indicated all the computations are going to be performed using time steps. This means that the whole simulation is advanced in certain time steps during which all the movement, collision and response tests are performed. As an example we can advanced a simulation 2 sec. on each frame. Based on Euler equations, the velocity and position at each new time step is computed as follows:

Velocity_New = Velocity_Old + Acceleration*TimeStep

Position_New = Position_Old + Velocity_New*TimeStep

Now the objects are moved and tested against collision using this new velocity. The acceleration for each object is determined by accumulating the forces which are acted upon it and divide by its mass according to this equation: Force = mass * acceleration.

Every time a collision takes place an explosion is triggered at the collision point. A nice way to model explosions is to alpha blend two polygons which are perpendicular to each other and have as the center the point of interest (here intersection point). The polygons are scaled and disappear over time. The disappearing is done by changing the alpha values of the vertices from 1 to 0, over time. To be correct we had to sort the polygons from back to front according to their eye point distance, but disabling the Depth buffer writes (not reads) also does the trick. Notice that we limit our number of explosions to maximum 200 per frame, if additional explosions occur and the buffer is full, the explosion is discarded.

4 Test Scenarios

The following are the various test scenarios considered in our experiments.

4.1 Collision Gap Accuracy Test

Newton's third law of motion states that when two bodies collide they exert equal and opposite forces on one another, we can clearly see that human perception greatly relies on the dynamic properties of the world like mass of the objects, direction in which collision happens and the like. For example, a collision of 2 moving objects on collision will cause both the objects to move but there will be definitely change in direction of movement. The human brain appears to have very low-level processes that allow people to distinguish between animate and inanimate objects. Do we need each and every object in an animated environment to exactly collide? Or are we ok on some level of accuracy of collision? Let's say that as a user I'm ok with two objects if they overlap each other slightly. Using this test scenario for an animation environment we would be able to adjust the collision algorithm so that we could use the computational resources for other purposes.

4.2 Collision Velocity and Direction Test

This test scenario is used to determine how fast the objects movement should be so that users are good on the animation. We provide various adjustable parameters like variation in after collision speed, variation in initial velocity. We are also using factors such as size and speed of objects to choose the levels of detail at which to render objects in a scene before and after collision. Based on user's input like "I liked them better when they were moving with maximum speed" or, contradictorily: "I liked them better when they were side by side and moved with normal speed"; "They looked more bouncy when they were moving at medium speed"; we were able to change the collision algorithm to obtain the best collision effect.

4.3 After Collision Change in Angle Test

This test scenario is used to determine if the users are ok with a slight deviation in the collision angle. By default let's say the angle of deviation of objects is zero. Are the users ok with a 20 degree deviation of the colliding object after collision? Or the users are more particular on the degree of deviation of the colliding object saying it should be always zero. For this test we provide the users with series of varying sliders. We also note the direction of the deviation after collision. The users can also do combination of all the above test scenarios.

5 Human Perception Threshold

The collision effects can be exploited in a real-time application by tracking the user's fixation position. When the viewer is looking directly at a collision, it would be given a higher priority than a collision occurring at a slight different location, which itself would receive a higher priority than other collisions presented more peripherally. To achieve more realistic animations we need to conduct many user surveys to get the human perception of animation. Several user surveys were carried out for the various collision handling scenarios like accuracy of separation between the colliding objects, velocity of the moving objects, angle of deviation after collision.

5.1 Accuracy

In this case objects that do not touch each other but leave a gap, decreases with increasing strangeness of the collision point. 2 Spheres Red and White were used. Collisions were presented at various accuracy levels ranging from -100% to 100%. There were many replications at each accuracy level, each in a different screen location. The order of presentation was also randomized. The users have the ability to pause or stop the collision at any time. The pause collision button can be turned on to view the exact accuracy. To verify the accuracy level we added accuracy detail radio button which gives us the exact accuracy when the collision gap goes over 0%. The spheres appeared and moved towards each other for 1 second, and then moved apart for 1 second after touching each other, leaving a small gap of 3 millimeters (mm) when the accuracy is set to 20%, or a larger gap of 2.8 centimeters (cm) when the accuracy is set to 100%. Based on the user survey we are able to see that people are more particular in accuracy.

5.2 Velocity

The objects that had more speed or velocity are compared to objects with minimum or normal speed. 2 Spheres Red and White were used. Collisions were presented at various post collision ball speed levels ranging from 0% to 200%. Default setting is at 100% of ball speed after collision. The users have the ability to pause or stop the collision at any time. The show direction button displays the pre and post collision velocities and direction. To verify the optimal velocity level we added show direction button which gives us the exact direction of the balls pre and post collision. The spheres appeared and moved towards each other for 1 second, and then moved apart for 1 second after touching each other, the after collision speed is doubled when the percentage is made to be 200% while it is reduced to half when we set it to 50%. The ball speed can be changed for one of the balls or for both.

5.3 Bounce off Angle

The objects that had 90^0 deviations are compared to objects with 0^0 or with objects that deviated 45^0. 2 Spheres Red and White were used. Collisions were presented at various post collision ball angle deviations ranging from -90^0 to 90^0. Default setting is at 0^0 of deviation post collision. The users have the ability to pause or stop the collision at any time. The show direction button displays the pre and post collision velocities and direction. To verify the bounce off angle we added show direction button which gives us the exact direction of the balls pre and post collision. The spheres appeared and moved towards each other for 1 second, and then moved apart for 1 second after touching each other, the after collision angle is made based on the selection.

6 Experiments

Collision events can be classified into different types listed below:

1. Objects approaching each other
2. Objects touching each other and then reversing
3. Objects showing repulsion effect.
4. Objects penetrating

By varying the accuracy percentage we are able to simulate the above collision events.

6.1 Collision Event

In our experiments, three types of collisions may occur (see figure 1): - "True" collisions, where entities touch, the collision is detected, and fully accurate collision response occurs. We may consider this as being the control situation for experimental purposes. (Figure 1)

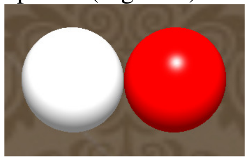


Figure 1: True Collision – 0% Collision Gap Accuracy

- Interpenetrations, where the entities also touch, but the collision are not detected or are ignored by the application. The entities are therefore allowed to continue on their previous path, even though it causes them to merge into each other to a greater or lesser degree. (Figure 2)

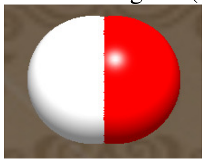


Figure 2: Interpenetrations – Negative Collision Gap Accuracy

- Repulsions, where the entities are close to each other but have not actually touched. (Figure 3)

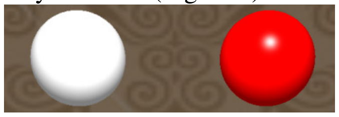


Figure 3: Repulsions – 100% Collision Gap Accuracy

The application decides to take the chance that they are actually touching, and accepts this situation as a true collision. This may then cause the entities to change their paths, causing a repulsion effect. The application can control which type of collision anomaly will occur most often, if not always. It could be argued that a detailed user study conducted led us to our approach into which type of anomaly is most preferred by the viewer. However, based on our user study we were able to see that most of the users preferred true collisions as opposed to other collision events.

6.2 Effect of Velocity

In our experiments, two types of velocity changes are being considered:
1. Initial velocity Pre Collision for the two balls.
2. Post Collision Speed of One ball or two balls.

Our first experiment was to keep the velocity by default that is the initial velocity is 100%. The figure below shows the direction of the objects here both the balls have the same initial velocity but in the opposite direction. (Figure 4)

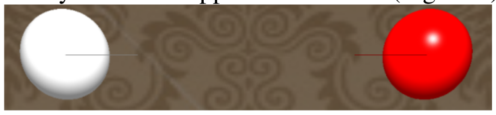


Figure 4: Initial Velocity: 100% for both objects

Now let's see the case when the initial velocity is increased to 200% the direction of the velocity is represented by the line. Here it's obvious that the velocity is higher. This is shown in (Figure 5).

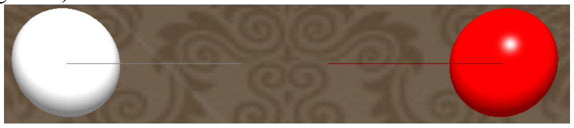


Figure 5: Initial Velocity 200% for both objects

Similarly we did experiments on varying the after collision speed from 0% to 200% for one ball and also for 2 balls. Based on the user survey we are able to see that people are ok with a speed variation of 0% to 20% on an average.

6.3 Effect of Post Collision Angle

In this experiment, we study the effect of post collision angle change on collision detection. We could change the deviation angle from -90^0 to 90^0. (Figure 6) shows the collision effect with 0^0 angle of deviation

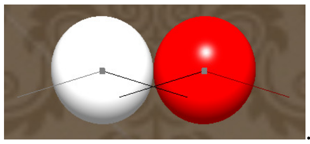

.

Figure 6: 0^0 Angle of Deviation after Collision

In the above figure the black lines represent the pre collision velocity and direction while the post collision is shown in grey and red colors. Now let's see the effect of changing the post collision angle from 0^0 to 90^0. The after collision angle is represented in the diagram below (Figure 7). Here we see that the new direction post collision is deviated 90^0.

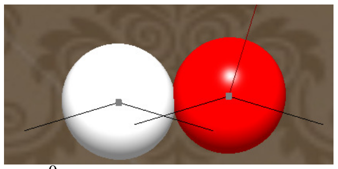


Figure 7: 90^0 Angle of Deviation after Collision

Based on the above experiments being held in our user study we were able to determine the tolerable limit for the post angle of deviation.

6.4 Location

In our experiments we also determine the useful field of view (UFV) for the perception of collision anomalies, i.e. to test what effect the location (e.g. above, below, above left etc...) of a collision has. In [Nies et al. 1998] they found that a 75 per cent confidence region for a visual search task was quite elliptic, with performance being better in the horizontal regions rather than in the vertical. Other effects were also evident, such as better detection to the right than to the left. They did find that there were big differences between subjects in the size and shape of the useful field of view. Hence, we will need to test collisions occurring in different regions, determined by the 9 cardinal directions, i.e. left, right, up, down, up-left, up-right, down-left, and down-right. The above experiments were conducted by dividing the screen into multiple sections 2, 4, 16 and studying the collision effect in specific areas. In the diagram below (Figure 8) we divide the screen into 4 sections with various accuracies of 0%, 25%, 50% and 100% accuracy.

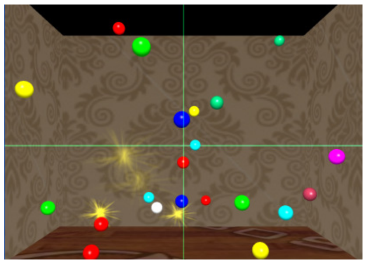


Figure 8: Screen with varying accuracy in 4 Section having 0%, 25%, 50%, and 100%

Based on our user survey we changed our algorithm by dividing the screen into sections and varying the accuracy at different sections, so that the accuracy in the corner of the room is < 100% while at the center of the screen we had 100%.

6.5 Results

It was essential to evaluate the techniques proposed, in order to justify and guide the later psychophysical studies. This set of experiments tested collision processing performance at the highest possible level of accuracy, i.e. processing every collision with 100% accuracy. In order to reach an overall target frame rate of at least 10 frames per second, the collision handing should take up only a part of the 100 milliseconds available per frame. Therefore, in the absence of any load balancing schemes, we can estimate that at most 50 milliseconds should be spent on collision handling per frame.

We can see that collision handling times are unacceptably high for 50, 70 and 100 objects (Figure 9). We can see from this data that with large numbers of objects, our frame times will be too slow and variable for real-time performance.

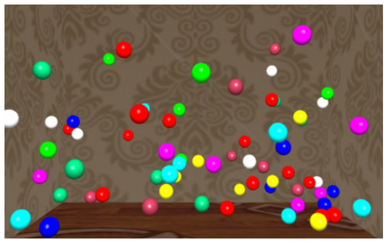


Figure 9: Collision handling with 70 bouncing balls

The parameters used for this experiment are: Time: Time spent on collision handling and # Of Collisions: Number of collisions. The mean results for all measured parameters are shown in (Table 1). The experiment is run for 10 seconds and the results are gathered. Based on these results its clear that as the number of objects in the screen increase the time spent on collision handling increases. (Figure 10) provides a graphical overview of these data.

Table 1: Results from collision with no interruption

	20	30	50	70	100
Time (ms)	1	16	252	300	646
Collisions #	2	24	92	67	84

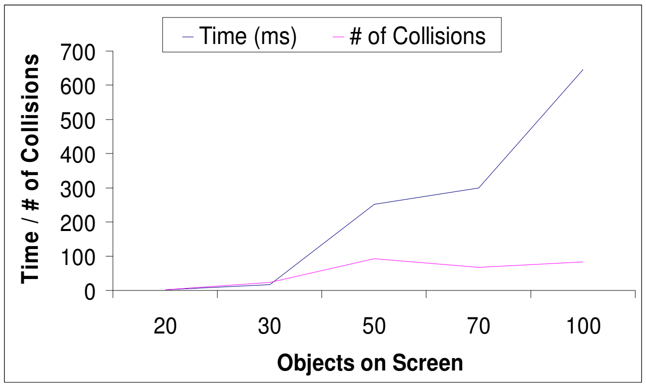


Figure10: Collision handling time with no interruption

The user survey shows us that viewers have the following tolerable level for the accuracy, velocity and angle. We took the mean of them all and we get accuracy tolerance = 1% that people are more concerned about accuracy; velocity tolerance = 5%; and angle tolerance = 20%. Taking into consideration of these values we did the analysis for 20, 30, 50, 70 and 100 objects with varying accuracy levels in the screen and our results are tabulated in the table below (Table 2). The experiment is run for 10 seconds and the results are gathered. Often binary search is used to detect exact collision time to precisely enforce contact conditions and repulsion. Finding of exact collision time becomes very complex when multiple collisions occur at a given time interval, and proper collision order sorting is also very time consuming. In the section below we see how the various criteria's promote the performance improvement of any animation system.

6.5.1 Collision Gap and Penetration

Collision gap adjustment, both gap and penetration allows us to eliminate costly backtracking to previous fraction of exact collision time search and collision order sorting operations. The ability to vary these criteria helps us to avoid spending more time on exact collision for every object. When we find the tolerable limit for collision gap or penetration we are able to apply the limit in the algorithm so that we could save the computational resources and lower the number of collisions.

6.5.2 Change of Angle and Velocity

By altering the bounce-off angles and velocity, we can intentionally avoid collisions that's about to occur. With some quick and approximate trajectory checks and collision time prediction, we can adjust the velocity and bounce off angles of post colliding objects to avoid eminent future collisions. This process is essentially altering the original behavior of simulated objects but since the alternation is within the differential threshold of human perceptual limit, they are unnoticeable and quite acceptable to users. Obviously we don't use an extensive search algorithm to avoid complicated secondary and tertiary level collisions, since our search algorithms is pretty simple one dimensional, i.e. looking into the direction of motion and see if there's any other ball in the direction of motion in a near field. Having a more sophisticated collision avoidance algorithm may further reduce the possible collisions globally but at the same time it takes more computation for the overhead, resulting in not much gain in overall throughputs.

The main advantage of this algorithm includes gain in computational power with the reduction in the number of collisions by adapting these limits. When the accuracy of collision is varied most collisions are ignored and therefore the numbers of collisions are reduced. But the animation is realistic since the actual collisions happen in the center of the screen. The behavior of the animation is different when the collisions in the peripherals are ignored. But since these surveys are based on human perception we find the collisions to be more realistic by considering collisions only in the center of the screen and ignoring the collisions at the peripherals. (Figure 11) provides a graphical overview of these data. From the two graphs we see a gain of about 7 times in the collision handling time spent when we adjusted our collision handling algorithm.

Table 2: Results of Collision with user study input

	20	30	50	70	100
Time (ms)	0	30	31	30	94
Collisions #	1	12	13	9	8

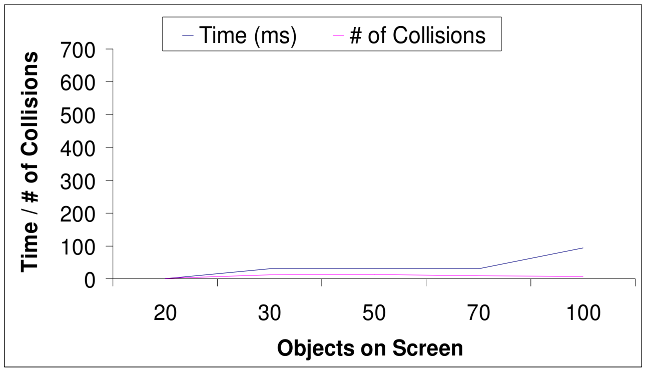


Figure 11: Collision handling time with user study input

7 Conclusions

In this paper we study and provide a solution to handle efficient collisions between objects by exploiting human perceptual threshold. We noticed that there is significant role of human perception by adjusting the various parameters to achieve speed up. We have developed the perceptually-adaptive collision detection algorithm that essentially alters the true correct object behaviors within the perceptual limit to minimize the number of collisions. The algorithm parameters are adjusted based on human subject test to get better speed up. The main algorithm is described in Section 3 and 4, along with a new model of human visual perception of collisions. In Section 5 and 6 we showed the feasibility of using this model as the basis for perceptual scheduling of collisions in a real-time animation of large numbers of homogeneous objects. It has been demonstrated that by using a ray based collision algorithm, perceived inaccuracy can be significantly improved. As described in Section 3 and 4, we validated this model psychophysically. This work is not only relevant for the problem of Collision Detection, but also for other applications where the processing of fine detail leads to a computational bottleneck. Our model could easily be adapted to accomplish perceptually-sensitive real-time shadows, shading, and other such techniques. Our psychophysical experiments can provide a starting point for other computer graphics practitioners who wish to enhance the realism of their real-time animations. For future work we would continue to work on multi level search, and global collision number optimization. We would also work on more in-depth studies for generic objects including deformable objects as our viable next steps. We think we have also contributed to the fields of Human Computer Interaction for interactive computer systems. The work in this paper can also be used in conjunction with analysis in functionality of human brain.

8 References

[1] Carol O'Sullivan. ; Collisions and Attention, ACM Transactions on Applied Perception, 2(3), pp309- 321, (2005)
[2]http://nehe.gamedev.net/data/lessons/lesson.asp?lesson=30
[3] John Dingliana and Carol O'Sullivan; Graceful Degradation of Collision handling in physically based animation, Computer Graphics Forum (Eurographics 2000), 19(3, pp239 - 247, (2000)
[4] Palmer, I.J.; and Grimsdale, R.L.; Collision detection for animation using sphere-trees. Computer Graphics Forum, v14 i2. 105-116
[5] C. Lauterbach; S. Yoon; D. Tuft; D. Minorca; RT-DEFORM: interactive ray tracing of dynamic scenes using BVHs, in: IEEE Symposium on Interactive Ray Tracing, 2006, pp. 39-46
[6] Carol O'Sullivan and John Dingliana; Collisions and Perception, ACM Transactions on Graphics. Vol. 20, No. 3. July 2001.
[7] Christer Ericson.; Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3-D Technology), Morgan Kaufmann Publishers Inc., San Francisco, CA, 2004
[8] C. Lauterbach; S. Yoon; D. Tuft; D. Minorca; RTDEFORM: interactive ray tracing of dynamic scenes using
BVHs, in: IEEE Symposium on Interactive Ray Tracing, 2006, pp. 39-46
[9] Carol O'Sullivan and Richard Lee; Collisions and Attention, ACM SIGGRAPH Symposium on Applied perception in graphics and visualization (APGV'04), pp165, (2004)

Conceptual Dynamic Collision Model For the Open Source Building Environment for Simulation and Training (OSBEST)

James Ross, Tulio Sulbaran, Ph.D., Andrew Strelzoff, Ph.D., Nan Wang, Ph.D.

Abstract— **Collision Detection** as it applies to games, virtual reality environments and simulations is the programmatic impedance of user avatars that prevents traversal through visible artifacts in the displayed field of view. When considering the user's sense of immersion in virtual reality environments for simulations or other purposes, collision is a critical feature in order to present the information in a realistic manner. In efforts to address collision, several studies have been performed in order to improve the user's experience by developing various methods for handling the computational intensive calculations associated with collision detection. These calculations include the user's position and heading relative to large numbers of fixed and moving objects in the field of view. The most widely used approach to collision detection is **collision mapping**, the use of an abstract map composed of occluded grid squares or simplified bounding boxes, which enormously reduces the computational requirements of collision detection. The result produced by the collision mapping approach is referred to as an **Expected Collision Map** (ECM) which is a map of a virtual environment's expected areas that a user cannot pass. For example, a user wants to walk up to a car but can't because the existing bounding box is larger than the actual size of the car. In most cases, once the ECM is created it remains static throughout the life of the simulation or virtual environment. However, in collaborative virtual reality environments (CVRE) like Open Source Building Environment for Simulation and Training (OSBEST), a static generated ECM will not handle the case where users are permitted to import 3D-models created in external modeling software. In this case, an automated generation of ECM is performed but has proven to be highly inaccurate. When users discover they can walk through visible objects in an un-expected manner or fail to traverse an area restricted by the ECM, the result is a loss of the user's immersion. The current solutions to handling inaccurate ECMs are either to limit users with a small library of pre-made models or have system administrators periodically check the ECM for coherence and usability which is a manual process. Rather than try to re-create or refine the well-known and used automated ECM creation tools, the solution this paper proposes is to dynamically modify the automatically generated ECM in an iterative manner by using a database logged history of user interactions with the ECM.

Keywords*— *Virtual Reality, Dynamic Expected Collision Map Modification, Collision Detection, Axis-Aligned Bounding Box, Virtual Simulation

James Ross is with The University of Southern Mississippi, School of Computing, 118 College Drive Box 5106, (corresponding author phone: 601-266-4949; fax: 601.266-5792; e-mail: James.E.Ross@eagles.usm.edu).

Dr. Tulio Sulbaran is with The University of Southern Mississippi, School of Construction, 118 College Drive Box 5138 (email: Tulio.Sulbaran@usm.edu).

Dr. Andrew Strelzoff is with The University of Southern Mississippi, School of Computing, 118 College Drive Box 5106 (email: Andrew.Strelzoff@usm.edu).

Dr. Nan Wang, is with The University of Southern Mississippi, School of Computing, 118 College Drive Box 5106 (email: Nan.Wang@usm.edu).

I. INTRODUCTION

Before providing a solution to inaccurate dynamically created ECMs there are several terms that must be defined.

Firstly, an ***object*** is a shape that is visually represented within the CVRE and usually made of one or more triangles connected together. As the number of connected triangles used increase so does the potential to improve the visual of the represented shape.

A ***bounding box*** is a set of geometrically based coordinate plane associated vertices that defines the area of an object in which no other objects in the CVRE can enter. Bounding boxes are the building blocks for the ECM in collaborative virtual reality environments like OSBEST. In an ideal case, no vertices of the visually displayed object lie outside the border defined by the bounding box.

The ***border*** of a bounding box as defined in this paper is all points lying on the perimeter of the area defined by the well-known trigonometric formula for calculating the area of a non-irregular polygon:

$area = \frac{S^2 N}{4tan(\frac{\pi}{N})}$	***S*** is the length of any side ***N*** is the number of sides ***π*** is PI, approx. 3.142 ***TAN*** is the tangent function calculated in **radians**

Table 1: The Well known Trigonometric formula for calculating the area of a regular shaped polygon

A bounding box is considered to be ***ideal*** when it contains the minimum number of vertices containing all vertices of the object within its border and the area of the bounding box is exactly equal to the area of the object contained within its border. The ideal case is rarely found in CVREs where users are allowed to import objects created in the various modeling software. In a normal case, there are large amounts of space in between the bounding box and the object it contains.

An ***avatar*** is made up of both an object and a bounding box. The object is used to give the user a way to visually represent their self within the CVRE. Every avatar in the CVRE also has a bounding box assigned to it. which contain as little a number of vertices as possible while still possessing the axis aligned property of an Axis Aligned Bounding Box (AABB) and having no vertices of the avatar's object beyond its bounding box's border.

The **Axis Aligned Bounding Box** (AABB) is perhaps the most widely used family of expected collision mapping methods and is represented in the ECM as **Minimum Bounding Rectangles** (MBR) which serves as a simplified proxy for an object's spatial extent in the ECM.

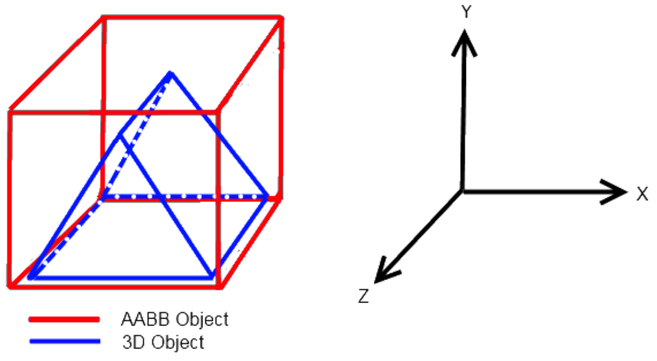


Figure 1: AABB containing a 3D Object

Another well known type of bounding box creation is **oriented bounding box** (OBB). The oriented bounding box functions in almost exactly the same way as an AABB but is not required to maintain its alignment with the geometric axis at all times. This means an OBB can be angled as seen Figure 2.

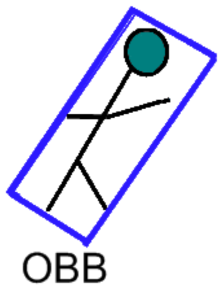


Figure 2: Oriented bounding box

As a result of the OBB not having to maintain its alignment with the axes, more calculations are needed for AABB collision detection.

A **voxel** is analogous to a pixel in a 2D image but lies in 3D space also has size and volume. Voxels are

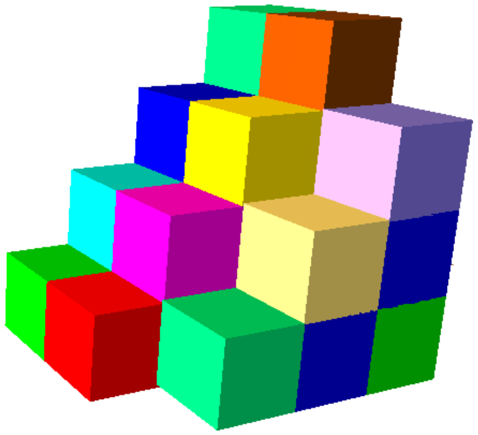


Figure 3: One highlighted voxel in a stack of voxels

useful in spatial division methods especially when they are required to satisfy the axis aligned property of an AABB. Normally in a voxel spatial division method or VSDM, voxels are used for dividing the entire environment domain such as in the 'Parallel Collision Detection Algorithm' proposed by Lawlor (3). However, in the case of OSBEST, the environment space is very large and requires a relatively large number of voxels in order to fully divide the virtual space. Also, OSBEST is a dynamic virtual environment allowing for users collaboratively add new objects and requires voxels to be constantly reinitialized to check for new objects. This initialization check would be in addition to the processing required for constantly checking of objects within the environment that have moved. It would be very difficult to perform all of the necessary calculations to perform these tasks and send these calculated results to every connected user within the networked virtual environment due to the bottleneck imposed by today’s networking speeds. In order to avoid creating so many voxels, we plan to only use voxels within the bounding box of our identified problem bounding boxes. The plan we use to partition the object space with voxels will be further discussed in Algorithm Phase 2 of our Methodology.

Virtual reality is a computer generated representation of a 3 dimensional space. In all actuality, the computer is really generating a 2D image that is distorted to give a 3 dimensional feel to the image. However, the 3 dimensional feel is enough to allow users to perceive the computer generated image as a 3 dimensional space in which they may or may not navigate. In our case, OSBEST is a virtual reality environment in which users are allowed to interact with the 3D generated scene and are also allowed to modify the different elements of the scene.

According to Ng, spatial data mining “plays an important role in extracting interesting spatial patterns and features” from spatial data such as photographs and video cameras (2). As a result of these interesting relationships that are found using spatial clustering algorithms, we intend to use the special clustering algorithms provided by the MATLAB data mining toolbox in order to identify problem collision boxes within our virtual environment.

Open Source Building Environment (OSBEST) is a free open source on-line virtual environment platform that allow users to create and share simulations and training experiences in Architecture Construction, Engineering and Computing. It is the first Web-GL based JavaScript virtual environment platform released under a BSD License, making it both open source, and commercially friendly to embed in products.
In summary, it is a collaborative 3D virtual reality environment that allows users to interact and dynamically modify objects within the environment. In addition to environment element modification, users are able to dynamically add and modify JavaScript based code to environment elements. An example of an added script would allow users to rotate an object. Users of OSBEST are allowed to freely navigate a 3D virtual environment and add objects created in an external modeling software. Users are given the ability to modify their imported models by resizing, modifying shaders, and relocating their models to other areas of the environment space. OSBEST is also based completely in a webpage accessible from any WebGL supporting web browser. In order to begin using OSBEST, the user is not required to download or install any software other than a WebGL supporting browser. In the case of OSBEST and other collaborative virtual environments like RealXtend and Opensim, user imported objects normally possess inaccurate bounding boxes when initially imported into the environment. Modifying these inaccurate bounding boxes to more accurately represent the users expectations for collision is the problem which this paper provides a solution.

The last term to be discussed is the **Expected Collision Path** (ECP). As a user moves their avatar through the CVRE, they will tend to develop expectations for where they should collide with a particular object and where there should be no collisions. This developed expectation for the paths through and around an object that the user may move their avatar is the ECP. For example, if a user moves their avatar through several objects that visually represent an archway, the next time the user encounters an object or objects that visually looks like an archway, they will expect to be able to move their avatar through the visually represented opening in the archway.

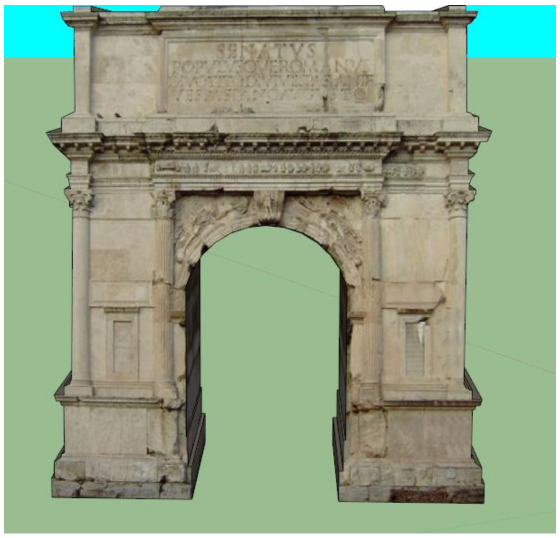


Figure 2: Example visually represented archway in a CVRE

II. OBJECTIVE

The objective of this work is to create a better collision map dynamically without either restriction to a small set of assets or manual modification of every bounding box in the CVRE. The objective is three-fold: (1) Develop an algorithm that will dynamically modify an ECM object's bounding box, (2) Identify and apply a Data-

Mining technique to discover problem bounding boxes, (3) Test the ability of the algorithm to successfully modify a bounding box resulting in less unexpected object collisions.

III. Methodology

There are three phases in this research methodology for completing the goal which are develop, identify and apply, and test.

In the develop state the objective is to develop an algorithm that will dynamically modify an object collision box. The algorithm needs the ability to handle the addition of new objects in the CVRE. It will accept these new objects and log all collisions that occur between their bounding boxes and the bounding boxes of other objects within the CVRE.

In the second phase, identify and apply, users will be asked to explore a CVRE with the intention of not colliding with any objects in the environment. Every collision that occurs between the user's avatars bounding boxes and the objects within the CVRE will be recorded by the algorithm. The date from this algorithm will be placed in a spatial database. A spatial clustering data-mining algorithm will be applied to the information in the database to *identify* problem bounding boxes within the CVRE. Once the problem bounding boxes are known, the algorithm will then be *applied* to modify the existing bounding boxes.

In the last phase, testing, users will be asked to explore the modified CVRE to further identify problem bounding boxes. The data from the users navigation through the CVRE will be recorded in a database format by the algorithm introduced in phase 1 and have the same data-mining technique applied. Phase two and three of the methodology will continue to be repeated until no further problem bounding boxes are identified.

In summary of the three phases of the methodology, the idea will be to place obstacles within OSBEST. The users will explore the environment with the intention of not running into any of the objects. As the users move through the environment, everywhere that they run into collision boxes will be stored in a dataset. As the multiple users move through the environment, the areas of the map producing a high number of collisions will be dynamically modified to more accurately match the users ECP. A data mining special clustering algorithm will be used in order to separate accidental contact with a collision box and collisions with an unexpected collision box.

Since bounding box creation in 3rd party 3D modeling software cannot be relied upon to create a bounding box that will remain accurate when imported into a CVRE such as OSBEST, incorrect bounding boxes have the potential to be generated. In order to help remove problem bounding boxes before users ever notice there is a problem, an axis aligned bounding box should be generated by the algorithm. The AABB generated by the algorithm will insure that the bounding box for every user imported object is at least a MBR. In some cases, a MBR will be accurate enough to satisfy the users ECP and no further calculations will be required. However, in most cases, an inaccurate collision box is created and will need to be identified. As users explore the virtual environment all collisions will be logged into a MYSQL database according to the object on which a collision occurred, and by the position of the avatar that collided with the object. A spatial clustering data mining algorithm will be run on the MYSQL logged collisions to determine when a cluster of collisions are developing around a user imported object. The accidental user collisions will be ignored by defining a collision cluster to be expected to satisfy the following two criteria. The first criteria is met by establishing a minimum number of collisions to identify a cluster and the second criteria is satisfied by defining a maximum distance in between the points that make up the collision cluster. Once the spatial clustering algorithm has determined that a cluster of collisions is developing by using the previously mentioned criteria, an incorrect bounding box flag is checked and the partitioning of the object's collision box space can begin.

In order to improve the rate in which a more accurate bounding box is developed and to help identify areas of the model the user would not expect collisions to occur, special attention is going to be given to the space in-between a 3D model and the bounding box in which it is contained. In most cases, when a MBR is created there is a large amount of empty space between the model and its bounding box. It is very important according to Bandi that "the ratio between object volume to bounding box volume be as high as possible" ( ). In order to help maintain this ratio, we intend to use the "Spatial Subdivision of the Object Space" method proposed by Bandi. This method describes the process of *Digitizing* objects by dividing them into 2^*d* voxels in which the *d* represents the extent into which the object space is divided. The process of *digitizing* objects takes place every render cycle. The main difference in our use of this method will be observed in the *digitization* process of the models within the environment. In Bandi's method all moving objects are *digitized* every render cycle. Since the process of *digitizing* a 3D model adds overhead to each render cycle in which it is used, we intend to only *digitize* objects within the environment that have been identified as possessing problem bounding boxes.

Once an object's bounding box has been *digitized* and therefore identified to be a problem, we plan to perform a method of hierarchical division of a MBR based on the positions of the points the make up the cluster used in the identification of the problem bounding box. Once the object has its problem MBR divided into smaller bounding boxes, the *digitized* object space within each smaller bounding box will have its voxels examined to determine whether or not object geometry is present. If there is no geometry present in any of the voxels contained within a smaller bounding box, the bounding box is determined to no longer be necessary and is removed. The empty bounding box areas of the originally generated problem MBR are removed, thus improving the likelihood of the bounding box areas surrounding the object more accurately representing the user's ECP.

```
1  Generate MBR for user imported object
2  On Render
3     Log collisions;
4     Examine MYSQL DB for clusters
5     If  cluster exists
6         Perform digitization on offending object
          space
7         Divide bounding box into smaller
          bounding boxes
8         Examine voxels for object geometry of
          smaller bounding boxes eliminating the
          bounding boxes that do not posses
          object geometry
9     End IF
10 End Render
```

IV. Conclusion

This work will produce a new approach to the problem of inaccurately generated ECMs from user contributed objects into a CVRE. The expected improvement to the collaborative virtual reality expected collision map rests upon automatically gathering and mining user interactions with the ECM and iteratively improving MBRs in areas where unusual patterns of user-ECM interaction are detected. In areas of the ECM where users are repeatedly colliding with bounding boxes the algorithm will automatically generate a new trial MBR for the offending object which will then be re-examined at the next iteration to either accept or further refine the ECM thus automatically improving the users' sense of immersion without either restricting users import of new objects or requiring tedious manual adjustment of MBRs.

The development of a new approach to automated ECM improvement in CVRE will enable a new generation of collaborative virtual environments in which users can contribute arbitrary objects to the environment and regular manual revisions of bounding boxes unnecessary.

The method of modifying the AABB to dynamically modify a collision map will improve the speed of which collision maps are created. Normally the developers ask testers to explore an environment and make notes of where the collision boxes are out of place. The developers then go into the environment and modify the collision boxes to more accurately represent the ECM. The process of making these modifications can be extremely tedious and time consuming. The proposed solution will help to take this modification out of the hands of the developer and modify the environment based on the ECP of the user.

V. Limitations

A limitation of this research is that the OSBEST CVRE is only currently able to maintain fifty visually represented avatars. As a result, testing on how more than fifty avatars would affect the expected collision map is not possible.

Another area for potential improvement is intentional collision with objects by users. If one user intentionally collides with a particular object then the data-mining technique will consider it accidental contact, but if multiple users run intentionally into one object then the proposed method for modifying collision boxes will create very inaccurate ECMs.

The method for how to handle intersecting collision boxes would also be an area for research. An example case would be when a user places a wall within the CVRE and then places a door object within the bounding box of the previously created wall. We will assume for the sake of the example that both the collision boxes for the wall and the door objects are ideal. When the user tries to move through the door the walls collision box will restrict the user's movement even though there is space for the user to pass through in the door's collision box.

Another good example a possible future work for the proposed dynamic bounding box modification method would be the case where a single regular shaped bounding box could not be created that contained all vertices of the object and prevented the user from experiencing unexpected collisions. A good example of this case would be when a user adds an archway to the CVRE. In order to create a reasonable ECM a single regular shaped polygon bounding box would not be able to include all visible components of the archway while still providing a path for the user to move their avatar through the archway.

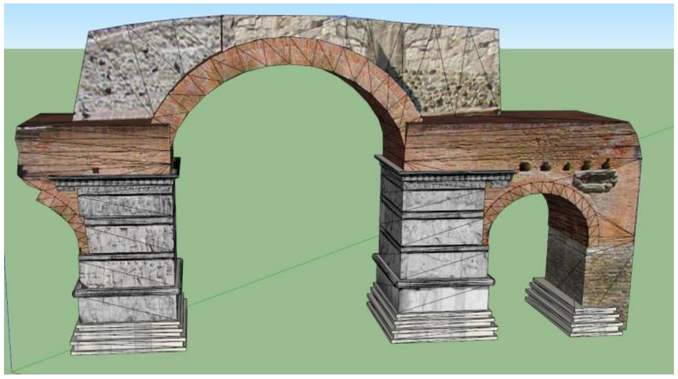


Figure 3: Archway in 3D environment

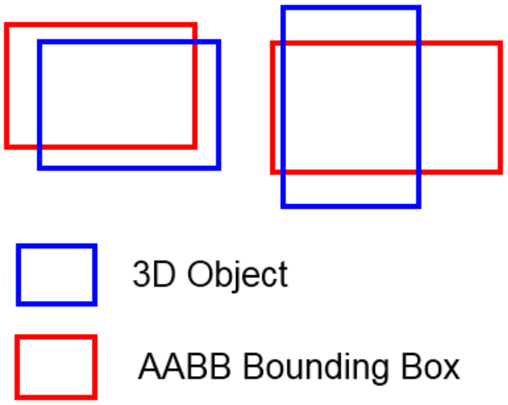


Figure 3: Example of poorly assigned bounding box

Another good example of a potential future work of the proposed bounding box modification method would be the case where an irregular-shaped polygon bounding box was desired over using multiple independent regular-shaped bounding boxes. As mentioned in the previous limitation, an archway would also be a good example of the need to handle irregular shaped polygons.

The last example that is going to be discussed is the case where objects are being relocated by users. If a user places and object in the CVRE and the object is then sent in motion, the current algorithm will probably not correctly identify the problem collision box. This issue could be handled by the algorithm with modifications to keep track of where the object was for previous collisions and where the object was for new collisions.

REFERENCES

[1] Bandi, S. & Thalmann, D. An Adaptive Spatial Subdivision of the Object Space for Fast Collision Detection of Animating Rigid Bodies. DOI: 10.1.1.14.8193.pdf

[2] Chapel, Mike. Data Mining with the k-means Algorithm. Retrieved from http://databases.about.com/od/datamining/a/kmeans.htm

[3] Ditchburn, K. Collisions. 2004-2010. Retrieved from http://www.toymaker.info/Games/html/collisions.html

[4] Ester, M. , Kriegel, H.-P. , Sander, J. & Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In Proc. of the Second Int'l Conference on Knowledge Discovery and DataMining, Portland, OR, 1996.

[5] FRANQUESA-NIUBO, M., AND BRUNET, P. 2003. Collision prediction using mktrees. Proceedings of CEIG, 217-232. DOI: 10.1.1.9.8002.pdf

[6] Hshieh, H., Tai, W., & Chang, C. 2010 GPU-based Collision Detection and Response for Particles on 3D Models. Journal of information Science and Engineering, 26, 1619-1335. Located at: http://www.iis.sinica.edu.tw/page/jise/2010/201009_04.pdf

[7] Lawlor , O. & Kal, L. 2002. A voxel-based parallel collision detection algorithm. In Proceedings of the I16th international conference on Supercomputing(ICS '02). ACM, New York, NY, USA, 285-293. DOI: 10.1145/514191.514231

[8] Ng, R. & Han, J. 1994. Efficient and Effective Clustering Methods for Spatial Data Mining. Technical Report. University of British Columbia, Vancouver, BC, Canada. DOI: 10.1.1.95.4297

[9] Sander, J.. Ester, M.,. Kriegel, H.-P., &. Xu. X. Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications. Data Mining and Knowledge Discovery, 2(2):169–194, 1998.

[10] Tan, Steinbach, Kumar, & Ghosh. The K-Means Algorithm. Retrieved from : http://www.cs.uvm.edu/~xwu/kdd/Slides/Kmeans-ICDM06.pdf

[11] Velmurugan, T. & Santhanam, T. 2010. Computational Complexity between K-Means and K-Medoids Clustering Algorithmsfor Normal and Uniform Distributions of Data Points. Journal of Computer Science 6 (3) 363 – 368. POI: 10.1.1.165.9474-1

[12] Yong-ping, K., Yun-lan, T., Chang-xin, L. & You-ru, Y. November 2009. The optimization design of collision detection for barrage games. Journal of Communicaiton and Computer, 6, 11. Retrieved from: http://www.informatics.org.cn/doc/ucit200911/ucit20091102.pdf

Interval type-2 Fuzzy Sets-based no-reference quality evaluation of synthetic images

Samuel Delepoulle,André Bigand,Christophe Renaud

LISIC, ULNF, Calais Cedex, 62228, France. Email: {delepoulle,bigand,renaud}@lisic.univ-littoral.fr

Abstract— *No-reference image quality assessment needs no prior knowledge of reference image. A new fuzzy image quality measure (built from interval type-2 fuzzy sets (IFS2)) is compared with experimental psycho-visual data. The proposed measure is based on IFS2 entropy applied on synthetic images. A recently performed psycho-visual experiment provides psycho-visual scores on some synthetic images, and comprehensive testing demonstrates the good consistency between these scores and the quality measures we obtain. The proposed measure has been too compared with full-reference quality measures (or faithfullness measures) like SSIM and gives satisfactory performance.*

1. Introduction

No-reference image quality measures are very important to characterize images visual quality. They are of great interest in image compression (JPEG models) and in image synthesis. They are still in investigation. Some recent papers [1], [2] proposed no-reference quality assessment of JPEG images. Although the authors obtained good results, these reported quality measures have their limitations. At the moment, the classical model to characterize image quality remains psycho-visual experiments (Human in the loop experiment [3]).
A reference image quality measure is often expensive to obtain (or not available). This is the case in image synthesis using global illumination methods. The main goal of global illumination methods is to produce *synthetic images with photorealistic quality*. For this purpose photon propagations and light interactions with matter have to be accurately simulated. Stochastic methods were proposed for more than 20 years in order to reach these goals. They are generally based on the Path Tracing method proposed by Kajiya [4] where stochastic paths are generated from the camera point of view towards the 3D scene. Because paths are randomly chosen the light gathering can change greatly from one path to another generating high frequency color variations through the image [5]. The Monte Carlo theory however ensures that this process will converge to the correct image when the number of samples (the paths) grows. But no information is available about the number of samples that are really required for the image being considered as *visually* satisfactory. Indeed the final use of these images is to be seen by human observers who are generally very sensitive to any image artefact. The human visual system (HVS) is endowed with powerful performances but is a very complex process! Consequently and due to the high computational cost of global illumination algorithms, perception-driven approaches were proposed. The main idea of such approaches is to replace the human observer by a vision model. By mimicking HVS such techniques can provide important improvements for rendering. They can be used for driving rendering algorithms to visually satisfactory images and to focus on visually important features [6]–[8]. Most HVS models provide interesting results but are complex and still incomplete due to the internal system complexity and its partial knowledge . They generally require relatively long computation times and are often difficult to use and to parameterize.
So, this paper focuses on the use of a new perceptual index to replace psycho-visual index in the perception-driven model. We propose a novel no-reference image quality, based on type-2 fuzzy sets (to modelize uncertainty brought by noises affecting the image synthesis).
Type-1 fuzzy sets (or FS1) are now currently used in image processing and particularly for gray scale segmentation [9]–[11]. Recently, fuzzy region oriented techniques for color image segmentation have been presented [12], [13], defining a region as a fuzzy subset of pixels, where each pixel in the image has a membership degree to each region. These techniques are based on fuzzy logic with type-1 fuzzy sets. Other techniques have been presented to perform color clustering in a color space [14], [15], based on type-1 fuzzy sets and a homogeneity measure (homogeneity of the "paths" connecting the pixels [16], or fuzzy homogeneity calculated by fuzzy entropy [14]).
The major concern of these techniques is that spatial ambiguity among pixels has inherent vagueness rather than randomness. However, there remain some sources of uncertainties in type-1 fuzzy sets (see [17]): the meanings of the words that are used, measurements may be noisy, the data used to tune the parameters of type-1 fuzzy sets may also be noisy. Imprecision and uncertainty are naturally present in image processing [18], and particularly these three kinds of uncertainty.
Techniques that are not much used for the moment in images analysis are type-2 fuzzy sets. Mendel [17], [19], [20] shows that type-2 fuzzy sets (or FS2) may be applied to take into account these three kinds of uncertainty (measurement noise,

data-generating mechanism, and description of features that are all nonstationary, when the nature of the nonstationarities cannot be expressed mathematically), and we have investigated this new scheme in this paper. The concept of a type-2 fuzzy set was introduced first by Zadeh [21] as an extension of the concept of an ordinary fuzzy set (type-1 fuzzy set). Type-2 fuzzy sets have membership degrees that are themselves fuzzy. At each value of the primary variable (universe of discourse X), the membership is a function (and not just a point value) - the secondary membership function - whose domain (the primary membership) is in the interval [0,1] and whose range (the secondary degrees) may also be in [0,1]. Hence, the membership function of a type-2 fuzzy set is three dimensional, and it is the new third dimension that provides new design degrees of freedom for handling uncertainty.
In this paper we will focus on gray-scale images and use FS2 for image quality evaluation. The paper is organized as follows:

- Section 2 describes the experimental setup we use;
- Section 3 briefly describes the type-2 fuzzy sets;
- Section 4 introduces image quality evaluation using type-2 fuzzy sets;
- In section 5 we will present some results;
- Finally, the paper is summarized with some conclusions in section 6.

2. Experimental setup

Unbiased Global Illumination (G.I.) methods use randomly chosen paths for sampling the illumination of visible objects. This process generates stochastic noise which is perceptible by any human observer. Image denoising techniques, used *a posteriori*, are largely present in the literature [22]–[24]. Noise models and noise estimation from images are however more difficult. Anyway these models are based on theoretical models of noise like additive white noise. However in G.I. algorithms, noise is not additive and arises from an unknown random distribution function. To our knowledge there is no existing model able to detect and to quantify stochastic visible noise in an image. We detail in the following the different steps of our approach for solving this problem by using a new image quality measure.

2.1 Overview

Our goal is to mimic the human visual detection of noise by way of a no-reference image quality measure. So it is necessary to provide to the proposed model some examples of what human judgment consider to be noisy images or noiseless ones. After validation on all these examples the method will generate a model that will then be used on images that have to be analyzed.

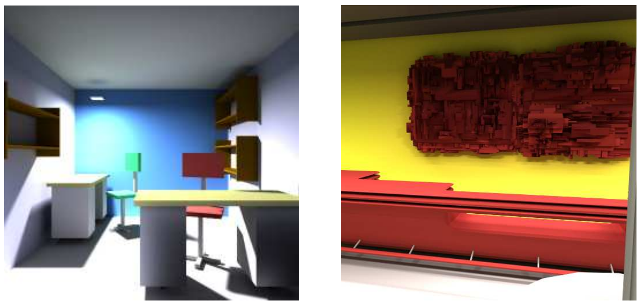


Fig. 1: Two used (reference) scenes.

2.2 Data acquisition

2.2.1 The images dataset

The model is built on data corresponding to images of globally illuminated scenes. We used (as a first approach) a Path Tracing with next event algorithm [5] which computes several images from the same point of view by adding successively N new samples[1] equally for each pixel. For each scene and each point of view we thus have several images available, the first ones being strongly noisy and the last ones being converged. The images were computed at 512×512 resolution, the number of additional samples between two successive images was $N = 100$ and 12 scenes were used. The largest number of samples per pixel was set at 10.000 which appeared to be sufficient for generating visually converged images. Figure 1 presents 2 of these scenes that were used during the validation stage of the model. These scenes highlight different illuminations and various geometrical and textures complexities.

2.3 Psycho-visual scores acquisition

Because we have to evaluate the noise level present in each generated image, some experiments were necessary in order to specify the noise threshold that is used as stopping criterion in images synthesis. But considering the entire image for noise thresholding has two main drawbacks: on one hand, it requires evaluation methods to work on very large data sets; this has been experimentally shown to reduce their learning efficiency. On the other hand the noise is generally not equally perceived by human observers through any part of an image; noise thresholds are thus different for each location in each image and the use of a global threshold would reduce the efficiency of the approach by requiring the same number of samples to be computed for each pixel of the image. We thus defined a very simple protocol in which pairs of images are presented to the observer. One of this image is called the *reference image* and has been computed with $N_r = 10.000$ samples per pixel. The second image so called the *test image* is built as a stack of images, from very

[1]In the following we will call sample a stochastic path between the view point and a light source.

noisy ones above to converged ones below: by calling N_i the number of samples in the stack's image i, with $i = 100$ at the top of the stack and $i = max$ at its bottom, we thus ensure the property $\forall i \in [100, max[, N_i < N_{i+100} \leq N_r$. Each of these images are opaque and virtually cut into non-overlapping blocks of size 128×128 over the entire image. For the used 512×512 test images we thus get 16 different blocks (clockwise sorted from 0 to 15, top image of figure 3) for each of the stack's images. During the experiments the observer is asked to modify the quality of the noisy image by pointing the areas where differences are perceived between the current image and its reference one. Each point-and-click operation then causes the selection and the display of the corresponding $i + 100$ level block thus visually reducing noise in this image's subpart. This operation is done until the observer considers that the two images are visually identical. Note that for reducing experiment artefacts this operation is reversible meaning that an observer is able to go down or up into the images stack. The pair of images that is presented to the observer is chosen randomly but we ensure that each pair will be presented two times. Obviously the block grid is not visible and all the observers worked in the same conditions (same display with identical luminance tuning, same illumination conditions, ...). The results were recorded for 33 different observers and we computed the average number of samples $\tilde{N}$ that are required for each subimage to be perceived as identical to the reference one by 95% of the observers. We got experimentally $\tilde{N} \in [1441, 6631]$ with often large differences between subimages of the same image (see figure 4). So now we present the no-reference image quality index we propose.

3. Type-2 fuzzy sets

3.1 Definition

Ordinary fuzzy sets used in image processing are often fuzzy numbers. A fuzzy set defines the meaning representation of the vagueness associated with a linguistic variable A in a natural language. However, it is not possible to say which membership function is the best one. The major motivation of this work is to remove the uncertainty of membership values by using interval-valued fuzzy sets.
Let X be the universe of discourse. An ordinary fuzzy set A of a set X is classically defined by its membership function $\mu_A(x)$ (with $x \in X$) written as:

$$\mu_A \quad : X \to [0, 1] \tag{1}$$

where the membership function denotes the degree in which an event x may be a member of A. A point x for which $\mu_A(x) = 0.5$ is said to be a crossover point of the fuzzy set A. Vagueness, or fuzziness, is absolutely specific. When a fuzzy set X is created we have perfect evidence of the degree to which a certain element x belongs to A. Let $S([0, 1])$ denote the set of all closed subintervals of the interval $[0, 1]$. An Interval type-2 fuzzy set (IFS2 or interval-valued fuzzy set (IVFS): the two terms are equivalent in the following) A in a non-empty and crisp universe of discourse X is a set such that [21], [25]:

$$A = \{(x, M_A(x) = [\mu_{AL}(x), \mu_{AU}(x)]) \mid x \in X\}. \tag{2}$$

The membership function (MF) M_A defines the degree of membership of an element x to A as follows:

$$M_A : X \longrightarrow S([0, 1]) \tag{3}$$

For each IVFS A, we denote by $\delta_A(x)$ the amplitude of the considered interval ($\delta_A(x) = \mu_{AU}(x) - \mu_{AL}(x)$). So non-specific evidence (an interval of membership values) for x belonging to a linguistic value A is identified by IVFS.

3.2 Uncertainty representation

The uncertainty of membership function of a precise FS is modelled using the length of the interval $\delta(x)$ in an IVFS (the longer $\delta(x)$ the more uncertainty), so choice of functions $\mu_U(x)$ and $\mu_L(x)$ is crucial. Tizhoosh [11] applied IVFS to gray scale image thresholding. He used interval-valued fuzzy sets with the following functions $\mu_U(x)$ and $\mu_L(x)$:

- upper limit: $\mu_U(x) : \mu_U(x) = [\mu(x; g, \sigma)]^{1/\alpha}$, (with $\alpha = 2$),
- lower limit: $\mu_L(x) : \mu_L(x) = [\mu(x; g, \sigma)]^{\alpha}$

where $\mu(x; g, \sigma)$ is a Gaussian (FS) fuzzy number ($\mu(x; g, \sigma)$ is represented in Fig. 2) centered on g and which support is set using a free constant parameter σ ($x \in [0, G - 1]$):

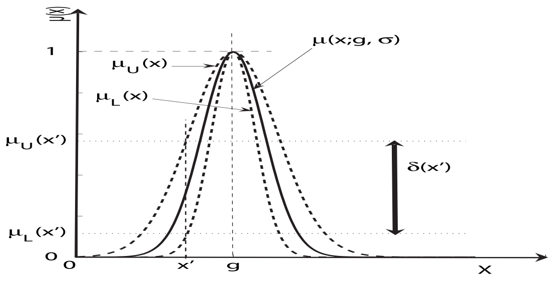
$$\mu(x; g, \sigma) = exp\left[-\frac{1}{2}\left(\frac{x - g}{\sigma}\right)^2\right]. \tag{4}$$

The study he carried out on these functions showed that they are well adapted in image processing. So, in the sequel, we shall use the same functions in image filtering. Fig. 2 presents (Gaussian) membership function of an IVFS in X. For each element $x \in X$ of the IVFS, the imprecision of the FS is defined by closed intervals delimited by the upper membership function $\mu_U(x)$ and the lower membership function $\mu_L(x)$. These membership functions $\mu_L(x)$ and $\mu_U(x)$ are two FS membership functions, which fulfill the following condition:

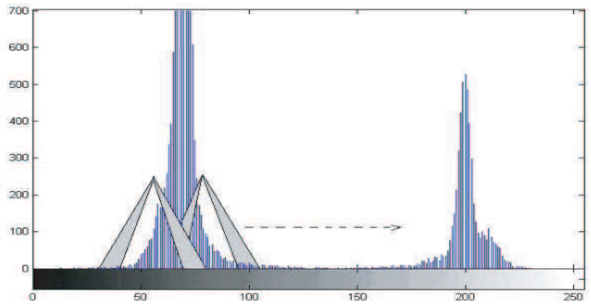
$$0 \leq \mu_L(x) \leq \mu_U(x) \leq 1. \tag{5}$$

3.3 Interval-valued fuzzy sets entropy

The process of selecting the necessary information for image processing must lead here to the correct estimate of the image quality index. The present work demonstrates an application of fuzzy set theory to evaluate this index, with the highest accuracy possible (so a good evaluation of uncertainty is essential). Three kinds of uncertainty are currently established [26]: fuzziness (vagueness), nonspecificity and



(A) Membership function of a IVFS



(B) Shifting of the membership function

Fig. 2: Membership function and shifting of a IVFS.

discord. The terms *fuzziness degree* [27] and *entropy* [28] provide the measurement of fuzziness in a set and are used to define the vagueness degree of the process. These well-known concepts have been developped in a previous paper ([24]). Nevertheless, the total amount of uncertainty is difficult to calculate in the case of fuzzy sets (FS), and particularly when images (represented using a FS) are corrupted with noise, so we introduce the IVFS imprecision degree (imprecision of approximation of a fuzzy set; Many authors named it interval-valued fuzzy entropy for historical reasons). Burillo [29] presented an interesting study relative to the entropy of an IVFS A in X. The defined entropy measures how interval-valued a set is with respect to another fuzzy set. A real function $Ind(A) \quad : IVFS(X) \rightarrow R^+$ is called an entropy on IVFS(X) (where IVFS(X) is the set of IVFSs on X) if Ind(A) satisfies the following conditions:

- entropy is null when the set is a FS,
- entropy is maximum if the set is totally undetermined,
- entropy of an IVFS is equal to its respective complement,
- if the degree of membership and the degree of non-membership of each element increase, this set becomes fuzzier and therefore the entropy should decrease.

Among all the possible fuzzy entropies characterized by Burillo, we can choose the following measure (this measure is the same Sambuc gives for the indetermination degree for IVFS (he named Φ-fuzzy set) [30]):

$$\begin{aligned} Ind(A) &= \sum_{i=1}^{q} \delta_A(x_i) \\ &= \sum_{i=1}^{q} [\mu_U(x_i) - \mu_L(x_i)] \end{aligned} \tag{6}$$

For a $M \times N$ image subset $A \subseteq X$ with G gray levels $g \in [0, G-1]$, the histogram $h(g)$, Tizhoosh [11], intuitively proved that it is very easy to extend the previous concepts of FS (linear index of fuzziness proposed by Pal [31]) for IVFS, and to define the (linear) *index of ultrafuzziness* as follows:

$$\begin{aligned} \Gamma(x) &= \frac{1}{M.N} \sum_{g=0}^{G-1} [h(x).(\mu_U(x) - \mu_L(x))] \\ &= \frac{1}{M.N} \sum_{g=0}^{G-1} \Big[h(x).\Big([\mu(x;g,\sigma)]^{1/\alpha} - [\mu(x;g,\sigma)]^{\alpha}\Big)\Big] \\ &= \frac{1}{M.N} \sum_{g=0}^{G-1} [\Delta_g(x)] \end{aligned} \tag{7}$$

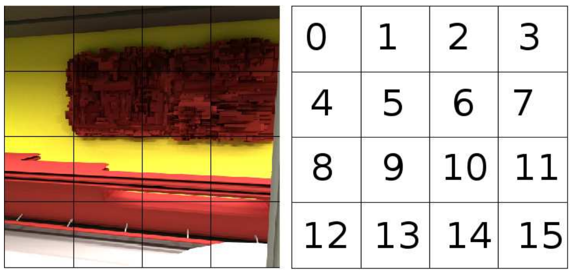
The previously defined IVFS (Fig. 2) is shifted over the interval $[0, G-1]$ of the histogram of the image by varying g but keeping σ fixed. When σ is fixed, the whole function $\Delta_g(x)$ can be determined uniquely given g. Once functions $\Delta_g(x)$ are calculated for every g, it is possible to obtain the linear index of ultrafuzziness $\Gamma(x)$. This basic definition is in accordance with Burillo's work and verifies the four conditions proposed by Kaufmann [27] for the measure of fuzzy set uncertainty. Among the numerous frameworks of uncertainty modeling, this last equation seems to be an interesting tool for image processing. Using the same definition of the linear index of ultrafuzziness, we used this performing index to propose a generalization of Gaussian kernel filtering [24]. Considering the interesting ability of IVFS to filter noisy images, ultrafuzziness is now proposed to be used for noise level measure.

4. The design of the type-2 fuzzy sets image quality evaluation

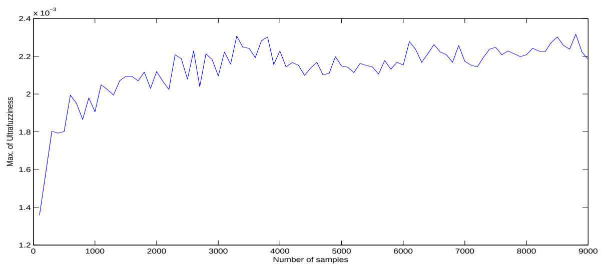
We propose a fuzzy entropy approach (ultrafuzziness) to take into account simultaneously the local and global properties of the pixels and extract a new noise level index.

4.1 Proposed scheme

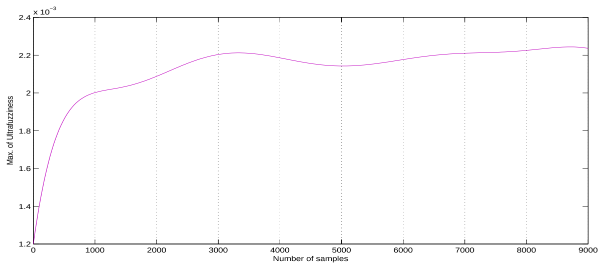
The noise level measure scheme is divided into two steps. In the first one, we perform histogram analysis using the IFS2 ultrafuzziness index $\Gamma(x)$ applied to a block of the processed image I (obtained with Ni samples). The processed image is analyzed using type-2 fuzzy set (both the occurence of the gray levels and the neighboring



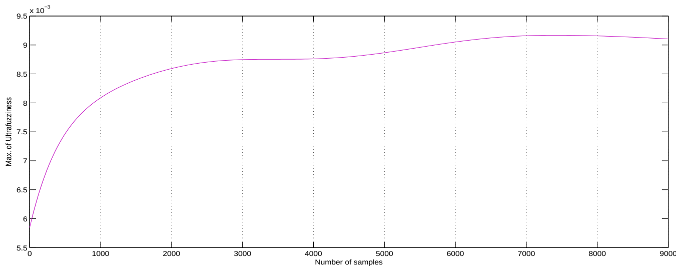
(A) Reference Image



(B) Ultrafuzziness evolution vs. samples number, block 0



(C) Fitted Ultrafuzziness evolution vs. samples number, block 0



(D) Fitted Ultrafuzziness evolution vs. samples number, block 8

Fig. 3: Original image and ultrafuzziness evolution.

homogeneity value among pixels is considered) and then fuzzy entropy is computed. So local and global information are employed in the algorithm. We use the efficient peak detection method (dominating peaks represent regions in the histogram analysis) presented by Cheng [32] to obtain the maximum ultrafuzziness for that block. In the second step, we applied the same treatment on that block with N samples more, and so forth until ultrafuzziness index remains constant: the stopping threshold is reached.

4.2 Type-2 fuzzy set entropy

The membership function of the IVFS is shifted over the gray-level range and the amount of fuzziness is calculated (using equation 7). So we are able to transform an image into fuzzy domains with maximum fuzzy entropy. The proposed image quality evaluation could be described as a system in which the input is an image and the entropy threshold value. The output of the system is the stopping threshold.

4.3 Algorithm

The implementation of image quality evaluation based on type-2 fuzzy sets and measure of ultrafuzziness Γ is given by the following algorithm:

Algorithm 1 Image quality measure

```
Require: a $M \times N$ gray-level image I, $Ni$ samples
 1: Select the shape of MF
 2: Compute the image histogram
 3: Initialize the position of the membership function
 4: Shift the MF along the gray-level range (see Figure 3)
 5: $\Gamma_{max} \leftarrow 0$
 6: for each position $g$ do
 7:     Compute $\mu_U(g)$ and $\mu_L(g)$
 8:     Compute $\Gamma(g) = \frac{1}{M \times N} \sum_{g=0}^{L-1} h(g) \times [\mu_U(g) - \mu_L(g)]$
 9:     if $\Gamma_{max} \leq \Gamma(g)$ then
10:         $\Gamma_{max} \leftarrow \Gamma(g)$
11:     end if
12:     Keep the value $\Gamma_{max}$
13: end for
14: Iterate the number of samples: $Ni + N$
```

4.4 Experimental results with a synthetic image

In order to test the performance of the proposed technique, some results obtained with the synthetic image named "Bar", figure 3, are shown is this presentation (other images were tested and same behaviors were observed, so they are not presented here due to the lack of space). This image is composed of homogeneous and noisy blocks and is interesting to present some results. We also present the ultrafuzziness curves for some blocks of this image. The first figure represents the "Bar" image, the second one represents the ultrafuzziness curve for block number "0", the third one the fitted curve for the same block and the fourth one the same curve for another block (block "8"). These curves exhibit an important slope when the samples number is low (the noise is huge) and then the slope becomes gentle (the noise decrease and the image becomes visually satisfactory). The stopping criterion (noise threshold) is obtained when the slope becomes lower than a chosen value (obtained from a large quantity of tests). *How does it work?* It is now well-known that images and fuzzy sets can be modeled in the same way. The

proposed noise-level evaluation method generates 255 fuzzy partitions (X_i) of the image when shifting the MF (FS1) along the gray-level range. To each pixel x_i of the partition X_i corresponds a membership value $\mu(x_i)$, when using FS1. Using FS2, the method generates 255 fuzzy partitions L_i and 255 fuzzy partitions U_i, corresponding to the shifting of the MF's (U and L are the "bounds" of the IFS2) along the gray-level range. To each pixel x_i of the partition L_i (U_i respectively) corresponds a membership value $\mu_L(x_i)$ (and $\mu_U(x_i)$, respectively), when using FS2. Let us consider one pixel and its neighbors. If these pixels are uncorrupted by noise, they have about the same gray level, and consequently the same membership values. So we obtain about the same results with FS1 and FS2. On the other hand, if some pixels in the neighboring areas are noisy, the FS1-entropy consider only $\mu(x_i)$, while FS2-entropy consider the difference $\mu_L(x_i)$ - $\mu_U(x_i)$. The maximum difference in each fuzzy partition L_i and U_i represents the uncertainty about the gray-level value of the pixel considered and consequently the noise-free accepted range of that pixel. So noise is taken into account (the greater the difference $\mu_L(x_i)$ - $\mu_U(x_i)$ is, the bigger the entropy is).
The **main idea** of the paper is the following: synthesis process is started with a certain number of (random) samples. In our experiments this initial number is 100 and corresponds to a great noise level and a certain entropy value. So the initial position of the synthesis process is unknown but the observed behavior measured at each iteration of the image synthesis process brings us information. The average information quantity gained at each iteration is entropy. The measured entropy using IVFS, named ultrafuzziness, seems to be an interesting measure of noise level and supplies a no-reference quality evaluation used as stopping criterion in the proposed image synthesis process. This proves the advantage of this approach qualitatively (more uncertainty is taken into account using type-2 fuzzy sets, as is previously suggested).

5. Performance comparisons of image quality measures

The algorithm has been implemented on the well-known software "Matlab" on PC (it is important to note that the software has not been optimized). We have applied the unsupervised algorithm we propose further, and extensive experiments have been conducted on a variety of test images to evaluate the performance of the proposed image quality index. Some stopping thresholds are presented fig. 5 where psycho-visual index and proposed index are compared for some blocks of the "Bar" image of fig. 4, and some blocks of the first image of fig.1 ("Deskroom" image). It is easy to notice that these values are very close to each other: we regard these results as an intuitive but instructive comparison. Anyway, with respect to the psycho-visual measure, we would like to highlight the advantages of the proposed measure: this measure is simple; it is parameter free and avoid additional procedures and training data for parameter determination. The measure of structural similarity for

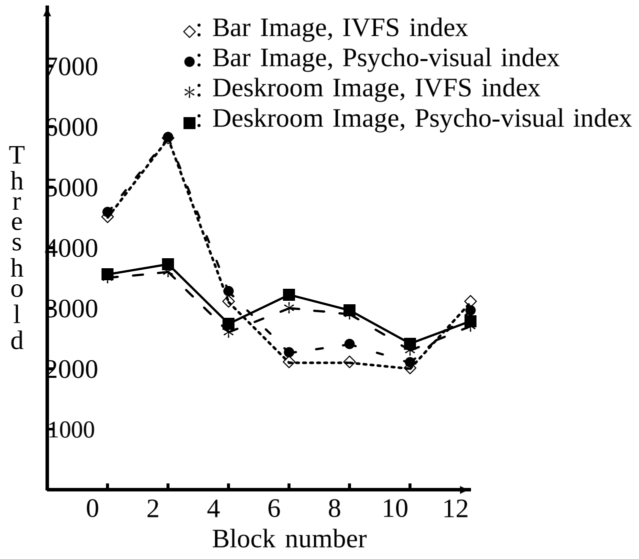


Fig. 4: Stopping threshold vs. block number.

images (SSIM quality index [33]) has also been calculated on the course of iterations as shown fig. 5 (for block "0"). This (supervised) measure is based on the adaptation of the human visual system to the structural information in a scene. The index accounts for three different similarity measures, namely luminance, contrast and structure. The tested image is compared to the reference image: the closer the index to unity, the better the result. It is easy to see that this curve exhibits the same behavior that the ultrafuzziness curve presented figure 3 and confirms the performance of the proposed index.

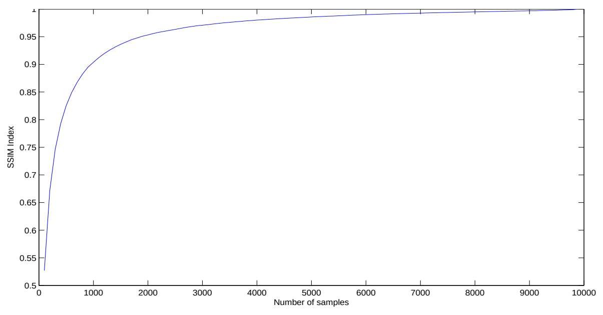


Fig. 5: SSIM index versus samples number.

6. Conclusion

The central idea of this paper was to introduce the application of type-2 fuzzy sets, to take into account the total amount of uncertainty present at the image synthesis stage, and this idea seems to be very promising. The method effectively combines image histograms information with the spatial information about pixels of different gray levels by using a FS2 entropy technique. Like other techniques based on image histogram, this technique is simple and computationally efficient and makes it possible to be used as automatic stopping criterion in image synthesis. Particularly, it assumes no "a priori" knowledge of a specific input image, no learning, no numerous tunable parameters, yet it has good performance compared to a psycho-visual method. Uncertainty is correctly treated, but there remains some open questions to effectively establish a link between the characteristics of noise affecting the image (noise level) and the choice of the FS2. We have processed IFS2, but other types of FS2 could be more effective, as proposed in the papers [34] and [35], for visual results could still be improved. In particular, more extensive investigations on other measures of entropy and the effect of parameters influencing the width (length) of the interval of IVFS are under investigation (to establish a link between this interval and level and type of noise) towards an automatic noise quantification and will be presented in the future.

References

[1] R. Ferzli and L. Karam, "No-reference objective wavelet based noise immune image sharpness metric," in *International Conference on Image Processing*, 2005.

[2] J. Zhang, S. Ong, and M. Thinh, "Kurtosis-based no-reference quality assessment of JPEG2000 images," *Signal Processing: Image Communication*, vol. 26, pp. 13–23, 2011.

[3] O. Faugeras, "Digital color image processing within the framework of a human visual model," *IEEE Trans. on ASSP*, vol. 27, pp. 380–393, 1979.

[4] J. Kajiya, "The Rendering Equation," *ACM Computer Graphics*, vol. 20, no. 4, pp. 143–150, Août 1986.

[5] P. Shirley, C. Wang, and K. Zimmerman, "Monte Carlo techniques for direct lighting calculations," *ACM Transactions on Graphics*, vol. 15, no. 1, pp. 1–36, 1996.

[6] D. P. Mitchell, "Generating antialiased images at low sampling densities," in *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press, 1987, pp. 65–72.

[7] J. Farrugia and B. Péroche, "A Progressive Rendering Algorithm Using an Adaptive Perceptually Based Image Metric." *Comput. Graph. Forum*, vol. 23, no. 3, pp. 605–614, 2004.

[8] P. Longhurst, K. Debattista, and A. Chalmers, "A gpu based saliency map for high-fidelity selective rendering," in *AFRIGRAPH 2006 4th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*. ACM SIGGRAPH, January 2006, pp. 21–29.

[9] A. Bigand, T. Bouwmans, and J. Dubus, "Extraction of line segments from fuzzy images," *Pattern Recognition Letters*, vol. 22, pp. 1405–1418., 2001.

[10] H. Cheng, C. Chen, H. Chiu, and H. Xu, "Fuzzy homogeneity approach to multilevel thresholding," *IEEE Trans. Image Process.*, vol. 7, no. 7, pp. 1084–1088, 1998.

[11] H. Tizhoosh, "Image thresholding using type 2 fuzzy sets," *Pattern Recognition*, vol. 38, pp. 2363–2372, 2005.

[12] R. Demirci, "Rule-based automatic segmentation of color images," *Int. J. Electron. Commun.*, vol. 60, pp. 435–442, 2006.

[13] S. Philipp-Foliguet, M. Vieira, and M. Sanfourche, "Fuzzy segmentation of color images and indexing of fuzzy regions," in *Int. Conf. CGIV 2002*, Poitiers, 2002, pp. 507–512.

[14] H. Cheng, X. Jiang, and J. Wang, "Color image segmentation based on homogram thresholding and region merging," *Pattern Recognition*, vol. 35, no. 2, pp. 373–393., 2002.

[15] T. Chen and Y. Lu, "Color image segmentation – an innovative approach," *Pattern Recognition*, vol. 35, pp. 395–405, 2002.

[16] B. Prados-Suarez, J. Chamorro-Martinez, D. Sanchez, and J. Abad, "Region-based fit of color homogeneity measures for fuzzy image segmentation," *Fuzzy Sets and Systems*, vol. 158, pp. 215–229, 2007.

[17] J. Mendel and R. B. John, "Type-2 fuzzy sets made simple," *IEEE Trans. on Fuzzy Systems*, vol. 10, no. 2, pp. 117–127, 2002.

[18] I. Bloch, "Information combination operators for data fusion: a comparative review with classification," *IEEE Trans. on SMC - Part B*, vol. 26, pp. 52–67, 1996.

[19] H. Wu and J. Mendel, "Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems," *IEEE Trans. on Fuzzy Systems*, vol. 10, no. 5, pp. 622–639, Jan. 2002.

[20] Q. Liang, N. Karnish, and J. Mendel, "Connection admission control in ATM networks using survey-based type-2 fuzzy logic systems," *IEEE Trans. on Systems, Man and Cyber. - Part B*, vol. 30, no. 3, pp. 329–339, 2000.

[21] L. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning," *Information Sciences*, vol. 8, pp. 199–249, 1975.

[22] P. Heinonen and Y. Neuvo, "FIR-median hybrid filters," *IEEE Trans. ASSP*, vol. 35, no. 6, pp. 832–833, 1987.

[23] K. Arakawa, "Median filters based on fuzzy rules and its application to image restoration," *Fuzzy Sets and Systems*, vol. 77, pp. 3–13., 1996.

[24] A. Bigand and O. Colot, "Fuzzy filter based on interval-valued fuzzy sets for image filtering," *Fuzzy Sets and Systems*, vol. 161, pp. 96–117, 2010.

[25] H. Bustince, E. Barrenechea, M. Pergola, and J. Fernandez, "Interval-valued fuzzy sets constructed from matrices: Application to edge detection," *Fuzzy Sets and systems*, vol. To appear, 2009.

[26] G. Klir and B. Yuan, *Fuzzy sets and fuzzy logic. Theory and applications*. Prentice Hall, 1995, no. 5/6.

[27] A. Kaufmann, *Introduction to the theory of fuzzy set - Fundamental theorical elements*. New York: Academic Press, 1975, vol. 28.

[28] A. Deluca and S. Termini, "A definition of a nonprobabilistic entropy in the setting of fuzzy set theory," *Information and Control*, vol. 20, no. 4, pp. 301–312, 1972.

[29] P. Burillo and H. Bustince, "Entropy on intuitionistic fuzzy sets and on interval-valued fuzzy sets," *Fuzzy Sets and Systems*, vol. 78, pp. 305–316., 1996.

[30] R. Sambuc, "Fonctions ϕ-floues, application a l'aide au diagnostic en pathologie thyroidienne," Ph.D. dissertation, These de doctorat en medecine, Marseille, 1975.

[31] N. Pal and J. Bezdek, *Measures of fuzziness: a review and several classes*. New York: Van Nostrand Reinhold, 1994.

[32] H. Cheng and Y. Sun, "A hierarchical approach to color image segmentation using homogeneity," *IEEE Trans. Image Process.*, vol. 9, no. 12, pp. 2071–2081, 2000.

[33] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process*, vol. 13(4), pp. 600–612, 2004.

[34] J. Mendel and H. Wu, "Type-2 Fuzzistics for symmetric Interval Type-2 Fuzzy Sets: Part 1, Forward problems," *IEEE Trans. on Fuzzy Systems*, vol. 14, no. 6, pp. 781–792, 2006.

[35] J. Mendel, "Advances in type-2 fuzzy sets and systems," *Information Sciences*, vol. 177, pp. 84–110, 2007.

Brain Segmentation from Volumetric MR Images and Cortical Surface Characterization Using Discrete Curvature Classification

MS. Naceur[1], **K. Aloui**[12]

[1]LTSIRS laboratory, National Engineering School of Tunis (ENIT), Tunisia.
[2]LiSSi EA 3956 laboratory, Paris-Est Créteil Val de Marne University (UPEC), France.

Abstract - *This paper presents a 3D brain segmentation from volumetric Magnetic Resonance Images (MRI) and cortical surface characterization. Brain surface extraction firstly based on low-level segmentation tools to generate an initial interface between gray matter (GM) and cerebral-spinal fluid (CSF). Then level-sets approach is applied to obtain final cortical surface. Afterward pipeline procedure for brain characterization/labeling using discrete curvature classification is developed. This is based on estimating curvature information at each vertex in the surface mesh related to cortical surface.*

Keywords: Brain, Curvature, level-Sets, MRI, Surface mesh.

1 Introduction

Segmentation of anatomical structures of the intracranial cavity is a preliminary and main step for the most analysis procedures for brain MRI images. When, the brain surface, also called cortical surface, is viewed, a human brain appears as a volume with a highly wrinkled boundary surface having numerous long furrows. The term sulci (plural of sulcus) is associated with these furrows and the term gyri (plural of gyrus) designates the regions between the sulci. Geometric modelling, mesh representation and mesh segmentation of anatomical structures in 3D images are becoming increasingly important processing steps. Segmentation of cortical surface should reduce the mesh into meaningful connected pieces. “Meaningful” implies that the partitioned areas are relevant to the application at hand. In addition, visualization of brain structures such as brain surface, the segmentation allows the automatic identification and labeling of cortical sulci, which will be used in neuronavigation applications, understanding brain anatomy and function, neurosurgeon can easily track the features of interest. Also, segmented sulci from a brain mesh can serve as landmarks, which can be used to register the mesh with other brain meshes to make intra or inter-comparisons. For example, this could serve to measure brain growth and identify diseases.

The most common characterization or labeling of a cortical mesh is into sulal and gyral regions. The brain gyri can be defined as the top surfaces of the brain folds (Ridges) or as convex regions. The barin sulci can be defined as the area within the brain folds (valleys) or as concave regions. Segmentation of a cortical surface in terms of sulci and gyri can occur in several ways. Classification or partitioning of the cortical surface in concave and convex regions can reduce the size of the Laplacian matrix, in the case of spectral analysis [1]. Also, it is useful to avoid the risk of main memory congestion, improve rendering speed and reduce the computational cost of rendering surface process that to be instantaneous.

In the last several years, many algorithms have been proposed in this growing area, offering several methods. For example, Rettmann’s works [2] involving the segmentation of sulci using watersheds. This work focuses on segmenting sulcale regions. This paper uses the geodesic depth of mesh points in the sulci regions as the height function of the watershed algorithm. Also Mangan and Whitaker [3] proposed an approach using the watershed algorithm for partitioning 3D surface meshes when total curvature is computed and used in segmentation.

We propose in this paper a pipeline steps for the problem of 3D mesh characterization of the brain in magnetic resonance images. Our approach is based on two stages. Initially, volumetric MRI data is processed to generate a discrete representation of cortical surface. In second stage, we segment the obtained meshes using a criteria based on discrete curvature.

2 Brain surface extraction

Manually segmentation of volumetric images is a complex process which requires lot of time and much concentration to achieve a good quality extraction of regions of interest. For this reason, it is generally interesting to deal with automatic segmentation algorithms. For this purpose, a range of methods including edge based, region based, and knowledge based have been proposed for semi-automatic or automatic detection of various anatomical brain structures. Recently, several attempts have been made to apply deformable models [4], [5], [6] to brain image analysis. Indeed, deformable models refer to a large class of computer vision methods and have proved to be successful segmentation techniques for a wide range of applications. Moreover, they constitute an appropriate framework for merging heterogeneous information and they provide a consistent geometrical representation suitable for a surface based analysis.

In some particular Level-sets [7], geometric deformable models provide an elegant solution for medical images processing [8], [9], [10], [11]. In this paper, to extract brain surface from volumetric MRI images, we will use a region-competition level-sets method as described in [12], [13]. This algorithm overcomes classical level-sets problems by modulating the propagation term with a signed local statistical force, leading to a stable solution.

We propose a method which operates on 3D MRI scans to extract brain surface. First, the data volume is pre-processed with an anisotropic diffusion filtering method [14] to reduce noise and preserve edges. Then no-brain tissues are removed from the data volume using a simple thresholding. Afterwards, a 3D binary morphological erosion and dilation processes [15] are applied to get an initial brain surface and finally, we refine brain region extraction by using region-based information into the level-sets framework. These computational steps are illustrated in figure 1.

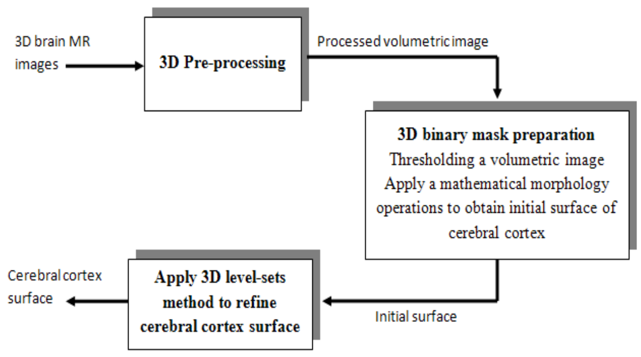

3D brain MR images
3D Pre-processing
Processed volumetric image
3D binary mask preparation
Thresholding a volumetric image
Apply a mathematical morphology operations to obtain initial surface of cerebral cortex
Cerebral cortex surface
Apply 3D level-sets method to refine cerebral cortex surface
Initial surface

Fig. 1 Proposed method for brain cortical surface segmentation.

The aim of the third step is to extract “exactly” the cerebral cortex surface representing the interface between GM and CSF. For this reason, we propose to use a deformable model algorithm based on the level set technique. We propose also to drive our model by region information instead boundary information, because it is more robust. The process requires an initialization step and speed function. Theoretically, the level-set snake is defined as the zero level set of an implicit function ϕ defined on the entire volume. This function will change over the time according to the speed term F. The evolution of ϕ is defined as in [7] via a partial differential equation:

$$\frac{\partial \phi}{\partial t} = F|\nabla \phi| \tag{1}$$

The classical speed term is defined as in [16], [17]:

$$F = g(|\nabla I|)(v + k) \tag{2}$$

Speed term is coupled with the image data through a multiplicative stopping term $g(|\nabla I|)$. The curvature k and the constant force v propagate curve near the region of interest surface.

In this work, we use a simplified version of the level-sets formulation [13]. The model shrinks when the boundary encloses parts of the background, and grows when the boundary is inside the brain region. Here, the speed function usually consists in a combination of two terms: curvature term for smoothness and data term for evolution. The snake evolves using the following equation:

$$\frac{\partial \phi}{\partial t} = \alpha D(x)|\nabla \phi| + (1-\alpha)k|\nabla \phi| \tag{3}$$

Where D is a data term that forces the model to expand or contract toward desirable features in the input 3D MRI data. This was done by making D positive in desired regions or negative in undesired regions. The term k is the means curvature of the surface, which forces the surface to have less area (and remain smooth), and $\alpha \in [0, 1]$ is a free parameter that controls the degree of smoothness.

The speed function depends on the grayscale value input MRI data denoted I at the point x:

$$D(I) = \varepsilon - |I - T| \tag{4}$$

Where T controls the brightness of the region to be segmented and ε controls the range of grayscale values around T that could be considered inside the object. A model situated on voxels with grayscale values in the interval $T \pm \varepsilon$ will expand to enclose that voxel, whereas a model situated on grayscale values outside that interval will contract to exclude that voxel.

As represented in Figure 2, T describes the central intensity value of the region to be segmented, and ε describes the intensity deviation around T that is a part of the desired segmentation. Therefore if a voxel has an intensity value within the $T \pm \varepsilon$ range, the model will expand; otherwise it will contract.

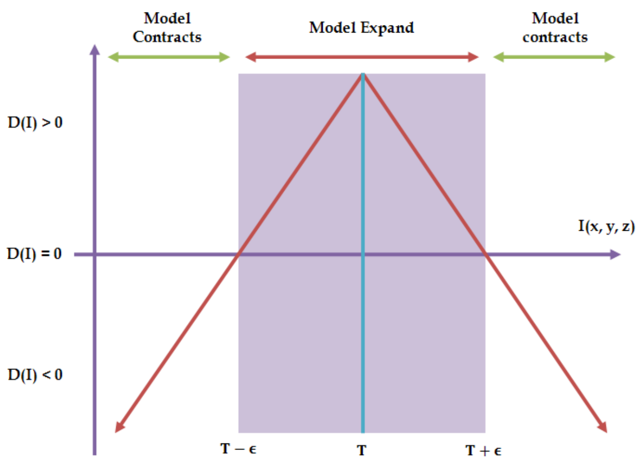


Fig. 2 The speed term from [13]

Consequently, the three user parameters that need to be specified for the segmentation are T, ε and α. The initial surface obtained after pre-processing must be transformed into a signed distance [18] for the level-sets function is also required (FIG. 3). The level-sets iteration can be terminated once ϕ has converged, or after a certain number of iterations.

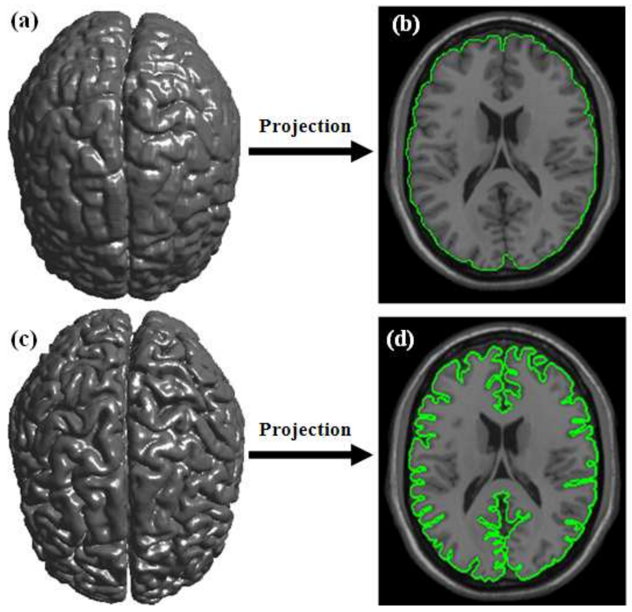


Fig. 3 Cortical surface obtained after 3D segmentation of volumetric brain MR Images. (a) Initial cerebral cortex surface, (b) show 2D projection into 2D brain slice of initial surface, (c) Cerebral cortex surface obtained using 3D level-sets method, (b) 2D projection of brain refined surface into 2D brain slice.

3 Cortical surface characterization

Curvatures can also be used as a height measure. The idea is that ridges and valleys have opposite signed curvatures, and the cortical surface is naturally divided between crest lines or ridges (gyri) and valley (sulci). For this reason, we compute gaussian curvature, mean curvature and the two principal curvatures, and later use them to classify the surface type of vertices. The major complication is that curvature cannot be directly evaluated for triangle meshes because it is mathematically defined for smooth surfaces only. However, discrete differential geometry operators have been developed which can estimate curvatures on triangulated manifolds [19], [20]. We apply here some operators, which are derived recently by Meyer et al [21], to estimate curvature information at each vertex in the mesh. We denote H and G, the Mean and the Gaussian discrete curvature operators. k_1 et k_2 are the two principal curvatures operators. We calculate these operators according to the following formulas:

$$h(x_i) = \frac{1}{2A} \sum_{j \in N(i)} (\cot(\alpha_{ij}) + \cot(\beta_{ij}))(x_i - x_j) \quad (5)$$

Mean curvature

$$H(x_i) = \frac{1}{2} \| h(x_i) \| \quad (6)$$

Gaussian curvature

$$G(x_i) = \frac{2\Pi - \sum_{j=1}^{f} \theta_j}{A} \quad (7)$$

As shown in Figure 4, α_{ij} and β_{ij} are the two angles opposite to the edge in the two triangles sharing the edge (x_i, x_j), θ_j is the angle of the j^{th} face at the vertex x_i and f denotes the number of faces around this vertex. The two principal curvatures are expressed as shown by the following equations:

$$k_1(x_i) = H(x_i) + \sqrt{\Delta(x_i)} \quad (8)$$

$$k_2(x_i) = H(x_i) - \sqrt{\Delta(x_i)} \quad (9)$$

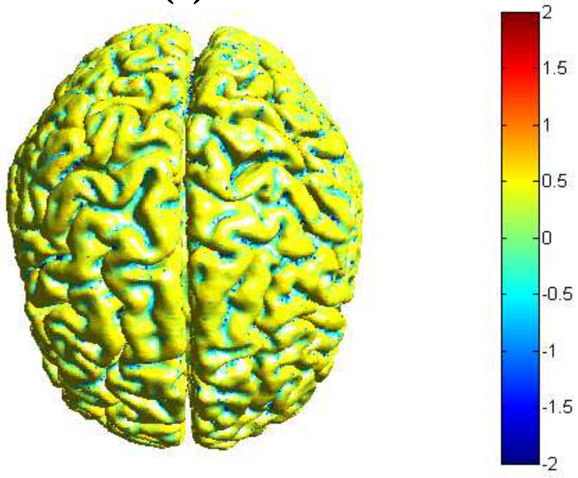
Where

$$\sqrt{\Delta(x_i)} = H^2(x_i) - K(x_i) \quad (10)$$

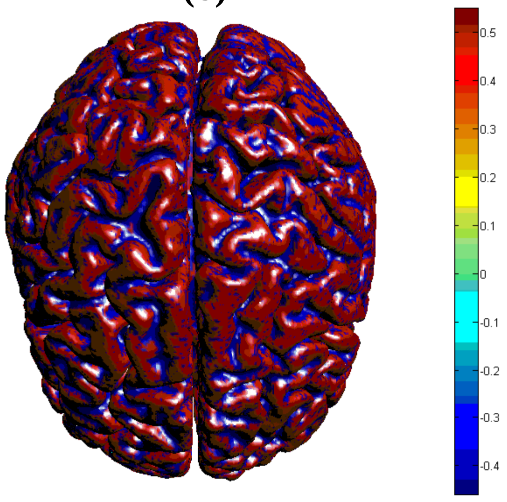
By examining discrete curvatures on triangular meshes, one can achieve the following analysis:

– Concave regions/valleys (sulci) :

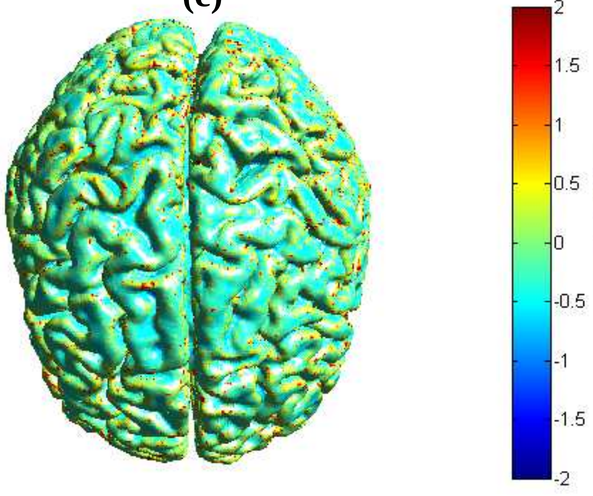
(a)


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2

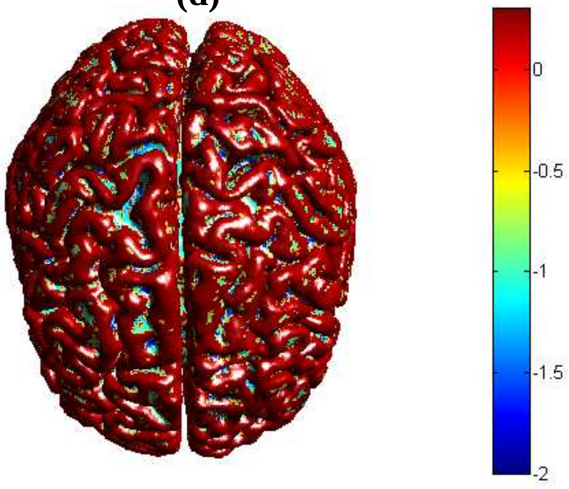
(b)


0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4

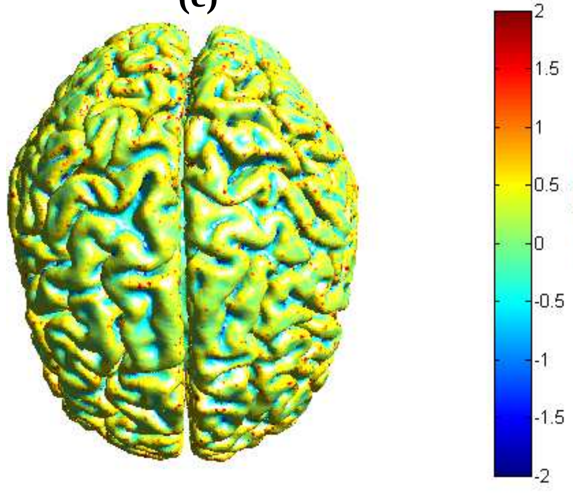
(c)


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2

(d)


0
-0.5
-1
-1.5
-2

(e)


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2

(f)

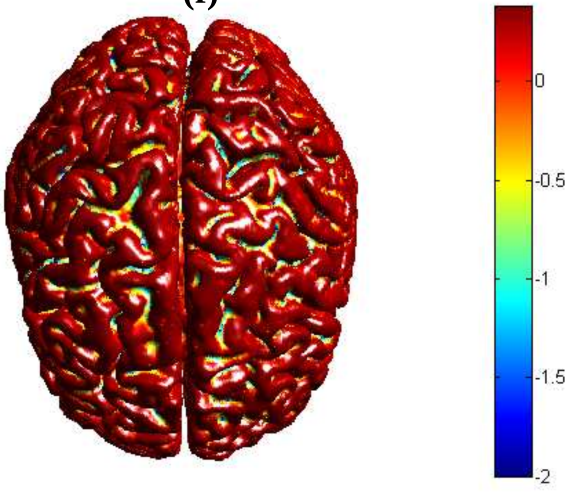

0
-0.5
-1
-1.5
-2

Fig 5. Corticale surface segmentation using discrete curvature classification into sulci and gyri. (a) Principal curvature k_1, (b) Principal curvature k_1 after thresholding, (c) Principal curvature k_2, (b) Principal curvature k_2 after thresholding , (a) Mean curavature H, (b) Mean curvature H after thresholding.

$$k_2 \prec\prec 0 \text{ and } k_1 \simeq 0$$

– Convex regions/ crest lines (gyrus) :

$$k_1 \succ\succ 0 \text{ and } k_2 \simeq 0$$

– If the H value is negative, then we have a convex behavior (gyri), otherwise it is concave (sulci).

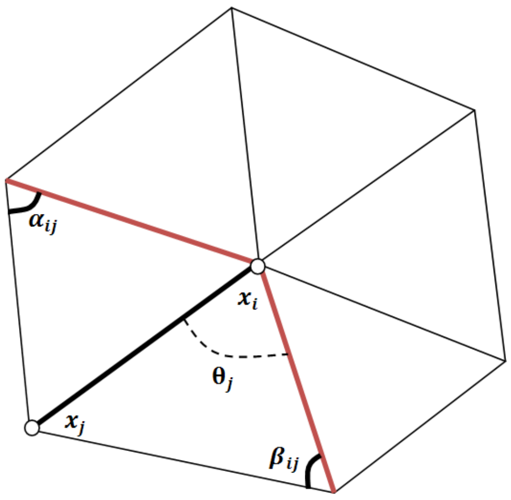


Fig. 4 1-ring neighbors and angles opposite to an edge.

We have performed a series of experiments on brain MR images from MeDEISA database [22]. Results of cortical surface characterization by curvature classification are depicted in Figures 5. Through this classification process, it will be easy to distinguish between gyri that are the surfaces top of the brain folds (ridges), and sulci that are the surfaces within the brain folds (basins). Each label corresponds to a different color for each curvature operator.

4 Conclusions

This paper proposes a novel method to decompose cortical surface represented by triangle meshes into separate parts based sulci and gyri using discrete curvature classifiaction. Even if they have not yet been compared to manual or other automatic segmentation results, we think that they are encouraging and faster than manual procedures. Thus, the prospects of this work would be to use our approach to address studies on asymmetry of brain anatomy, the interindividual variability of brain anatomy, neurological dimension of certain mental diseases such as autism and schizophrenia, or, in the context of longitudinal studies on the characterization of brain development for healthy or pathological subject.

5 References

[1] G. Lavoué. “Compression de surfaces, basée sur la subdivision inverse, pour la transmission bas débit et la visualisation progressive“. Thèse de doctorat, Université Claude Bernard Lyon 1, 2005.

[2] M.E Rettmann, X. Han, C. Xu, and J.L. Prince. “Automated Sulcal Segmentation Using Watersheds on the Cortical Surface”. NeuroImage, No.15, p. 329-344, 2002.

[3] Mangan and R. Whitaker. “Partitioning 3d surface meshes using watershed segmentation”. IEEE Transactions on Visualization and Computer Graphics, 5, 1999.

[4] T. McInerney and D. Terzopoulos, “Deformable models in medical image analysis: A survey,” Medical Image Analysis, vol. 1, pp. 91-108, 1996.

[5] M. Kass, A. Witkin, and D. Terzopoulos, “Snakes: Active contour models,” International Journal of Computer Vision, vol. 1, pp. 321-331, 1987.

[6] C. Xu, D. Pham, and J. Prince, “Image segmentation using deformable models,” In Handbook of Medical Imaging, vol. 2: SPIE, 2000, pp. 129-174.

[7] J.A. Sethian, “Level Set Methods and Fast Marching Methods,” Cambridge University Press, 1996.

[8] S.K. Warfield, M. Kaus, F.A. Jolesz and R. Kikinis, “Adaptive, Template Moderated, Spatially Varying Statistical Classification,” Medical Image Analysis, 20(8):43–55, 2000.

[9] C. Baillard and al. “Cooperation between Level Set Techniques and 3D Registration for the Segmentation of Brain Structures,” Int. Conf on Pattern Recognition ICPR, 2000, 991-994.

[10] J.S. Suri and S. Singh and L. Reden, “Computer Vision and Pattern Recognition Techniques for 2-D and 3-D MR Cerebral Cortical Segmentation (Part I): A State-of-the-Art Review,” Pattern Anal. and App., 5:46-764, 2002.

[11] A.W.C. Liew and H. Yan, “Current Methods in the Automatic Tissue Segmentation of 3D Magnetic Resonance

Brain Images," Current Medical Imaging Reviews, 2:91-103, 2006.

[12] S. Ho, E. Bullitt and G. Gerig, "Level set evolution with region competition: Automatic 3-D segmentation of brain tumors,' 16th Int. Conf on Pattern Recognition ICPR, 2002, 20(8):532-535.

[13] A.E. Lefohn, J. Cates and R. Whitaker, 'Interactive, GPU-based level sets for 3D brain tumor segmentation," Medical Image Computing and Computer Assisted Intervention, 564-572, 2003.

[14] P. Perona and J. Malik,"Scale-space and edge detection using anisotropic diffusion," IEEE Trans. Med. Imaging, 12:629-639, 1990.

[15] J.P. Serra, "Image Analysis and Mathematical Morphology," Academic Press Inc, 1982.

[16] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic Active Contours," Inter. Journal Comput. Vision, 22(1):61-97, 1997.

[17] R. Malladi, J.A. Sethian, and B.C. Vemuri, "Shape modeling with front propagation: A level set approach," IEEE Trans. Pattern Anal. Machine Intell, 17(2):158-174, 1995.

[18] Stanley Osher and James A. Sethian, "Fronts propagating with curvature dependent speed: algorithms based on hamilton-jacobi formulations," Journal of Computational Physics, 79:12-49, 1988.

[19] B. Hamman."Curvature approximation for triangulated surfaces". In: Gerald Farin et al., editors,1993 Geometric Modelling, Computing Supplementum, Springer-Verlag, Vienna, Austria, pages 139–153.

[20] N. Max. "Weights for computing vertex normals from facet normals". Journal of Graphics Tools, 1999 4(2), pages1–6.

[21] M. Meyer, M. Desbrun and P. Schroder. "Discrete differential-geometry operators for triangulated 2-manifolds". International Workshop on Visualization and Mathematics, Berlin, 2002, Germany.

[22] Medical Database for the Evaluation of Image and Signal processing (MeDEISA), www.medeisa.net.

Real Time Terrains Realistic Illumination

Belaiche Hamza[1], **Babahenini Med Chaouki**[2], **and Zidani Abdelmadjid**[2]
[1]Department of computer, University of Batna, Batna, Algeria
[2]Department of computer, University of Biskra / University of Batna, Batna, Algeria

Abstract— *The research work reported here deals with landscape illumination techniques. We describe a new technique for computing the terrain shadows (illumination). We try to augment the realism without augmenting considerably the algorithm's complexity and without using global illumination model. In this paper we will first discuss the main research works made in the landscape lighting area, we will explain the outdoor lighting of terrain in the real world, and give two techniques for simulating these lighting in real time ; the first is very fast but not realistic, the second one treat the sun as a simple directional light source, except for the possibility of partial occlusion but a very heavy phase to calculate the horizon angles is necessary . We present after our approach.*

Keywords: Terrains, illumination, shadows, realistic, real time, horizon angles.

1. Introduction

Many games have scenes set in outdoor environments. In these environments, the most visually dominant object is the terrain. The model of a Landscape is a two dimensional grid of heights (heightmap [1]) that can be stored as an image [2], [3], [4], [5].Succeed in view the terrain is a good thing. But it lacks one important thing: the shadows in order to show the relief. It is associated with each pixel of the heightmap a coefficient of light [1]. It will be a floating point number between 0 and 1 and we multiply this number by the color for the final color [6].

1.1 Main works

There is much published work in the game development community on efficiently generating visible terrain geometry, but relatively little on lighting it[8]. In this talk, we will describe tow techniques on terrain lighting.

1.1.1 The raytracer method

The algorithm is quite simple in fact [7]. For every grid point, we check if the ray from the light position to the point intersects the map. This is done very quickly by checking only the points that are under the projection of the vector L, as shown in Figure 1 illustrated below.

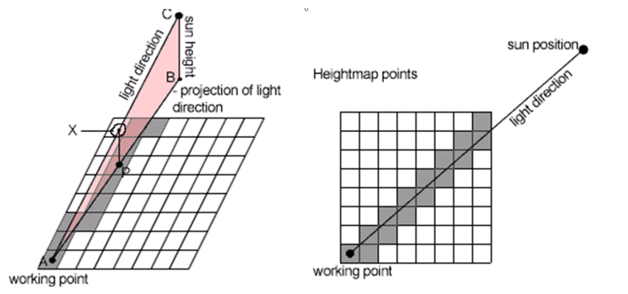


Fig. 1: Ray tracer principle.

Let us consider the following notations:

- A = working point;
- B = coordinates of the sun position projection;
- C = coordinates of the sun position;
- L = light direction vector $(L = A - C)$;
- P = any point that lays under the projection of vector L;
- $X(P)$ = coordinates on the L vector whose projection is point.

The points P are generated using a $2D$ line algorithm from the working point (A) to the projection of the sun position point (B). Then, for every point P, you check to see if the height map value for point P is bigger than the y value of point $X(P)$. If it turns out to be bigger, then we know that the ray L intersects the map, and the illumination at point A will be equal to '0', and we can safely move to the next working point. If all the P-type points have been tested and no intersection was found, then the illumination at point A is computed using the following formula:

$$ColorFin = Color * (\vec{L} \cdot \vec{N}(A)) \qquad (1)$$

Where $\vec{N}$ is the normal at the point A.

The sun is very remote, so we consider it as a directional light source, something not considered in this interactive algorithm, the angular diameter of the sun which produces the effect of shadows with soft edges.

1.1.2 Real-Time analytical computation of outdoor lighting

This technique makes use of several simplifying assumptions, approximations and precomputed data to calculate a complete outdoor lighting solution in real-time [8].

For each texel in the lightmap, we precompute and store horizon angles in the same plane as the sun's arc, thus creating a horizon map (first introduced by Max in [9].

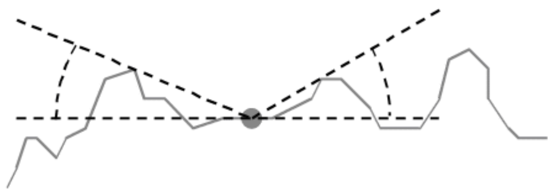


Fig. 2: Horizon angles.

Then at runtime, as the sun's position and color changes, we calculate maximum and minimum angles each frame:

$$\theta_{Min} = \theta_{Center} - \frac{1}{2}\alpha \tag{2}$$

$$\theta_{Max} = \theta_{Center} + \frac{1}{2}\alpha \tag{3}$$

Where θ_{Center} is the elevation angle of the center of the sun disk, and α is the angular diameter of the sun (though to achieve pleasing soft-shadow effects).

Then we looking up the precalculated lighting, and multiplying it by the visibility/occlusion factor. This factor is calculated by:

$$\theta_{Sun} = \frac{\theta_{Max} - \theta_{Horizon}}{\theta_{Max} - \theta_{Min}} \tag{4}$$

This technique considers that the terrain model and horizon map are already stored, otherwise a very heavy phase to calculate the horizon angles is necessary.

Sometimes it is useful to model and to render simultaneously and interactively (real time), why, we must look for an approximate and interactive method to simulate the effect of the angular diameter of the sun without using the horizon map.

The sun is not a point light source, so they exists points in the land for which the sun is partially visible, then the solution of lighting for these points should be multiplied by a sun factor of visibility, the problem therefore lies in the calculation of this factor.

2. Our methodology

The technique which we will try to propose here is based on the ray tracer method.

Here we will simulate the effect of shadows with soft edges, set by the angular diameter of the sun. After multiplying the color by the coefficient of light, we will multiply the result by a visibility factor V. So the problem lies in the calculation of this factor. We consider that the light vector is a cylinder which has a perpendicular diameter $(R = 2r)$. The visibility factor should approximate the ratio between the visible surface and the total area of light vector, then the algorithm is as follows:

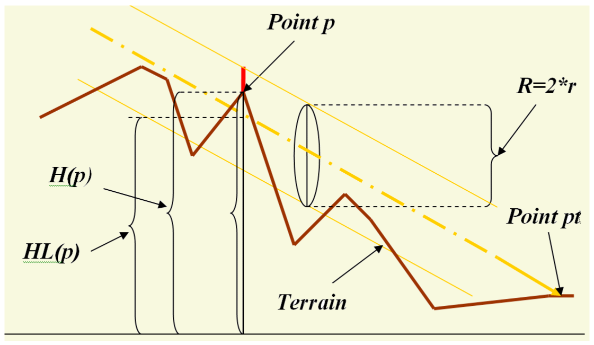


Fig. 3: Cylinder principle.

From the point Pt that interests us, we will follow the light vector. If there is no point with a height $H(p)$ such that $H(p) > (HL(p) - r)$, or $HL(p)$ is the height of the center of light vector at this point; this means there is not any point on the land which hide a portion of the surface of the light vector, so the sun is fully visible, the visibility factor of sun will be equal to 1 $(V(pt) = 1)$ and we get the final color:

$$ColorFin = Color * (\vec{L} \cdot \vec{N}(pt)) \tag{5}$$

Where $\vec{N}(pt)$ is the normal at the point pt and $\vec{L}$ the light vector.

If there are points which satisfy the condition $H(p) > HL(p) - r$, we have two possible cases:

- If $H(p) \geq (HL(p)+r)$, the sun is invisible $V(pt) = 0$, so $ColorFin = 0$.
- If $(HL(p) - r) < H(p) < (HL(p) + r)$, then the sun is partially visible, therefore:

$$ColorFin = V(pt) * Color * (\vec{L} \cdot \vec{N}(pt)) \tag{6}$$

The visibility factor is the ratio of the visible surface (length of the red line) on the total area of light vector (R), then:

$$V(pt) = \frac{(HL(p) + r) - H(p)}{R} \tag{7}$$

If we have several points for which the sun is partially visible we keep the minimum visibility factor.

2.1 Limitation

From the Figure 4 we note that from the outset, a part of the surface of the light vector is hidden by the neighbors of the working point pt, which makes the sun still partially visible, why we have the shadow everywhere as shown in Figure 5.

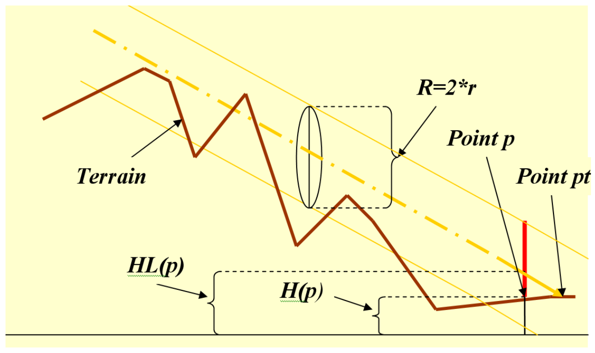
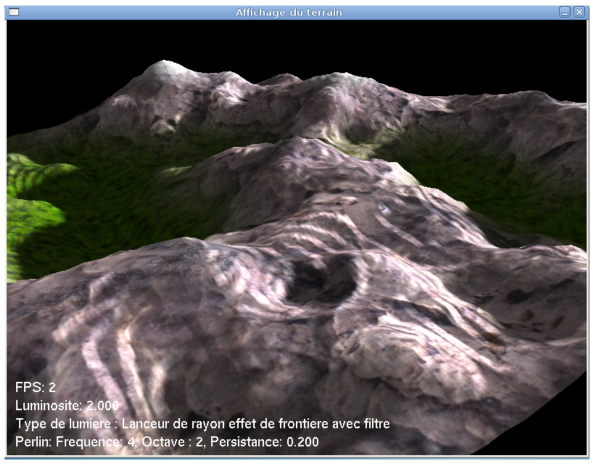


Fig. 4: Problem of the first version.

Here is a picture illustrating the result of this solution:



```
Affichage du terrain
FPS: 2
Luminosite: 2.000
Type de lumiere : Lanceur de rayon effet de frontiere avec filtre
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200
```

Fig. 5: Solution (false) of the first version.

2.2 Method improvement

As we have seen if we consider that the light vector is a cylinder, we have the problem that the neighbors of the working point obscure the light vector; in this section we will solve this problem.

Instead of seeing the light vector is a cylinder, we assume that is a cone with a diameter of variable length. The length diameter starts from value 0, the value increases linearly with the displacement of the light vector at every step Δx adds value Δr for the light vector diameter.

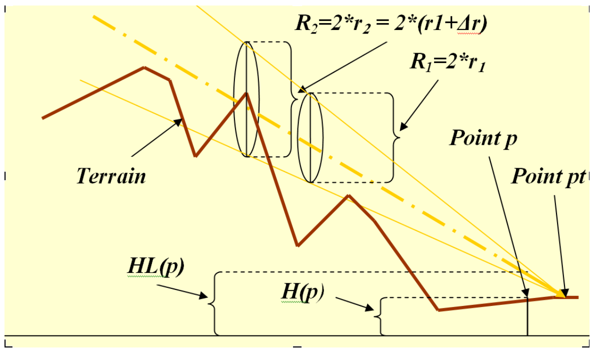
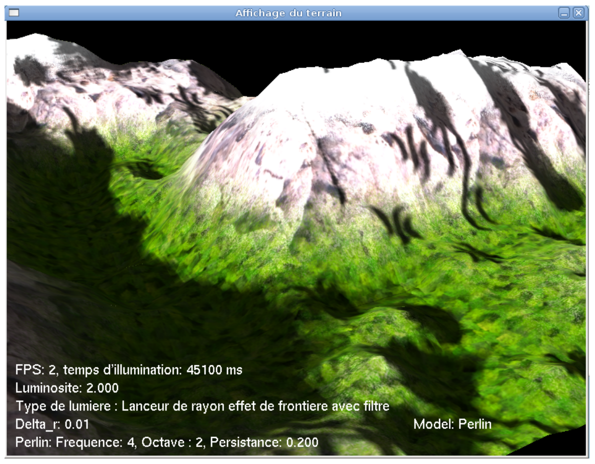


Fig. 6: Principle of Cone.

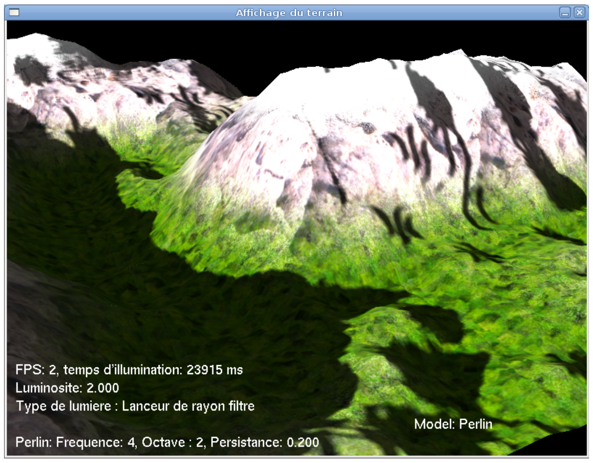
The Figure 7 is a picture illustrating this solution:



```
Affichage du terrain
FPS: 2, temps d'illumination: 45100 ms
Luminosite: 2.000
Type de lumiere : Lanceur de rayon effet de frontiere avec filtre
Delta_r: 0.01                                                     Model: Perlin
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200
```

Fig. 7: Final Result.

If we compare this image with the result of ray tracer in Figure 8:



```
FPS: 2, temps d'illumination: 23915 ms
Luminosite: 2.000
Type de lumiere : Lanceur de rayon filtre
                                              Model: Perlin
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200
```

Fig. 8: Result of ray tracer.

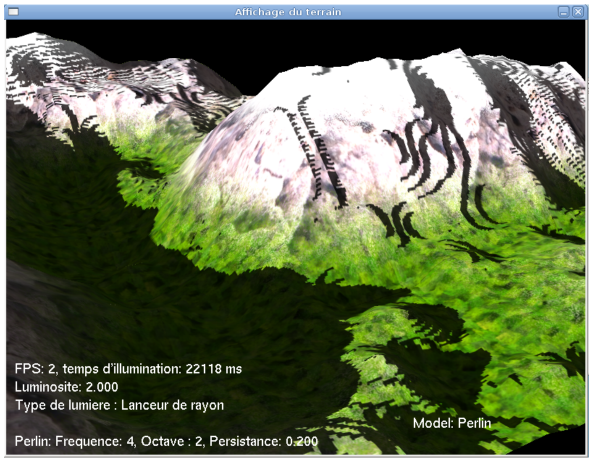
We note that the difference is clearly visible in the proposed method which has shadows with soft edges.

3. Results

In what follows we will evaluate our approach by comparison with the method raytracer, then exploiting our method for modeling the night light.

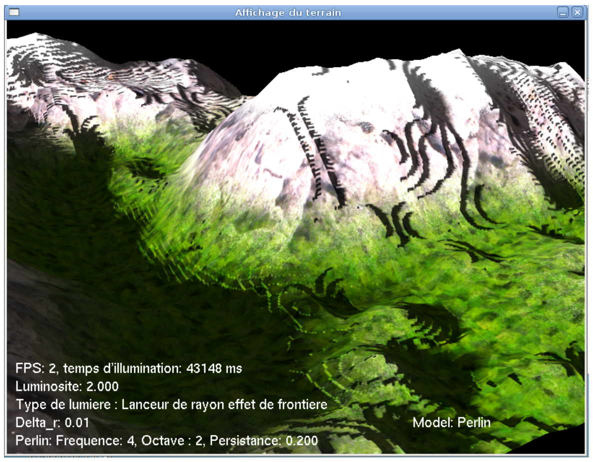
For rendering we have used a luminosity of 2.0 with a terrain of Perlin with $Frequency = 4$, $Octave = 2$, $Persistence = 0.2$, for any method using of smoothing filter [2] is necessary to remove the imperfections.

Now, we will compare our method with the Raytracer method.



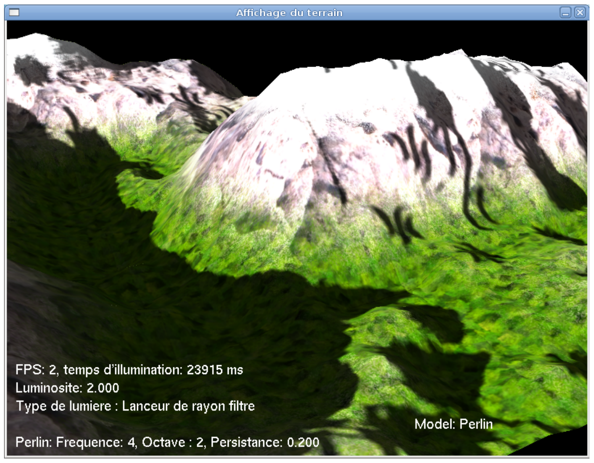
```
FPS: 2, temps d'illumination: 22118 ms
Luminosite: 2.000
Type de lumiere : Lanceur de rayon
                                              Model: Perlin
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200
```

Fig. 9: Result of ray tracer without smoothing filter.



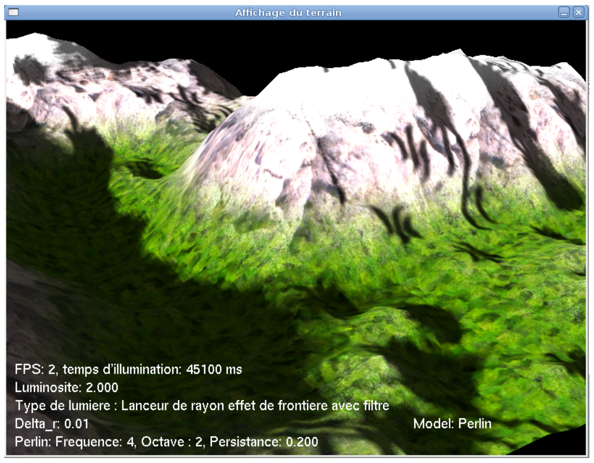
```
FPS: 2, temps d'illumination: 43148 ms
Luminosite: 2.000
Type de lumiere : Lanceur de rayon effet de frontiere
Delta_r: 0.01                                 Model: Perlin
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200
```

Fig. 10: Result of our method without smoothing filter.



FPS: 2, temps d'illumination: 23915 ms
Luminosite: 2.000
Type de lumiere : Lanceur de rayon filtre
Model: Perlin
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200

Fig. 11: Result of ray tracer with smoothing filter.

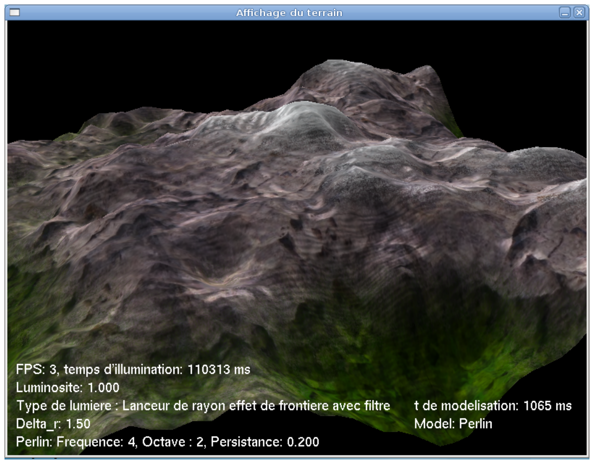


FPS: 2, temps d'illumination: 45100 ms
Luminosite: 2.000
Type de lumiere : Lanceur de rayon effet de frontiere avec filtre
Delta_r: 0.01
Model: Perlin
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200

Fig. 12: Result of our method with smoothing filter.

We note that the proposed method (Raytrace effect of borders) has an additional effect of shadows with soft edges, and is quick and simple than the technique based on the horizon map.

Now, we will begin the modeling of night light. Typically the light sources of night as the moon lights are very close than the sun, then their angular diameter is larger, and they have a low light intensity, so to model the light of the night just to increase the parameter Δr $(\Delta r = 1.5 here)$ in our method, and reduce the brightness to 1.
The Figure 13 is an image that illustrates the solution:



FPS: 3, temps d'illumination: 110313 ms
Luminosite: 1.000
Type de lumiere : Lanceur de rayon effet de frontiere avec filtre
t de modelisation: 1065 ms
Delta_r: 1.50
Model: Perlin
Perlin: Frequence: 4, Octave : 2, Persistance: 0.200

Fig. 13: Night light.

4. Conclusion

In this paper we have introduce a new method for calculating shadow for land, we have increased the realism without increasing the complexity of a considerable degree, not too many additional calculations except the visibility factor, and without need for additional and complex structures as the horizon map.

The proposed method is more realistic than the ray tracer method, and is quick and simple than the technique based on the horizon map.

The overall objective of the work is reached. However, some perspectives are beginning to be considered:

- Simulate more realistic effects such as interstitial reflections without using global models and increase complexity.
- To remove the imperfections. We have used a filter, then the solution obtained is not the real solution we can reduce the imperfections by increasing the density of the points by increasing the size of heightmap.

References

[1] André Lamothe, *Focus on 3D Terrain Programming,*, Premier Press, 2003.

[2] MEI X., DECAUDIN P., HU B, "Fast hydraulic erosion simulation and visualization on gpu,". in *Pacific Graphics*, 2007, pp. 47-56.

[3] N EIDHOLD B., WACKER M., DEUSSEN O, "Interactive physically based fluid and erosion simulation,". in *Eurographics Workshop on Natural Phenomena*, 2005, pp. 25-32.

[4] JENS SCHNEIDER T. BOLDTE R. W, "Real-time editing, synthesis, and rendering of infinite landscapes on gpus,". in *Conference on Vision, Modeling, and Visualization*, 2006, pp. 153- 160.

[5] A. Peytavie, E. Galin, J. Grosjean, S. Merillou , "Modélisation de terrains complexes 3D ," Toulouse, 2008.

[6] Y. Amara and X. Marsault, "A GPU Tile-Load-Map architecture for terrain rendering : theory and applications,". in *The Visual Computer*, paper 25.8, 2009, p. xx-yy.

[7] Philip Dutré, "Global Illumination Compendium,". *Program of Computer Graphics*, Cornell University, March 16, 2000.

[8] Naty Hoffman, Kenny Mitchell, "Real-Time Photorealistic Terrain Lighting,". *Westwood Studios*, 2003.

[9] N.L. Max, "Horizon Mapping: Shadows for Bump-Mapped Surfaces," in *The Visual Computer*, 1988, paper 4.2, p. 109-117.

SESSION

VIRTUAL REALITY

Chair(s)

TBA

The Effect of Global Illumination on Presence in a Virtual Environment for those with Autism Spectrum Disorder

Justin Ehrlich
School of Computer Sciences, Western Illinois University
Macomb, Illinois, US

Abstract - *Presence, or the feeling of being there, has been attributed as a driving force for the effectiveness of virtual environment (VE)-based interventions for treating those with Autism Spectrum Disorder (ASD). Therefore, this research sought to measure the extent to which visual realism can induce the presence of those with ASD within a VE intervention, enumerate the specific characteristics of rendering that promote the sense of presence and the ability to generalize, and statistically verify enhanced outcomes from using these techniques. A between-group study with 24 individuals with ASD, ranging between ages of 9-17, utilized a quantitative questionnaire. Illumination realism was found to have a positive effect on the presence felt by these individuals. This presence has been found to correlate with the ability to generalize. This work provides empirical evidence supporting the claim that illumination realism increases the presence felt by users with ASD when interacting with VE-based intervention.*

Keywords: virtual reality, autism, presence, illumination, global illumination

1 Background

There have been numerous studies [1-3] demonstrating the efficacy of using virtual reality to treat those with Autism Spectrum Disorder (ASD) along with numerous discussions of why these virtualized interventions are so effective. One suggested reason for the success of virtual reality (VR) is that as a user walks through a virtual environment, that user is forced to use their imagination to gain the perspective of their virtual avatar. This not only increases the ability to imagine, but also enhances the ability to take perspective outside of the virtual environment and into the real world [1]. Another suggestion [2] is that kids with ASD tend be visual learners and virtual environments (VE) offer learning in a visual manner. Still another suggestion [4] is that kids with ASD tend to enjoy computers and are most comfortable at interacting with an environment that they understand and can control.

While there have been numerous studies [1-3] demonstrating the effectiveness of VEs on those with ASD, there is a gap in knowledge of finding ways of improving the effectiveness of these interventions. It has been suggested [1] that one such improvement may be increasing the presence felt by a user. This is because a VE with a high level of presence appears real to the user, and any lessons learned within the virtual experience is easily connected to the real world. Further, there is evidence that an presence can increase the ability to generalize skills learned in VEs for not only neurologically typical (NT) individuals [29], but also for those with ASD [5], and because illumination realism has been found to affect presence, increasing illumination realism may enhance this generalizability. One reason why generalization is thought to work in VR for individuals with ASD is that VEs encourage symbolic generalization because the user must imagine the 3D scene and avatars to represent real life in order to successfully navigate and understand the environment [5].

Presence is dependent on both the realism of the graphics and immersive technology (place illusion) and the interaction of the entities in the scene (plausibility) while interacting with the user, yet improving any of these dimensions has proven problematic for those with ASD, as those on the spectrum are acutely aware of subtle differences between those that are real and virtual characters – and any differences notices, no matter how small, will likely break the presence felt. The uncanny-valley has an exaggerated effect on those with ASD, so improvements to both geometric realism (place illusion) and interaction tend to have negative effects preventing improvements to VEs. However, recent work within computer graphics has surfaced linking illumination realism to presence [5-7], which is a subtle effect that includes global illumination and, while important for presence being formed by the subconscious, usually goes unnoticed by the observer. The goal of adding realism to a VE should be to increase presence, and this presence is the best way of measuring the effectiveness of the added realism [8], although the uncanny valley effect [9] ironically causes an inversion to the correlation between presence and geometric realism, which is the art of incorporating high resolution models and textures to VEs. In essence the uncanny valley introduces a hurdle to improving presence through geometric realism, but Slater et al. [6] demonstrated that *illumination* realism increases presence in virtual environments (VE) for neurologically typical (NT) individuals. This illumination realism looks to be perfect solution for increasing presence in those with ASD, because this form of realism is not affected by the uncanny valley effect.

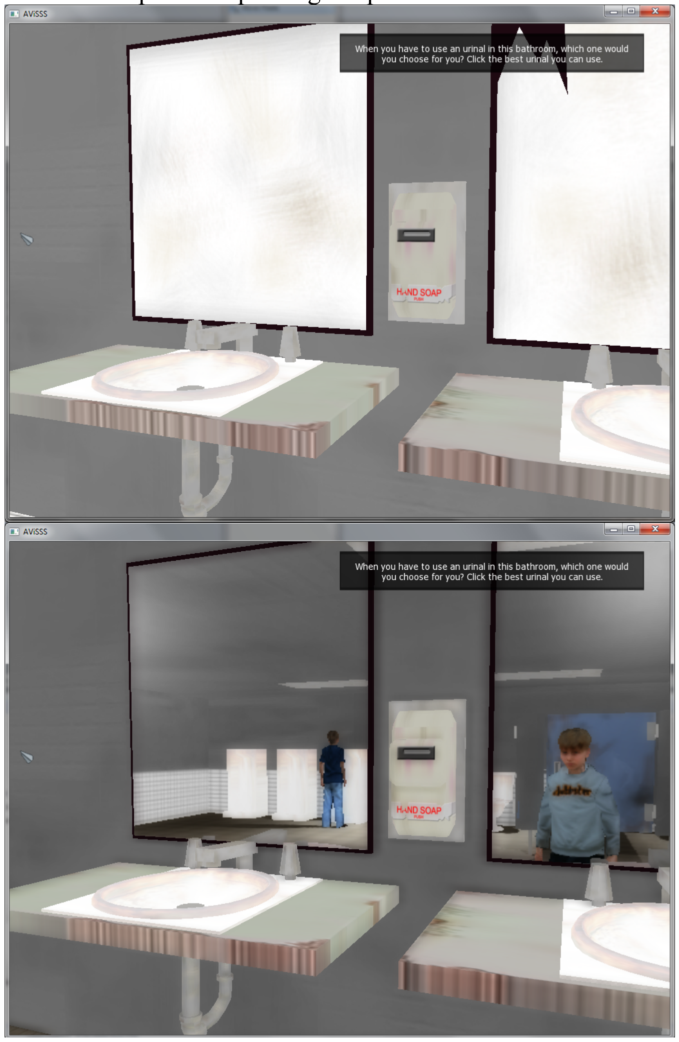
Thus, the purpose of the present study is to explore to what extent illumination realism affects the presence and the ability to generalize of those with ASD within a virtualized intervention. To accomplish this, the study measured the extent to which visual realism can induce the presence of those with ASD within a VE intervention, enumerated the specific characteristics of rendering that promote the sense of presence and the ability to generalize, and statistically verified enhanced outcomes from using these techniques.

Further, since head mounted displays (HMD) are intrusive to those with ASD and since there is a lack of research about VEs without HMDs, this study focused on the use of standard desktop displays. Up until now, there were gaps in the literature comparing illumination realism and user presence using standard displays, and methods for improving presence in VEs for individuals with ASD. This study will attempt to help fill these gaps and to improve VE interventions by increasing the community's understanding of illumination realism and induced presence. In order to improve illumination realism, I incorporated cutting edge research from the computer graphics community to extend our initial intervention to provide an enhanced version featuring real-time global illumination.

1.1 Global Effects

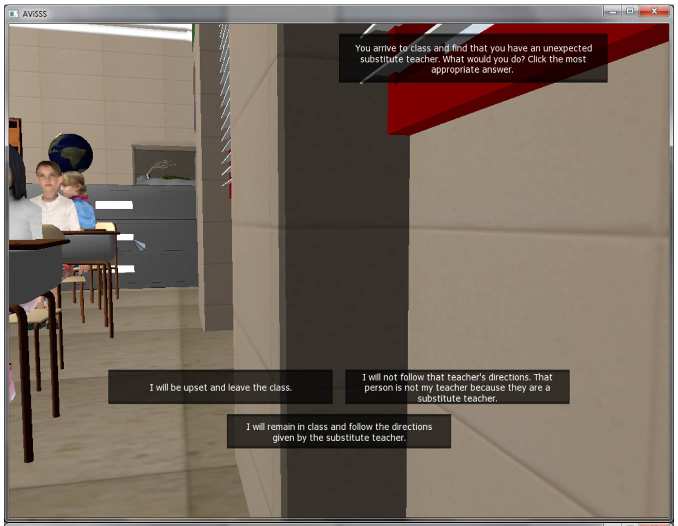
Illumination realism is the simulation of how light recursively bounces through the scene. To accurately simulate lighting, global lighting must be calculated, which is how light propagates across a scene, going from one surface to the next, yet this usually requires prohibitive processing power that is reserved for rendering farms. When light from a primary source hits a surface in that scene, the light is scattered out towards other surfaces. Simulating this light propagation is necessary for realistic images, but it requires extensive processing that up until recently was reserved in its full glory for offline renderers whose goal is to render an image realistically without the constraint of interactive frame rates. Offline renderers are usually used to produce visualizations and animations that do not need interactive controls. Offline renderers can take minutes, hours, or even days to complete depending on the accuracy desired. True global illumination remains elusive for general scenes rendered by interactive applications since each frame needs to be rendered in a fraction of a second, although methods are available for simulating this light in real-time.

Recently the graphics community has been abuzz introducing algorithms that are capable of real-time global illumination using shortcuts and approximations suitable for shaders on raster-based graphic pipelines. Three of these algorithms were used in this study to render mirror reflections, inter-surface reflections, and multi-sourced soft shadows. For the specular mirror reflections, two rendering passes rendered the floor and restroom mirror reflections; the first pass rendered the geometry reflected into the mirror onto a texture, which was used to texture the surfaces during the final pass. The shadows were rendered using the shadow map algorithm introduced by Williams [10], by rendering the shadows from the camera and using these shadow depth buffer to determine the surfaces occluded. Finally, Screen Space Direction Occlusion, an extension of Screen Space Ambient Occlusion (SSAO) [11, 12], as introduced by Ritchel, et al.[13], was used to render ambient occlusion and interreflections between the surfaces. These algorithms, shown in figures 1-4, were used to render global illumination within a virtualized intervention, which dramatically increased the illumination realism in hopes of improving the presence felt.


When you have to use an urinal in this bathroom, which one would you choose for you? Click the best urinal you can use.

When you have to use an urinal in this bathroom, which one would you choose for you? Click the best urinal you can use.

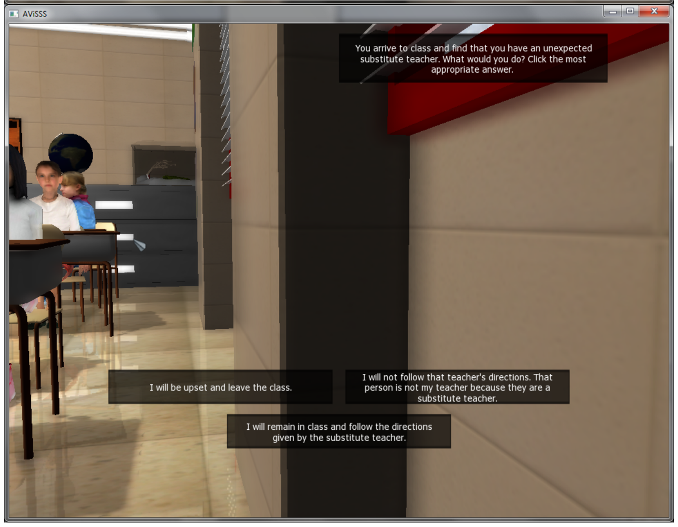
Figure 1-2: Comparisons of Local and Global AViSSS. The top figure shows the restroom with local lighting and the bottom shows the restroom with global lighting. Notice the accurate rendering of the mirrors and shadows in the global version of AViSSS.


You arrive to class and find that you have an unexpected substitute teacher. What would you do? Click the most appropriate answer.

I will be upset and leave the class.

I will not follow that teacher's directions. That person is not my teacher because they are a substitute teacher.

I will remain in class and follow the directions given by the substitute teacher.


You arrive to class and find that you have an unexpected substitute teacher. What would you do? Click the most appropriate answer.

I will be upset and leave the class.

I will not follow that teacher's directions. That person is not my teacher because they are a substitute teacher.

I will remain in class and follow the directions given by the substitute teacher.

Figures 3-4: Comparisons of Local and Global AViSSS. The top figure shows the classroom with local lighting and the bottom shows the classroom with global lighting. Notice the shadows, floor reflections, and SSAO in the global version of AViSSS.

2 AViSSS

In order to assess the effectiveness of presence of virtualized interventions, I repurposed and extended an existing intervention that I helped to develop for the University of Kansas' Dept of Special Education[14]. This intervention, named Animated Visual Supports for Social Skills (AViSSS), is intended to teach social and perspective-taking skills and is partially based upon the social narrative family of interventions. AViSSS simulates an average school day for a student with ASD by presenting various social scenarios with choices to be made within typical school environments. The rendering algorithm used in this application was developed to simulate elaborate global illumination (e.g., shadows and reflections). Such global illumination algorithms are well known to greatly increase illumination realism.

3 The Study

The population was randomly divided into two groups. The first group, which is henceforth called the global lighting (GL) group, used AViSSS with global illumination turned on. The second group, which is called the local lighting (LL) group, used AViSSS without the global illumination turned on. For the effect of illumination realism on presence, a questionnaire was administered during both groups' experience in their respective VE. The common practice of gathering reported presence is through a five-point Likert scale based upon the questionnaire introduced [15] and justified [8] by Slater et al. and verified by Usoh, et al. [16], yet during the pilot study I found that individuals with ASD have difficulty conceptualizing abstract analog scales (Likert item) and responses tend to be biased towards the extremes which, after a discussion with a number of colleagues, led me to use discrete multiple choice questions instead. While converting the spectrum questions into discrete multiple-choice questions, I carefully retained the nature of the questions, based on the recommendations from past studies by Mania et al. [17-20] and Slater et al. [6, 8, 15, 21]. To quantify the choice, each response was given a value from 0-3, with 0 defining no presence and 3 defining the most presence. The following is one such question from the hallway scenario:

What was your experience like in the hallway?

It felt like I was in the hallway.	*Value = 3*
It usually felt like I was in the hallway.	*Value = 2*
It usually did NOT feel like I was in the hallway.	*Value = 1*
It never felt like I was in the hallway.	*Value =0*

To find possible differences of generalization between the two groups, I measured any differences in effectiveness that global illumination caused in AViSSS. To determine this effectiveness, a graphical driver application (DA), which I developed for this study, played twenty different videos to both groups before exposure to AViSSS. Each video related to a different social situation that AViSSS was attempting to address. The DA enquired of the users what they would do in each video situation. For instance, the first video set might contain a video of a cafeteria scene where the user must choose a seat. The DA then subjected the user to AViSSS, which included a scene that guided the user in choosing a seat in the cafeteria. After the intervention, the users viewed the twenty videos again, along with six different videos of situations not addressed within AViSSS. The second part was a scene based in the school library, which was never addressed by AViSSS. This demonstrated the effectiveness of AViSSS at teaching the generalization of the targeted skills. The DA again asked the user to make a choice for each situation, which the responses were recorded.

3.1 Study Environment

We recruited special education teachers to administer AViSSS to their own students. This allowed the student to remain in a familiar environment while interacting with a known and trusted individual. An initial questionnaire was sent to interested teachers with questions pertaining to their special education knowledge, computer accessibility, type and speed of computer available, and number of students. I used this information to select the subject population. Once I selected the teachers to administer the test, I took a computer loaded with AViSSS to the teacher's classroom and trained the teacher how to conduct the test.

The teacher then invited his or her students with ASD, whom received parental permission, to interact with AViSSS. The DA guided the students through the program and the questionnaire; the teacher did not need to intervene during the test. At a few schools, I conducted the study myself to accommodate the teachers' busy schedules. The students' testing times were staggered to fit the teacher's schedule, e.g. one student a week.

4 Data

The DA collected the data on each test machine as the students were experiencing AViSSS. The independent variable is the group membership, which is one of two groups: local lighting and global lighting. The dependent variables consist of presence, number correct, and generalizability of which presence is accumulated in table 1, analyzed in table 2, and the correlation between other variables is presented in table 3 (the rest of the data is presented within my dissertation [22]). The presence value is an average of the results from the questionnaire given to each subject. Each response was ranked from zero to three, with zero representing the least felt presence and three representing the most. The number correct variable is the number of correct responses on the posttest, excluding the generalized questions. The generalizability is the number of correct generalized questions. There were six generalizability questions asked of each subject.

Looking at table 1, the average reported presence of those using AViSSS with local lighting reported 1.406 while those experiencing global illumination reported an average of 1.885, a 34% increase. While the p-value (table 2) of the study's average user response is 0.175, the effect size is 0.6, which is large enough to show a medium difference between the two groups. Some questions were more telling than others, especially questions 4 and 5 with question 4 receiving a large effect size of 1.4268 (p-value: 0.003) and with question 5 receiving an effect size of 0.7058 (p-value: 0.113). The mean difference for question four is 1.41667, with the average of the local lighting group reporting 0.75, and the average of the global lighting group reporting 2.1667, an increase of 189%. The mean difference for question five is 0.8333, with the average of the local lighting group reporting 1.0, and the average of the global lighting group reporting 1.8333, an increase of 83%. When creating questionnaires gathering user reported presence, it is important to choose questions that differentiate the population, for individual interpretations of the questions and responses to the questions are subjective and can vary widely [19], especially those with ASD.

The subject extraneous variables include school, gender, age, and spectrum placement. The age groups were on a scale as follows: 1=9-11 years; 2=12-14 years; 3=15-17 years and the mean for both the groups were 13.09 years. The control group's mean age was 13.36 years while the AViSSS group's mean age was 12.82 years. The functionality spectrum placement was measured by the following scale: 1=low, 2=medium low, 3=medium high, 4=high, with the mean falling between 2 and 3. Consistent with the current rate of gender differences among those diagnosed with ASD, the CG had two females while the AG had one, where male=1 and female=2. To remove any bias that the extraneous variables may have on the dependent variables, the extraneous variables must be balanced between the two groups, which they are. An analysis of the extraneous variables demonstrated no significant differences between the two groups for school (p=0.854), gender (p=0.557), age (p=0.842), and spectrum placement (p=0.603).

The correlation between presence and the other dependent variables are shown in table 3. Reported presence was found to have a correlation of 0.427 with the number of generalized questions correct (p-value: 0.037). Similarly, there is a correlation of 0.323 between presence and the number of posttest questions (p-value: 0.124) found to be correct, which excludes the generalized questions.

Group Statistics

	Group	N	Mean	Std. Deviation	Std. Error Mean
Average Presence	Local lighting	12	1.40625	.803269	.231884
	Global lighting	12	1.88542	.870048	.251161

Table 1: Average Reported Presence Between the Two Groups

Independent Samples Test

	t-test for Equality of Means				
	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference
Equal variances assumed	-1.402	22	.175	-.479167	.341836
Equal variances not assumed	-1.402	21.861	.175	-.479167	.341836

Table 2: Independent Samples Test of Reported Presence Between the Two Groups

Presence Correlations					
	Functioning	Age	Generalizability	Num Correct	ASD Placement
Pearson Correlation	.340	-.189	.427*	.323	-.080
Sig. (2-tailed)	.104	.376	.037	.124	.711
N	24	24	24	24	24

Table 3: Correlation Between Presence and the Other Dependent Variables

5 Discussion

The results demonstrate not only a positive effect of global illumination on reported presence, but also a correlation between reported presence and correct generalized questions. While the p-value of the study's average user response is high (0.175), the effect size is 0.6, which is large enough to show a medium difference between the two groups. Since the power analysis recommended 21 subjects in each group and since this study could ultimately only provide 12 subjects in each group due to the teachers' time constraints, the high p-value is probably due to the small user subject size causing a low confidence level. Another reason for the high p-value may be a ceiling in the questions for individuals placed in a higher region on the autism spectrum, which could explain the negative correlation between ASD placement and learning. However, because the effect size is medium, a conclusion can still be reached, which is that there is evidence for a positive effect of illumination realism on presence felt for those with ASD. Further, looking at the questions individually shows that the two questions discussed above differentiate the group significantly, suggesting these questions were the most telling. Based upon the evidence provided by this study, illumination realism has been found to enhance user presence.

Additionally, reported presence was found to have a positive correlation with both the number of targeted skill based questions and generalized questions answered correctly. This supports the original claim, which is the motivation for this study, that presence increases the effectiveness of interventions at teaching and generalizing social skills. Therefore, this study provides evidence that increasing illumination realism increases the ability to learn and generalize social skills targeted by a virtual intervention.

6 Conclusion

This study found that increasing illumination realism has a positive effect on reported presence based on a between-group study with 24 subjects with autism spectrum disorder (ASD) assigned randomly to two groups. Increasing the illumination realism caused reported presence to increase an average of 34%, which demonstrates a trend that increasing illumination realism increases presence in students with ASD. This is significant because these individuals typically lack the ability to imagine and experience presence. It is thought that an increased presence enhances the generalization in VEs for those with ASD [5], and because illumination realism has been found to affect presence, the ability to generalize skills can be enhanced by increasing illumination realism. This study's analysis supports this conclusion since a statistically significant positive correlation was found between presence and generalization.

This study has shown that an intervention with increased illumination realism increases presence, which enhances the effectiveness of the intervention by enhancing the user's ability to generalize. This finding is of major importance not only in the field of special education, but also in the field of computer graphics. By providing evidence that illumination realism increases presence, researchers creating VE interventions can now know that, even on a standard desktop display, increasing illumination realism may improve presence, which in turn enhances the user's ability to learn and generalize targeted skills. If utilized, this will increase the effectiveness of future interventions, including AViSSS. The full extent of this work can be found in Ehrlich's dissertation [22].

7 References

[1] G. Herrera, *et al.*, "Development of symbolic play through the use of virtual reality tools in children with autistic spectrum disorders: Two case studies," *Autism: The International Journal of Research & Practice,* vol. 12, pp. 143-157, 2008.

[2] D. Strickland and L. M. Marcus, "Brief report: Two case studies using virtual reality as a learning tool for autistic children," *Journal of Autism & Developmental Disorders,* vol. 26, pp. 651-659, 1996.

[3] P. Mitchell, *et al.*, "Using Virtual Environments for Teaching Social Understanding to 6 Adolescents with Autistic Spectrum Disorders," *Journal of Autism & Developmental Disorders,* vol. 37, pp. 589-600, 2007.

[4] K. S. Exkorn, *The autism sourcebook : everything you need to know about diagnosis, treatment, coping, and healing*. New York: ReaganBooks, 2005.

[5] M. Slater, "Place illusion and plausibility can lead to realistic behaviour in immersive virtual environments," *Philosophical Transactions B,* 2009.

[6] M. Slater, *et al.*, "Visual Realism Enhances Realistic Response in an Immersive Virtual Environment," *IEEE Computer Graphics and Applications,* vol. 29, pp. 76-84, May/June 2009.

[7] M. Slater, *et al.*, "How we experience immersive virtual environments: the concept of presence and its measurement," ed: Universitat de Barcelona, 2009.

[8] M. Slater, "Measuring Presence: A Response to the Witmer and Singer Presence Questionnaire,"

Presence: Teleoper. Virtual Environ., vol. 8, pp. 560-565, 1999.

[9] M. Mori, " Bukimi no tani The uncanny valley," *Energy,* vol. 7, pp. 33–35, 1970.

[10] L. Williams, "Casting curved shadows on curved surfaces," *SIGGRAPH Comput. Graph.,* vol. 12, pp. 270-274, 1978.

[11] M. Mittring, "Finding next gen: CryEngine 2," presented at the ACM SIGGRAPH 2007 courses, San Diego, California, 2007.

[12] P. Shanmugam and O. Arikan, "Hardware accelerated ambient occlusion techniques on GPUs," presented at the Proceedings of the 2007 symposium on Interactive 3D graphics and games, Seattle, Washington, 2007.

[13] T. Ritschel*, et al.*, "Approximating dynamic global illumination in image space," presented at the Proceedings of the 2009 symposium on Interactive 3D graphics and games, Boston, Massachusetts, 2009.

[14] J. A. Ehrlich and J. R. Miller, "A Virtual Environment for Teaching Social Skills: AViSSS," *IEEE Comput. Graph. Appl.,* vol. 29, pp. 10-16, 2009.

[15] M. Slater*, et al.*, "The influence of body movement on subjective presence in virtual environments.(Special Section: Virtual Environments: Models, Methodology, and Empirical Studies)," *Human Factors,* vol. 40, p. 469(9), 1998.

[16] M. Usoh*, et al.*, "Using Presence Questionnaires in Reality," *Presence: Teleoperators & Virtual Environments,* vol. 9, pp. 497-503, 2000.

[17] K. Mania, "Connections between lighting impressions and presence in real and virtual environments: an experimental study," presented at the Proceedings of the 1st international conference on Computer graphics, virtual reality and visualisation, Camps Bay, Cape Town, South Africa, 2001.

[18] K. Mania and A. Chalmers, "The Effects of Levels of Immersion on Memory and Presence in Virtual Environments: A Reality Centered Approach," *CyberPsychology & Behavior,* vol. 4, pp. 247-264, 2001.

[19] K. Mania and A. Robinson, "The effect of quality of rendering on user lighting impressions and presence in virtual environments," presented at the Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry, Singapore, 2004.

[20] K. Mania*, et al.*, "The Effect of Visual and Interaction Fidelity on Spatial Cognition in Immersive Virtual Environments," *IEEE Transactions on Visualization and Computer Graphics,* vol. 12, pp. 396-404, 2006.

[21] M. Slater*, et al.*, "The influence of dynamic shadows on presence in immersive virtual environments," presented at the Selected papers of the Eurographics workshops on Virtual environments '95, Barcelona, Spain, 1995.

[22] J. A. Ehrlich, "The effect of desktop illumination realism on a user's sense of presence in a virtual learning environment," Ph.D. dissertation, University of Kansas, Lawrence, KS, 2010.

A Training Program of Psychopathological Exploration of Somatoform Disorders Based on Virtual Reality and Artificial Intelligence

Jose Gutierrez-Maldonado, Angel Aguilar, Marta Ferrer, Claudia Peñaloza
University of Barcelona, Spain

Abstract - *A training program of differential diagnosis skills was developed to enhance the learning of the psychopathological exploration of Somatoform Disorders using Virtual Reality (VR) and Artificial Intelligence (AI) based simulations. The sample of the study consisted of 89 psychology students. Comparisons between the experimental (VR and AI training) and the control group (role-playing training) showed that students trained with the VR-AI system obtained better scores than students trained with a traditional method. These findings suggest that the acquisition of psychopathological exploration skills can be improved by utilizing alternative learning methods that consider interactive and meaningful experiences that allow for recreation of real situations.*

Keywords: diagnostic interview, psychology, psychopathological exploration, somatoform disorders, virtual reality, artificial intelligence

1. Introduction

A virtual environment allows students to change their point of view by observing a phenomenon from different perspectives, enhancing the recall of objects and their localization [1,2,3]. Similarly, students can play different roles in a social virtual environment which enhances their social skills. Hence, virtual reality can be used to perceive and to experience a situation in different manners.

The way individuals perceive reality is determined by their biological characteristics and the conditioning experiences previously acquired. The more a virtual environment adjusts to previous experiences of a student the more meaningful it will become and as a consequence, interaction with the virtual environment will also produce a change in experience. Virtual reality offers a number of possibilities to see the world from different points of view, to break cognitive and social barriers that limit perception of reality, to evolve its mental representation and to improve more adapted behaviours.

Many studies found that immersion enhances learning [1] while others state that when not followed by an increase of presence (sensation of being in a virtual environment and not in the laboratory or the classroom where the experience takes place) might be ineffective [4]. Physical immersion may only require a student to carry an HMD (Head Mounted Display) and move in a virtual environment. However, in order to achieve a high level of psychical immersion (presence) it is also necessary that the environment is provided of personal meaning for the student interacting with it. There is evidence on the positive correlation between presence and learning [4] and there are no available reports on presence as an interfering element in learning.

The main purpose of this project was to develop a new method to improve the acquisition of psychopathological exploration skills, in psychology students, for the diagnosis of somatoform disorders, by developing a computerized resource based on VR and AI.

Somatoform Disorders include those where the symptoms suggest a medical condition but where no medical condition can be found by a physician. A person with a somatoform disorder might experience significant pain without a medical or biological cause, or they may constantly experience minor aches and pains without any reason for these pains to exist. The disorders included in this category are: Body Dysmorphic Disorder; Conversion Disorder; Hypochondriasis Disorder; Pain Disorder; and Somatization Disorder. Patients with this disorders often become worried about their health because the doctors are unable to find a cause for their health problems. Symptoms are sometimes similar to those of other illnesses and may last for several years. Somatoform disorders are not the result of conscious malingering (fabricating or exaggerating symptoms for secondary motives) or factitious disorders (deliberately producing, feigning, or exaggerating symptoms) - sufferers perceive their plight as real. The exploration of these symptoms in order to develop an appropiate diagnosis requires the learning of assessment skills in psychologists and psychiatrists.

The study consisted on measuring the effects of a training program based on virtual reality and artificial intelligence in the psychopathological exploration skills of psychology students who had to perform diagnostic explorations of simulated patients with Somatoform Disorders.

2. Methods

2.1. Sample

The sample consisted of pre and post graduate students from the University of Barcelona. Voluntary agreement to participate of the study was required. The final sample of the study consisted of 89 students.

2.2. Instruments

A software based on virtual reality and artificial intelligence, *Simulated Interviews*, was developed to enhance skills learning. Virtual environments were developed with 3D studio. Agents were modelled with Poser and Character Studio. Interaction and navigation were programmed with Virtools Dev. Voices were directly recorded from actors. AIML (Artificial Intelligence Mark-up Language) was used to elaborate the knowledge base. The basic unit of knowledge in AIML is called a category. Each category consists of an input question, an output answer, and an optional context. The question, or stimulus, is called the pattern. The answer, or response, is called the template. The two types of optional context are called "that" and "topic." The AIML pattern language is simple, consisting only of words, spaces, and the wildcard symbols _ and *. The words may consist of letters and numerals, but no other characters. The pattern language is case invariant. Words are separated by a single space, and the wildcard characters function like words. AIML is not exactly the same as a simple database of questions and answers. The pattern matching "query" language is much simpler than something like SQL. But a category template may contain the recursive <srai> tag, so that the output depends not only on one matched category, but also any others recursively reached through <srai>.

A cycle of supervised learning was required to improve the initial knowledge base. The program was applied to a group of 300 students of psychology and other health related sciences. Entrances suggested by students that did not match an answer in the system were registered and classified as targets. Targets were subsequently revised by professors in order to select new matches that could be entered to the knowledge base to offer correct answers.

The knowledge base developed for the AI system and the multiple choice question system resulted from a data base matching question classes and answer classes according to the DSM-IV hierarchical system of differential diagnosis (Diagnostic and Statistical Manual of Mental Disorders) [5] for the diagnostic group considered for the training (Somatoform Disorders).

A series of flowcharts were initially designed in which operators referred to each phase of the diagnostic exploration and vectors were associated to the answers that allowed a participant to move forward on the process of differential diagnosis. It was necessary to have a large number of alternative questions in each operator so that common contents could be extracted from a series of target words. Vectors should also contain a large number of answers so that the system would not get interfered while interpreting the natural language.

The development of a graphic engine and an AI engine was required so that the simulated interviews could be held in any local computer and within a virtual reality environment. Both were also required to be operated through internet. Thus, the program could be distributed in DVD or CD-ROM format. They include 3D models with realistic textures and illumination, and avatars that play the role of virtual patients whose facial expression matches the verbal contents according to the psychopathology simulated (Somatoform Disorders).

Finally a diagnostic interview skills test was also used to evaluate the psychopathological exploration and differential diagnostic skills acquired. The final score was calculated taking into account the correct answers converted on a 10 point- scale.

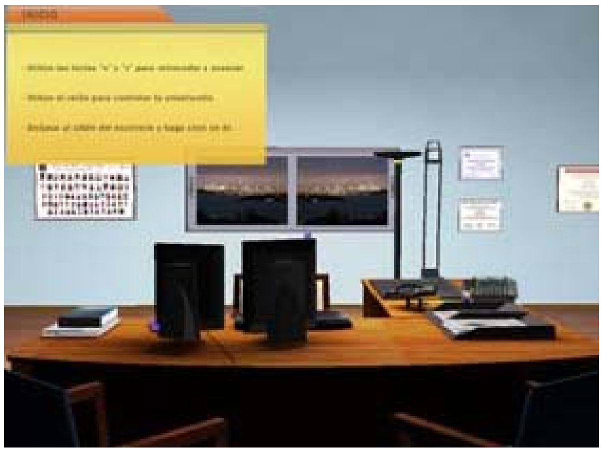


Figure 1. Virtual environment.

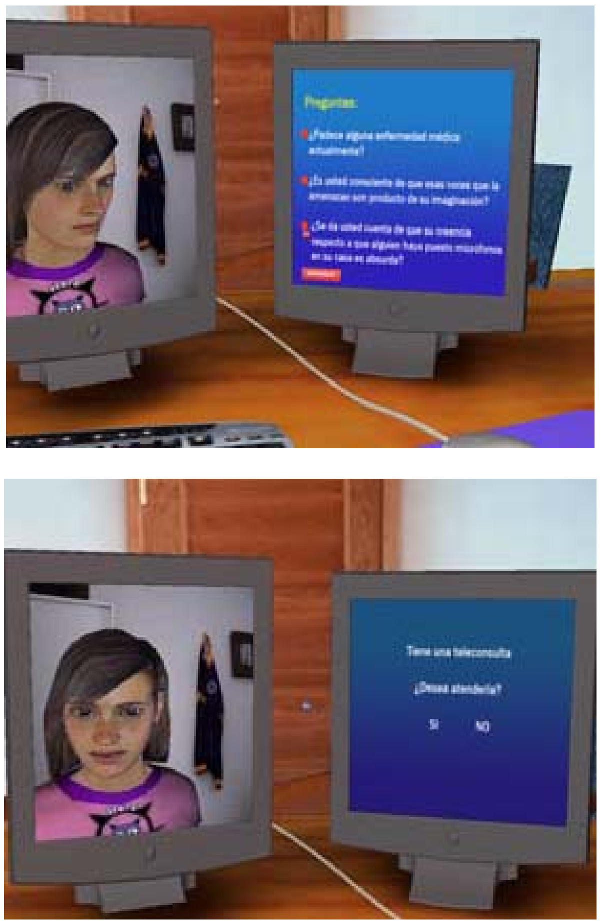


Figure 2. Virtual interview: The patient appears on the left screen while the answer choices and the diagnosis hypothesis are displayed on the right screen.

2.3. Procedure

Participants were randomly assigned to one of the following conditions: 45 students who received psychopathological exploration skills training using the simulated interviews (experimental group) and 44 students who received psychopathological exploration skills training using the traditional method of role playing (control group).Three professors were available for each group.

Students in the experimental group were requested to assist to the laboratory for four consecutive sessions of 30 minutes with a five minute pause. Every student received from the professor in charge a basic explanation of the main characteristics of the Somatoform Disorders. Later, students had to interact with the virtual simulations of four patients who displayed the disorders (Body Dysmorphic Disorder; Conversion Disorder; Hypochondriasis Disorder; Pain Disorder; and Somatization Disorder).

The same procedure was applied to the students in the control group, however, instead of interacting with the simulated interviews training program they received a traditional training based on role playing in which the professor in charge played the role of the patient and every student had to perform an interview to identify the correct diagnosis. Finally, it was necessary to evaluate the effects of the training program on the students learning compared to the traditional teaching method, thus, students in both groups were required to do the diagnostic interview skills test.

3. Results

First, the homogeneity in both groups was examined, by comparing their distributions in gender and age. Similar values were found on both variables (See tables 1 and 2).

Table 1. Gender of participants in each group

			Gender		
			Man	Woman	Total
	Simulated Interviews	Count	17	28	45
		% of Total	19,1%	31,5%	50,6%
	Role Playing	Count	15	29	44
		% of Total	16,9%	32,6%	49,4%
Total		Count	32	57	89
		% of Total	36,0%	64,0%	100,0%

The chi-square test demonstrated the distribution among men and women between groups was not significantly different (0,13; p= 0,71). The t-test did not find age differences between groups (0,48; p= 0,63).

Table 2. Age of participants in each group

	Group	N	Mean	Std. Deviation
Age	Simulated Interviews	45	22,2667	4,57960
	Role Playing	44	22,7500	4,83267

A t-test for independent samples was conducted to evaluate the differences between the scores obtained by students in the experimental and control group in the diagnostic interview skills test. Students who were trained with the simulated interviews obtained better scores than the students trained with the traditional method of role-playing (t=3,25, p= 0,002) (See table 3).

Table 3. Scores in the diagnostic interview skills test

	Group	N	Mean	Std. Deviation
Test	Simulated Interviews	45	7,6444	1,76011
	Role Playing	44	5,2045	2,09739

An analysis of variance (ANOVA) was conducted to determine if gender modified the performance of students on the simulated interviews.

The interaction between the variables group and gender did not reach statistical significance (F = 0,14; d.f.=1 p = 0,7). Thus, training based on virtual simulation has the same effect on the performance of male and female students (see figure 3).

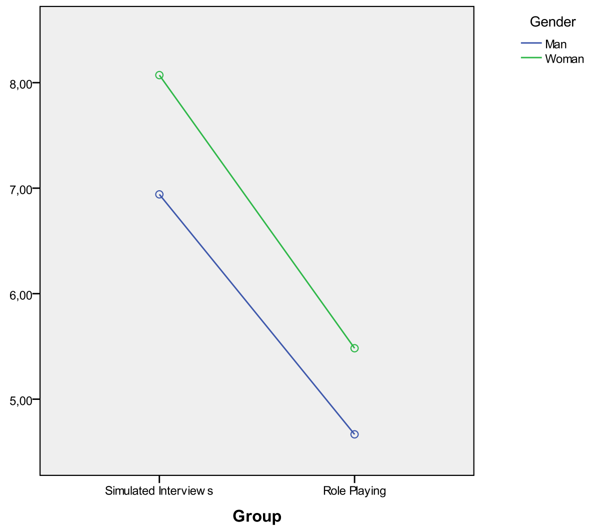


Figure 3. Interaction between group and gender

4. Discussion

The group that received the training program by means of simulated interviews reached higher scores in the diagnostic interview skills test than the group that received traditional training based on role playing. This difference was statistically significant. Given that both groups were homogenous in terms of age and gender, the differences found in the final scores obtained in the diagnostic interview skills test can be attributed to the different training programs applied to the experimental and control group.

The results of the analysis of variance (ANOVA) performed allows to conclude that the effect of learning enhanced by the simulated interviews is effective for both genders. According to the existing literature on this issue, it could be expectable to find a larger enhancing effect on male participants since they could have a stronger motivation towards this type of learning resources based on computer mediated communication, whereas female participants seem to prefer real face to face interaction. Previous studies suggest that female participants tend to feel more awkward than men when non verbal signals of communication are lost [6]. Face to face communication is rich in this type of signals that seem to get easily lost while interacting with a computer. The trend to use phrases that clarify the intentions underlying verbal expressions to compensate this lack of non-verbal signals is more commonly found in women than in men [6].

The simulation of emotions in our virtual patients, by means of facil expressions, can explain the finding of a similar enhancing effect on the learning of male and female participants.

Although no differences were found between men and women in our study, more research has to be done in this topic in the future. One possible limitation of our study is the different number of men and women in the sample. This is a typical situation in Psychology schools. A more balanced sample in relation with gender should be used in future studies in order to replicate the results obtained here.

Enhancing the learning of differential diagnosis and psychopathological exploration skills in general is important for psychology students who wish to work in the clinical field. The developed system may result promising when adapted to other fields of psychology, education, or human resources, in which communication skills play an important role.

5. References

[1] Arthur, E.J., Hancock, P.A. & Chrysler, S.T. (1997) The perception of spatial layout in real and virtual worlds. *Ergonomics 40*:69–77

[2] Dalgarno, B., Bennett, S. & Harper, B. (2010). The Importance of Active Exploration, Optical Flow, and Task Alignment for Spatial Learning in Desktop 3D Environments. *Human–Computer Interaction, 25(1),* 25-66. doi:10.1080/07370020903586670

[3] Lee, K. M., & Jung, Y. (2005). Evolutionary nature of virtual experience. *Journal of Cultural and Evolutionary Psychology, 3,* 159-178.

[4] Winn, W.D., Windschitl, M., Fruland, R. & Lee, Y.L. (2002). When does immersion in a virtual environment help students construct understanding? In *Proceedings of International Conference of the Learning Societies*, Seattle, WA.

[5] American Psychiatric Association (1994). *Diagnostic and Statistical Manual of Mental Disorders.* Masson

[6] Gutiérrez, J. Mora, M., Edipo, P y García, S. (2001): Personalidad, sexo y comunicación mediada por ordenador. *Anuario de Psicología, 32 (2)*, 51-63

Implementing Stereo Vision of GPU-Accelerated Scientific Simulations using Commodity Hardware

T.S. Lyes and K.A. Hawick
Computer Science, Institute for Information and Mathematical Sciences,
Massey University, North Shore 102-904, Auckland, New Zealand
email: { t.s.lyes, k.a.hawick }@massey.ac.nz
Tel: +64 9 414 0800 Fax: +64 9 441 8181

ABSTRACT
Stereo vision technology is becoming more and more commonplace in the movie and gaming industries. It has applications in many other fields as well, one of these is viewing scientific data. We develop a stereo vision system using commodity priced hardware and portable graphics software. Hardware and software details are described, as well as some resulting visualisations and performance issues. C++ and OpenGL are employed to create the stereo visualisation, using Nvidia 3D glasses and a professional GPU graphics card and driver. Key code fragments are presented, and we discuss some of the difficulties in setting up the stereo vision for scientific use. We also present some ideas for future development of scientific visualisation of voxel data in stereo.

KEY WORDS
stereo vision; OpenGL; Nvidia; Quadro; quad-buffering; simulation; interaction.

1 Introduction

Volume rendering[1, 2, 3, 4, 5] of scientific data sets is a challenging problem, particularly for interactive simulation. It is computationally expensive but very valuable to be able to interact directly with a simulation program to steer it through a non-trivial parameter space but also to detect and analyse subtle spatial features that emerge - particularly in complex systems simulations.

A number of sophisticated visualisation algorithms and techniques[6, 7] involving cut-away surface identification[8] and shading[9, 10] have been developed bit it is also possible to support various user-interactive ways of navigating through a solid model. Expensive state-of-the art interactive environments have now been feasible for immersive visualisation for some years[11] , but only very recently has it been economically feasible to build a stereo visualisation system from commodity priced desktop components and portable graphics software.

Figure 1: The authors using the stereo visualisation system with a test data set.

A simple program – known as “cubes” [12] – was developed in OpenGL[13] and C++ and allows a user to load in two- or three dimensional hyperbrick files and to render them in a number of simple ways, with options to transform, shift, rotate the data set, cut-away sections of it to see inside, selectively render only some of the voxel values present, and to produce various output graphics formats including raw portable pixel maps as well as input files suitable for use in more sophisticated rendering tools such as vtk[14] or povray[15].

This present paper describes how we added stereoscopic 3d support to the ’Cubes’ program to further enhance visualisation of the hyperbrick data sets. Stereoscopic vision is already becoming prominent in the video gaming and movie industries, however it has also been applied in areas such as vehicle detection [16], medical practices [17] [18], and manufacturing [19], and it would be interesting if the technique could be used to better visualise and understand scientific data sets. For example, a scientist could be ”inside” the data set adjusting the parameters, gaining insights immediately on what is happening as the data changes.

While only fairly recently (the last 5 years or so) gaining mainstream success in the arts[20], and in the video gaming and movie industries, the stereoscopic vision problem has been around for a far greater amount of time. In 1838, Charles Wheatstone [21] realized that each eye viewed an object from slightly different positions; the images projected onto each eye differ in the horizontal position, thus giving the illusion of depth (this is known as horizontal, retinal or binocular disparity). Da Vinci[22] also recognized this phenomenon and stated this as a reason why painters would never be able to realistically portray depth on a single canvas. More recently, stereoscopic vision was used in conjunction with photography to produce stereograms (stereo images seen through a stereoscope), and later "autostereograms" such as the popular Magic Eye pictures.

The original 'Cubes' was written in the C/C++ programming language using the OpenGL and GLUT libraries, and it therefore made sense to code the stereoscopic 3d programming in the same language using the same graphics API. OpenGL has built-in stereo functionality which can be used well if the required hardware can be found. This paper will firstly describe the sort of voxel data sets that we are attempting to visualise (Section 2) and in Section 3 we describe the hardware and software necessary to view the program in stereo, as well as giving some key code fragments. In Section 4 we present some results in the form of screenshots and in Section 5 discuss the perceived performance. Finally we offer some conclusions and areas for further research in Section 6. work in the

2 Volume Datasets

For simplicity we employ an easily coded and portable file format for passing voxel data between simulation programs and our stereo visualisation system. The hyperbrick file format[12] – with file ending ".hbrk" – was inspired by the incredibly useful portable pixmap format family (often known as "NetPBM") designed by Poskanzer and developed by Henderson[23]. The ppm and pbm formats have been available to programmers for over two decades and their value is largely due to their simplicity. One can off the top of one's head code up C/C++/Java to generate, or read and write these formats. The "H1" hyperbrick format is a generalisation of the pgm 2D greymap image file format, for the case of 3D (and in principle even higher dimensional) data.

Figure 2 shows the .hbrk file format, consisting of a two character textual header "H1" followed by a newline and an optional series of comment lines starting with a hash character. The subsequent integer – in this case a

```
H1
# a comment or header line
1
3 64 64 32
hyper-raster-of-raw-unsigned-chars
```

Figure 2: The .hbrk hyperbrick file format for a 3-d data set of unsigned chars with spatial extent $x = 64 \times y = 64 \times z = 32$, where the x-coordinate changes fastest, and z slowest.

size of "1" denotes the number of bytes in each payload entity. If one wanted voxels to be allowed to take on $2^2 4$ different levels - like portable pixmap pixels, then one could use a size of "3" to denote three bytes per voxel. The next line gives the dimensionality d of the hyperbrick – usually $d = 3$ for examples discussed in this document, followed by exactly d integer edge lengths in order of increasing significance – so in the example shown $L_x = 64, L_y = 64, L_z = 32$. This line is terminated by a newline and the remainder of the ".hbrk" file is a set of binary characters in the "hyper-raster" order implied by the dimensionality and lengths. So in the example thei_x index would move fastest, the i_y next fastest and so forth.

3 Stereo Rendering

In order to run a graphics simulation in stereoscopic 3d in OpenGL the computer needs to be set up using the right hardware - specifically a graphics card able to support OpenGL Stereo. Standard consumer graphics cards (such as Nvidia GeForce) do not support this. Nvidia Quadro cards are professional versions of the Nvidia GeForce cards and contain additional support for various professional software packages such as Maya and 3ds Max. It is puzzling why open source software such as OpenGL Stereo are included in this package and do not just come standard in all graphics cards. Hopefully this will change in the future. For our program, Quadro cards need to also come with quad-buffering support. Quad-buffering allows the OpenGL program to render and swap buffers for each eye (back and front buffers for right and left sides makes four buffers in total). This allows for a smooth picture in each eye and each left or right set of buffers can only be seen through the corresponding eye when seen through the glasses. Unfortunately only a select few Quadro cards support quad-buffering (specifically the Quadro FX range, such as the Quadro FX 5800, FX 3700 and the older FX 380).

The glasses used are the standard Nvidia 3d Vision shutter glasses such as those featured in Figure 3 (left).

Figure 3: NVidia stereo 3D glasses and Infra Red transmitter which synchronizes the glasses' shutter rate with the monitor's frame rate

The glasses lenses are able to flicker on and off very quickly (undetectable by the human eye) and an infrared transmitter Figure 3(Right) is used to synchronize the glasses with the frame rate of the running application. Also needed to view the stereo program smoothly is a monitor with high refresh rate capabilities, as the glasses shuttering each image separately for each eye will effectively halve the frame rate of the program that would be otherwise be running in non-stereo mode. A monitor with a refresh rate of at least 120 hertz (typical monitors display at only 60 hertz) is good enough for a nice, smooth display (we used the Samsung 2233 RZ model). If the monitor does not have the required refresh rate the images will seem choppy and flicker constantly.

Nvidia 3d Vision is currently only compatible with Windows and (more recently) Linux operating systems. While ideally we would have liked to make the program on a Linux machine, the hardware required to run it on a Linux machine was very expensive so we chose to run it on Windows - specifically Windows 7 32 bit, however it can work on all versions of Windows 7, Vista and XP. We used an Nvidia Quadro FX 380 graphics card with driver version 266.45. OpenGL version 3.7 or higher is needed, as OpenGL Stereo requires OpenGL Game Mode to work, which was introduced in the 3.7 update.

Figure 4: The Nvidia Quadro FX 380 graphics card

We employed the Nvidia Quadro FX 380 graphics card (as shown in figure 4). This device is capable of managing two screens although for the purposes of the work reported here we only employed a single 120Hz screen. Nvidia manufacture a range of more powerful Quadro cards, but this was the most affordable one that could drive the stereo glasses system.

The C/C++ and OpenGL code used to render the hyperbricks in stereo is based on the code explained in [24]. Although one might think that rendering an object in stereo might be much more complicated than rendering simply in 2D, it is not so difficult. Instead of rendering the hyperbrick image on a single buffer, the program renders two images on two buffers, each image seen from a slightly different viewpoint. Algorithm 1 shows a generalized method of rendering objects in stereo.

Algorithm 1 the general method of rendering a hyperbrick in stereo

```
if rendering in stereo then
  update stereo camera, normals, viewpoints
  select right buffer
  render hyperbrick
  select left buffer
  render hyperbrick
end if
for all buffer in buffers do
  swap buffer
end for
```

Each buffer (left and right) is only seen by the corresponding lens in the stereo glasses. If one were to shut one eye, a full 2D image of the hypercube would still be seen through the other. Because of the slightly different viewpoints from each eye, an illusion of depth is achieved and thus the rendering in stereo is complete. These different viewpoints are the key to rendering a good stereo image - if they are not correct, the image may appear to have 'shadows' from the other buffer, in general just be an unconvincing 3D image, or even just appear to be rendered in 2D. Note that both left and right buffers are swapped after the hyperbrick has been rendered - this is not to say that the left buffer is swapped with the right buffer and vice versa, rather, the left buffer is swapped with another buffer on the left and the same with the right. Usually this swapping of buffers refers to each set of left and right buffers as back and front, making four buffers altogether: back left, back right, front left and front right, and hence the term 'quad-buffering'. As stated previously, this technique allows the program to render smoothly, in the same way as the double-buffering technique allows non-stereo graphics programs to render smoothly.

So how do we calculate the viewpoints in order to get a convincing 3d stereo projection? As described in [24], there are two main ways of doing this. The first is what is known as the "toe-in" method, where both camera views are pointed at a single focal point, the second is what is known as the "off axis" method, where the views of each camera are parallel to each other. The "off-axis" method is considered the superior of the two methods as the discomfort levels are lower than the toe-in method due to the fact it does not introduce a vertical parallax [25].

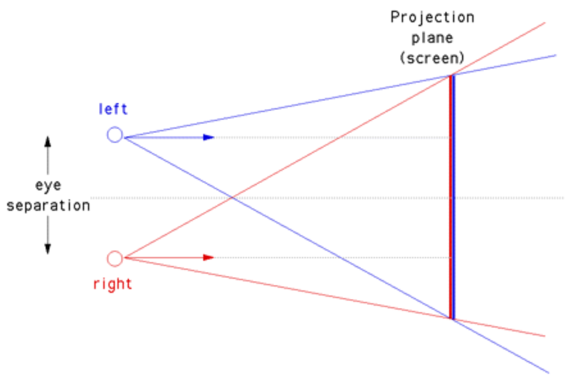


Figure 5: The off-axis stereo projection geometry.

Figure 5 demonstrates the geometry for the "off-axis" method to project the image in stereo 3D, showing the key distances of eye-separation and approximate eye-screen distance.

The code listed in Figure 6 shows how one would typically select the left buffer to be ready for rendering. The function glFrustrum() is used to calculate the correct viewing projection for the left "eye" and the glLookat() function is used to position the "eye" in the correct place and pointing in the correct direction. All variables not declared in the function (they are members of the camera class) are directly dependent on the camera's position, focal length, aperture and eye separation variables, all of which can be altered slightly by the user to get the optimum 3D image. The code for selecting the right buffer is very similar, with only minor changes to the glLookat() function making sure the right "eye" is "mirrored" to the left counterpart.

4 Stereo Visualisation Results

The resulting stereo visualisation of the Cubes data sets was successful. It is difficult to show the results properly on paper for obvious reasons; it is not possible to show proper stereo 3d displays on paper. The Cubes program can load in any compatible hyperbrick data set in standard non-stereo mode, and the user is able to toggle stereo mode on and off as needed. The code to display the Cubes data in stereo is completely separate

```
void Camera::LeftBuffer(){
  glDrawBuffer(GL_BACK_LEFT);
  glClear(GL_COLOR_BUFFER_BIT |
          GL_DEPTH_BUFFER_BIT);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  double ratio = scrWidth / scrHeight;
  double wd2 = near * tan(RAD*aperture/2);
  double ndfl = near / focalLength;
  double left = -ratio * wd2 + 0.5
    * eyeSeparation * ndfl;
  double right = ratio * wd2 + 0.5
    * eyeSeparation * ndfl;
  double top = wd2;
  double bottom = -wd2;
  glFrustrum(left, right, bottom, top,
    near, far);
  glMatrixMode(GL_MODELVIEW);
  glLoadIdentity();
  glLookat(-Pos.x-norm.x, -Pos.y-norm.y,
        -Pos.z-norm.z,
        -norm.x, -norm.y, -norm.z,
        Up.x, Up.y, Up.z);
  glRotatef(-Rotation.x, 1.0, 0.0, 0.0);
  glRotatef(Rotation.y, 0.0, 1.0, 0.0);
}
```

Figure 6: OpenGL Code outline for preparing Left "eye" buffer for rendering.

to the Cubes program code itself, and no functionality was sacrificed in order to display it in stereo. Any hyperbrick data set can be rendered in stereo, provided the same data set is rendered on each buffer, as shown in Algorithm 1 .

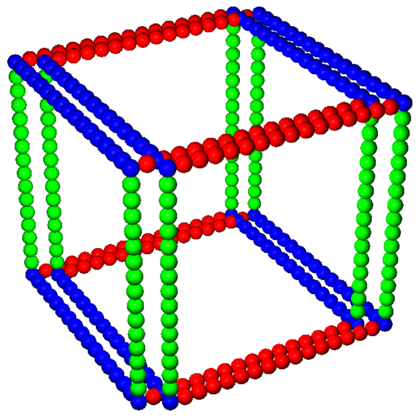


Figure 7: The 'scaffold' predefined test data set as it would appear on screen to the naked eye.

For the stereo display to be most effective it is recommended that the Cubes rendering properties be changed to a sphere rendering model, with grids and frames surrounding the voxels removed and only non-zero data visible. It is also recommended to have some sort of lighting model switched on for added effect. While these properties produce the best results when viewing the data in stereo, these properties can be changed and the stereo display will still work. These properties are the ones chosen for the following screen-shots showing some examples of the program.

Figure 7 shows the pre-defined test data set "Scaffold" as how it might appear on screen to the naked eye; this is not a true representation of how it would look, as the image would appear slightly 'faded' as each buffer flickered on and off. Because at any one time only one buffer is being displayed, it was necessary to render the images of both eyes on only one buffer when taking screenshots, otherwise only the currently displayed buffer would be shown and the image would appear as it would in a non-stereo display mode. The scaffold is simply a hyperbrick of voxels, rendered as Phong-lit[6] spheres, and with all but the edge voxels switched off, and with the x-y-z axes coloured red-green-blue to aid orientation. This data set illustrates well how the stereo display works - the image for each eye is not simply displaced horizontally by a certain amount, but rather the object is viewed at slightly different angles for each eye. This is shown in the scaffold data set by the cube's edges being almost but not quite parallel to each other. The camera's aperture angle can be increased to show this more clearly (this would give the effect of the camera moving closer to the object, or in stereo mode the object would appear further outwards of the screen and closer to the person viewing it).

A typical scientific use of stereoscopic vision is to aid identification of phenomena taking place in a simulation. The example we resent here is discussed further in[26] and involves growing a model of electro-deposition on a physical surface. Figure 9 shows the red particles grown on a green flat surface. The tendrils are hard to analyse and to see what is taking place in the simulated model. Cluster labeling techniques are used to identify separate clusters of connected tendrils, and the "biggest" is visualised separately in Figure 8. The stereo technology allows a much greater sense of the structure and morphology of the simulation that is possible with flat or shaded monocular rendering.

Figure 8 shows the different images for each eye - the left buffer is coloured blue while the right is coloured red. It features the hyperbrick "biggest", referring to the biggest tree generated by an invasion percolation. This data set is particularly effective in being displayed in stereo, as the voxels can be shifted in the x, y and z directions at the user's discretion. This allows parts of the hyperbrick to appear in the extreme front, back, top, bottom or sides which looks great in stereo mode.

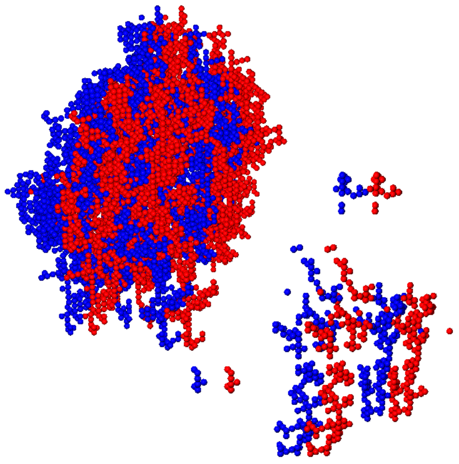


Figure 8: Red and blue colours for each eye for the predefined dataset 'biggest' - which is a cluster pulled out of a larger simulation dataset.

In Figure 8 the small group of voxels to the left and bottom of the image appear extremely close to the viewer, while the rest of the data set is much further away.

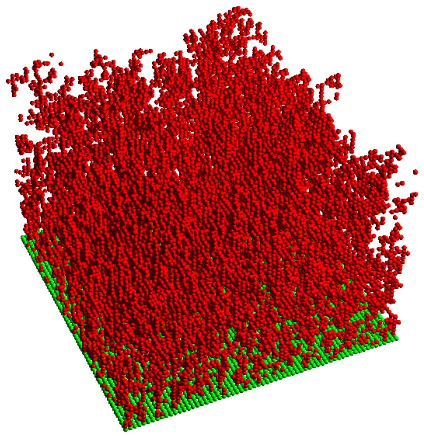


Figure 9: The 'original' hyperbrick dataset from a surface deposition simulation - showing how difficult it is to identify the structure of the individual clusters.

Figure 9 features the "original" hyperbrick dataset, referring to the entire data set generated by the epitaxial growth model (the "biggest" data set in Figure 8 is part of this data set). This data set was a lot more dense and contained a lot more voxels than most other hyperbricks, however the stereo display still worked fine. Because of the heavy clustering it was not practical to display a screenshot of both buffers, as it would be difficult to distinguish which voxels belonged to the left and right buffers. It is important to mention that this data set really challenged the graphics cards capabilities; displaying so many voxels as spheres as well as adding lighting effects and stereo buffering meant the program ran very slowly on the Quadro FX 380.

5 Discussion

In general the program performed well, however as the number of voxels in the hyperbrick is increased (for example a 64 x 64 hyperbrick instead of the standard 16 x 16) the performance levels decline. As mentioned previously, this is most noticeable when rendering the individual voxels as spheres, and even more so when lighting is turned on. This is mainly due to the fact that the graphics card which we used (Quadro FX 380) was a much older model and thus could not keep up with the large workload; however, rendering such large amounts of voxels as spheres and under lights can be taxing on even good graphics cards, and rendering the images twice to obtain the stereo image effect only adds to the workload. It would be good to see if we can find a way to reduce that while still maintaining a clear and convincing looking stereo image.

Another thing to note is that different people seemed to prefer different settings for the stereo camera in order to get the most comfortable image. For this reason the focal length, aperture and eye separation settings for the camera were enabled to be changed on the fly by the user using keyboard support while they were viewing the stereo image.

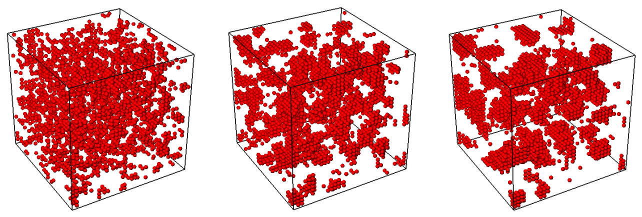


Figure 10: A sequence of voxel sets from simulation of a $32 \times 32 \times 32$ cell Kawasaki spin-exchange model. The particles clump together with time.

It is useful to view stereo-rendered animation sequences. Our system supports this by allowing a set of voxel sets to be cycled through dynamically. Typically a few thousand hyperbrick voxel set can be loaded in memory and used in this way. Figure 10 hows some sample renderings of a sequence of 1024 time-steps from a simulation of the Kawasaki Ising model[27] on a 32^3 lattice. In this model the red and white “spins” are initialised randomly and with time, the system anneals to forms clumping structures, whose precise shape depends upon the model parameters. Stereo visualisation aids identification of the clumping regimes and insights into the structures formed.

6 Conclusions

We have shown how a capability for stereo visualisation of voxel data can be readily implemented with commodity hardware and portable graphics software. Our cubes system offers this capability and users can choose whether to enable the stereo feature or not at run time. The stereo camera's focal lengths needed to be changed depending on the person viewing the program, otherwise the visualisation would leave a 'shadow' effect in each eye. Keyboard support for this was added. Although the stereo vision works well, it is only compatible with specific graphics cards (we used the Nvidia Quadro FX 380). The graphics card needs both OpenGL stereo support and quad-buffering support. OpenGL stereo support comes with all Quadro cards, which are the 'professional' equivalent to the Nvidia GeForce consumer cards. Quad-buffering support allows OpenGL to write to four buffers instead of just two. Only a selection of Quadro cards have this support. Hopefully in the future all or most graphics cards will come built in with both of these functionalities as stereo 3d becomes more prominent in the industry.

For future work it would be desirable to run the simulation on a Linux box. We believe the current generation Nvidia Quadro cards now support Linux. Other interesting interactive commodity-priced hardware such as Microsoft's Kinect device offer some promise for enhanced user-interactivity with a running simulation. Haptic devices such as the Kinect would allow users to manipulate or navigate the voxel data set using hand movements or other methods. It should be possible to take into account the user's head movements, and as the user moves around the image the projection would rotate accordingly, giving the effect of a 'virtual reality' environment. Several studies have been done [28][29][30] using this method. It has been found that coupling this method with stereo vision techniques gives better results than stereo vision on its own, so some experiments on the Cubes program using this method would be very interesting to do.

In summary, it is now economically and technically feasible to expect to have stereo vision and other user interactive devices commonly available on desktops to enhance the interactive experience and computational steering of a simulation experiment.

References

[1] Drebin, R.A., Carpenter, L., Hanrahan, P.: Volume rendering. In: ACM SIGGRAPH Computer Graphics. Volume 22. (1988) 65–74 ISSN:0097-8930.

[2] Wang, W., Sun, H., Wu, E.: Projective volume rendering by excluding occluded voxels. Int. J. Image Graphics **5** (2005) 413–432

[3] Stolte, C., Tang, D., Hanrahan, P.: Multiscale visualization using data cubes. IEEE Transactions on Visualization and Computer Graphics **9** (2003) 176–187

[4] Sherbondy, A., Houston, M., Napel, S.: Fast volume segmentation with simultaneous visualization using programmable graphics hardware. In: Proc. IEEE Visualization 2003. (2003) 171–176

[5] Levoy, M.: Display of surfaces from volume data. IEEE Computer Graphics and Applications **8** (1988) 29–37

[6] Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F.: Computer graphics: principles and practice (2nd ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1990)

[7] Hearn, D., Baker, M.P.: Computer Graphics with OpenGL. Third edition edn. Number ISBN 0-13-015390-7. Pearson Prentice Hall (2004)

[8] Hawick, K., Playne, D.: Spectral and real-space solid representations and visualisation of real and complex field equations. Technical Report CSTN-101, Computer Science, Massey University (2009)

[9] Fletcher, P.A., Robertson, P.K.: Interactive shading for surface and volume visualization on graphics workstations. In: VIS '93: Proceedings of the 4th conference on Visualization '93, Washington, DC, USA, IEEE Computer Society (1993) 291–298

[10] Schott, M., Pegoraro, V., Hansen, C., Boulanger, K., Bouatouch, K.: A directional occlusion shading model for interactive direct volume rendering. Computer Graphics Forum **28** (2009) 855–862

[11] Cruz-Neira, C., Sandin, D.J., DeFanti, T.A.: Surround-screen projection-based virtual reality: the design and implementation of the cave. In: Proceedings of the 20th annual conference on Computer graphics and interactive techniques. SIGGRAPH '93, New York, NY, USA, ACM (1993) 135–142

[12] Hawick, K.: 3d visualisation of simulation model voxel hyperbricks and the cubes program. Technical Report CSTN-082, Computer Science, Massey University (2010)

[13] Woo, M., Neider, J., Davis, T., Shreiner, D.: OpenGL Programming Guide: The Official Guide to Learning OpenGL. 3rd edition edn. Addison-Wesley (1999) ISBN:0201604582.

[14] VTK Team: Visualization Toolkit (VTK). See Website: `www.vtk.org` (2010)

[15] Persistence of Vision Pty. Ltd.: Persistence of Vision (TM) Raytracer. Williamstown, Victoria, Australia. (2004)

[16] Bertozzi, M., Broggi, A., Fascioli, A., Nichele, S.: Stereo vision-based vehicle detection. In: Proceedings of the 2000 IEEE Intelligent Vehicles Symposium. (2000)

[17] Maupu, D., Horn, M.H.V., Weeks, S., Bullitt, E.: 3d stereo interactive medical visualization. IEEE Computer Graphics and Applications **25** (2005) 67–71

[18] Webster, R., Haluck, R., Ravenscroft, R., Mohler, B., Crouthamel, E., Frack, T., Terlecki, S., Sheaffer, J.: Elastically deformable 3d organs for haptic surgical simulation. In: Medicine Meets Virtual Reality. IOS Press (2002)

[19] Aguilar, J., Torres, F., Lope, M.: Stereo vision for 3d measurement: accuracy analysis, calibration and industrial applications. Measurement **18** (1996) 193–200

[20] Ahearn, L.: 3D Game Art f/x & Design. Coriolis, Scottsdale, Arizona (2001) ISBN: 1-58880-100-4.

[21] Wheatstone, C.: Contributions to the physiology of vision - part the first. on some remarkable, and hithero unobserved, phenomena of binocular vision. Philosophical Transactions **128** (1838) 371–394 `www.stereoscopy.com/library/wheatstone-paper1838.html`.

[22] Beck, J.: Leonardo's Rule of Painting - an unconventional approach to modern art. 2nd edn. Phaidon Press (1979)

[23] Henderson, B.: Netpbm history. Web Site (2007) Last Visted October 2010.

[24] Bourke, P.: 3d stereo rendering using opengl (and glut). Website PDF (2002)

[25] Bourke, P.: Calculating stereo pairs. `http://paulbourke.net/miscellaneous/stereographics/stereorender` (1999)

[26] Hawick, K.: Morphology and epitaxial growth in a directed diffusion model. In: Proc. International Conference on Modelling, Identification, and Control (MIC), Innsbruck, Austria, IASTED (2011) 1–8

[27] Baillie, C.F., Gupta, R., Hawick, K.A., Pawley, G.S.: Monte Carlo renormalization-group study of the three-dimensional Ising model. Phys. Rev. B **45** (1992) 10438

[28] Ware, C., Franck, G.: Evaluating stereo and motion cues for visualizing information nets in three dimensions. ACM Transactions on Graphics **15** (1996) 121–140

[29] Ware, C., Franck, G.: Viewing a graph in a virtual reality display is three times as good as a 2d diagram. Technical report, Faculty of Computer Science, University of New Brunswick (1994)

[30] Deering, M.: High resolution virtual reality. Computer Graphics **26** (1992) 195–202

Low-cost Driving Simulator for Driver Behavior Research

Koskela, K.[1], Nurkkala, V-M.[1], Kalermo, J.[2], and Järvilehto, T.[1]
[1]Kajaani University Consortium, Psychology Research Laboratory SkilLab, University of Oulu, Kajaani, Finland
[2]Kajaani University Consortium, CEMIS-Oulu, University of Oulu, Kajaani, Finland

Abstract - *Driving simulator is an important tool for driving behavior studies. It offers a safe and replicable virtual driving environment where it is possible to create scenarios that are ethically, logistically and monetarily impossible to carry out in real environment. The development of simulator technology and especially the reduction of prices in certain key areas have made it possible to create low-cost driving simulators with the features that are usually found only in very expensive high-fidelity driving simulators. In this article a low cost driving simulator (pricing below € 20.000) for the research of the drivers' behavior is presented.*

Keywords: Low-cost, driving simulator, driver's behavior, motion-base, stereoscopic view, curved screen

1 Introduction

There are some commonly recognized reasons why simulation is used in driving performance or behavior studies. First, repeatability provides the researcher with the ability to study the phenomena numerous times, which in the real world would be hard to accomplish [1]. Second, safety plays a critical role when studying, for example, unexpected driving conditions or driving under the influence of alcohol. Third, tracking of the most operations of the driver becomes possible in a simulator [2], and fourth, a simulator enables the use of versatile research equipment, thus providing a comprehensive recording of multivariate data for detailed analysis. This was also the basis for the development of the present low-cost driving simulator.

Although simulation provides several benefits for research, it also generates a new problem known as simulator sickness. Simulator sickness has been reported to be less frequent in a motion based driving simulator compared with a fixed-based one [3], [4]. Slob [5] argues that the main reason why a motion system is important is the prevention of simulator sickness. Thus, one of the requirements for the development of the present driving simulator was that at least a moderate motion platform should be included.

There are impressive, high-fidelity driving simulators that mimic the real environment rather closely, such as NADS-I at the University of Iowa [6], Toyota's driving simulator [7], Daimler AG driving simulator [8], Swedish National Road and Transport Research Institute (VTI) simulators [9] and a recently developed simulator at the University of Tongji in Shanghai [10]. These simulators are also in the high-end of the budget. In the low-end of the budget, a simulator can simply be an off-the-shelf computer game with a steering wheel and pedals. Then again, there can be thousands of variations depending on the quality and price.

It is impossible to create a simulator that would in all aspects equal a real driving situation. No matter how realistically the simulation environment is created, the subject still knows that s/he is sitting in a simulator. All that can be done is to try to create as realistic a simulation environment as possible, in which the actions of the driver mimic those in the real driving situation. The main goal of the present project was to implement, as far as possible, features to the simulator that can usually be found in simulators belonging to the high-end of the budget. In this article, a technical description of a low-cost simulator solution which fits this description is presented.

2 Simulator hardware

A single custom-built high-performance desktop computer was used for running the simulator environment. The computer has an Intel 3.07 GHz i7 950 quad core with 12 GB of DDR3 RAM memory and Windows 7 operating system. Three nVidia GTX 470 graphic adapters were used for the video output. Two of these cards provide the 3D view for three 3D projectors (Optoma GT720) in Scalable link interface (SLI) parallel processing mode. Third graphic adapter was dedicated for calculating PhysX and providing video input for a LCD screen containing driving telemetry.

While the driving simulator environment was being developed, different display solutions were experimented. It became clear that for successful navigation and driving performance, a side view is imperative. Particularly, turning at the crossroads and detecting incoming traffic at intersections becomes increasingly difficult when the scene is limited to the front view only. An in-house designed 220 degree curved display with radius of 2.5 meters (8.2 feet) was constructed. The resolution for the projected image is 3840x720 and the actual size of the image in the curved screen is 10 x 1.9 meters (32.8 x 6.2 feet). For combining the image from three different projectors and making the view look seamless, a program called Nthusim is used.

nVidia's 3D Vision Home kit [11] with shutter glass technique was selected for the presentation of stereoscopic 3D view. It was expected that the stereoscopic view makes estimating distances easier and increases the feeling of realism by adding depth to the view. No driving simulator studies where stereoscopic shutter glass view was used were found. In a flight simulator research [12] stereoscopic 3D view was considered valuable in situations in which the aircraft is very close to the ground or to another airplane in taxiing, formation flying or air to air refueling, for example.

Some details of the real car were added to the simulator in order to increase the realistic driving feeling. These include a middle console with the gear shifter and hand brake from a Volvo S60, blinkers attached to the back of the seat and a high impact speaker installed inside the driver's seat in order to add a feeling of vibration while driving. Logitech Z-5500 high quality 5.1 Speaker system is used for creating realistic driving sounds. A small display including gages for speed, RPM, fuel level and engine temperature was added behind the steering wheel in order to create an illusion of real car indicators.

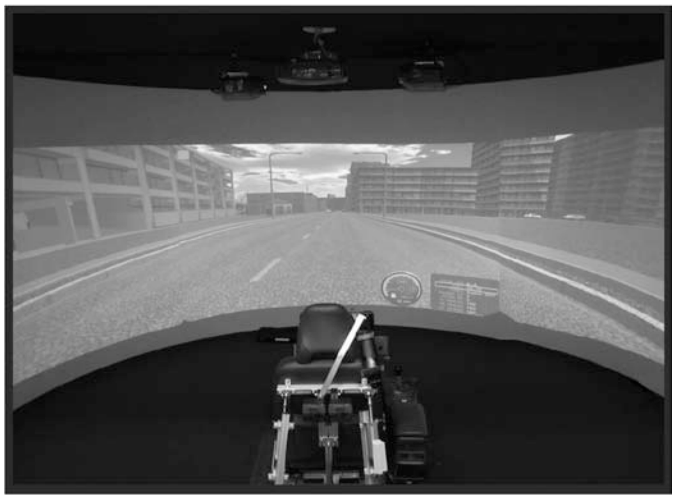


Figure 1. The driving scene of the low-cost driving simulator

The base for the motion platform and driving control system is manufactured by Frex GP (Osaka, Japan). This system was modified for more realism, including three degrees of freedom (3 DOF) motion platform, high quality steering wheel, pedals and gear shifter. The original linear actuators from Frex GP, which were only capable for moving 10 kg each, were replaced by actuators capable of moving 50 kg each. One of the actuators from the Frex GP system was used to create the rear sliding effect.

Controllers were connected to the simulator computer via USB. Linear actuators for the motion control were fastened into a car seat manufactured by Volvo, and the weight distribution of this "vehicle" is in the middle of the driver's seat. While driving in the simulator, traffic rules and speed limits (40–80 km/h; 25–50 mph) are to be followed. While driving under these rules the feeling of motion created by this light motion platform is adequate. For situations that contain a lot of fast movements, like off-road racing, the limitations of the movement platform become evident.

3 Simulator Software

rFactor, a commercial game developed by Image Space Inc., was used as the platform for the driving simulator. The game runs on an engine called gMotor-engine2. There are several substantial advantages on running the simulator using a commercial game. The price of the game is about 40 euros (55 dollars), which makes it a very cheap solution compared to the normal simulator software pricing in the tens of thousands of euro. The game developers have offered the users many tools for controlling the game environment, which makes the simulator environment highly modifiable.

There were also other reasons why rFactor was selected as the main software for the driving simulator. The game physics are generally considered to be realistic. No research data on the accuracy of the game physics were found, although the subjective feeling of the driving is quite realistic. For the present research project the feeling of realism is more important than the accuracy of the mechanics of the simulator.

Getting reliable telemetry data (recording of different mechanical aspects of driving) is imperative for research purposes. When using rFactor this is quite easily accomplished with an in-house programmed plugin. Another plugin is used for real-time telemetry monitoring of driving. The telemetry data includes large variety of variables which are important for our research purposes, such as steering, braking, accelerating, and even tire temperature or wheel rotation speed.

One of the most important factors in the selection of the simulation software was the ability to create custom-made driving scenarios. For the needs of varying research themes, it was essential that the simulation environment could be built from scratch without limitations. Bob's track builder (BTB) offers this option. BTB has a very accurate and easy-to-use interface for building roads and landscape. GPS location data can be imported and road maps can be created based on those coordinates. If the GPS data includes height data, this can be transferred directly to BTB. The road network can also simply be drawn and the height can be adjusted manually.

When the road network has been designed or imported, the road can be modified in varying ways. The width and the camber of the road can be changed and bumps and shapes can be added to the surface. The appearance of the road can be modified by importing road textures and more details can be added by including specular and bump maps to the road. These maps make the surface look more uneven or change the lighting and reflection settings of the texture. The friction properties of the road can be altered, thus creating for example black ice for unexpected slippery conditions. After the road network design is finished, a landscape can be added around the roads. The height, texture or the driving feel of the landscape can be modified.

One of the main goals for the simulator was to create driving scenarios which include elements from the Finnish traffic environment, such as buildings, trees and traffic signs. For the modeling, several different programs were used. Most of the 3D modeling was done using Google SketchUp because of its good compatibility with Bob's Track Builder. Autodesk 3Ds Max and 3Dsimed were used for the more detailed graphics and animated textures. 3ds Max road object set was purchased and used in order to improve the outlook of roads. The package includes high quality roads with realistic sidewalks and street lights.

The main challenge with the designing of 3D objects is to make the objects look realistic, but still keep the structure of the models simple enough to ensure the smooth rendering of the world. This requires reducing some graphical improvements, such as transparent windows on distant houses and 3D model trees in the distance. Detailed models were used near the driving lane, whereas low detail objects were used in the background. The model's level of detail is also altered depending on the drivers distance from the object.

The default control program for the actuators provided by Frex GP was replaced by a program called X-Sim. It offers a more profound control over the movement of the actuators. X-Sim program runs on a separate computer, and it receives motion data from the game engine. This data is transformed into movement based on set multipliers, which define the intensity of the movement that the actuators perform. The data is then sent to the motion platform via Ethernet cable. Because the drivers have to follow moderate speed limits and traffic rules, the G forces do not rise very high. Therefore the G force data received from the game was dampened, and most of the movement of the motion platform was based on the values of pitch, roll and yaw.

4 Research equipment

For the successful research on human driving behavior, measurements of physical actions of simulator are required (telemetry data) synchronized with different kinds of recorded data from the driver. The motion control software X-Sim provides an option to create a trigger signal for outside measuring devices. A data channel with no value for the researcher (tire temperature) was selected for the synchronizing purposes (triggering). Once the driving situation starts the data is automatically sent to the X-Sim program, which transforms it to a start signal for the NeurOne EEG / EMG device [13], EyeLink II eye movement device [14] and Basler high speed cameras with Vicon motion analysis software [15][16]. NeurOne is used for most of the physiological measurements, including the measurements of EEG, EMG, GSR and heart rate. These physiological measurements offer a comprehensive outlook on the neurophysiological aspects of driver's behavior during tasks in the simulator.

5 Data Analysis

In the driving situation a set of multimodal and time-stamped synchronized data is collected. These consist of 1) experiential data (subjective reports), 2) behavioral data (observable changes in the subject's behavior), 3) electrophysiological recordings (EEG, EMG GSR, heart rate), 4) eye movement recordings, 5) mechanical data from the vehicle, including the use of different control parts (steering wheel, brake, throttle), and more continuous events, such as use of gas, and 6) environmental situation (video recording of the road and driving environment, other vehicles, pedestrians, traffic signs, weather conditions).

The setup offers the possibility for the integrated analysis of the factors important in driving. Thus, we develop methods that make comprehensive use of all the multimodal data. The aim is to sort out from the wealth of data factors that explain, e.g., successful driving or driving accidents. This kind of knowledge may then be applied to the planning of the traffic environment or to the teaching of driving.

6 Discussion

The price of the present simulator does not exceed € 20.000 ($ 28.500). However, with such a low price, there are some limitations especially with the software that restricts the type of research that can be carried out with the simulator. Lack of an intelligent triggering mechanism prevents the research on more complicated driving situations. The traffic in the environment cannot be controlled, and other vehicles follow a pre-programmed route responding accordingly to objects in the driving environment. They may brake, for example, if there is another car or an object on the road, but they do not react to the objects themselves (e.g., to the meaning of the traffic light, or rail road crossing beams). Such limitations make it necessary to upgrade the simulation software in the future if more complex traffic situations will be studied.

Research on the advantages of 3D stereoscopic view while driving would be very important. At the moment, 3D view is used because it is expected to have a positive effect on the estimation of distances and making correct turning in the intersections of the road easier. In addition, it is not known whether 3D scene is more susceptible to simulation sickness. In this respect more research is needed.

The present simulator provides an excellent tool for performing physiological (EEG, EMG) and eye movement measurements (for the latter especially the large driving scene is important). EEG and EMG measurements are very sensitive to mechanical and electrical disturbances. However, the motion platform, despite of being rather moderate, has its advantages, as it does not cause too much mechanical or electrical ambient noise. With good grounding no problems have been encountered in the recording of electrophysiological data.

ACKNOWLEDGEMENTS

The development of the driving simulator environment is funded by European Regional Development Fund, the Joint authority of Kainuu and Sunit Oy.

7 References

[1] Brooks, J.O., Goodenough, R.R., Crisler, M.C., Klein, N.D., Alley, R.L., Koon, B.L., Logan, W.C., Ogle, J.H., Tyrell, R.A. & Wills, R.F. (2010). Simulator Sickness during Driving Simulator Studies. Accident Analysis and Prevention 42, 788–796.

[2] Wang, Y., Zhang, W., Wu, S. & Guo, Y. (2007). Simulators for Driving Safety Study – A Literature Review. in R. Shumaker (Ed.): Virtual Reality. Lecture Notes in Computer Science, 2007, Volume 4563/2007, 584-593, Springer Berlin / Heidelberg. DOI: 10.1007/978-3-540-73335-5_63.

[3] Curry, R., Artz, B., Cathey, L., Grant P. & Greenberg J. (2002). Kennedy ssq results: fixed- vs motionbased FORD simulators. Proceedings of Driving Simulator Conference, pp.289–300, 2002.

[4] Watson, G. (2000). A synthesis of simulator sickness studies conducted in a high fidelity driving simulator. Proceedings of Driving Simulation Conference, pp. 69-78, 2000.

[5] Slob, J.J. (2008). State-of-the-Art Driving Simulators, a Literature Survey. DCT 2008.107. DCT report. Eindhoven University of Technology.

[6] National Advanced Driving Simulator. http://www.nads-sc.uiowa.edu/. Retrieved March 1, 2011.

[7] Toyota Driving Simulator. Toyota Motor Corporation. http://www.toyota.com/esq/articles/2010/Driving_Simulator.html/. Retrieved March 1, 2011.

[8] Daimler AG. http://www.daimler.com/. Retrieved March 3, 2011.

[9] Swedish National Road and Transport Reserch Institute (VTI). http://www.vti.se/. Retrieved March 1, 2011.

[10] Tongji University Driving Simulator. http://www.tongji.edu.cn/english/inc/index.asp/. Retrieved March 1, 2011.

[11] nVidia 3D Vision Home Kit. http://www.nvidia.com/object/3d-vision-main.html/. Retrieved March 1, 2011.

[12] Menéndez, R.G. & Bernard, J.E. (2000). Flight Simulation in Synthetic Environments. Digital Avionics Systems Conferences, 2000. Proceedings. DASC. The 19th , vol.1, no., pp.2A5/1-2A5/6 vol.1.

[13] NeurOne EEG/ERP Device. Mega Electronics Ltd., Kuopio, Finland. http://www.megaemg.com/.

[14] EyeLink II eye movement recording system. SR Research, Canada. http://www.srresearch.com/index.html/. Retrieved March 3, 2011.

[15] Basler high speed cameras. Basler AG, Ahrensburg, Germany. http://www.baslerweb.com/.

[16] Peak Motus, Vicon Motion Systems, Oxford, UK.

SOFTWARE

Autodesk 3ds max. http://usa.autodesk.com/3ds-max/.

Bob's track builder. http://www.bobstrackbuilder.net/.

Google SketchUp. http://sketchup.google.com/.

Nthusim. Nthusim Pty Ltd. http://nthusim.com/.

rFactor game. Image Space Incorporated, USA. http://rfactor.net/.

X-SIM software. http://www.x-simulator.de/forum/.

3Dsimed software. http://www.sim-garage.co.uk/.

A Study of User Experiences with Various 3D Interfaces for a Mobile Application

David Redding, Benjamin Bishop
Department of Computer Sciences
University of Scranton
Scranton, PA USA
redding.dave@gmail.com, bishop@cs.uofs.edu

***Abstract*--With the availability of OpenGL ES and low-power specialized graphics hardware, 3D graphics are increasingly incorporated into software for mobile devices. However, there is only a small body of research regarding effective user interfaces in this environment. The goal of this study is to analyze the effectiveness of several 3D mobile interfaces by conducting an experimental survey. Study participants were asked to solve a 3D graphical number placement logic-based puzzle using four different interfaces. Quantitative survey results are reported as well as conclusions to aid in the development and implementation of effective user interfaces for 3D software for mobile devices**

***Keywords*--3D graphics, user interface, mobile device, mobile development, OpenGL ES**

I. Introduction

Industry groups report that the worldwide smart phone applications market grew more than $2.2 billion dollars within the first six months of 2010 and application download numbers reached a total of 3.8 billion in only six months, compared to 3.1 billion in 2009[1]. However, prior work suggests that the mobile market is becoming saturated with repetitive and low quality applications[2]. The present study attempts to improve this situation by exploring options for effective 3D mobile interfaces. This is approached by creating a 3D puzzle and implementing different user interfaces to manipulate the 3D object

II. Prior Work

A number of related projects are reviewed and discussed below.

A. Lab vs. Field Testing

Kaikkonen et al. explored whether mobile device research can be conducted in a lab without adverse consequences compared to field tests[3]. It was concluded that mobile device research may be conducted in the lab, without construction of a complex field test.

B. Different User Interfaces

Bucolo et al. implemented three different controls to navigate a ball through a maze using the integrated camera that came with the mobile phone. The first control scheme used the traditional joystick to move the ball through the maze, while the second required users to “pan” the phone itself, and finally the third control scheme used a “tilt” action of the phone to move the ball. Results from the study showed that the traditional joystick control provided the fastest times but the least amount of user interaction with the mobile device, while the tilt controls were the most challenging but provided the greatest amount of user involvement [4].

C. Interaction in the 3D World

Chehimi et al. tested using the built-in accelerometer on the mobile device as a different control scheme to use while playing a 3D multiplayer game[5]. The accelerometer would be used to pilot a space ship through space. The accelerometer was used to turn left or right as well as up or down.

Winkler et al. used the built-in camera from the mobile device to play a 3D racing game and for map navigation[6]. The camera was found to be effective for this purpose.

D. Finger Gestures

Tests by Stößel et al. showed that people between the ages of 20-35 preferred more fingers to be used for a single gesture while people between the ages of 60-75 would prefer more finger presses to achieve the same goal[7]. For example, to zoom in on a map on a mobile device, the younger age group preferred the “pinch to zoom” finger gesture while the older age group preferred to double tap on the screen. For 3D interface research, it may be most effective to keep finger gestures to a minimum while interacting with 3D objects in the hopes that subjects of all ages will be able to focus on the 3D user interface rather than gestures.

III. Implementation

The 3D puzzle was developed on a mobile device that supported the Android OS. Android OS is open source, is well documented, and has a large support community[8]. Android OS currently supports OpenGL ES 2.0+ libraries, which were used to create the 3D and 2D look-and-feel of the puzzle. A 1x1x1 block used 12 triangles to create its overall cube shape, followed by 64 1x1x1 blocks to make the 4x4x4 puzzle. A textured image was imposed onto each side of the 1x1x1 block to show the user what value was associated to it. A separate thread was used to pick up finger presses from the user. This thread returned the screen coordinates and finger actions to the software to be handled appropriately. For the 3D games, the individual blocks were moved using simple linear and non-linear functions. For the 2D effect, all the 1x1x1 blocks were placed on the same plane with the camera placed on a vector normal to the plane.

When the user presses onto the screen, a vector is drawn from the camera. If this vector collides with a 1x1x1 block, a notification would appear requesting an action from the user. Upon each successful collision, the software checks to verify if the puzzle has been solved.

IV. Methodology

The present study required data from human subjects. Authorization for this study was received from the University of Scranton Institutional Review Board under protocol # 20-10A. The research tested different user interfaces on a mobile device used by students from The University of Scranton.

A. Population Surveyed

The students were drawn from required general education courses. Before participating in the research, the students were asked to answer a few background questions. These questions asked them their gender, age, and to rate themselves from 1-5 their experience with computers, mathematics, mobile devices, mobile games, and figure gestures on a touch screen. Figure 1 shows the results that were collected by this survey. The students varied in age (18-22) as well as major. From this population, 6 women and 12 men were surveyed. The students that participated in this study received extra credit in their general education course.

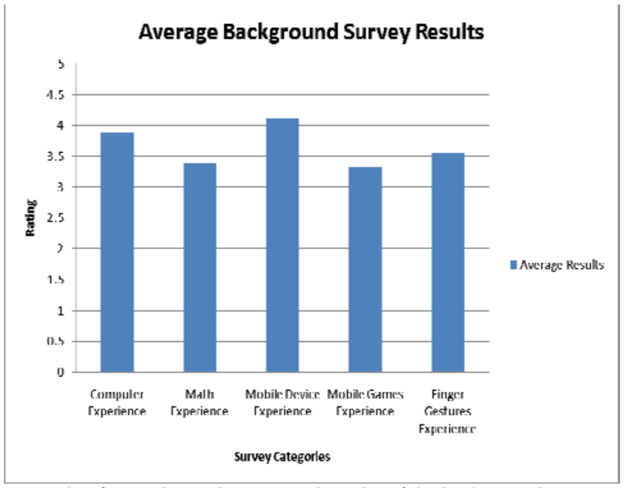


Fig. 1 This figure shows the averaged results of the background survey given to the participants before the experiment

B. Mobile Device

The mobile device used during the study was the HTC Incredible (Figure 2.) This mobile device contained a 1GHz Snapdragon processor, 8GB of internal memory, an 800 x 480 screen resolution, and ran Android 2.2.

Fig. 2 The front and back of the HTC Incredible smart phone

C. 3D Logic-Based Number Placement Puzzle

The puzzle being created is a 4x4x4 Latin cube[9] that follows three rules. The rules are as follows: each row must contain the numbers 1-4 exactly once, each column must contain the numbers 1-4 exactly once, and along the remaining axis the numbers 1-4 must occur exactly once.

Each time a game starts, the program randomly creates a solved puzzle, based on the rules noted above. Then, the program traversed each of the individual numbered blocks and unassigned it with 10% probability. The testers were asked to correctly assign values to these unassigned blocks in order to solve the puzzle.

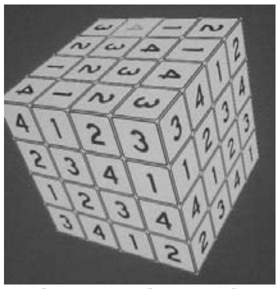


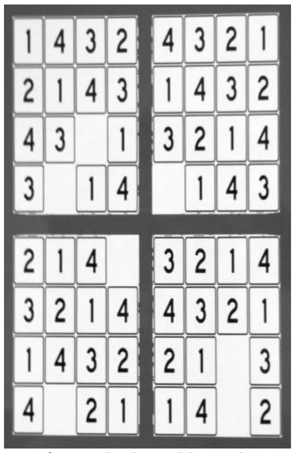
Fig. 3 A complete Latin cube using the numbers 1-4

D. User Interfaces

Focusing on how the testers would be able to interact with the inside of the puzzle, we created four different user interfaces.

1) *2D User Interface:* This user interface showed each layer at once on the mobile device. This interface gave a traditional look and feel for the testers to try out. If the puzzle were increased to a size larger than 4x4x4, this user interface would become difficult to interact with due to the small screen size of the mobile device. An example of this is as follows: if the puzzle were a

6x6x6, there would be six 6x6 Latin squares that need to be solved following the rules of a 6x6x6 Latin cube.



1	4	3	2	4	3	2	1
2	1	4	3	1	4	3	2
4	3		1	3	2	1	4
3		1	4		1	4	3

2	1	4		3	2	1	4
3	2	1	4	4	3	2	1
1	4	3	2	2	1		3
4		2	1	1	4		2

Fig. 4 The 2D user interface; which is able to show each individual layer on the screen at one time.

2) *Exploding Cube:* Upon a double tap, the larger cube containing all the numbers would split into smaller blocks containing their individual number. Upon another double tap, the smaller blocks would return and form the original cube.

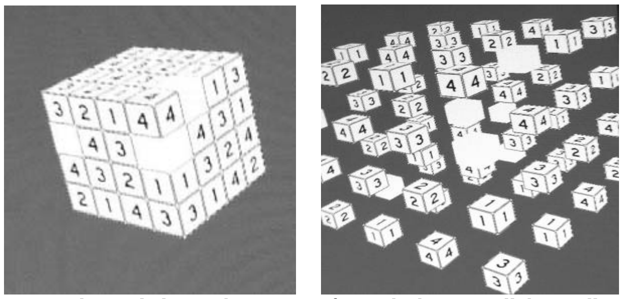


Fig. 5 The Exploding Cube user interface; which causes all the smaller blocks to "explode" outwards upon a double tap allowing the user to see the inner blocks

3) *FlipBook:* This user interface peeled off one layer at a time to expose the next inner layer. Once this front layer was removed, it was returned to the opposite side of the cube. Due to the Latin cube qualities, this does not violate any of the rules placed on the game since each individual row, column, and depth should still contain only one occurrence of the numbers 1-4.

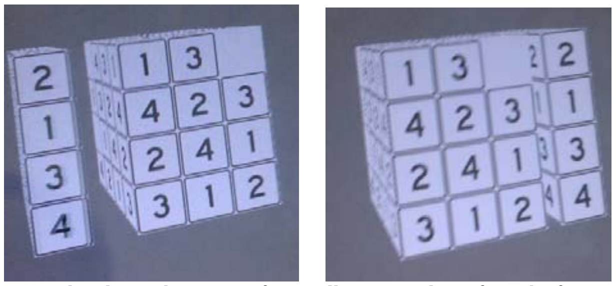


Fig. 6 The FlipBook user interface; will remove a layer from the front of the cube and replace it on the back of the cube ever time the user does a double tap finger gesture.

4) *Transparent Cube:* Upon a double tap, this user interface made the outer layer of the cube become transparent, which exposed the inner 2x2x2 cube. At this view, the tester could only interact with the inside cube. The outside cube is shown in wireframe to help display to the tester that they were looking inside the cube. If this puzzle were increased to a larger size, the user interface would have to make more layers become transparent. For example, in the 4x4x4 puzzle, the participants only needed to remove one layer. In a 5x5x5 puzzle, after removing one layer, there will still be a 3x3x3 puzzle remaining. Therefore, another layer would have to be removed to show the inner 1x1x1 block.

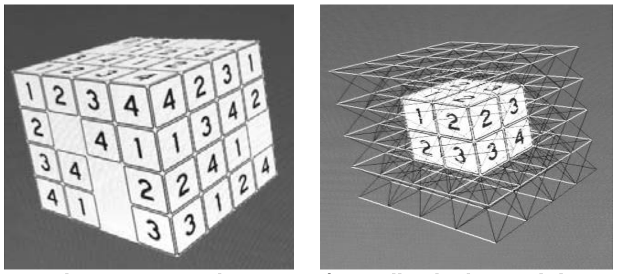


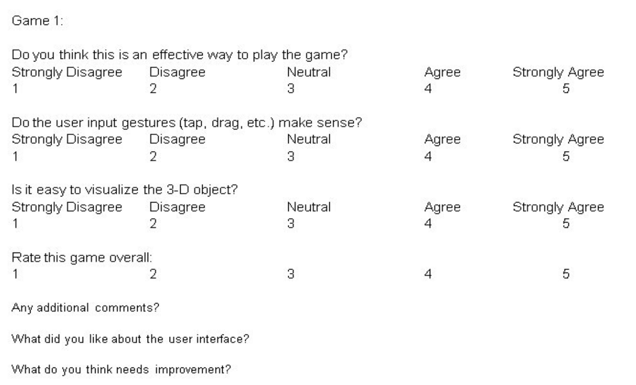
Fig. 7 The Transparent cube user interface; will make the outside layer of the cube become clear to allow the user to interact with the inner cube. The wireframe models remain on the screen to help illustrate to the participants that they are looking inside the cube

E. Conducting the Experiment

Each subject was given a brief introduction which explained what the study was about and what was expected of them. The subject was presented with the survey which obtained non-identifying information about his/her experience with smart phones and his/her mathematical background. Each subject was then given all of the user interfaces in a random order to solve a randomly generated puzzle. Each puzzle solution attempt was timed and documented.

F. Obtaining Data

After the participants solved all puzzles using the various user interfaces, they were given a questionnaire which asked them to answer various questions on each user interface. The results of the questionnaire were analyzed to draw conclusions.


Game 1:

Do you think this is an effective way to play the game?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

Do the user input gestures (tap, drag, etc.) make sense?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

Is it easy to visualize the 3-D object?

Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
1	2	3	4	5

Rate this game overall:

1	2	3	4	5

Any additional comments?

What did you like about the user interface?

What do you think needs improvement?

Fig. 8 After the participant completed the puzzle with each user interface, they were presented with this questionnaire-one for each user interface listed above.

V. Results

A. Preferred User Interface

From the results of the quantitative survey for each user interface the participants tested (Figure 9), the 2D user interface was preferred overall.

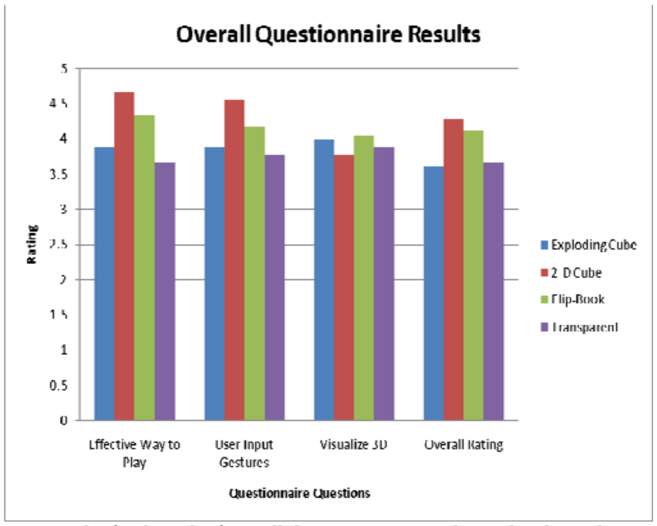


Fig. 9 The final results from all the participants in the study; shown here, the 2D interface is most preferred.

B. Discussion

Due to the small puzzle size, the 2D user interface provided a clear look at all the layers at any given time. It is suspected that performance using the 2D user interface would deteriorate for larger puzzle sizes, say 6x6x6. This would be due to congestion; for example, in the 6x6x6 case, the user would be interacting with 216 very small blocks. It may have been more meaningful to test these larger puzzle sizes, however, survey time constraints dictated a smaller size.

For time to puzzle completion, the mean (M) and standard deviation (SD) were calculated. The 2D and FlipBook (FB) user interfaces showed the characteristics [M_{2D} = 34.6s, SD_{2D} = 18.4s, M_{FB} = 52.2s, SD_{FB} = 14.8s] respectively. For the Exploding Cube (EC) and Transparent (T) interfaces, their times were more variable [M_{EC} = 80.7s, SD_{EC} = 62.3s, M_{T} = 78.6s, SD_{T} = 57.7s]. A small number of participants made time-consuming errors while using the Exploding Cube and Transparent interfaces.

C. User Reactions

While using the Exploding Cube user interface, participants made remarks on how the spacing between the cubes should be changed and slowed down. Although it was visually pleasing, some participants found it to be confusing and difficult to interact with. The FlipBook user interface implemented a realistic animation to peel off the layer and move it to the other side. This animation, which incorporates inertia, may have helped to give the user interface a more realistic feel than the others.

D. Different Populations

Further analysis from the results have shown that, in the personal background survey, subjects who reported higher numbers[1] on mobile device experience (MDE), mobile game experience (MGE) and finger gestures experience (FGE) favored the FlipBook user interface over all the others (Figure 10). The subjects with lower experience in those categories preferred the 2D user interface (Figure 11).

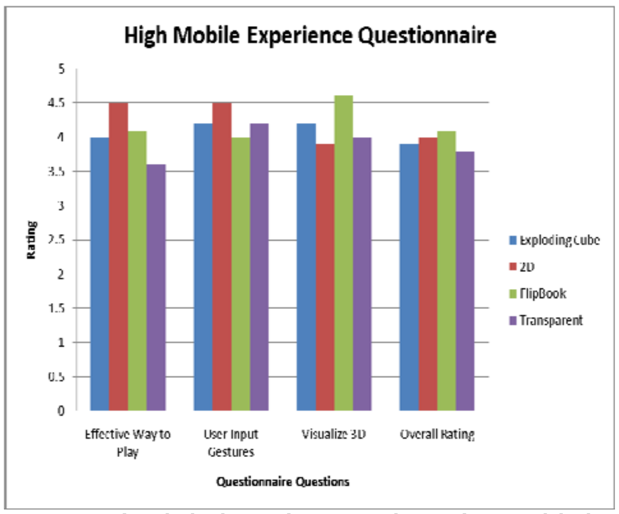


Fig. 10 Based on the background questions, the population with higher mobile experience preferred the FlipBook user interface and was able to visualize the 3D cube much better than using alternative interfaces

[1] Overall Mobile Experience is mobile device experience + mobile games experience + finger gesture experience. If Overall Mobile Experience is greater than 11 out of 15, the participants were considered to belong in the high mobile experience population.

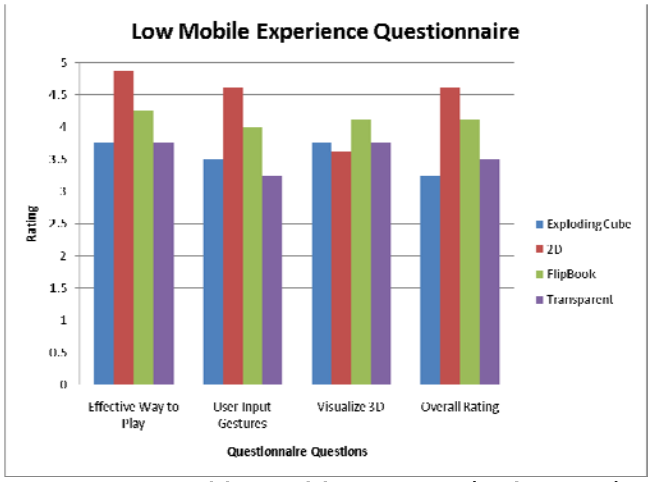


Fig. 11 Participants with lower mobile experience preferred a 2D interface even though it was slightly harder to visualize the 3D object

The average times to puzzle completion for males using the Exploding Cube and Transparent interfaces [$M_{EC,Males}$ = 61s, $SD_{EC,Males}$ = 21.4s, $M_{T,Males}$ = 63.7s, $SD_{T,Males}$ = 48.3s] were less than females [$M_{EC,Females}$ = 120s, $SD_{EC,Females}$ = 97s, $M_{T,Females}$ = 108.3s, $SD_{T,Females}$ = 67.7s], however, the average times for females using the 2D and FlipBook interfaces [$M_{2D,Females}$ = 29.2s, $SD_{2D,Females}$ = 2.6s, $M_{FB,Females}$ = 48.5s, $SD_{FB,Females}$ = 16.5s] were slightly less than the males [$M_{2D,Males}$ = 37.3s, $SD_{2D,Males}$ = 22.3s, $M_{FB,Males}$ = 54.1s, $SD_{FB,Males}$ = 14.2s]. This may be due to differences in background between participants of each gender. Differences in technical background (computing, mathematical, etc.) were not controlled for in this study. Males reported a stronger computer and mathematical background as well as more experience using a mobile device, playing mobile games, and using finger gestures (Figure 13).

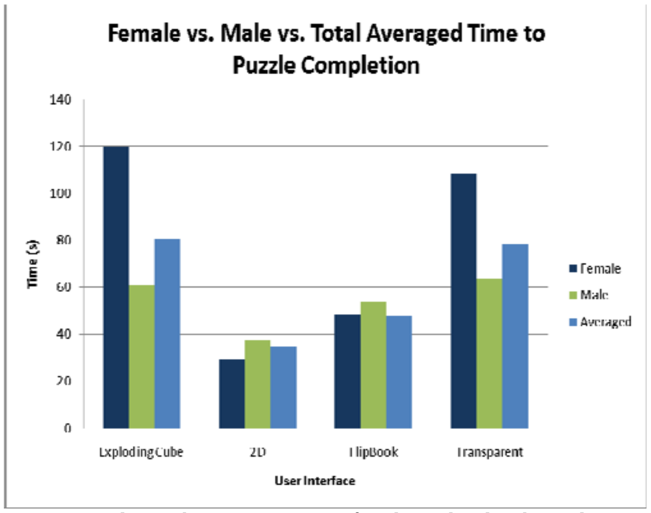


Fig. 12 Splitting the participants into females and males show a large difference in overall time between the two populations, especially involving the 3D user interfaces. Total Average time was generated by totaling all the times and dividing it by the number of participants

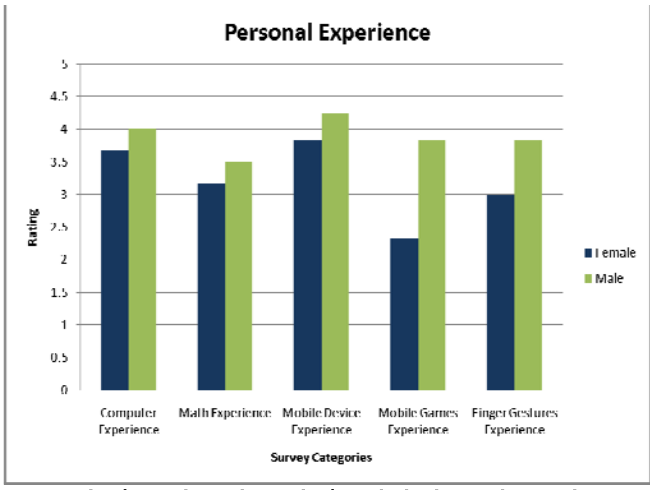


Fig. 13 This figure shows the results from the background survey between the females and males that participated in the study.

VI. Conclusions

From the research conducted, the 2D user interface produced the best results from the participants in the study. The screen size allowed a clear view of all four layers to solve the 4x4x4 puzzle. However, if this puzzle were to grow, say a 6x6x6 puzzle, it would not be practical to show all the layers on the screen at one time. Due to the small screen size, the layers would be far too small to interact with or observe.

The FlipBook user interface scored the best among the 3D user interfaces and was preferred by the participants who reported a higher overall mobile experience. The natural animation of the FlipBook, may have caused it to be preferred over the Exploding Cube and Transparent Cube user interfaces. This could be attributed to the FlipBook animation being realistic - incorporating inertia. Another factor may have been that the animation was not as abrupt as the other two 3D interfaces. For example, the Transparent Cube immediately made the outside layer into a wireframe layer as opposed to a gradual transparency over time.

It is important to note the time difference between the male and females. The males had relatively constant times on all the user interfaces where as the females took very different times on each one. Specifically on the 3D user interfaces, with the Exploding Cube interface, on average, the females took 120 seconds to solve the puzzle where, the males took 61 seconds. A similar effect was observed with the Transparent Cube; the females took 108.3 seconds and the males took 63.6 seconds to complete the puzzle. In the FlipBook user interface, the females' average time was 48.5 seconds and the males' average time was 54 seconds. A possible explanation for this effect may be that the FlipBook resulted in smaller spatial changes to the 3D object. Alternatively, the FlipBook animation was also less abrupt than the other 3D interfaces.

VII. References

[1] Research2guidance, The Mobile Research Specialists, http://www.research2guidance.com/ the-smartphone-application-

market-has-reached-more-than-2.2-billion-dollars-in-the-first-half-of-2010/

[2] Chehimi, F., Coulton, P., and Edwards, R.: Evolution of 3D mobile games development, Personal and Ubiquitous Computing 12, volume 12, Springer London, 19–25, (2008)

[3] Kaikkonen, A., Kekäläinen, A., Cankar, M., Kallio, T., and Kankainen, A.: Usability testing of mobile applications: A comparison between laboratory and field testing. Journal of Usability Studies, pp.4-16 November (2005)

[4] Bucolo, S., Billinghurst, M., and Sickinger, D.: User Experiences with mobile phone camera game interfaces. Mobile and Ubiquitous Multimedia, pp. 87-94. ACM Press, New York (2005)

[5] Chehimi, F., and Coulton, P.: Motion controlled mobile 3D multiplayer gaming. Advances in Computer Entertainment Technology, pp. 267-270. ACM Press, New York (2008)

[6] Winkler, S., Rangaswamy, K., Tedjokusumo, J., and Zhou, Z.: Intuitive application-specific user interfaces for mobile devices. Mobility '07 Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology, pp. 576-582. ACM Press, New York (2007)

[7] Stößel, C. and Blessing, L.: Mobile device interaction gestures for older users. Human-Computer Interaction: Extending Boundaries, pp. 793-796, ACM Press, New York (2010)

[8] Google, Android Developers, http://d.android.com/

[9] Dougherty, S. and Szczepanski, T.: Latin k-hypercubes. The Australiasian Journal of Combinatorics, pp. 145-158. (2008)

Low-Cost, High-Fidelity Virtual Landmine Detection Training System

W. Zhu[1], M. C. Leu[1], X. F. Liu[2], R. Kotikalapudi[2], H. He[1], S. Surisetty[3], J. D. Plunkett[4], G. Pierson[4], B. M. Davis[5]

[1]Department of Mechanical and Aerospace Engineering, Missouri University of Science and Technology, Rolla, Missouri, USA

[2]Department of Computer Science, Missouri University of Science and Technology, Rolla, Missouri, USA

[3]School of Electrical Engineering and Computer Science, Oregon State University, Corvillis, Oregon, USA

[4]Advanced Military Equipment, Dixon, Missouri, USA

[5]Army Research Laboratory, Huntsville, Alabama, USA

Abstract - *The development of a low-cost motion capture based landmine detection training system is presented in this paper. Two Nintendo Wii Remotes (Wiimotes) are used to form a stereo vision system to obtain the position and orientation of the simulated physical landmine detector. The position data is used to compute the distance between the detector and a virtual mine, and the distance is used to generate a synthetic sound based on a sound model. Metal detector sounds have been recorded for several types of landmines as well as for clutter. From the recorded sound data, an artificial neural network is implemented for sound modeling. During the training session, a trainee is asked to identify the landmine's type and buried location. The virtual training system can help train a trainee's ability to discriminate among different landmines and clutter, and help refine his/her sweeping skill. This system also provides additional visual cues based on a color scheme to improve the training effectiveness.*

Keywords: landmine detection, virtual training, motion capture, sound model, color scheme

1. Introduction

Landmines continue to pose a real threat to civilians and military forces worldwide. Current landmine detection technologies, including the military's AN/PSS-14 Mine Detector [1], serve as a mainstay in combating this threat. The AN/PSS-14 Mine Detector has proven an invaluable asset to today's war fighter and will remain so for the foreseeable future. However, field training of mine detection using AN/PSS-14 is tedious and expensive and may be dangerous in nature. Due to limited resources, many soldiers cannot get sufficient training to build and maintain the skills needed to operate this system. Additionally, some units have not been issued the AN/PSS-14 even though they are authorized to have this equipment. They are among the problems which cause long delays between training sessions. Research has suggested that, when mines are buried in a cluttered space, soldiers are able to find fewer than 65% of low-metal, anti-personnel mines in as few as 60-90 days after completing Army's 40-hour training program [2]. This problem is compounded by many types of mines the US military forces encounter. Each type of mine produces a different sound based on such variables as metal content, buried depth, and proximity to other high-metal objects.

The Sweep Monitoring System (SMS) developed by Carneige Mellon University has been on the market for several years [3]. In this system, two video cameras are used to track the movement of a target in 3D space. The target used in the SMS is a big, brightly colored foam ball mounted atop the detector. Hence the detector does not closely resemble a real landmine detector. The SMS provides the trainer on the computer monitor a graphic display of the detector sweep pattern to monitor whether the trainee leaves some gaps in the coverage area during a training session. Although showing gaps in the sweeping coverage is useful, the display of the sweep pattern in the SMS does not include information on the distances of the detector from the surrounding mines. Furthermore, a mine lane up to 2 m wide and 25 m long is needed on a plot of ground to allow inert training or simulated mines being buried at selected locations throughout the lane. Due to the above limitations, the SMS is limited in its capability, portability, and fidelity besides the fairly high cost of that system.

In this paper, we present the design and development of a low-cost virtual landmine detection training system to realistically replicate the functionality of the AN/PSS-14 landmine detector. Two Nintendo Wii Remotes

(Wiimotes), which are essentially low-cost IR cameras are used to build a portable and wireless motion tracking system to obtain the position and orientation of a detector head, which is mounted with four LEDs to track the detector head movement. Our simulated detector closely resembles the real AN/PSS-14 landmine detector, and our virtual landmine detection training system is more capable, portable, and realistic than the SMS. The paper describes how we use Wiimotes to develop a virtual landmine detection training system, with the ability to track the detector movement with a good accuracy of 2.7mm and a frame rate at 75 fps (versus the 30 fps provided by the SMS). Also detailed in the paper is how we develop a sound model utilizing an artificial neural network to generate a synthetic sound given the location of the simulated detector relative to the virtual mine for various types of landmines. In addition, the paper describes how we built a graphic simulation based on a color scheme to provide additional visual cues for the trainee. The software of the virtual training system is developed using the Higher Order Object-Oriented Modeling Technique.

2. Overview of the landmine detector simulation training system

The virtual landmine detection training system consists of two major components: a motion tracking system and a virtual training environment, as shown in Fig. 1. The motion tracking system consists of motion tracking sensors and motion tracking software; the sound model and the landmine detection simulator together form the other part of the virtual training environment. The system obtains the positions of four LEDs mounted on the detector head by computing the locations of their images on the IR cameras of two Wiimotes. These LED positions are used to compute the distance between the detector head and each virtual mine, as well as the orientation of the detector head. The distance data together with the mine type and the buried depth is used to generate a synthetic sound by computing the sound frequency and amplitude values based on a sound model that is developed utilizing an artificial neural network. A sound model is built for each type of landmine from the Metal Detector (MD) sounds recorded for several types of low-metal and high-metal landmines as well as clutter. The Ground Penetrating Radar (GPR) tones are also recorded. The rendering of sound is done using the OpenAL software, which is commercially available. In a virtual training session a trainee is asked to sweep the detector and to identify the specific landmine type and its buried location based on the computer synthesized sound. In addition to sound synthesis, the virtual training system uses a color scheme to generate a graphic simulation, which helps training by providing an additional sense through visual cues for the trainee. The virtual training system does not require any physical objects for the simulated landmines, and it requires only a 1.5m ×1.5m physical area for the trainig.

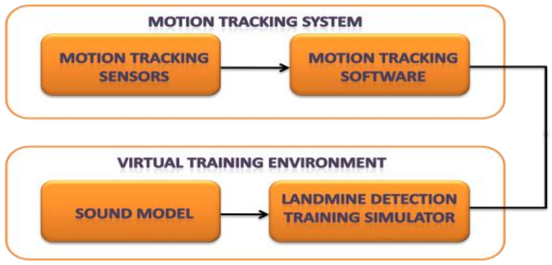


Figure 1: System architecture

3. Motion tracking system

A Wiimote based motion tracking system has been developed previously as described in [4], but applying the system to virtual landmine detection training is first discussed in the present paper. In the virtual training system, two Wiimotes are used to form a stereo vision system as shown in Fig. 2(a). Four LEDs are mounted on the detector head as shown in Fig. 2(b). The LEDs can be seen by the IR cameras of the two Wiimotes, and their images are used to calculate the position and orientation of the detector during a training session.

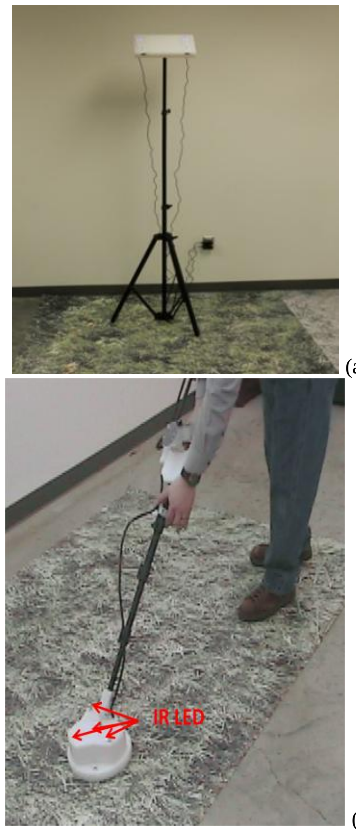


Figure 2: (a) Two Wiimotes mounted on a tripod, and (b) four LEDs mounted on the detector head

3.1 Motion tracking system principle

The basic principle of stereo vision is as follows. Assume a simplified configuration of two parallel cameras as shown in Figure 3. Also assume a straight line connecting the two cameras' optical centers being coincident with the x_c-axis. In the figure, point P has the coordinates (X_C, Y_C, Z_C) and projects to both cameras.

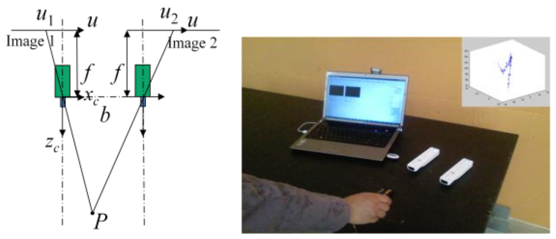


Figure 3: Principle of the system

The pair of image points u_1 and u_2 of point P on the two cameras are referred to as conjugate points. The difference between the image locations of the two conjugate points is the disparity $d=u_1-u_2$. The Z_C coordinate, which is the distance of P from the stereo vision system, can be calculated as

$$Z_C = \frac{f.b}{d} \tag{1}$$

where f is the focal length of the Wiimote, and b is the basis length (distance between the two cameras). Thus the position of the point can be calculated from the image data from the two cameras using the triangulation principle.

3.2 Motion tracking coverage

An important parameter in the stereo vision system is its tracking coverage, which is the overlap of the covered regions of the two Wiimotes. As shown in Fig. 4, the two Wiimotes are placed such that their centers are connected by a horizontal line. The overlapping area at the distance of H from this line is represented by the length L and width W. The length L and width W can be calculated as

$$L = 2H\tan\frac{\theta}{2} - b \qquad W = 2H\tan\frac{\alpha}{2} \tag{2}$$

For the Wiimote, $\theta=41^0$ is the horizontal view angle, b=10 cm is the basis length, and H is the height. When the Wiimotes are set 2 meters from the ground, the coverage is 1.6m ×1.8m.

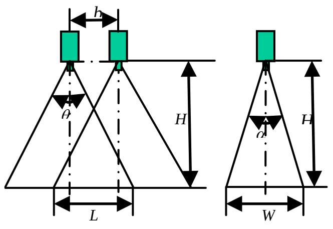


Figure 4: System coverage

Figure 5: Camera holder

3.3 Stereo vision system calibration

To get the position and orientation of the detector in 3D space, it is necessary to determine the intrinsic and extrinsic parameters of both cameras, which can be achieved through calibration of the camera system. A camera's intrinsic parameters should be determined only once ideally, but the extrinsic parameters must be re-determined if the relative position between two cameras has changed.

It is desirable to have the motion tracking system that can be calibrated in the lab before putting into the training and does not need to be calibrated again in its service life. To achieve this goal, the camera holder is designed as shown in Fig. 5. The two Wiimotes are in two slots 10 cm apart on a plate that is mounted on a tripod. The plate for holding the two Wiimotes can be adjusted to any angle with respect to the ground. Since the two Wiimotes are fixed very well and they do not move during the use, the stereo vision system needs to be calibrated only once.

A calibration plate with 36 LEDs as shown in Fig. 6 is used for calibration the intrinsic and extrinsic parameters of the Wiimotes. In the calibration process, the plate is moved to different locations in the field of view of the two Wiimotes. Figure 7 indicates a pictorial representation of the 36 LEDs at two different locations as seen by the two Wiimote in the camera coordinate system. The pixel coordinates of each LED at each location of the calibration plate are obtained by each of the two Wiimotes. The intrinsic and extrinsic parameters of the Wiimotes are determined using a calibration algorithm developed by Zhang [5], which has been coded by Bouguet [6] for the Camera Calibration Toolbox of MATLAB software.

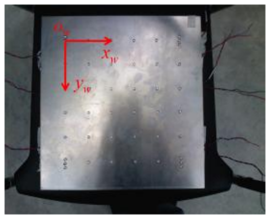


Figure 6: Calibration plate

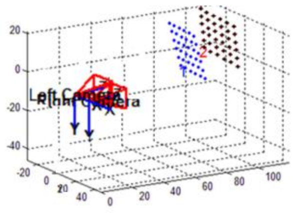


Figure 7: Data collected by a stereo system

3.4 Motion tracking accuracy

To investigate the tracking accuracy of the Wiimote based stereo vision system, we use a straight wand with two LEDs 19.5 cm apart. The wand is moved randomly in 3D space at the distance of 2-2.5 m from two Wiimotes and the distance between the two markers is calculated from the obtained image data by the Wiimotes for many locations of the wand. The RMS distance error observed is 2.7 mm, as shown in Figure 8.

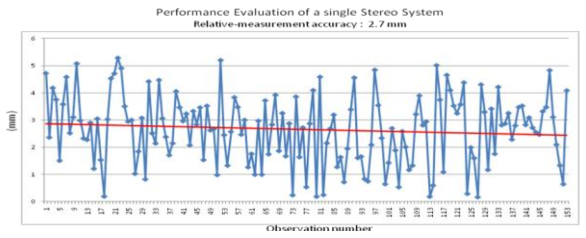


Figure 8: Measurement accuracy

with a single stereo system

4. Sound modeling

The primary objective of sound modeling is to synthesize the metal detector sounds a function of the distance between the buried mine and the detector head for each of the various landmines of interest. The research efforts consist of sound clips acquisition in the real world, sound data analysis, mathematical model generation, and sound rendering in a virtual landmine detection training system. The block diagram is given in Fig. 9.

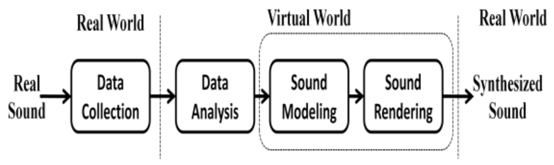


Figure 9: Sound modeling method and procedure

From analyzing the frequency content of the sound acquired by the real landmine detector using Fast Fourier Transformation (FFT) [7], we find out that all the peak frequencies have a “base” and several “harmonic” components as shown in Fig. 10. Therefore, it is possible to determine the audio frequency and amplitude as a function of distance, depth and target type.

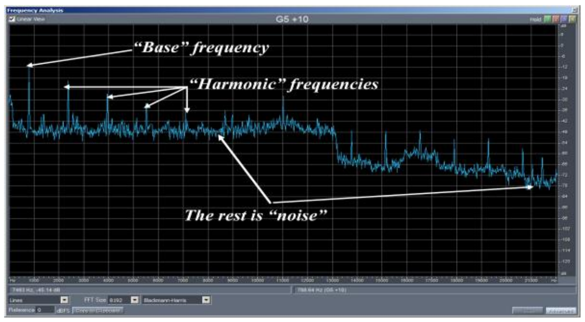


Figure 10: An FFT spectrum of the recorded sound

4.1 Sound data collection

Sound data is collected for several landmine targets. Each dataset contains around 500 instances, each instance representing a different radius and height of the detector head from the target. The sound amplitude and main frequency for each target are collected by placing the AN/PSS-14 detector at various distances from the target starting from 0 cm to 20 cm, with 0.625 cm increments. Sound data was also collected at various heights from 0 cm to 10 cm, with 0.625 cm increments. The dominant frequencies and amplitudes of the sounds obtained at different radii and heights can be read directly from the graphic user interface of the Spectral Surplus software. We investigate representing the sound frequency and amplitude using a mathematical model with the least mean squared error by utilizing an artificial neural network (ANN).

4.2 Utilizing artificial neural networks for sound modeling

The ANN modeling techniques are advantageous compared with conventional modeling methods. It has the adaptability necessary to represent a strongly nonlinear system.

The multilayer perceptron (MLP) is a popular ANN structure. In this technique the neurons are grouped into different layers. The first layer is the input layer and the last layer is the output layer. Between them is the central part of the neural network consisting of hidden layers. The number of hidden layers and the number of neurons at each layer can be varied. In our study, a two-layer (one hidden layer and one output layer), feed-forward, supervised ANN is utilized as shown in Fig. 11. Two features, distance and depth, together with a bias equal to 1, are input into the ANN as the input layer. The neural model is then trained to learn the input–output relationship from the training data. Training is used to determine the neural model parameters, i.e. neural network weights w*ij* and v*i*, where w*ij* represents the weight of the link between the jth neuron of the input layer and the ith neuron of the hidden layer, and v*i* represents the link between the hidden layer and the output layer.

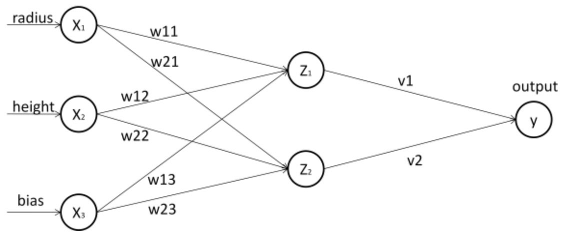


Figure 11. The multilayer perceptron

For a given input X, the output of a three-layer MLP neural network can be computed using the following equations:

$$y = \sum_{i=1}^{2} v_i \, \sigma(\sum_{j=1}^{3} w_{ij} \, x_j) = [v_1, v_2][z_1, z_2]^T \qquad (3)$$

where $z_1 = \sigma([w_{11} \quad w_{12} \quad w_{13}] \cdot X)$

$z_2 = \sigma([w_{21} \quad w_{22} \quad w_{23}] \cdot X)$

$X = [x_1 \quad x_2 \quad x_3]^T$

and $\sigma(\cdot)$ is an activation function. The overall nonlinear relationship between the input and the output is realized by various activation patterns of the neurons whose activation functions typically have the form of a smooth switch function, e.g., the sigmoid function:

$$\sigma(\alpha) = \frac{1}{1+e^{-\alpha}} \qquad (4)$$

For the input layer, x_1 is the horizontal distance (length) between the detector head and the virtual mine, x_2 is the vertical distance (height) between the detector and virtual mine, and x_3 is the bias, which is set to be 1. In order to train the vaules of w_{ij}, and v_i, the particle swarm optimization (PSO) method is utilized in our system since it has been shown to result in a better accuracy in comparison to the back propagation (BP) and the genetic algorithm (GA) techniques [8].

PSO is a stochastic global optimization method based on simulation of social behavior [9]. PSO exploits a population of potential solutions to probe the search space. It relies on the exchange of information between individuals (called the particles) of the population (called the swarm). Each particle adjusts its trajectory towards its own previous best position, and also towards the previous best position attained by any member of its neighborhood [10]. The previous best value of the particle position is called the pbest. It has another value called the gbest, which is the best value of all the pbest positions in the swarm. The basic concept of PSO is that each particle in the swarm moves toward its pbest and gbest locations at each time step [9].

In our study, PSO is used to train the MLP neural network. To train this neural network, all neuron weights are put together as one particle. Each particle is updated toward the global best position, which minimizes the difference between the neural network output and the desired value.

5. System implementation

5.1 Software architecture

The virtual Landmine Detection Taining System has been developed using the Higher Order Object-Oriented Modeling Technique (HOOMT) [11]. HOOMT provides a way to analyze, design, and specify structure, functionality, and behavior of the system in a hierarchical manner. It is based on level-by-level decomposition of objects, functionality, and behavior of a system. It starts with a context diagram which defines a boundary of the target system, which is different from classical object oriented analysis and design technique using the Unified Modeling Language (UML) [12]. HOOMT can be used to specify interactions between external users and the system at the highest level. The highest level of the model also describes the functions and properties of the system as a whole. At each lower level, the model is characterized by subsystems or objects with more specialized units and relationships. An object which has been decomposed to the lowest level is a primitive. The objects representing system components are ready to be implemented as hardware and software units.

The objects representing trainees include specification of their properties and operations. The objects representing landmines include their characteristics. The training scenarios for detection of landmines are modeled as use cases. A use case usually specifies a step-by-step process in which a trainee interacts with the virtual training system. It specifies a trainee's actions and responses to the system. The object-oriented models of configuration and simulation environment developed using HOOMT serves as a basis for implementation. They are mapped to hardware and software components of the system in the implementation. Interfaces between objects specified in the models dictate how actual hardware and software components of the system interact.

5.2 Selection of sound model

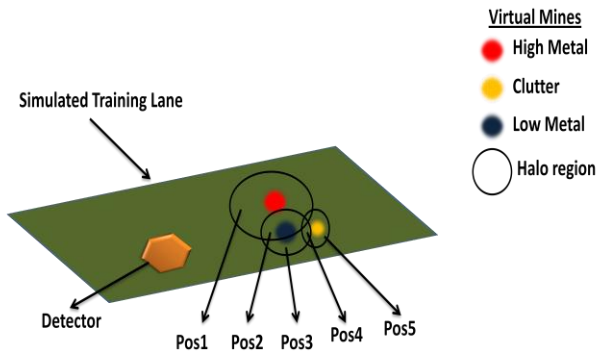


Figure 12: Landmines with intersecting halo regions

When the landmine detector moves, the distance between the detector and all the virtual mines are calculated. The shortest distance is identified and its information (distance, mine type, buried depth, etc.) is sent to the sound model to determine the corresponding frequency and amplitude. While selecting the sound model for auditory rendering, the high metal takes the highest priority, next the low metal and finally the clutter. Figure 12 shows a simulated training lane with overlapping halo regions of virtual mines. Based on the sound model selection algorithm, a high metal mine sound model is selected at Pos 1 and Pos 2, a low metal mine sound model is selected at Pos 3 and Pos 4 and the clutter sound model is selected at Pos 5.

5.3 Color scheme for simulation

During a training session, the trainee has to pay close attention the audio tone change in determining the center of the target. To improve training effectiveness, a color scheme is developed to provide the trainee with visual cues besides auditory cues, in order that the trainee can learn how to distinguish the change of auditory cues with the aid of visual cues. In the areas where there is no presence of metal, a cool color is presented with a trend to a warmer color as the distance from the detector head to the metal object decreases. It is a challenging task to maintain a framerate of over 60 frames per second (FPS) with continuous updating of objects in the graphic display and other overheads. An algorithm is developed to pre-compute all the expensive calculations before the start of simulation, thus improving the framerate.

The idea of the color scheme is to create squares consisting of different colors for the various pixels. The color of a square is based on the distance of the detector to the nearest mine. In our case, the color is modeled as a combination of green, red, and blue (GRB) values. Red and blue indices are fixed at 225 and 20 respectively, while the green index is varied from 0 to 255. The amount of green is proportional to the distance from the closest target. It is given by g = (255*actualDist)/maxDist, where maxDist is the radius of the gradient around a mine. If g exceeds 255, it is normalized back to 255.

To achieve high framerates, individual squares are colored by selecting pre-defined colors from the color list based on the above formula. This increases performance as all the colors are pre-calculated and cached during a call to build(). Hence, given mine positions, colors are generated with the amount of redness being inversely proportional to the distance of the detector from the mine. The number of colors equals the number of squares, with one color per square.

5.4 Training environment

Since the motion tracking system is based on IR cameras, it is not sensitive to the non-IR background. So the phyical training lane is designed as a vegetation background or a dessert background.

The difficulty level of the training can be specified in the setup procedure. There are 4 types of training lane configurations available from our system. They are: (1) Random-Easy: This lane configuration has a small number of landmines; (2) Random-Intermediate: This lane configuration has more landmines and clutter compared to the Random-Easy lane; (3) Random-Expert: This lane configuration has even more landmines and clutter. It also has some medal halo overlaps to make the mine detection more difficult; (4) Customized: This lane configuration is used by advanced users or for training a specific scenario such as foot printing. In this configuration the trainer is allowed to place as many landmines as he/she likes in the training lane to intensively test the trainees and familiarize them with different types of customized targets. Normarlly, 10 different configuration cells are set up in one trainig session. However, since only one cell is displayed at a time, the physical training area required is only 1.5 m ×1.5 m.

Figure 13 shows the graphics user interface where the actual training is done. It contains different options to show or hide landmines/clutter, detector height, detector speed, etc. In the figure, the virtual landmines have been set invisible. The figure also displays the visual cues generated with the color scheme.

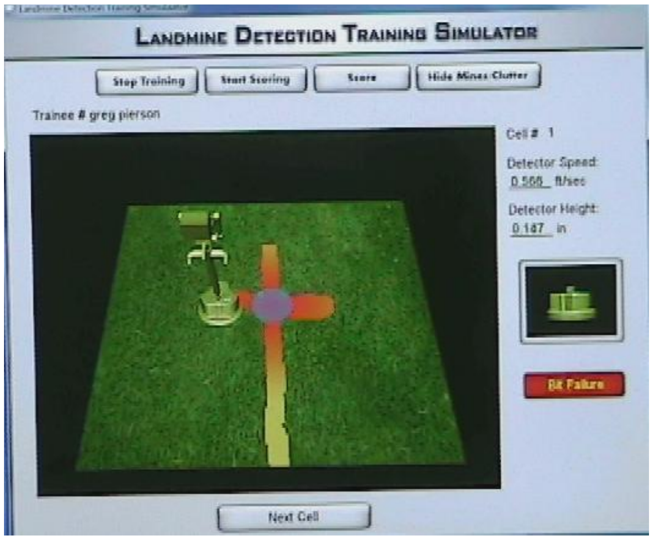


Figure 13: Simulation interface

6. Conclusion

This paper describes a successful development of a virtual landmine detection training system. By using Wiimotes for motion tracking, the training system can be made very inexpensive and easily portable, yet still with good accuracy. The system can generate the synthesized sound for the metal detector (AN/PSS-14) with mathematical models built from artificial neural networks based on the recorded sound frequency and amplitude data as a function of the detector position relative to the mine target, for various types of high-metal and low-metal landmines. The system can enhance a trainee's ability to identify a landmine's type and its buried location for different types of landmines as well as clutter. In addition to auditory rendering, the virtual training system includes a graphic simulation based on a color scheme to provide additional visual cues that can enhance the training effectiveness.

7. Acknowledgment

The authors would like to acknowledge the financial support for this research from Leonard Wood Institute with the contract LWI-191-029.

8. References

[1] Holbrook, D., "Mine Detection Moves into the Future: The AN/PSS-14 Mine Detector Requires a License," Army Chemical Review, pp. 48–50, 2007.
[2] Hancock, R.A., "Decay in AN/PSS-14 Operator Skill at 30, 60, 90 Days Following Training," Landmine Detection Research Center Technical Report, Department of Social and Behavioral Sciences, Lincoln University: Jefferson City, MO, 2006.
[3] Daniel, J., Hartman, P.E., Dean, O.C., "New training tools: enhancing mine detection performance", The Role of the Military in Mine Action Issue 8.1, 2004.
[4] Zhu, W., Chadda, A., Vader, A.M,, Leu M.C., Liu X.F., Vance J.B., "Low-cost versatile motion tracking for assembly simulation", Proceeding of 2010 international symposium on flexible automation, Tokyo, Japan, 2010.
[5] Zhang, Z., "A Flexible New Technique for Camera Calibration," IEEE Transactions on Pattern and Machine Intelligence, 22(11), pp. 1330-1334, 2000.
[6] Bouguet, J. Y., "Camera Calibration Toolbox for Matlab," http://www.vision.caltech.edu/bouguetj/calib_doc/index.html, 2011.
[7] Duhamel, P., Vetterli, M., "Fast Fourier transforms: a tutorial review and a state of the art," Signal Processing, Vol. 19, pp. 259–299, 1990.
[8] He, H., Leu, M.C., Zhu, W., Cheng, B., Liu, X.F., Pierson, G. and Davis, B.M., "Utilizing Artificial Neural Network To Model Sound For Virtual Landmine Detection Training," ANNIE 2010, St. Louis, MO, 2010.
[9] Liu, J., Xu, W., and Sun, J., "Quantum-behaved Particle Swarm Optimization with Mutation Operator," ICNC 2006, Part I, Springer–Verlag, pp. 959 –967, 2006.
[10] Kennedy, J., and Eberhart, R., "Particle Swarm Optimization," Proceedings of the IEEE International Conference on Neural Networks, Perth, Australia, pp. 1942–1945, 1995.
[11] Liu, X. F., Lin, H., and Dong, L., "High Order Object-oriented Modeling Technique for Structured Object Oriented Analysis," International Journal of Computer and Information Science (IJCIS), 2(2):74-96, 2001.
[12] Kulak, D. and Guiney, E., Use Cases: Requirements in Context, Second Edition, Addison Wesley, 2003.

An Approach to Maintaining Viewer Perspective in Interactive Virtual Tours

Thomas Carpenter, Gregory Doerfler, Thomas Way and Frank Klassner
Center of Excellence in Enterprise Technology
Department of Computing Sciences
Villanova University, Villanova PA 19085
thomas.way@villanova.edu

Abstract - *While virtual tours have been around for some time, the ability to get a true sense of the location has not been adequately developed. As part of a partnership between our department and the Vatican, we have created a virtual tour of the Sistine Chapel using multiple locations and interactive menus to allow users to feel as though they are actually there. Not only does the virtual tour give users a true sense of the place, but also it offers information about the chapel as well as the biblical history of paintings on the wall. The innovative aspect of this tour is the degree of attention given to maintaining viewer perspective at all times, so that movement appears both smooth and intuitive. In this paper, we discuss the overall development process of this virtual tour, and focus on the perspective maintaining techniques that were devised and evaluated.*

Keywords: Virtual Tour, Virtual Reality, Perspective, Stitcher, Krpano, XML.

1 Introduction

As part of an ongoing partnership between our home institution department and the Vatican, we created a virtual tour of the Sistine Chapel. This undertaking was a follow-up to a Vatican internship and study abroad experience in Rome during the fall semester of the junior year of a two student team (Carpenter and Doerfler). As a part of the internship, the team assisted Villanova research faculty with photographic shoots at various locations in and around the Vatican. The most notable location was the well-known and highly recognizable Sistine Chapel. At these photographic shoots, a special camera and rotating stand were used to take hundreds of images in the complete 360° panoramic range of directions from a central, fixed location, or in the case of the Sistine Chapel which has a central half-wall that does not make a single location practical, two fixed locations.

Once the photographic images data was gathered, the images were digital "stitched" together into a seamless, 360° panoramic view. To perform this image concatenation, a software program called Stitcher Unlimited was used. Stitcher Unlimited (stitcher.realviz.com) was used to stitch together the images from the Sistine Chapel into two panoramas, each centered on a fixed point within one of the two sides of the Chapel. These panoramas were then edited to remove minor imperfections in the stitching and developed further using the interactive 3D panorama viewing system and scripting language, Krpano (krpano.com). The result of this extensive development effort was the production of a 3D interactive tour that incorporates context-sensitive menus, visual and contextual "hotspots" and an intuitive navigation system. This combination of features and functionality, together with the innovative perspective maintaining panning algorithm we devised and the uniqueness of the Sistine Chapel as a setting, potentially marks this project as a significant contribution to the field of virtual tours.

2 Virtual Tours

Virtual tours work by offering a location (an axis point) from which the user can "look around" – changing the perspective, rotating the camera, and adjusting the zoom. Clickable "hotspots" link two or more locations together and allow the user to move to the next location.

What makes a virtual tour a tour of some location and not simply a single-location panorama is this ability to transport from one specific location to another within the boarder context of the location being virtually visited. In this project, a novel algorithm was devised that determines the visually "best," or most intuitive and visually pleasing, path when moving from specific location to specific location with the tour's virtual space. Additionally, an informative, hotspot-sensitive, interactive menu system was crafted to assist the user in navigating the space to meaningfully explore and learn about the many historical, architectural and artistic features within the Sistine Chapel.

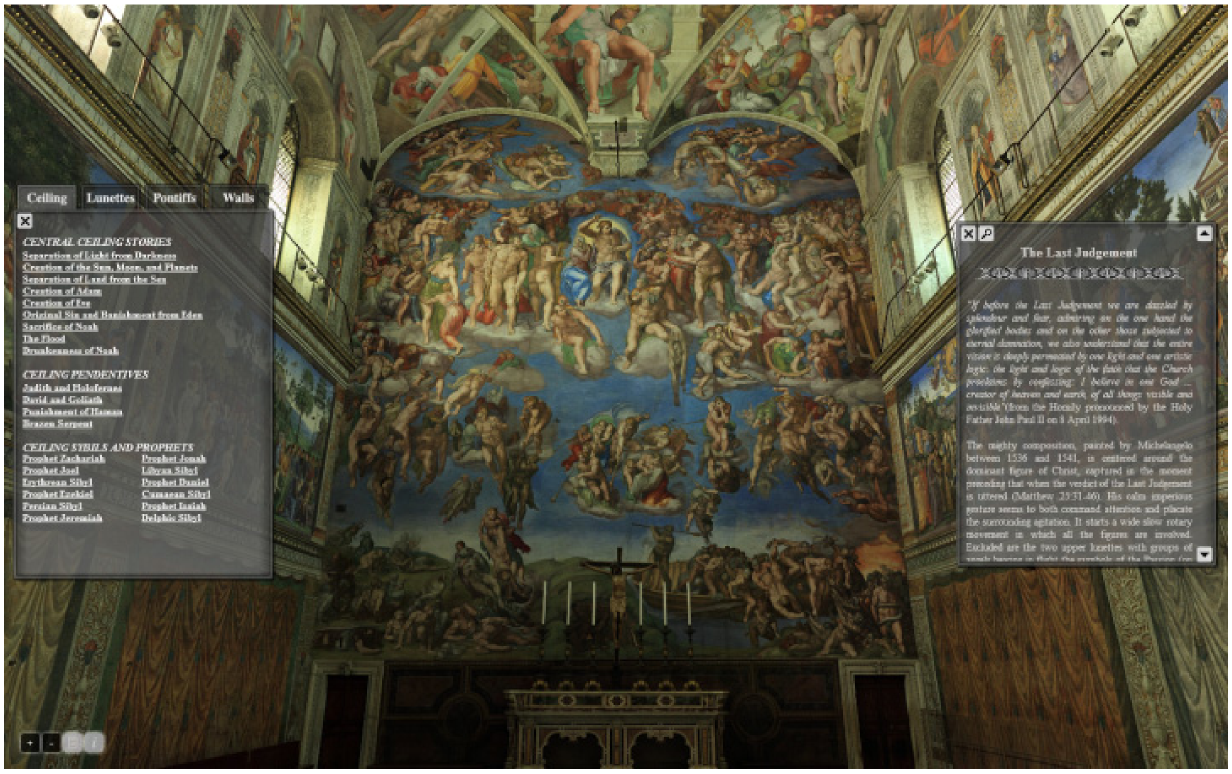
2.1 Photography

In order to produce panoramic images of the Sistine Chapel, a 360° camera rig and rotational mounting device were used to manipulate the camera and accurately capture the complete range of vision. The operation of this camera rig and mount was mechanized to produce highly consistent results. Once the rig had captured some pre-determined number of images in a 360° range (for example, 15 images), the camera would automatically adjust by an increment, either tilting up or down, and take another 360° rotation of images. The result of this exhaustive photography was two sets of over 200 very high quality images from the two fixed location, with each of the sets containing significant overlap within the collections of images.

Responsibility for setting up and configuring the image capture equipment fell to the research faculty involved due to their experience with the equipment. During the setup procedure, particular attention had to be paid to positing the camera properly on its mounting stand so that the back of the lens would sit precisely over the axis of rotation. If this careful calibration was not performed, images could be skewed slightly and may not line up later during the stitching procedure, producing a low-quality result. Extreme care was also taken in careful setting the focus and exposure levels of the camera so that images would be as clear as possible. The camera has manual focus capability only, with no ability to adjust focus during an image capture session, so a single focus setting had to be carefully chosen to produce the best overall results for the entire image-capture operation.

2.2 Stitching the Panoramas

The program "Stitcher Unlimited" was used to piece together the 200+ images that resulted from the image capture sessions inside the Sistine Chapel. While Stitcher does much of the work itself through "auto-stitching," its graphical user interface and supplementary tools proved helpful when the results of stitching contained minor imperfections and had to be manually repair using the available tools.



The Last Judgement

"If before the Last Judgement we are dazzled by splendour and fear, admiring on the one hand the glorified bodies and on the other those subjected to eternal damnation, we also understand that the entire vision is deeply permeated by one light and one artistic logic: the light and logic of the faith that the Church proclaims by confessing: I believe in one God ... creator of heaven and earth, of all things visible and invisible" (from the Homily pronounced by the Holy Father John Paul II on 8 April 1994).

The mighty composition, painted by Michelangelo between 1536 and 1541, is centred around the dominant figure of Christ, captured in the moment preceding that when the verdict of the Last Judgement is uttered (Matthew 25:31-46). His calm imperious gesture seems to both command attention and placate the surrounding agitation. It starts a wide slow rotary movement in which all the figures are involved. Excluded are the two upper lunettes with groups of

Figure 1. Screen capture of user interface, showing "The Last Judgment" in the Sistine Chapel.

The final output of this process was a massive, uncompressed digital image that is a flattened version of the entire 360º virtual tour. This uncompressed image was next processed using a plug-in tool of the Krpano program, which split the single tour image into over 500 images of many differently-sized rectangles. The Krpano program performed this splitting and selected rectangle sizes in order to best optimize the eventual online loading time of the finished, interactive, virtual tour product.

As a result of the vast quantity of images, and because the sets of images contained so much overlap, there was a significant amount of redundancy that needed to be managed. The solution to this redundancy problem involved selecting every second or third image in a sequence, which reduced the quantity of image input to Stitcher while providing ample information to form the complete 360º virtual tour image.

2.3 Interactivity

The next step in development of the virtual tour was incorporating interactivity, taking into account the needs and common behaviors of the viewer [6]. The initial task was to be able to view both of the single-location tours separately in the Krpano virtual tour viewing software. Internally, the tour is viewed by loading the web content for the tour into a web browser with also loads a corresponding file stored in the XML format that contains all content and commands to support the interactivity.

With the Krpano software, plug-ins were used such as the Options plug-in and Editor plug-in to obtain the coordinates for certain figures and features on the walls of the virtual tour and create interactive hotspots for these locations. An initial design component was the creation of a left-hand side, slide-out menu that listed wall figures and features. This initial menu was later abandoned in favor of a tabbed, fade-in and fade-out menu system that was more visually appealing.

Because the Krpano software provides no menu building plug-in, the menus were created by combining the capabilities of several plug-ins including the Text Fields, Images and Buttons plug-ins. The Text Fields plug-in provides limited HTML and CSS support for layout, so it was necessary to use multiple Text Fields to produce the illusion of a columnar layout.

Using Scroll Bars and Text Fields, a right-hand information menu was created that pops up when the user clicks on a hotspot, providing relevant information on the figure, or figures, of interest. On this menu is a magnifying glass icon, which when clicked gives the user a “better view” of the hotspot. Since the user may or may not want hotspots to be enabled, they are disabled by default but can be turned on or off using a toggle button on the bottom left of the user interface screen.

Much of the look and feel of the interactivity is the result of the algorithm that was designed for this specific project, which seeks to maintain visual perspective and make movements feel natural to the viewer. The algorithm automatically determines parameters such as direction of motion, degree of zoom out and zoom in for producing a better viewing experience, and how best to go through the center door of the half-wall when transitioning from one fixed, central location to the other location within the Sistine Chapel virtual tour.

2.4 Algorithm for Maintaining Perspective

The most significant contribution to the field of virtual tour creation is the development of perspective maintaining interactivity that allows the user to get a true sense of the location. While many other virtual tours attempt to use multiple locations and hotspots, the common approach is to produce a visual blur while transitioning from one area to another [7]. This blur effect prevents the user from maintaining a true sense of the movement.

For this project, an algorithm was devised that attempts to produce movement such that the user loses perspective throughout the viewing experience, a desirable feature [4,5]. The algorithm was developed in a series of steps. Initially, a basic algorithm was created that produced simple movement from a current hotspot to the desired hotspot. This was adequate in some cases, although if the viewer was already zoomed in closely on a feature, the resulting movement was confusing and dizzying and viewers found themselves feeling lost in the virtual space. The result of this observation was a more generalize movement algorithm that calculates a desired movement path proportionally based on the distance to the current and target hotspots (Figure 2).

The perspective maintaining motion algorithm works as follows. First, the x and y coordinates of the current position and the position of the hotspot are acquired.

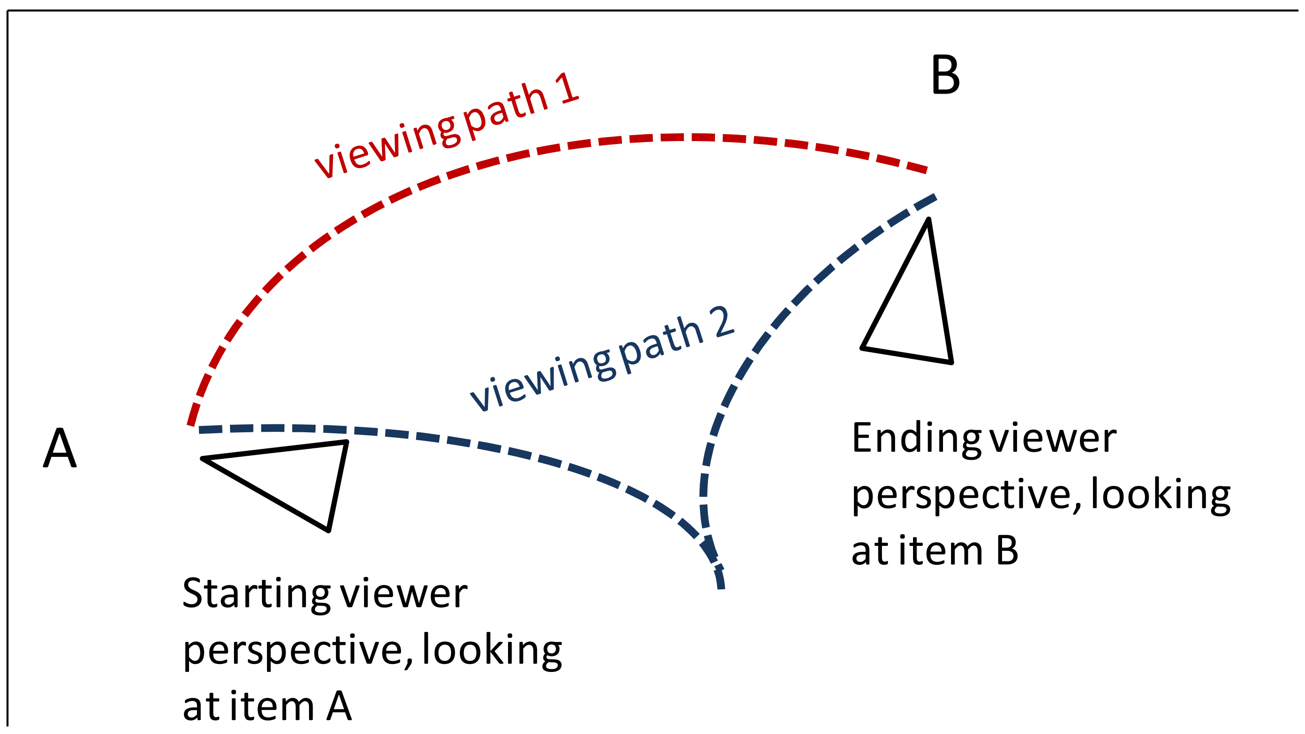

B
viewing path 1
viewing path 2
A
Ending viewer perspective, looking at item B
Starting viewer perspective, looking at item A

Figure 2. Comparison of naive, sweeping movement of the original viewing path 1 from viewing item A to viewing item B, which produces a less-desired viewer experience, with the improved back off, sweep, and zoom in movement of viewing path 2, which produces a consistent viewer experience that maintains perspective within context of the larger viewing area.

The Krpano viewer has a built-in function to get the current location, but the coordinate system can produce negative positions if the user's movement has been counter-clockwise. Therefore, if a position is negative, it is simply normalized by adding 360 to it, since with Krpano a location is guaranteed to be in the range of -360 to 360.

Next, the desired hotspot is compared with the bounds of each side of the Sistine Chapel tour to determine if it is in a different panoramic location.

The information for each hotspot is stored by simply creating an extra variable within the interactivity logic called **panoLoc**. If a hotspot is determined to be in the other location, motioned is created so that the viewer zooms toward the central door first before being panned to the new hotspot location.

The specifics of calculating this movement were determined to be that if the user is zoomed-in past a set zoom value of 40 and the current position is not already near the door, then first the view is zoomed out to a set value of 15 and then movement to the door is produced. Otherwise, the view can be moved directly to the door and then zoomed-through as the corresponding XML interactivity file for the other location is loaded.

If the desired hotspot is within the current panoramic location, then an appropriate zoom value must be calculated for use in the movement. Since the possible movement to a hotspot can be clockwise or counter-clockwise, the appropriate distance is always calculated to be the shorter rotational distance. Therefore, the x distance to move in degrees, or Δx, will always be between 0 and 180.

Because of this restriction, x must be normalized based on whether it is either greater than 180 or less than -180. The calculation is performed as follows:

```
If the value of x is greater
than 180, then calculate Δx to
be 360-curX+gotoX.

If the value of x is less than -
180 then calculate Δx to be 360
– gotoX + curX.

If the value of x is any value
in between -180 and 180, then
calculate Δx to be curX-gotoX.

Δy is calculated simply by
calculating curY-gotoY.
```

The x distance is calculated, and a ration of distance versus the distance to the hotspot is determined. The zoom values range from 15-55, making a range of 40. Multiplying the distance ratio by 40 produces a zoom value in proportion to the calculated distance. Because the zoom values start at 15 and not 0, the zoom value must be normalized by adding 15 to it to ensure that is in a valid range.

A final observation regarding zoom and movement was that no zoom is needed if the distance from the current hotspot to the desired hotspot is some minimal distance. Through objective observation and experimentation with a variety of values, it was determined that any rotational distance between hotspots that is less that 60° does not require zooming out in order to maintain visual perspective. For motion further than 60°, the calculated zoom value should be used as it produces a visually consistent experience for the viewer.

This carefully designed movement with the interactive virtual tour provides virtual visitors with a real sense of being within and looking around inside the Sistine Chapel. Using the perspective maintaining algorithm, movement throughout the Chapel is natural and viewers maintain a consistent sense of space and location.

3 Challenges

While developing the virtual tour, several issues were encountered that necessitated adaptations to the approaches that were used. First, an observation that designating regions that were greater than a certain size as hotspots did not produce the desired 3D effect. The solution was to divide many of the hotspots into groups of "linked" hotspots. The added benefit of this approach is that it was possible to mask out protruding objects such as railings and light fixtures. In this way, moving the mouse over any of the "linked" hotspots automatically invokes the presentation of the information associated with any of the associated "linked" hotspots so that information is presented to the user in a uniform and consistent manner.

Second, naming conventions were devised to produce consistency throughout the project. Because of the scale of the project, work was split between developers and when different naming conventions were used for variables or hotspots, significant effort was required to reconcile the functionality of individually produced components. By making naming uniform, unsurprisingly, the pieces of the project produced in parallel were integrated much more easily.

Third, it was realized that including all hotspot and data information into a primary XML interactivity file was cumbersome and could potentially reduce loading performance. The solution was to create a separate data XML file and two hotspot XML files, each of which were "included" into the main interface content as needed.

Finally, it was determined that incorporating JavaScript coding within the Krpano viewer scripting led to unreliability of the menu navigation system. The decision was made to program the interactivity purely within Krpano, which introduced additional challenge but resulted in a more dependable and stable result.

4 Conclusions

This project produced some significant contributions to the area of virtual tours, both in terms of the unique subject (the Sistine Chapel) and perhaps most importantly an algorithm for maintaining perspective when moving through a virtual tour space. While Krpano makes use of a developed scripting language, it is still evolving and therefore lacks some desired functionality and corresponding documentation.

The lack of working examples of other Krpano-based tours was one of the biggest challenges that made building the interactive menu system especially difficult. In our experience, this project has made the most extensive use of Krpano programming and functionality by a significant margin, and it is hoped that others producing such tours can benefit from this experience.

For anybody who has experienced a virtual tour, it is evident that interactivity of the tour enhances the experience for the user. It allows users to feel the real sense of the location as they, in this case, seamlessly move through the Sistine Chapel. Maintaining user perspective during automatic movement adds to the user experience, combining ease of navigation with a sense of being present in the tour.

5 Future Work

While the current version of this interactive tour is good according to users who have tried it, there are improvements that could be made to movement functionality and loading time. Once deployed, the tour is likely to draw considerable traffic on the Internet, so more efficient techniques for loading the visual and data content and for moving throughout the virtual tour are needed.

Mobile application of programs seems to be rapidly moving to many different parts of software development and programming. Another extension of this virtual tour project would be to develop it for a mobile platform such as the iPhone or Droid. Until recently, Krpano used Adobe® Flash™ in its rendering of the virtual tours. Recently, Krpano added support for using JavaScript and CSS in XML files in order to display a virtual tour. While this does not apply to the use of plug-ins (such as menus) and hotspots, it could increase opportunities for improved interactivity in the future, including mobile platforms.

6 References

[1] Andolina, S., A. Santangelo, and A. Gentile. "Adaptive Voice Interaction for 3D Representation of Cultural Heritage Site." Intelligent and Software Intensive Systems, 2010.

[2] Leigh, Jason, Andrew Johnson, Christina Vasilakis, and Thomas DeFanti. "Multi-perspective Collaborative Design in Persistent Networked Virtual Environments." Evl | Electronic Visualization Laboratory, 2010.

[3] User Documentation. Krpano.com, 2010. URL: http://krpano.com/docu, Accessed: Dec. 12, 2010.

[4] Niklas Elmqvist, M. Eduard Tudoreanu, and Philippas Tsigas. Tour generation for exploration of 3D virtual environments. In Proceedings of the 2007 ACM symposium on Virtual reality software and technology (VRST '07), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 207-210, 2007.

[5] Niklas Elmqvist, Mihail Eduard Tudoreanu, and Philippas Tsigas. Evaluating motion constraints for 3D wayfinding in immersive and desktop virtual environments. In Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems (CHI '08). ACM, New York, NY, USA, 1769-1778, 2008.

[6] Bastanlar, Y.. User Behaviour in Web-based Interactive Virtual Tours, In: *Proc. of 29th International Conference on Information Technology Interfaces*, Dubrovnik, Croatia, 2007.

[7] Eun-Young (Elaine) Kang, Ilmi Yoon, "Smooth Scene Transition for Virtual Tour on the World Wide Web," *Sixth International Conference on Computational Intelligence and Multimedia Applications* (ICCIMA'05), pp. 219-224, 2005.

Teaching Digital Camera Forensics in a Virtual Reality Classroom

Associate Professor Eamon P. Doherty Ph.D
[1]School of Administrative Science, Fairleigh Dickinson University, Teaneck, New Jersey, USA

Abstract - *A 2009 study shows that 77% of American households have a digital camera. Some of these digital cameras may be used by criminals at crime scenes to document their deeds for their own purposes or for proof of deed to collect a fee from others. Law enforcement or private security personnel may need training in how to collect existing or deleted pictures from a digital camera without compromising the data. Traditional classroom instruction demonstrating and explaining such techniques has worked well for students to the present time. However; there is a growing need for this type of training for students who are military personnel deployed overseas, foreign law enforcement personnel, and geographically diverse students with limited funds for travel. Virtual classroom instruction appears to be a cost effective solution to teach digital camera forensics to students around the globe who cannot get this training conveniently where they live or work.*

Keywords: VR classroom, digital camera forensics

1 Introduction

I am the Director of a Cybercrime Training Lab at Fairleigh Dickinson University. I often receive emails inquiring about in person and online classes with regard to the subject known as digital camera forensics. Some of the requests are from: divorce lawyers, law enforcement officers, private investigators, and people who have gone on vacation and lost all their digital pictures on their camera due to operator mishandling or hardware failure. Many of the law enforcement officers work rotating shifts and can only attend online due to the unpredictability of their work schedule.

On January 28, 2009 Kyle Schurman said “According to a recent study from the Consumer Electronics Association (CEA), 77% of American households now own at least one digital camera” [1]. People who take digital pictures often push the wrong buttons and delete pictures that are important to them. This segment of the public would like to be able to recover sentimental photos from a cruise or a relative’s wedding. This process is merely data recovery.

Then there is the law enforcement investigation where digital camera evidence must be gathered without altering the evidence. Such a case may involve a murder or rape. A rapist may use a digital camera to take a picture of the victim. A United Kingdom newspaper called the Daily Telegraph reports, “Lord Judge said “a pernicious new habit has developed” and photos were often taken either to show off, add further humiliation or to blackmail [2]. Law enforcement officers who arrest these rapists need to learn how to preserve and recover this evidence from a digital camera so that they may proceed to successfully prosecute the aggressor and get justice for the victim. For small town law enforcement officers who are far away from Regional Computer Forensics Labs (RCFL), we can now see the urgency for on demand digital camera forensics classes.

Divorce lawyers often receive camera cell phones or digital cameras with pictures of the unfaithful partner’s paramount. The cell cameras phones may also hold hostile text messages between husband and wife which are sometimes called “nastygrams.” The lawyer needs to be able to obtain this digital evidence, process it, and prepare it for a report to use for court or arbitration.

2 Moving an in person class to a virtual classroom

In my opinion, the digital forensics class that I teach in person would work well if moved online to a virtual environment such as Second Life. Some of my students are comfortable with virtual reality (VR) and have used VR environments or have seen family members use such VR gaming environments as “Everquest” and “America’s Army.” America’s Army is used by many young people as a recreational multiplayer online game that requires groups of people to work on a task as a team [3,4]. Many young veterans who are returning to school used an advanced version of America’s Army to train and in my opinion could effectively use avatars and a virtual environment to train about digital forensics.

Let us now consider the implementation of my digital camera forensics class to a VR classroom. Students could first register online at the University continuing education website. Then they would be instructed to create an avatar in Second Life and download Second Life on their laptop. My class could meet in a Second Life private island classroom for digital camera forensics. A variety of digital cameras could be shown to the students. I could then connect a cable from the laptop to a USB write blocker. Then I could connect a cable from the USB write blocker and to the digital camera and demonstrate how no temporary files or evidence would be changed on the digital camera. This concept of using a write blocker to prevent changing the data on an attached device is difficult to do if one only reads about it or sees a picture. In the virtual classroom, I would also prepare my examination machine by running antispyware, antivirus software, and disabling all connectivity to the computer so no evidence tampering occurred.

Then my virtual teacher avatar would demonstrate how to fill out a chain of custody form that takes the evidence from the time of digital camera seizure until the time it appears in court. If the camera had Wi-Fi or Bluetooth, I would keep it in a Faraday Bag to demonstrate that I was blocking all forms of potential connectivity to other networks or devices.

Then my virtual classroom teacher avatar would use Susteen Secure View or Paraben's Device Seizure to seize and organize all the data from the digital camera cell phone. A report would be generated with all the pictures having MD5 digital signatures. The virtual teacher would burn copies of the reports on CDs for both the prosecution and defense.

The virtual classroom teacher could then demonstrate how deleted data from a standard digital camera (not a cell phone camera), can be undeleted as long as it was not written over with new pictures. This type of camera often uses a FAT file system just like the ones found in Microsoft Windows or DOS. Perhaps some basic operating systems concepts could be displayed on a virtual blackboard. The avatar of the teacher could demonstrate that when a file is deleted, the first letter of the filename in the file allocation table is replaced with a sigma. If the file is restored, the sigma is often replaced to an underscore and the operating system will recognize the location of the file and display the picture.

The virtual teacher could then give a presentation how to recover the pictures by using a tool such as RecoverMyFiles, AccessData's Forensic ToolKit (FTK), or Guidance Software's Encase. The pictures would most likely be recovered if they were not overwritten. Sometimes the thumbnails can be recovered even if the full high resolution picture is deleted and overwritten.

The students could then be given a digital camera phone and digital camera in the virtual classroom and then asked to extract the data and create a report. Then they could take a test. If the reports from the devices and the results of the written tests are satisfactory, a credential could be given. This credential could then be downloaded and printed in the student's office in the real world.

2.1 Successes with virtual classrooms in Second Life

Some schools have a mix of part time and full time graduate students. Some are first responders while others are not and aspire for a career in emergency management. Dr. Bob Berry at the University of North Carolina has had success using a virtual classroom in Second Life to teach emergency management, conduct exercises, and then investigate what mistakes were made [5]. The virtual classroom takes time to set up but students like being able to talk to one another and interact with equipment and people in this novel environment. This appears to be a good learning modality for visual learners.

I had sat in Dr. Berry's presentation about using Second Life as a learning platform to teach emergency management and conduct exercises at the 13th annual Emergency Management Higher Education Conference in Emmetsburg, Maryland. In my opinion, Dr. Bob Berry had success because he was prepared with materials to help students create characters in Second Life and encouraged them to be persistent and learn to use the avatars the characters in the virtual environment. Whether one has distance learning students using Second Life or one has a group of students using Second Life in a computer lab, it is my opinion that it is necessary to have some type of tech support to help the people get started and then have a skilled and patient teacher to conduct the class.

2.2 Moving the class to a traditional online learning environment

We said earlier that some of my students are comfortable with virtual reality and have seen family members use such gaming environments as "Everquest" and "America's Army." However; many students I teach privately said that do not know what virtual reality is and would not select a class that used it. Most of those students said that they preferred an in person class but some said that online learning a text based virtual campus such as Blackboard was acceptable.

It is my opinion that the digital camera forensics class could be a success in Blackboard if students downloaded PowerPoint slides and took online tests. They could also have a link to a manufacturer's website for downloading a trial version of digital forensic software that they could use to examine their digital camera or digital camera cell phone. After successful completion of a task and test, they could obtain and print a certificate with their name on it. A very similar type of online learning and credentialing is done by FEMA [6].

3 Teaching digital camera forensics in person

I have taught digital camera forensics in person in the Cybercrime Training Lab at Fairleigh Dickinson University in New Jersey for approximately three years. In class, we first start with a display of some of the types of digital cameras available on the market. Then students are exposed to some of the digital cameras that are commonly embedded in devices such as binoculars and telescopes. Next we proceed to digital cameras found in cell phones.

In a digital camera forensics class, we then discuss the metadata that is in the digital picture. In Windows 7, one can right click on a picture and choose the properties tab. Then one can find information on the camera type and model. If the pictures are on a read only CD where they will not be altered, a non forensic tool such as Google's Picasa 2 or Picasa 3 can be used to show us the details about the lens and focal point of the camera.

One of the techniques that has been successful in getting the students enagaged in the conversation of digital camera forensics is when I show a 1945 digital picture of F. Castro and a former student's father at Okinawa (see figure 1.). I aquire the picture from my digital camera and ask the students if this picture was from the camera. The students will say that there were no digital cameras in 1945. Then I ask how it got on my computer. A younger student will often tell me that the digital camera is similar to a mass storage device connected to a digital / optical lense and that many files other than pictures may be stored there.

Then we will examine the metadata of the picture and notice that the resolution of the picture is not supported by digital camera. Someone else with my camera specifications may say that the picture format, GIF is not supported by my camera. We may examine the metadata further and learn that there is no camera make and model but actually a scanner model. A student will ask if I have such a scanner and I will tell them that I scanned a photograph and transferred it from my desktop to my digital camera.

After the technical conversation, we often go to the non techical discussion about the picture. I ask if F. Castro is Fidel Castro or could it be Frank Castro or Francis Castro. The students then talk about how they would investigate if the picture might be of President Fidel Castro of Cuba. I ask how old is President Castro now and how old was he in 1945. Then I ask them how old does the man in the picture appear to be.

Then students ask what President Fidel Castro might have been doing in 1945. We look online at some biographies and learn that he was in law school. Then we discuss how difficult law school is and that one cannot take much time from it and it most likely is not him. Students will also note that the biographies never say he visited Japan as a young man.

Then another student may notice a wedding ring in the picture and ask if there was a Mrs. Fidel Castro. Most students never heard of one and the online biographies do not mention an early marriage. The analysis of the picture gives students an understanding that there are many things to consider about a picture and that research plays a part in an investigation.

Another student may notice that the man in the picture has a beard but that it is shaped differently than other pictures of Fidel Castro. Another student might ask Fidel might want to visit Japan.

Some students might notice that the dog tag of the man in the picture looks different than the American ones and the mystery deepens again. They wil flip back to the technical discussion again and suggest running a facial biometrics program on the picture of F. Castro against a known picture of Fidel Castro at age 19. Other students who are investigators have suggested to write to the Cuban Embassy and ask.

I feel the same picture and discussion would work well in a virtual classroom and help students discuss technical and non technical considerations.

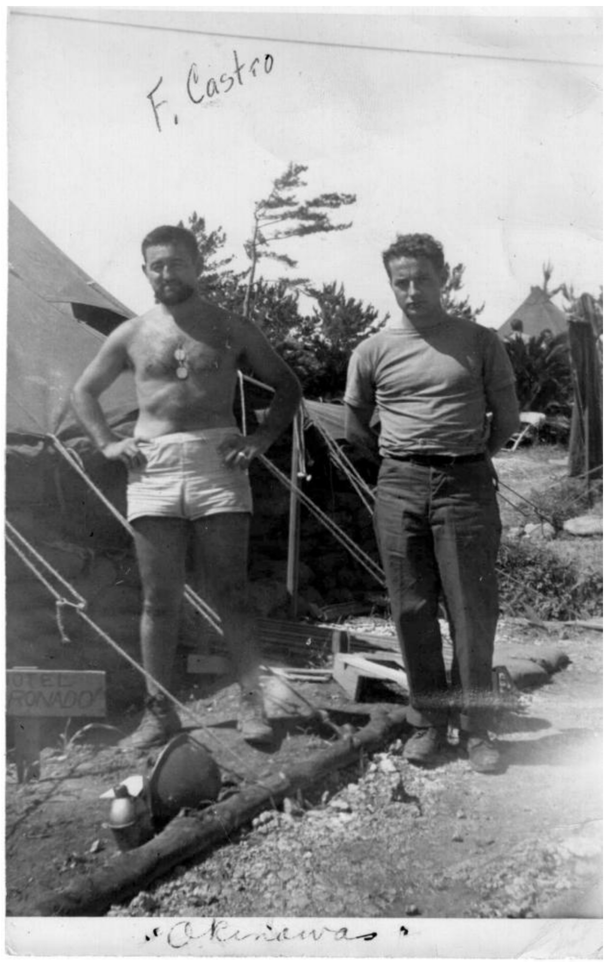


Figure 1 – F. Castro at Okinawa in 1945.

4 Credentials for the digital camera forensics teacher or practitioner

If someone is going to teach computer forensics, I would like to see a doctorate in computer science with some forensic certificates too. In my opinion, this type of teacher could go into the technical aspects of file recovery and discuss where files are stored and where parts of files may be found in the Microsoft Windows Swap File. However; a lawyer with some computer science and computer forensic training could also be a good professor because he or she could emphasize the legal aspects of examining a camera, especially if it was taken by an angry spouse without consent of the owner.

A computer crime detective with a forensic psychology doctor's degree could also be good because that person could also discuss the motivations of those who misuse digital cameras. The psychological aspect is important because it is important to know the patterns of a criminal and the mindset so that the investigator may understand the intent. Intent is an important component of crime.

I often tell people to become a Certified Computer Examiner (CCE) because it is a recognized credential in digital forensics [7]. The Microsoft operating systems that one learns about for the CCE examination uses the FAT 32 file system often found in digital cameras. This means that the file recovery techniques used for that credential can be used for the digital cameras. One should also try to find classes in digital camera forensics. One can also get Guidance Software's Encase certification which many people value for a wide area of digital forensics.

5 Suggested further education for digital camera forensic specialists

I would also suggest that people take any online or in person classes in Operating Systems and File Systems. These classes are found in most computer science departments. Brian Carrier has a very detailed book on file system forensic analysis which helps one understand how the operating system handles files that are being allocated to space or deleted [8].

The person who is the student needs to have a resume with his her education and work experience ready to discuss with the teacher. That resume should include formal university training, continuing education, and seminars. Then the student should also write briefly about his or her work experience including any investigations that he or she took part in.

The teacher can then see where the student needs more knowledge and then suggest further reading, webinars, or other methods of education to build that area of knowledge up. The teacher should be looking for report writing skills since the presentation of evidence for court or corporate officers is so important to an investigation. The teacher should also look for technical skills such as a knowledge of file systems, operating systems, and basic networking. The technical skills should also include the use of free Linux based tools that help the investigator acquire an image or complete copy of all the storage media located in the camera. Lastly, the teacher should evaluate the student's knowledge of legal principles that are relevant to the geographic area where the student may participate in investigations.

In my experience, many law enforcement students are often strong in report writing, investigating skills, and legal principles. However; they often need more technical knowledge such as operating systems and the advanced use of digital forensic tools.

In my experience, computer scientist students have excellent technical skills but need to learn more skills in conducting an investigation, interviewing people, and in legal principles. These students often need to be reminded that not following legal protocols, not having a proper chain of custody form, or not having properly licensed digital investigative tools can compromise an investigation.

The online teacher could conduct this type of conversation in a private chatroom, by telephone, or in a special section of a private island in a place such as Second Life. In certain circumstances, it might be best to conduct the discussion by email so that the student has a written statement of strengths and weaknesses and then can document ways to address those concerns. The point is that there are so many telecommunication modes that can work in tandem with the online virtual classroom to assist the student.

6 Conclusion

It is my opinion that students could be taught digital camera forensics by in person classes, by asynchronous online methods such as Blackboard, or by synchronous methods such as the virtual classroom in Second Life. In person classes work best for traditional learners who live near the university but this method only serves a small minority of people who need to study digital camera forensics. The asynchronous online Blackboard version of the class in my opinion serves the most people because it allows any size group of people anywhere and anytime to quickly take the class. They can also practice with their own digital camera or camera cell phone, and then print a credential after successfully completing the class. Lastly the synchronous version of the class in Second Life allows small groups of people anywhere to get personalized instruction but the drawback is that they have to wait until a scheduled class runs. If the class runs during business hours in the United States, it is nine hours later to a deployed soldier in Iraq. The time may not be convenient for the both the student and teacher. The main point is that online learning in any form allows students to obtain quality education that would normally not have been possible before the popularity of the Internet.

7. References

[1] URL Accessed March 5,2011
http://cameras.about.com/b/2009/01/28/stats-show-popularity-of-digital-cameras.htm

[2] Whitehead, T., (2011), U.K. Daily Telegraph, URL accessed March 5,2011
http://www.telegraph.co.uk/news/uknews/law-and-order/8360156/Rapists-who-attack-in-home-face-longer-in-jail.html

[3] URL Accessed March 5,2011
http://events.americasarmy.com/

[4] URL Accessed March 5,2011 Morris, C., (2002) "CNN Money, Your Tax Dollars at Play"
http://money.cnn.com/2002/05/31/commentary/game_over/column_gaming/

[5] Berry, B., (2010)," Pedagogy, Student Preparation and Technology Issues in a Simulated Environment Professor Carlie Merritt and Professor Robert Berry", Conference Proceedings of the 13th Annual Emergency Management Higher Education Conference, June 7-10,2010 at FEMA, Emmetsburg, Maryland

[6] Pine, J.,(2007),"Technology in Emergency Management", Wiley Publishing, Danvers, MA, p244, ISBN 978-0-471-78973-4

[7] Vacca,J.,(2011),"System Forensics, Investigation, and Response", Jones & Bartlett Learning, Sudbury, MA, p248, ISBN 978-0-7637-9134-6

[8] Carrier, B.,(2005),"File System, Forensic Analysis", Published By Addison Wesley, Upper Saddle River, New Jersey, ISBN 0-321-26817-2

SESSION

TOOLS AND APPLICATIONS

Chair(s)

TBA

Real-Time Spherical Panorama Image Stitching Using OpenCL

Wei-Sheng Liao[1], Tung-Ju Hsieh[1], Wen-Yew Liang[1], Yang-Lang Chang[2], Che-Hao Chang[3], and Wei-Yao Chen[3]
[1]Department of Computer Science and Information Engineering, National Taipei University of Technology, Taipei, Taiwan
[2]Department of Electrical Engineering, National Taipei University of Technology, Taipei, Taiwan
[3]Department of Civil Engineering, National Taipei University of Technology, Taipei, Taiwan

Abstract— *This paper presents a webcam-based spherical coordinate conversion system using OpenCL massive parallel computing for panorama video image stitching. With multi-core architecture and its high-bandwidth data transmission rate of memory accesses, modern programmable GPU makes it possible to process multiple video images in parallel for real-time interaction. To get a panorama view of 360 degrees, we use OpenCL to stitch multiple webcam video images into a panorama image and texture mapped it to a spherical object to compose a virtual reality immersive environment. The experimental results show that when we use NVIDIA 9600GT to process eight 640×480 images, OpenCL can achieve ninety times speedups.*

Keywords: image stitching, panorama, OpenCL

Fig. 1: Spherical projection system - Science on a Sphere. (Image courtesy of National Oceanic and Atmospheric Administration)

1. Introduction

Image stitching in the field of image processing technology is a popular topic, in many fields of science and the application will use this technology. In 1997, Shum [1] and others proposed a technology making one panorama image by overlapping multiple images which from the same scene but different parts, afterwards many related technologies and applications are extensions of this new technology. Nowadays there are many applications combing with panoramic images and virtual reality technologies to render stereo images covering a 360-degree angle with 3D, making users feel immersive. Such as Apple QuickTime VR technology, and recently very popular Google Street View are application cases combined with the panoramic images and virtual reality. Through virtual imaging technology, users can view panoramic images no longer a plain image but 360-degree three-dimensional images, such as the Google Street View offers map with panoramic views of three-dimensional, so users can find their destinations more easily while browsing the map.

In academic research, virtual reality is one of the popular, such Wyk [2], who proposed a virtual reality system that aims to simulate the real working environment of mining, to provide mining personnel training closer to the true way of mining jobs. Because of insufficient training of staff at the accident spot, the staff lack of adaptability, so authors propose to simulate an environment that close to the real mining situation to lower the damages and accidents. For now, it still require using the image stitching software to process the images from individuals to a panoramic photo, and the more the images, the more the time will take. Besides, image pixel is one of the factors that affect the performance and users cannot see result or make modifies in the real-time.

In this study, we aim to use simple photographic equipment to capture images and real-time convert them into a panoramic image. We aim to present panorama images in such a way that three-dimensional spherical images and can be manipulated by users interactively, allowing users to have a view of panoramic images like in an immersive environment. In this study, the goal is to develop an ideal image stitch system should be able to combine multiple images into a panoramic image. In addition, we aim to extend it to process video images taken using webcam (network cameras) captured images. We aim to develop a real-time coordinate system converting system to transfer multi-camera images to spherical panoramic images and to present it in three-dimensional spherical image. As shown in Figure 1, the science on a sphere has a spherical projection plane for displaying panoramic image or a panoramic video. Many academic institutions already used this approach to present three-dimensional panoramic video.

2. Related Work

OpenCL (Open Computing Language) is the first device for heterogeneity developed by the open-parallel computing language proposed by the Apple computer company, developed by the Khronos Group Management, and co-operation by several technology companies to participate. Compared to CUDA technology can only be used in the display chip developed by NVIDIA, since OpenCL is built on heterogeneous nature of the standard programming language that can run on multiple platforms, including the graphics processor, CPU, embedded devices and handheld devices, can support OpenCL parallel computing. There have been many scholars into OpenCL program development, such as Zhang [3], who used OpenCL open computing language parallel to the reconstruction of CT images, as opposed to CPU handling, performance enhanced about 100 times, but the experiments also shows that the performance of OpenCL operational is behind the CUDA version, the reason may be because OpenCL compiler is not yet well-developed, plus programs have not been optimized.

The open standard of OpenCL make it portable and make it independent of graphics system vendors. In addition, Sharma [4] use OpenCL parallel computing to accelerate the discrete wavelet transform calculated and compared the difference in performance with the OpenCL and CUDA parallel computing. The result shows that the performance on CUDA is better than OpenCL. In fact, we can asily move the OpenCL applications that are developed in the NVIDIA platform to ATI platform, as opposed to CUDA can only be operated on NVIDIA graphics cards. Therefore, OpenCL has the advantage of high portability. OpenCL could not only support basic and universal computing power but also provides extension and OpenGL, OpenGL ES and other graphics API for efficient interaction, making the 3D graphics imaging operations also have greatly improved.

The core architecture of OpenCL [5] can be divided into four main models, namely Platform Model, Execution Model, Memory Model and Programming Model, the following four subsections is going to introduce OpenCL model briefly. OpenCL platform model basically consists of a host, connected to single or multiple OpenCL computing device, OpenCL computing device can be one or more graphics processors, one or more multi-core CPU, or even can be handheld devices or embedded devices. OpenCL in a computing device can be divided into one to multiple computing units (CU), and each computing element and can be divided into a draw to deal with members of multiple (PE). Among the OpenCL [6] framework, the smallest unit is the processing members, so when a OpenCL computing device has multiple computing units, and each computing element contains a number of processing members, when amount of processing members processing the task at the same time, we can obviously see the advantage of parallel computing.

Generally, during the execution of OpenCL program, it is divided into two parts: kernel and the host program, one or multiple OpenCL computing device is responsible for the implementation of kernel, while the host program is executing on the host side, and responsible for defining the context of the kernel and the manage the executions of kernel. When the host issued a request for the implementation of the kernel, it will also define an index space, the index area is similar to the Grid in CUDA architecture, and generate an instance of the kernel corresponds to the index space and implement the kernel, the kernel instance is also known as a work item, a work item is equivalent to Thread in CUDA architecture, and each work item in the index space will be given an unique work item ID based on the location of the index space, known as the Global ID. Each work item in the kernel will execute the same code, but the execution path and access to information may be different according to their global ID. And the number of work items in the kernel will be the formation of a working group, the working group is equivalent to Block in CUDA architecture, the corresponding index for each working group will produce a unique work space group ID, and the same work group work item will also have their own work item ID, known as the local ID, the work item in the kernel can be identified via their global ID or local ID and the work item ID. During the execution of kernel, one calculation unit is responsible for the implementation of a working group and computing unit then distribute the work item to the members for parallel processing.

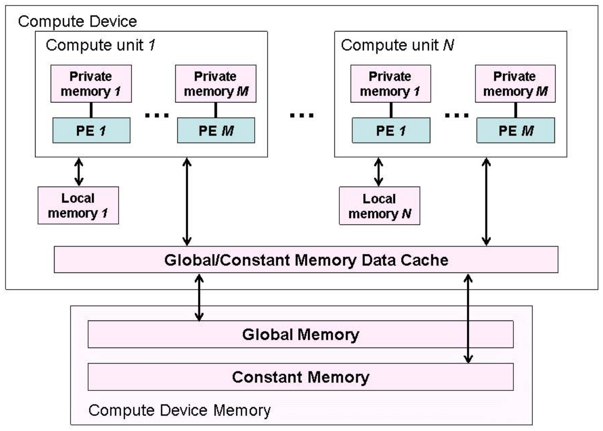


Fig. 2: OpenCL Memory Model. (Image courtesy of the OpenCL Specification)

OpenCL memory architecture model is similar to CUDA, using the hierarchical structure to distribute different kind of memory space. As shown in Figure 2, the main memory is divided into four types, namely, Private Memory, Local Memory, Constant Memory and Global Memory. Global

Memory belongs to Compute Device Memory, it has large memory capacity but slow access, each work item in the kernel can access any element declared in the Global Memory. Constant Memory is also in Compute Device Memory, the work item in the kernel can visit any element declared in the Constant Memory as well, but can only read. Local Memory is inside of Compute Device, the memory access speed is much faster than the Global Memory, but a relatively small capacity and elements announced in the Local Memory can only be access by the same working group, different working groups can't share the resources of Local Memory. Besides, Private Memory belongs to Compute Device Built-in memory as well, fast memory access, but the capacity is relatively very small, variables who declared in the Private Memory can only be possessed by their own work item, they cannot share Private Memory between work items and work items.

OpenCL execution model support two parallel models, one is data parallel and the other is task parallel, it also the support mixed use, usually data parallel programming model is mainly used in the OpenCL. Based on a set of instructions in the data parallel programming model defines a sequence of calculations, and this group instruction sequence will be applied to many elements in a memory object, and the index space related to the OpenCL execution model will defined the work items, and map the corresponding data to the relevant work items.

3. Spherical Panorama Image Stitching

Spherical panoramic imaging system is an interactive system, user input a series of photos or use a webcam as the image source, then the source image will be copied to the graphics memory, as parameters for OpenCL kernel to use, and converted the source images to spherical coordinates in real time and parallel computing way to achieve refresh the screen in real time, and provide users with manual adjustment of the image horizontal and vertical position, rotation, and other fine-tuning capabilities during the process.

Figure 3 is the flow chart of spherical panoramic image system, with OpenCL parallel computing technology and the OpenGL pixel buffer capacity of changing textures rapidly in this system, we can achieve the goal showing panoramic image in real-time, and the system not only can present the panoramic image in the way of three-dimensional ball but also can output the image to the host memory through pixel buffer and store them in the picture format. The following sections will introduce implementation process of spherical panoramic image system.

The obtained source images are converted to spherical coordinate so that images can be texture mapped on the sphere surface. To do this, we need to know the real-world coordinates and size of the source images. For each source image, the calculation of its corresponding location on the sphere is aimed to avoid excessive distortion.

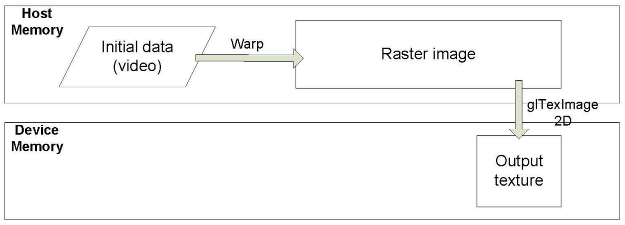


Fig. 3: Spherical projection system pipeline with CPU.

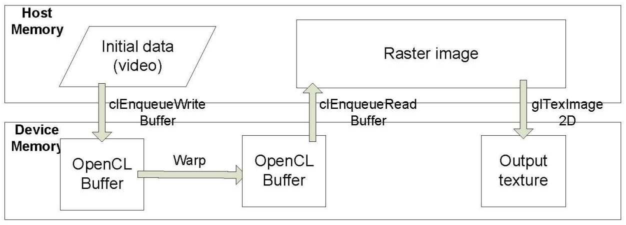


Fig. 4: Spherical projection system pipeline with OpenCL.

The method we used here is to calculate the perspective of the image, the so-called perspective is the scope a camera can contain. After obtained the perspective we can convert to the range that this image is placed, and obtained four vertices images of the rectangular plane coordinates in the image, and then convert the coordinate to the spherical coordinates.

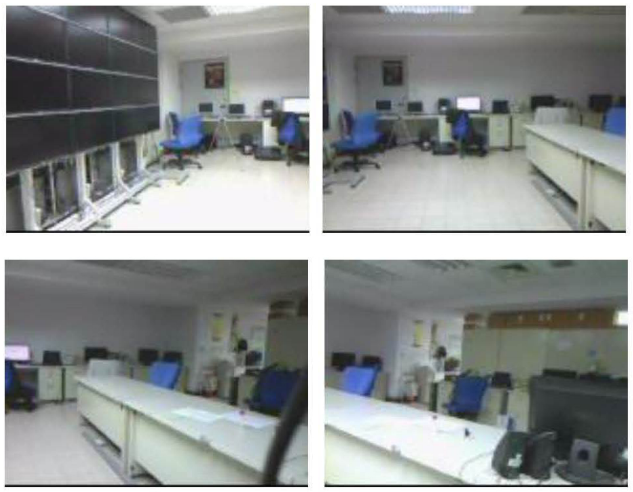


Fig. 5: Multiple webcam video images.

3.1 Spherical Coordinate Conversions

Image stitching in the first step required warping the image, and then adjacent images can be stitched together. Usually there are two kinds of deformation, one is the image projected onto the cylinder, and the other is the image projected onto the sphere. There are many related research about the ways to produce spherical panorama, such as Deng [7] and Tsai [8], they have produced the spherical panorama picture using the two fish-eye camera, the former provide a method to automatically generate the spherical panorama picture with two fish eye cameras without additional amendments. While the latter set two fish-eye cameras frame in the pano-head, users can spin the cameras through the user interface, and effectively establish spherical panorama pictures present it in the way of 3D rendering. Besides, Guan [9], who proposed a new video presentation to correct the visual distortion in the spherical panoramic image that is caused by the problem of differences of user location, allowing users to view the image from any angle and location in the spherical image environment, and can amend the visual distortion that is caused by the different angles in real time, so the visual just as immersed in the real life environment. And Kim [10], who proposed the spherical coordinates transformation to generate a from aspect image, and proposed a new method to compensate for image vignetting and illumination effects, experiments show that this method has worked well, making images and video look more real without any gap junction.

Then the world coordinates of four vertices were converted to image coordinates, followed by defining the location of the spherical coordinates corresponding to each pixel, which is calculated by the image pixel coordinates corresponding to longitude and latitude coordinates. As shown in Figure 6, we used the way that is called Ray Casting, we consider the center point of OpenGL coordinates as human eyes or a pinhole camera to emit the rays in the radiation way, cross the plane of each image pixel, finally reach to the sphere coordinates. Here we use sphere coordinate conversion to transform the rectangular coordinates of the pixel in the plane image to the spherical coordinate, the formula is as follows:

$$\theta = \cos^{-1}\left(\frac{z}{\sqrt{x^2+y^2+z^2}}\right) \tag{1}$$

$$\phi = \tan^{-1}\left(\frac{\sqrt{x^2+y^2}}{z}\right) \tag{2}$$

The x, y, z are coordinates of Cartesian coordinate system in image pixels, they can be converted by the formula of the spherical coordinates of zenith angle θ and azimuth ϕ. Then the original image plane coordinates by the rotation matrix can be obtained through the rotation of the plane after the image coordinates, which allows images to any move in the sphere, then the image plane coordinates as OpenCL core parameters to use and immediately converted to Spherical panoramic images. Figure 7 shows the intermediate step of coordinate conversion.

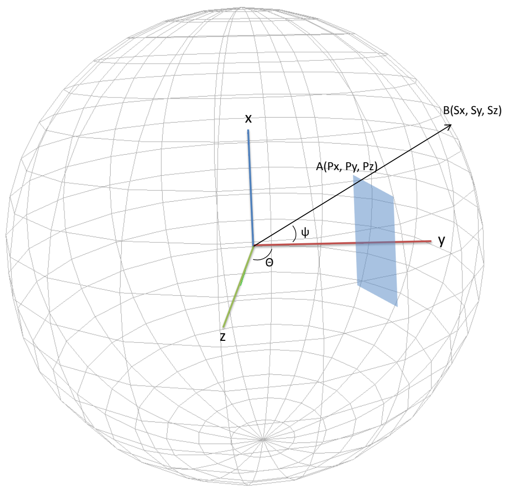


Fig. 6: Coordinate conversion.

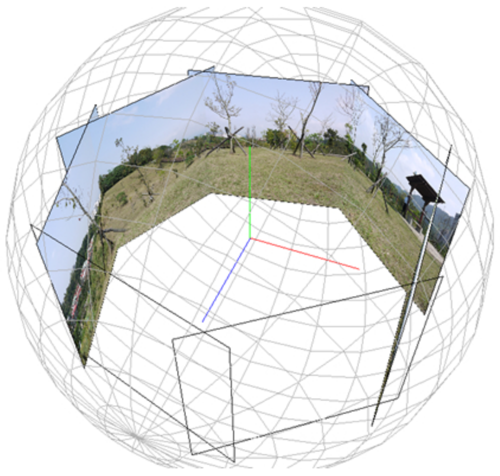


Fig. 7: Intermediate step of the coordinate conversion.

3.2 OpenCL Kernel

When obtained the source image and calculated angle of view, then you must copy the image to OpenCL objects in memory to use as the Kernel parameters, OpenCL Kernel will parallel processing and convert the source image to spherical coordinates and stored in the pixel buffer. With its powerful 3D processing capabilities of OpenGL, are now widely used in gaming, medical imaging, virtual reality and visualization program development and other fields. And now many GPU hardware support OpenGL and provides OpenGL shading language. In addition to supporting basic and universal computing power, OpenCL provides the extension and OpenGL, OpenGL ES and other graphics API for efficient interaction, making the 3D graphics images can also have a significant enhancement in operational aspects. So OpenCL not only have the development of general parallel computing capabilities, but also combine with the existing support OpenGL shading language to maximize performance and multi-applications. OpenCL1.0 provides GL extension function at the same time, allowing the context of OpenCL to establish the OpenGL context, so OpenGL buffer objects and textures can be used as OpenCL memory objects, making OpenCL and OpenGL can share resources and increase graphics performance computing efficiently.

The goal of spherical panorama image system is to establish a width of 2:1 ratio of the length of the spherical panoramic image, and present it in the way of the sphere textures, to improve the picture display speed, so this use double buffer to store pixel panorama images, in order to reduce system idle times and memory transmission. The Time between after establishing OpenCL buffer as spherical panoramic image memory space and before implement the core of OpenCL, we have to set the NDrange of the core, which means the work items and the number of working groups we mention earlier, work item is the smallest unit of implementation, and number of work items form a working group. In the implementation of the Kernel, the NDrange will be divided into two work intervals, namely Local work size and global work size, in this global work size refers to the total number of work items of implementation of this Kernel. And local work size refers to how many work items are there to form a working group.

In the experiment, our video source resolution is 640×480, and programs are designed to handle multiple images simultaneously, so here we set work size to three-dimensional, as for the global work size, the first two dimensions for the image are wide and high respectively, the third dimension is the number of input images N, so in the experiment, the total pictures number of global work items will be 640×480××N. As for local work size, since each image are located in different coordinates of the sphere, so the way our group divided is to set our images into many work-groups, to get their arguments more easily. Here, the first two dimensions are to given the same image work groups of the work items number, and the third dimension is fixed at 1.

After finishing the task of OpenCL Kernel, then we have to output the panoramic image to made into the textures, this step is also one of the important factor affecting the overall performance, the use of OpenGL pixel buffer will affect the performance efficiency, and whether OpenGL pixel buffer could shared resources with OpenCL buffer or not. And these two factors as the use of OpenGL and OpenCL device is supported or not, if not, when finishing the task of OpenCL Kernel, it must send back the panoramic images to the host memory, then read image pixel to the graphics card memory and convert it into texture. As shown in Figure 8, this situation will cost you twice the PCIE transmission time. In addition, when using the pixel buffer, but can't shared buffer and texture mapping of resources with the OpenCL device, after finishing the task of OpenCL Kernel, it must

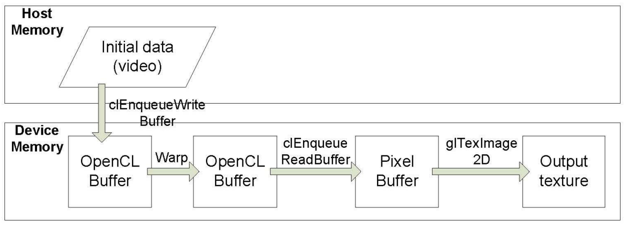


Fig. 8: System pipeline without pixel buffer.

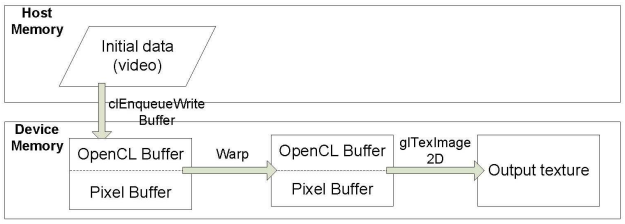


Fig. 9: System pipeline with pixel buffer.

copy the panoramic images from OpenGL pixel buffer to OpenCL buffer to be made of texture mapping then output them, in this case, it saved two PCIE memory transmission time, but still need to spend one transmission time between the graphics card memory, as shown in Figure 9. The final situation is to use pixel buffer and shared resources with OpenCL devices, as OpenCL buffer is directly established in the pixel buffer, so when the Kernel complete the task, it can output texture directly from the pixel buffer and made of panoramic images, present the panoramic images by way of three-dimensional in real-time, reducing unnecessary memory transfers, and effectively improve the overall performance, as previously shown in Figure 3.

4. Results and Discussion

In this section, we will test and compare the difference between the overall performance of spherical panorama image system of the OpenCL parallel acceleration technology and a single threaded CPU version, and using pixel buffer with or without the case on the overall performance Effects. Our test hardware environment has Intel Core 2 Duo 6320, clock 1.86 GHz, 2G DDR2 memory, and NVIDIA GeForce 9600GT graphics card. We use visual studio 2008 in the Windows XP operating system.

4.1 OpenGL and OpenCL Buffer

In this section, we examine the impact of overall performance on spherical panorama image system at OpenCL and CPU versions, and with or without OpenGL pixel buffer. Then we test pixel buffer for OpenCL parallel computing version of the image in the dynamic effects. As the test graphics card support the extended functions of OpenGL

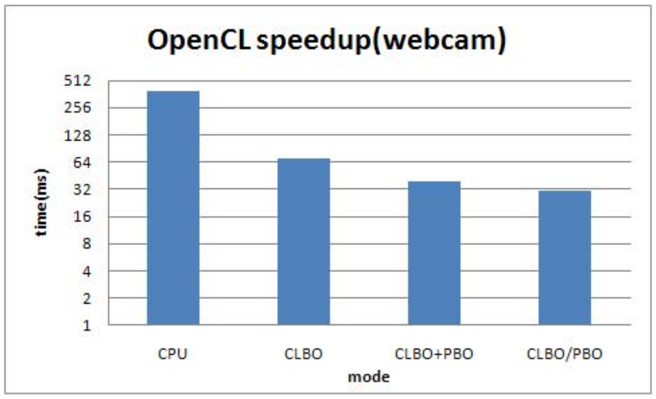


Fig. 10: Benchmark of OpenCL buffer.

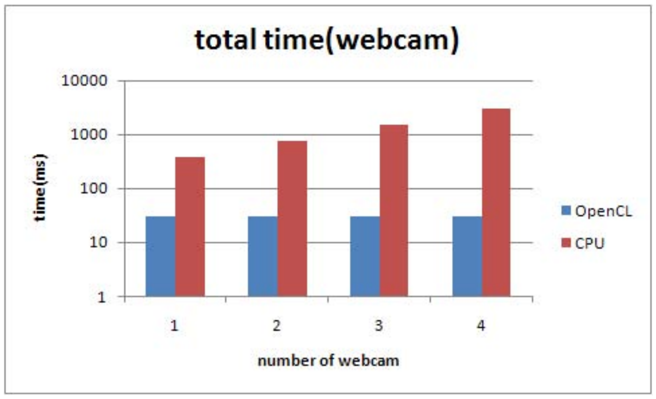


Fig. 11: Total computing time of OpenCL (CLBO/PBO).

can share the buffer and texture object with OpenCL, so in this experiment can be divided into three main parts of the discussion. The first part is not to use pixel buffer, in which the source must be read from the host memory to OpenCL buffer, and when the kernel finished the tasks, it must sent the data in the OpenCL buffer data back to the host memory, and then be read to the graphics card memory blocks to be made into texture mapping object. This approach will spend twice the memory by the PCIE transmission time, as shown in Figure 10, when not to use pixel buffer, use the 9600GT's performance can only achieve 14 FPS, experiments show that too much time was wasted on the transmission of memory, making the performance unsatisfactory.

Then we discussed the difference whether the pixel buffer in the buffer sharing resources with OpenCL. If the pixel buffer cannot share resources with OpenCL buffer, it must be read from the video source of the system memory buffer to OpenCL. And when the kernel finished the tasks, it must sent the data in the OpenCL buffer data back to the host memory, and then be read to the graphics card memory blocks to be made into texture mapping object. But when OpenCL can share resources with the OpenGL, we simply read the source image to the pixel buffer, and create OpenCL buffer from pixel buffer, and then complete the tasks in the Kernel. After the implementation, the data does not require copying, we can directly converted the data in the pixel buffer into textures, saving a lot of data transfer time. Figure 10 shows, in the case of 9600GT graphics card in combination with pixel buffer and OpenCL buffer, can be achieved in the performance enhancement of 2.2 times speedups, and use pixel buffer but not binding under the buffer OpenCL is 1.7 times fast in the performance. This is because in the case of using pixel buffer eliminates the need for two images in the PCIE on the transmission time, indirectly showing the memory transfer in the parallel computing important factor affecting the performance, also shows The advantage of resource sharing of OpenCL in combination with OpenGL, and makes the graphics better and more efficient in the implementation of the program as well.

4.2 Overall Performance

In this section, we compare the real-time image stitching system of CPU version and OpenCL version. The overall performance differences are measured using pixel buffer and buffer with OpenCL shared resources. In dynamic images, we measured performance time of using 1, 2, 4, and 8 webcam video images for real-time images stitching. The resolution of a video image is 640×480, and it is converted to spherical coordinates, and output panoramic images to be produced into textures. The final total performance time of outputting image into a spherical map exclude the time of coping the source image to the OpenCL buffer.

First, Figure 11 is the traditional way of handling CPU and OpenCL impact of converting the system's total execution time, we can see if we use CPU to process more source images, the process time will grow linearly, in this case just dealing with a image must spend almost 400 milliseconds, in other words that is presented only about 2.6 per second FPS, which makes the screen phenomenon presents a serious delay in the performance while can't achieve real-time interaction. In contrast to OpenCL parallel computing approach to image conversion processing performance than the CPU versions, the later have a very significant upgrade. As Figure 11 shows, in the best circumstances, processing a single image, as long as the cost on average about 16 milliseconds and can transform a source image as a texture for 3D images, and we can see that when the images of number increases, the process time of OpenCL did not increase, still maintained at the level of 16 milliseconds.

In Figure 11 we will compare the calculating time of CPU and OpenCL, we can see that when we process 4 webcam whose resolution is 640 x 480 simultaneously, we can enhance the computing performance to 98 times, the more the processing images are the better the performance will be. We can also achieve real-time video conversion, and user interaction. Figure 11 and Table 1 are the performance of the use of a webcam image as a source, when dealing with a single network Camera source image, the performance is 32 FPS, it costs almost half of the overall computation time compared to the source which are static images. The main reason is because the need to continue to capture images and send to the graphics card memory, which led to performance degradation.

Table 1: Overall performance (real-time imaging)

Number of Webcams	OpenCL(ms)	CPU(ms)
1	30.80	386.05
2	30.96	767.26
4	31.78	1535.50
8	39.84	3063.66

5. Conclusions and Future Work

In this paper, we implement a webcam-based spherical coordinate conversion system, through the combination of OpenGL which have the 3D stereo effect and OpenCL massive parallel computing power to achieve an interactive real-time 3D virtual scene image. The use of GPU calculation is different from the traditional CPU use of sequential processing loop to compute pixels. With the GPU multi-core architecture and its high-bandwidth data transmission rate of memory accesses, the programmable GPU and OpenCL language make it possible to handle multiple video images in parallel.

Experiments indicated that we can have a better performance to use OpenCL parallel computing in the implementation of image conversion than on the CPU, the more data to process the better parallel computing performance we can get, we used the NVIDIA 9600GT graphics cards to simultaneously handle 8 images whose resolution for each image is 640×480 and the computing performance is enhanced 95 times compared to CPU version, presenting a real-time three-dimensional spherical panoramic images and to achieve real-time interaction with the user results.

The main purpose of spherical panoramic imaging system is to achieve an interactive 3D virtual environment real-time video. Using OpenCL parallel computing power, we deal with large number of real-time video sources in parallel and get a significant enhancement performance compared to traditional sequential approach. Although operations have been able to achieve in real time, there are some system performance issues can be improved. First, the current panoramic image system has the brightness variation problems at joints between images. Second, low-resolution web cameras are used in this study. The future work is to use high-resolution photography equipment for real-time interactive in an immersive virtual reality environment.

Acknowledgements

This work was supported in part by the Taiwan National Science Council under grant NSC 99-2218-E-027-008-.

References

[1] H.-Y. Shum and R. Szeliski, “Panoramic image mosaics,” Microsoft Research, Tech. Rep. MSR-TR-97-23, 1997.

[2] E. van Wyk and R. de Villiers, “Virtual reality training applications for the mining industry,” in *AFRIGRAPH ’09: Proceedings of the 6th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*. ACM, 2009, pp. 53–63.

[3] W. Zhang, L. Zhang, S. Sun, Y. Xing, Y. Wang, and J. Zheng, “A preliminary study of OpenCL for accelerating CT reconstruction and image recognition,” in *Nuclear Science Symposium Conference Record (NSS/MIC), 2009 IEEE*, 2009, pp. 4059–4063.

[4] B. Sharma and N. Vydyanathan, “Parallel discrete wavelet transform using the open computing language: a performance and portability study,” in *Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, 19-23 2010, pp. 1–8.

[5] J. Stone, D. Gohara, and G. Shi, “OpenCL: A parallel programming standard for heterogeneous computing systems,” *Computing in Science & Engineering*, vol. 12, no. 3, pp. 66–73, 2010.

[6] K. O. W. Group, *The OpenCL Specification*, Khronos OpenCL Working Group, 2009.

[7] X. Deng, F. Wu, Y. Wu, and C. Wan, “Automatic spherical panorama generation with two fisheye images,” in *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, 2008, pp. 5955–5959.

[8] M.-J. Tsai, C.-L. Kao, and J. Liu, “The gentle spherical panorama image construction for the web navigation system,” in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, 2010, pp. 1578–1581.

[9] X. Guan, L.-K. Shark, G. Hall, and W. Deng, “Distortion correction for immersive navigation in spherical image environment,” in *CyberWorlds, 2009. CW ’09. International Conference on*, 2009, pp. 96–101.

[10] Y.-N. Kim and D.-G. Sim, “Vignetting and illumination compensation for omni-directional image generation on spherical coordinate,” in *ICAT ’06: Proceedings of the 16th International Conference on Artificial Reality and Telexistence Workshops*. IEEE Computer Society, 2006, pp. 413–418.

Initial Design of a Software-Based, Tremor-Reduction, Presentation Pointer

Anthony Dovelle, John Truitt and Thomas Way
Applied Computing Technology Laboratory
Department of Computing Sciences
Villanova University, Villanova PA 19085
thomas.way@villanova.edu

Abstract – *A significant portion of the population will at some point experience some form of hand tremor. These widely occurring hand tremors can range from barely perceptible, miniscule, rhythmic hand vibrations to large, spontaneous and virtually debilitating hand shaking. Recent hardware and software technological advances have lead to new computer input devices being available to accommodate general human-computer interaction needs. These new technologies also can be adapted to assist those with tremors. One area where a tremor can be particularly frustrating for some is when making a presentation using a laser pointer. This paper focuses on a software-based method for reducing the effect of tremors on computer users who are making presentations. Previous research in tremor reduction techniques is presented and used to motive the design of a software-hardware solution that combines a Nintendo® Wii™ game system controller with customized software. The design of our system is discussed and future directions for exploration are given.*

Keywords: Hand Tremors, Algorithms, Parkinson's, Input Devices.

1 Introduction

Technology continues to evolve at an astounding rate. As is obvious, this also significantly pertains to the topic of computing. This ongoing evolution produces benefits such as improved computer performance, increased user efficiency in completing tasks and innovative methods for human-computer interaction and user interfaces. Of importance is leveraging advancements in technology to improve access to computers and assist with their use by users with various disabilities. Technology continues to advance and provide access to computers and assistance with their use by persons who are blind, deaf, or with cognitive or functional disabilities. Researchers in the area of rehabilitation engineering and assistive technology routinely assess new human-computer interface technology and create innovative uses that can provide needed access to the populations they serve.

One particular example of recent research into assistive applications of technological advances focuses on reducing the effects of hand tremors. A hand tremor is a condition frequently correlated with factors such as advancing age, mental or physical, and a variety of other related pathological and physiological conditions including Essential Tremor and Parkinson's disease [1]. For many with tremor, its impact is minor and little adaptation is needed. For moderate and severe cases of tremor, regardless of the cause, reducing the impact of the tremor is an important quality of life goal, although it can pose a significant challenge to accomplish. Due to relatively recent advances in hardware power, interface devices and software techniques, the implementation of alternative interaction schemes to accommodate to those with disabilities is an active one. For example, a pilot study that explored mouse-motion smoothing techniques for computer users with a tremor shows significant promise [2]. Preliminary work in the area provides a foundation on which to build, and supports the need for innovative re-use of interface devices and software to provide access to computer users with varying degrees of tremor.

In the case of hand tremors, while most slight hand jitters for practical purposes are nearly insignificant as they effect the performance of common, daily routines, other user interface devices are more challenging to use and may require additional effort to use adequately, if they can be used at all. An example of a device that is difficult for a user with a tremor to use adequately is a pointing device, such as a mouse, light pen or laser pointers, which either are simulated in software or are of the hand-held, battery-powered type. Such pointing devices enable the user to interact with a computer by using the intuitive motion of pointing at the some part of the computer screen in order to move the cursor to that location or perform some other pointing and selecting task. This natural and direct interaction can be challenging for a user with a hand tremor as the shake induced by the tremor reduces pointing accuracy and therefore user efficiency, and in some cases leads to increased tremor due to performance-stress, nervousness, frustration and even embarrassment, however unwarranted.

Numerous results have been reported that attempt to address aspects of adapting technology to accommodate users with tremor. Some research focuses on measurement and problem characterization, such as analyzing the relationship between pointing accuracy and distance from the screen being pointed to [7]. Other research has focused on computational and digital signal processing algorithms that are perform mathematical analysis and calculation on pointer position and movement data, and attempt to analyze and react to user needs in order to provide smooth, responsive pointing motion. Notably, the "adaptive gain" method continually reacts and responds to user interaction with the device itself, and has successfully shown that the technique can "decide" whether or not a dampening of cursor or pointer movement in necessary and would be beneficial [4,6]. This adaptive gain technique shows promise as a foundation for solving a wide variety of motion-related filtering problems within the field of human-computer interaction, and appears to be particularly appropriate to tremor reduction of pointer motion.

In this paper, background information on hand tremor is presented and various software and digital signal processing techniques that have been developed to reduce the effect of hand tremor are reviewed. Next, the design of a software-based tremor reduction prototype is introduced and its effectiveness is discussed. Finally, future directions for continued development of this software technique are presented and a number of related projects and applications of this technique and software are provided.

2 Hand Tremor

The term "tremor" is defined as an involuntary shaking movement that is most often noticeable in hands but can also affect other parts of the body including the head or voice [9]. In this paper, the focus is on tremor that impact hand movement. Symptoms of hand tremor may include slight to severe shaking of the hand, which can be caused by several medical or psychological conditions. More specifically, a tremor is an involuntary, sometimes rhythmic, muscle contraction and relaxation that creates an undulating movement pattern within the effected body part. There are benign and natural tremors that occur in the hand with no real underlying cause and they are generally minor in significance. There are also tremors that occur only during certain activities and these are generally minor tremors. The condition known as Essential Tremor is one that typically occurs in people over the age of about 65. An Essential Tremor is naturally occurring, is not normally debilitating, and is not associated with any medical illness or condition. It is believe to be caused, at least in part, by the tendency for muscle weakness gradually to increase with advancing age. Essential Tremor, though typically not dire or debilitating, can still be a major frustration, and this can be particularly noticeable in more public situations such as when using a laser pointer while making a presentation. [9]

More severe tremors are frequently the result of a more serious medical condition. These severe tremors can be a symptom or by-product of diagnosed medical disorders, and there exist several other medical conditions that can cause tremors in a person, as well. Tremors are often recognized as a symptom of mental disorders in parts of the brain, which manage muscle simulation and motion throughout the body, or in specific areas, such as the hands. Examples of neurological conditions that can cause tremor include multiple sclerosis, stroke, traumatic brain injury, chronic kidney disease, as well as neurodegenerative diseases such as Parkinson's disease. Characteristics of tremor can include rhythmic shaking in an area of the body, difficulty writing or drawing, or even problems holding and controlling utensils. While these are the most common causes, there is a significant variety of types of tremors that affect the body in varying ways and to differing degrees. As with Essential Tremor, more severe tremors also can negatively impact a person with a tremor from being comfortable when in public situations, such as when making a presentation while using a pointing device. [9]

3 Tremor Reduction Techniques

While tremor can be a barrier to a person being comfortable or effective in public situations where the tremor is most noticeable as when giving a presentation, significant research has been done in the area of tremor reduction. [2-6]

3.1 Mouse Motion Smoothing

Research in the form of a pilot study on the effects of a smoothing approach on the usability of a computer by users with Essential Tremor restricted its examination to techniques for tremor reduction when using a common mouse input device [2]. The experiments conducted for this research were human-oriented, allowing for people with differing levels of tremor intensity to participate. These participants were asked to perform various tasks, using a specialized mouse, ranging from target clicking, i.e., clicking on buttons in a specific order, to holding the mouse over a target area for a given amount of time. After the participant had completed the testing, they were given a survey to express their opinions on ease of use of the specialized mouse. This mouse-based approach provides an important starting point in development of a more generalize technique that can apply to our software-based laser pointer approach. The need for user testing is crucial, and led to our desire to conduct preliminary, ad hoc experiments on our software approach as it was developed.

3.2 Adaptive Gain

The paper, "Toward Goldilocks' pointing device: determining a 'just right' gain setting for users with physical impairments" [4] focuses on tremor reduction through the use of personalized gain settings when using computing pointing devices. The authors examined the Adaptive Gain algorithm, a method of analyzing movement patterns and delivering a personalized setting of sensitivity for each individual user. Through this method's use, one can create a significant improvement in pointer accuracy. This paper presented the definition of gain as the proportional difference between cursor movement and actual physical movement, a concept that we draw upon heavily in our prototype design and implementation, and forms the basis for our experimental design. The adaptive gain method reported in this paper can enable pointing devices to reduce how tremors are read by the device and how the cursor reacts on screen. [4]

3.3 Adaptive Calibration and Pointing

In research into an adaptive pointing technique that uses a gain adaptation approach to intuit filtering of pointing device motion, the authors examined the effectiveness of adaptive gain adjustment in general. Adaptive gain adjustment means that the pointing device automatically adjusts the gain, or reactivity, of the device on the fly based on positional data and velocity of pointer motion. The benefit of this approach is that it does not require calibration. Rather, this adaptive technique automatically adjusts based on the presenter's movements. This research builds upon other work in Adaptive Gain, and presents a strong case for incorporating an adaptive approach into the design of our solution, reinforcing how this approach can be an effective and exceptional method in support of software-based tremor reduction. This research also discussed three related approaches to tremor reduction that preceded the Adaptive Gain method. These three methods are target-oriented, manual-switching, and velocity-oriented. These methods are described in-detail in a later section of our paper. Their research also mentioned a Wii-based application that could be used as an inexpensive alternative to the more expensive medical based methods. This idea of using the Wii for testing became an invaluable part of our work. [6]

As previously mentioned, many attempts have been made to increase both efficiency and accuracy when computing with pointing devices. Many of these schemes can be classified under three distinct categories of approaching the problem:

- Target-Oriented Approach
- Manual-Switching Approach
- Velocity-Oriented Approach

These each are examples of pertinent approaches on which more sophisticated tremor reduction techniques can be built. Each of these methodologies pertains to a particular set of actions to which the method is beneficially. The upsides, as well as corresponding drawbacks to each of these approaches are discussed here, providing a foundation for further refinement of our prototype system.

3.3.1 Target-Oriented Approach

The Target-Oriented Approach is associated with a focus on computing "targets" or hotspots. This approach takes popular points of user interaction into consideration when proposing a solution. These points include general point-and-click interaction routines including that of buttons, hyperlinks, and menus. The proposed solution is as the cursor passes near or over these objects, they become "sticky" in response. Sticky, of course, refers to the reduction in the level of apparent on-screen reaction to physical motion. A cursor within the vicinity of a clickable object will adjust its position automatically to stay within its boundaries. Variations of this technique exist in which the cursor responds to objects of interest by dampening the proportion of on-screen motion to device movement.

The Target-Oriented Approach proves a vast improvement in efficiency for most mundane computing tasks. These tasks often associated with basic operating system manipulation, e.g., clicking buttons, file directories, icons, and adjusting control panel options. These tasks become little to no challenge as the technique reduces the duration of time spent on accurate position readjustments.

When applied to other forms of interaction, this technique falls short of user necessities. This applies especially to the dilemma of presentation pointing. There exist no hotspots to which the cursor may "snap." More complex tasks, such as highlighting key points in text and posting drawn lines become difficult, if not impossible. These tasks require line smoothing from the commencement of the physical stroke up towards the end. Furthermore, this technique does not compensate for individual user needs, but rather provides a more general solution.

3.3.2 Manual-Switching Approach

Manual-switching primarily refers to the ability of the user to select the amount of tremor reduction he or she needs at a given point in time. Such a method presents a choice between two fundamental corrective schemas: the absolute and relative approaches. The absolute approach represents a direct relationship between devices manipulation and on-screen cursor position. A pointer directed at a particular spot on a screen will inevitably force the screen to display the cursor at that precise location. In contrast, the relative scheme adjusts this proportion of physical and apparent movement. As an example, a user may decide to use a relative approach. In doing so, the on-screen cursor only reacts slightly to more extravagant motions of the hand.

Both of these approaches are appropriate in particular settings. Allowing users to select between these two approaches provides the opportunity to assist oneself during difficult maneuvers. Furthermore, when one requires swift, vague motions, the option for an absolute movement scheme remains available. It is assumed that the user may determine a "correct" setting which does not need to be modified at too great an extent.

However, it is not always optimal to relinquish control to users. Find an optimal or "best" movement translation setting may not always possible. Needs change due to many unknown variables. Increase in stress levels or change in temperature may serve to heighten the intensity of tremors for example. Factors, such as traversing a stage while giving a presentation, modify the distance between the user and screen. This further increases the potential for on-screen jitters. In other words, changes in environmental influences make the task of decision-making entirely more difficult for the user. The level of cognitive involvement is not beneficial when the mind is already strained and an inefficient environment is generated.

3.3.3 Velocity-Oriented Approach

One term, which has been commonly integrated with this discussion, is the concept of "adaptation." The two major deficits in both approaches hint at the need for a far better, more intricate solution. Target-Oriented approaches present us with the problem of strict, singular behavior for all circumstances. Manual-Switching approaches allow more room for user error, although they can present additional challenges and complexity to the implementation. Regardless, adaptation becomes key as both previous schemes are lacking this capability.

Thus, we come to the Velocity-Oriented Approach. This approach intends to address the two above issues simultaneously by introducing a more universally applicable concept. Velocity-Oriented schemes rely on movement speed analysis to determine what sort of "gain" is required at the present time. Gain specifically refers to the particular proportion of hand speed motion to pointer display reaction distance. Much like the aforementioned relative scheme, gain represents the dampening or intensifying of user input on screen. This technique is simple in that it relies on a simple principle: the higher the velocity of the pointer motion, the lesser the degree of accuracy required to follow through with the motion.

Simply put, this technique represents a concrete foundation for tremor reduction in complex motion schemes. As a whole, the velocity-oriented approach is the first step in the adaptive "evolution" process of software. In other words, the software evolves as the user utilizes the software by responding to stimuli: in this case, velocity. [4]

The only real problematic circumstances involved with the velocity-oriented approach are those with significant lag times or "latency". Latency, that is the duration of time necessary to compute and modify the cursor coordinates, is particularly significant when implementing a velocity-oriented approach. The more time spent modifying input, the more distracting and difficult the algorithm becomes when fully utilized.

These papers all touch on a primary theory called Fitts' Law. Fitts' Law states that the time necessary to move to a location is proportional to the distance travelled along with the size of the target location.[3,5] This law directly impacts tremor reduction research when it comes to large movements made by the presenter. One major issue that has to be addressed in any attempt to reduce tremors is that of lag. Assigning a computer to make adjustments to the cursor position as the presenter moves the pointing device can cause it to lag as it performs adjustment calculations. This lag can become a hindrance to any presentation and could cause any method or technology for tremor reduction to be too much of a hassle for efficient utilization. We attempted to take all of this into consideration during our research.

The adaptive gain technique is a form of the Velocity-Oriented Approach as it is based on the principle of analyzing velocity patterns before adapting to user input. This scheme considers each user as an individual with unique needs. These needs are subject to change with various external inputs. In this sense, an adaptive gain approach would continuously analyze user input and determine an amount of gain sufficient to stabilize the cursor in different situations. In this sense, users are less of a generalization and more of a collection of individual profiles. [4,8]

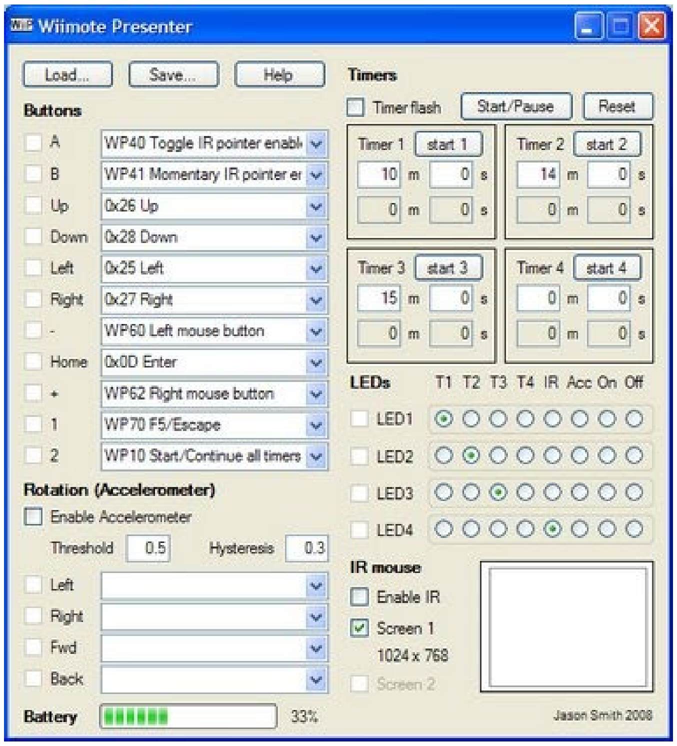
The adaptive gain approach is beneficial to those who require the most assistance. That said, users with moderate to severe hand tremors of various motion patterns will profit the most from such a scheme. This is mostly due to the propensity of adaptive gain software to accurately respond to most any situation. As tremor intensity increases, adaptive gain software adapts by decreasing the motion sensitivity. In this way, the software eliminates smaller vibrations while dampening more erratic shakes. It is this adaptive nature that makes the adaptive gain approach particularly attractive.

4 Prototype Tremor Reduction System

In this paper, a variety of background is provided on tremor and on the most recent advancements in algorithmic technique of tremor reduction. In order to implement a solution to tremor reduction, a software-based tool was created that enables a user with tremor to make use of a simulated laser pointer device while making a PowerPoint type presentation.

4.1 Software Foundation

To develop a software platform for experimentation with tremor reduction and digital signal processing techniques, an existing open source project was identified that provides a Nintendo Wii-based mechanism for simulating a laser pointer on a presentation screen (Figure 1). The software is called "Wiimote Presenter" and is implemented in the C# programming language. [10]



Buttons
A: WP40 Toggle IR pointer enabl
B: WP41 Momentary IR pointer er
Up: 0x26 Up
Down: 0x28 Down
Left: 0x25 Left
Right: 0x27 Right
-: WP60 Left mouse button
Home: 0x0D Enter
+: WP62 Right mouse button
1: WP70 F5/Escape
2: WP10 Start/Continue all timers

Timers
Timer flash
Timer 1: 10 m 0 s; 0 m 0 s
Timer 2: 14 m 0 s; 0 m 0 s
Timer 3: 15 m 0 s; 0 m 0 s
Timer 4: 0 m 0 s; 0 m 0 s

LEDs	T1	T2	T3	T4	IR	Acc	On	Off
LED1	●	○	○	○	○	○	○	○
LED2	○	●	○	○	○	○	○	○
LED3	○	○	●	○	○	○	○	○
LED4	○	○	○	●	○	○	○	○

Rotation (Accelerometer)
Enable Accelerometer
Threshold 0.5 Hysteresis 0.3
Left
Right
Fwd
Back

IR mouse
Enable IR
✓ Screen 1 1024 x 768
Screen 2

Battery 33%

Figure 1. Wiimote Presenter screen.

The Wiimote Presenter software was modified to incorporate a variety of movement filtering algorithms. Variations of the adaptive gain technique seemed to be most effective, although evaluation of techniques was a secondary goal at this point, the production of a viable research platform for experimentation of these techniques being the primary goal.

4.2 Prototype System Cost

For the system being produced ultimately to be useful to any user, it is desired to consider cost to that user. It is assumed that the user owns or has access to a computer and projection equipment, so those costs are not included in the estimate. In order to use the modified Wiimote Presenter software (free), a Wiimote remote control device (Figure 2, $24 US) a Bluetooth adapter and receiver (Figure 3, $16 US) are needed. Thus, the overall investment in a working, tremor-reduction solution is about $40 US, which is relatively affordable.

Figure 2. Nintendo® Wii™ Controller.

Figure 3. Bluetooth USB adapter.

4.3 Filtering Example

Although the prototype software is under active development, preliminary results showing the visual impact of tremor reduction algorithms are encouraging. To characterize the effect of various tremor reduction approaches, the current technique is to gather descriptions of subjective observations of the effect. The goal of filter is illustrated in a before and after diagram (Figure 4).

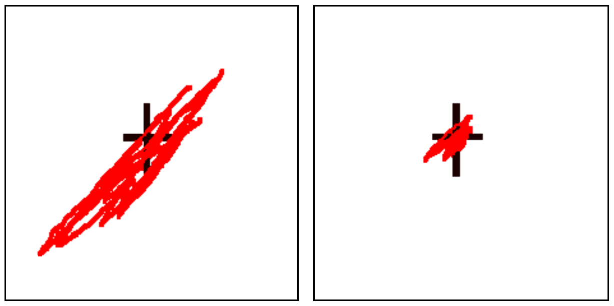


Figure 4. Original and Filtered Tremor trail.

Other measures we intend to incorporate are both empirical and subjective. Empirical measures of pointer coordinates in relation to a central "target" position, measure of velocity and angular regularity of movement will be examined. Subjective measures such as effectiveness of pointer movement smoothing, ability of the pointer to be controlled effectively enough to be useful and not be a distraction, and general ease of use of the software and pointing device while giving a presentation will be explored, as well.

5 Conclusions

There is a definite utility to providing tremor-reduction in software form, as it can apply even as hardware technology and pointing device technology continues rapidly to evolve. A smoother and less jumpy pointer can help eliminate a distraction for an audience during a presentation and provide a presenter who has a tremor of some form with a more comfortable and effective presentation experience. Pointing devices using tremor-reduction software can help those with natural tremors such as Essential Tremor or those with medical conditions that cause other mild to severe tremors. In fact, an approach to tremor filtering may be more widely applicable and useful to a general user, as the tendency to become nervous, and therefore have some mild hand tremor, when making a presentation before an audience is a natural tendency for many speakers.

6 Future Work

The continued refinement of the modified Wiimote Presenter software is of primary interest to us. In addition, some related applications and experiments are on the horizon. A series of experiments is planned to determine the most effective method for tremor-reduction, including the possibility that different forms of tremor will respond better to different tremor filtering algorithms.

Healthcare professionals who treat tremor need more tools to quantify to characterize tremor and therefore measure the effectiveness of various treatments. By extending the functionality of the prototype system, an accurate means of measuring factors such as tremor frequency, amplitude, standard deviation of position in relation to a target, etc., can be developed.

Clearly, a more formula human subjects evaluation is needed, and this will require following rigorous experimental design and approval procedures. Testing the software and a variety of tremor-reduction approaches on subjects with a range of tremor types will provide a means to evaluate which techniques work best for which types of tremor.

7 References

[1] J. Timmer et al., "Characteristics of Hand Tremor Time Series," Biological Cybernetics, vol. 70, no. 1, pp. 75-80, May, 1993.

[2] C. Bodine et al., "Effects of Mouse Tremor Smoothing Adapter on Ease of Computer Mouse Use by Individuals with Essential Tremor: A Pilot Study," Lecture Notes in Computer Science, vol. 4554/2007, pp. 632-636, 2007

[3] E. J. Frett and K. E. Barner, "Accuracy and frequency analysis of multitouch interfaces for individuals with Parkinsonian and essential hand tremor," ACM SIGACCESS Conference on Computers and Accessibility, Baltimore, MD, 2005, pp. 60-67

[4] H. H. Koester et al., "Toward Goldilocks' pointing device: determining a 'just right' gain setting for users with physical impairments," ACM SIGACCESS Conference on Computers and Accessibility, Baltimore, MD, 2005, pp. 84-89

[5] A. Pavlovych and W. Stuerzlinger, "The tradeoff between spatial jitter and latency in pointing tasks," Symposium on Engineering Interactive Computing Systems, Pittsburgh, PA, 2009, pp. 187-196

[6] W. A. Konig et al., "Adaptive pointing: implicit gain adaptation for absolute pointing devices," Conference on Human Factors in Computing Systems, Boston, MA, 2009, pp. 4171-4176

[7] D. Vogel and R. Balakrishnan, "Distant freehand pointing and clicking on very large, high resolution displays," Symposium on User Interface Software and Technology, Seattle, WA, 2005, pp. 33-42

[8] R. S. Adapathya et al., "Device Driver System for Minimizing Adverse Tremor Effects During Use of Pointing Devices," August, 2002

[9] Tremor article, Medline Plus Medical Encyclopedia, National Institutes of Health, URL: http://www.nlm.nih.gov/medlineplus/medlineplus.html, Accessed: May 1, 2011.

[10] J. Smith, Wiimote Presenter software, URL: http://sites.google.com/site/jasonlpsmith/wiimotepresenter, Accessed: February 15, 2011.

Reconstructing 3d Scene From 2d Footage In Architectural Visualization For Camera Tracking And Site Investigation

Ugwummadu Victor Ejiofor.
Artarc Limited, Abuja, Nigeria

Abstract - *The success of camera tracking in 3D studio max is hinged on world measurement of tracked objects where footage is shot. Although, the other packages can project objects' positions automatically and some can generate a file that could be imported into 3D max, but where the footage site is unknown or proves unreachable, a manual operation within the 3D studio max package can reconstruct 3D scene from the 2D footage.*

Keywords: Architectural visualization, camera tracking, 3D studio max, computer graphics, 3D scene, 2D footage.

1 Introduction

In most media arts including architectural visualization, a Computer Graphics (CG) camera that would match the motion of a live camera is obtained through camera tracking. It provides the only route to rotoscoping and aid site investigation where the original site is lost or proves unreachable. Voodoo, boujou, maya, and 3D max are some CG packages with the camera tracking utilities. In any case, a fundamental operation in camera tracking is to reconstruct objects' positions in 3D scene as they are in world space. 3D max among other packages provides very accurate tracking operation but does not reconstruct objects' positions automatically rather it relies on track markers, manually reconstructed 3D scene and taking world measurement where footage is shot. This is a huge problem because the CG artist scarcely knows where his desired footage is shot. Even if he does, circumstances could prevent access and where tracked objects are far apart, taking measurements is discouraging. The big question therefore is how can objects' positions in 3D scene be derived where the only available information is a photo or video footage?

Although, the other packages can project objects' positions automatically and some can generate a file that could be imported into 3D max but the accuracy and reliability for architectural visualization are often not satisfactory. Besides, these packages only project the positions of some selected points but do not really reconstruct the 3D scene as might be required for site investigation. Again, they are licensed packages that make affordability very remote especially for low income earners as often obtained in African continents.

This paper therefore uses diagrams to demonstrate in clear steps how a 3D scene can be reconstructed from a 2D footage to adequately serve the purpose of camera tracking and site investigation in a completely affordable way using only 3D max. Meanwhile, in my paper "Overcoming Premature Terminated Rendering Operation In Disadvantaged Working Environment" for Worldcomp 2009 proceeding, I mentioned that every student of architecture must have had cause to project a 3 dimensional view of an architectural design from the plan and elevations. As it is possible to project 3D view from a point of view by tracing rays of light, it follows that a point of view can be identified by tracing back rays of light from a 3D view. This paper demonstrates that the same principle makes it possible to project scene objects' positions in 3D and further reconstruct objects' forms.

2. Understanding how 3d view of an architectural design is projected from plan and elevations

In order to understand how scene objects, an architectural design for example, can be reconstructed from a footage it would be necessary to understand how such footage is projected by tracing rays of light. Below are the floor plan, front and left side elevation of a simple two bedroom bungalow.

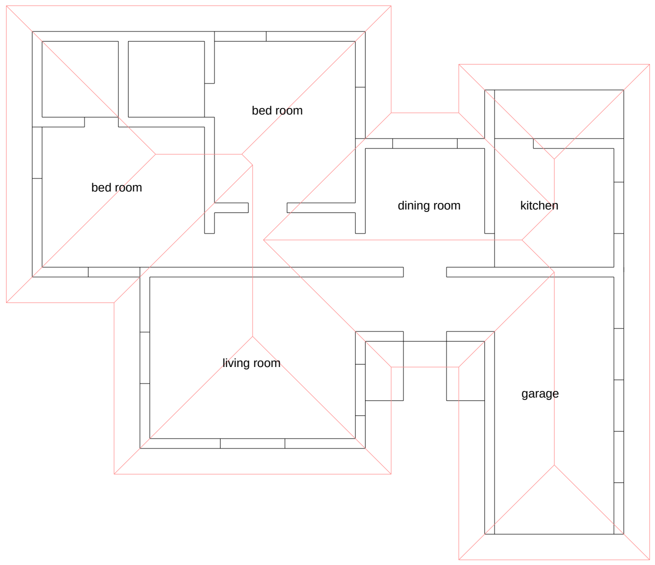


Fig. 1. Floor plan

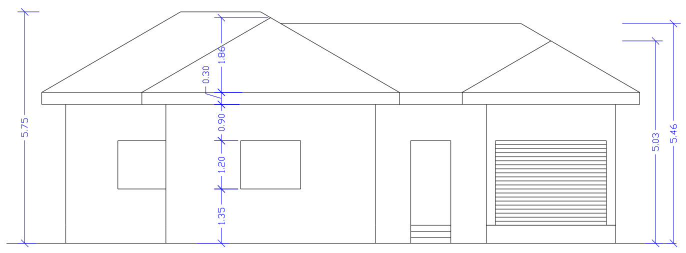


Fig. 2. Front elevation

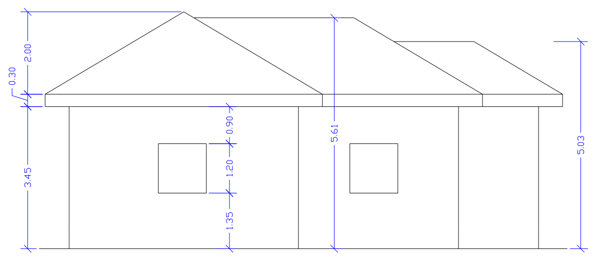


Fig. 3. Left side elevation

2.1 Procedure

2.1.1. Position the floor plan and draw your Direct Line (DL) as shown in fig 4. Choose your View Point (VP) and draw lines PLA and PLB so that they are parallel to the sides of the plan and also pass through VP.

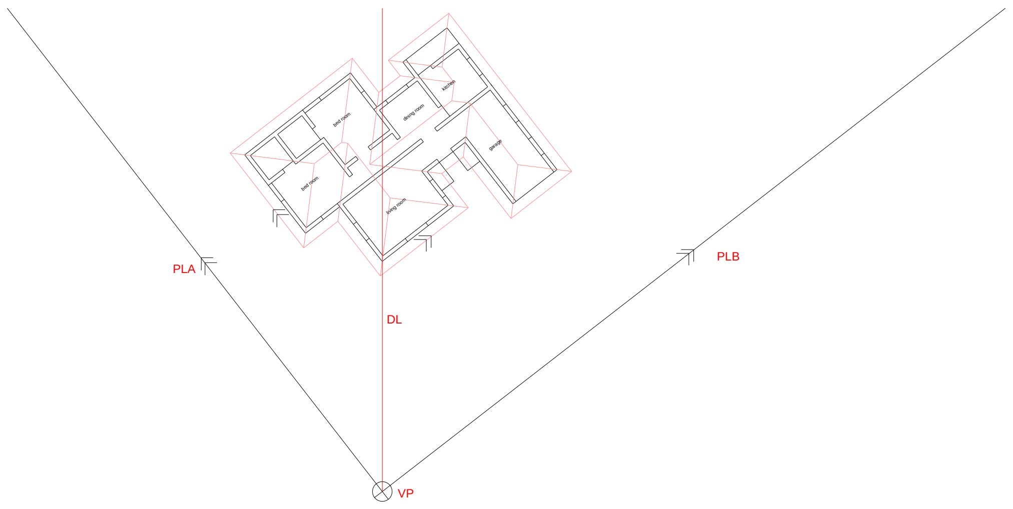


Fig. 4

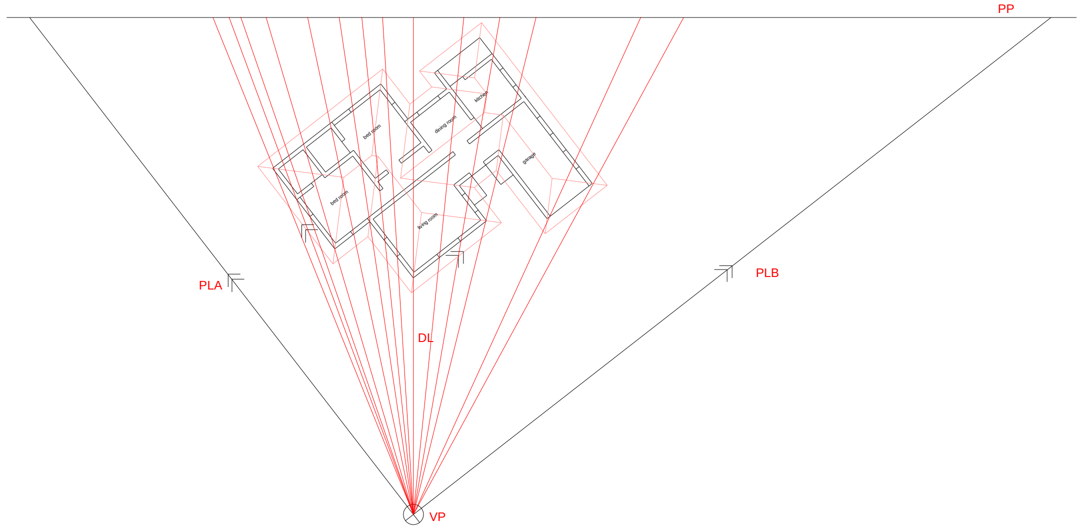


Fig. 5

2.1.2. Draw the horizontal line to represent the Paper Plane (PP) and draw lines that would represent light rays from VP to PP, each line passing through a feature in the floor plan like windows, doors and wall corners as shown in fig 5.

2.1.3. Draw the Eye Level (EL) line and vertical lines from the intersections of PLA and PLB with PP to meet EL at VP1 and VP2. VP1 and VP2 would form the two varnishing points. Draw the other vertical lines from where the light rays meet PP as shown in fig 6.

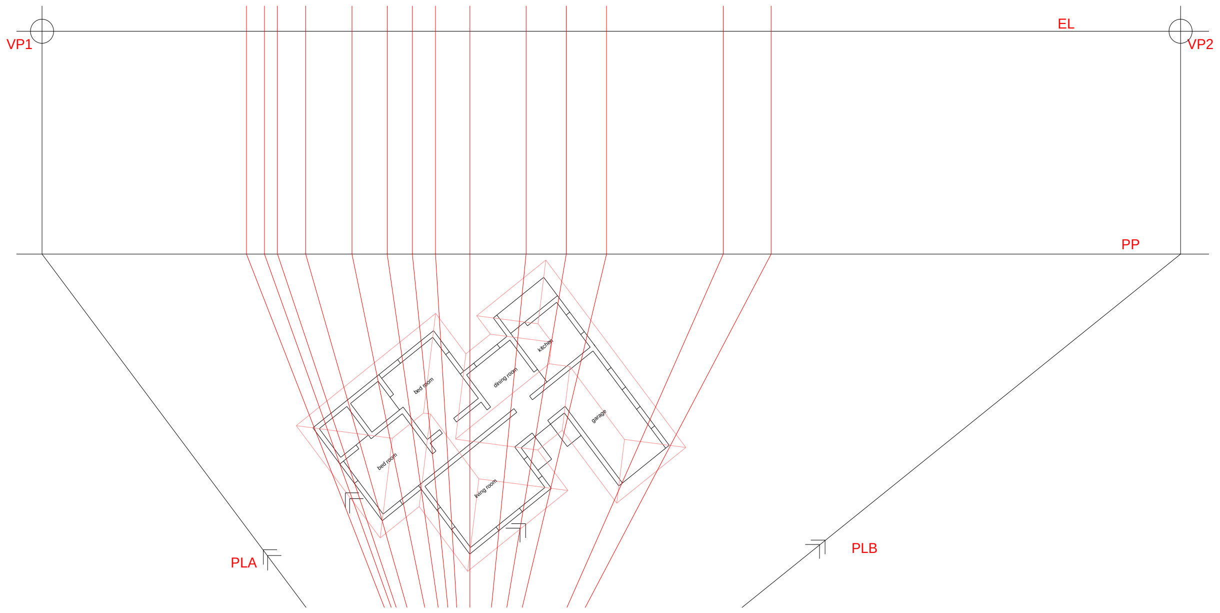


Fig. 6

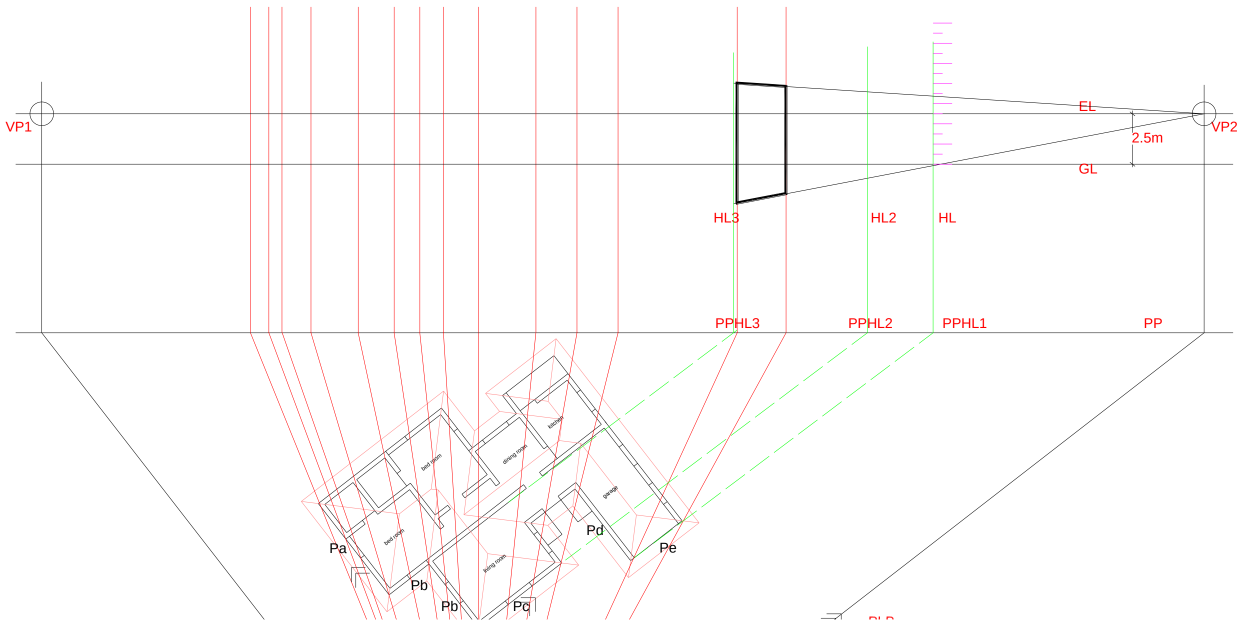


Fig. 7

2.1.4. Create the Ground Line (GL) about 2.5 meter below the eye level EL as shown in fig7 and project the length of a wall to meet PP at PPHL1. The vertical projection would form the building Height Line (HL) from which most height measurements would be taken.

Tips: For convenience, a number of height lines could be made for different wall planes as shown in fig 7 and the height line could further be calibrated. The figure also shows that Pe' plane is projected by drawing lines from the varnishing point P2 through the height HL to the verticals representing the corners of that plane.

2.1.5. Project the other planes from VP2 if the plane is parallel to PLB but if parallel to PLA, project it from VP1 as shown in fig 8 and project the details as in fig 9

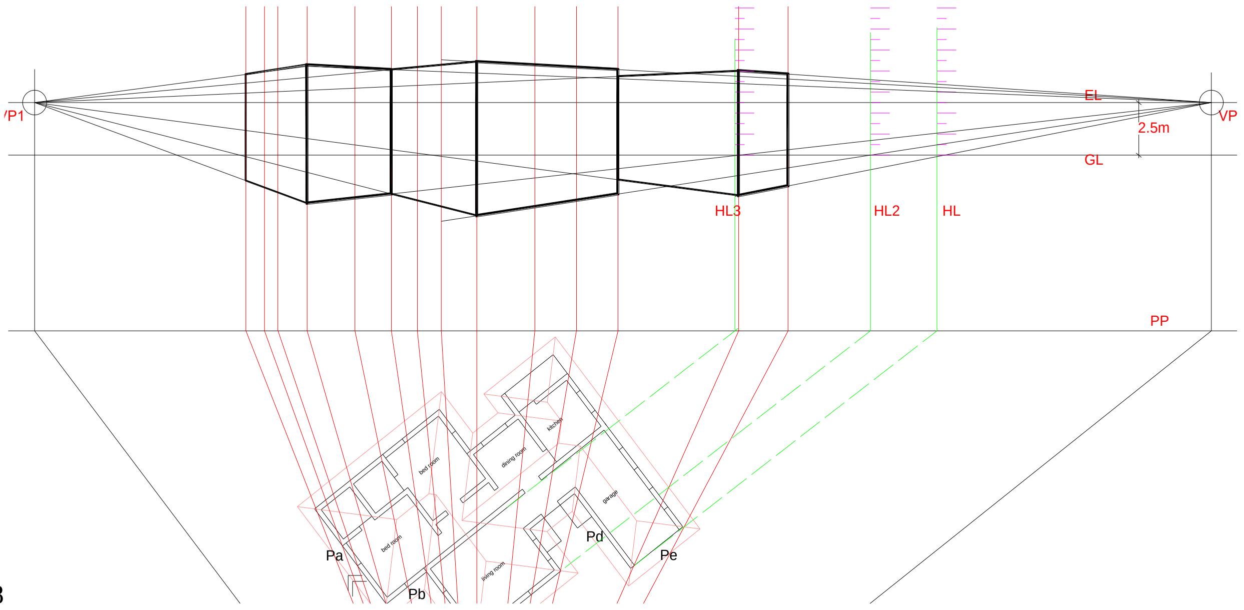


Fig. 8

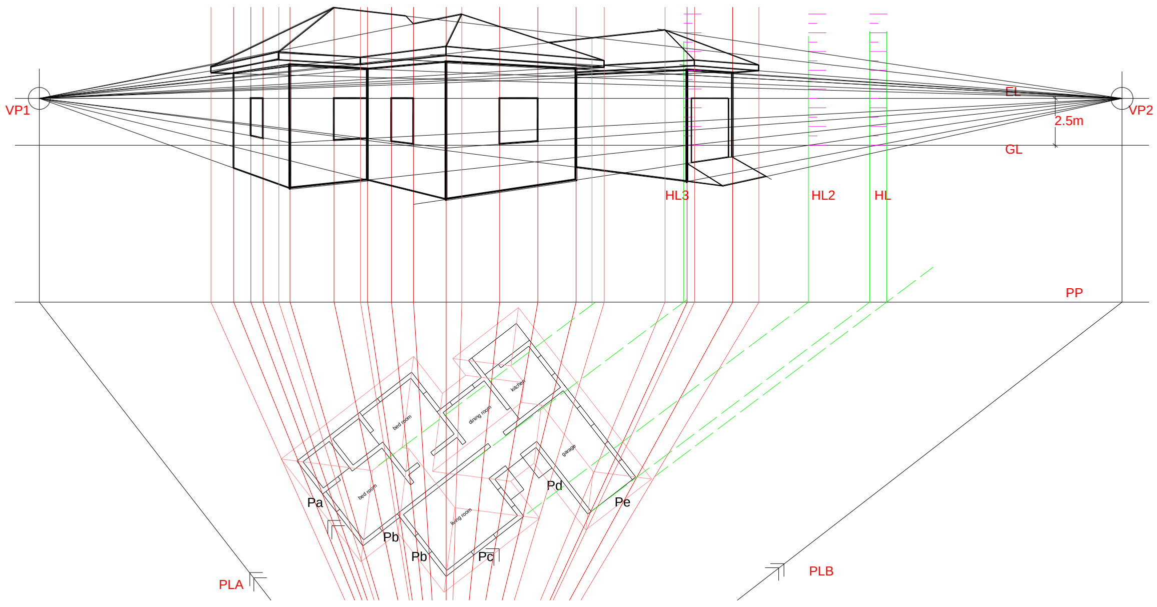


Fig.9

2.1.6. Clean up.

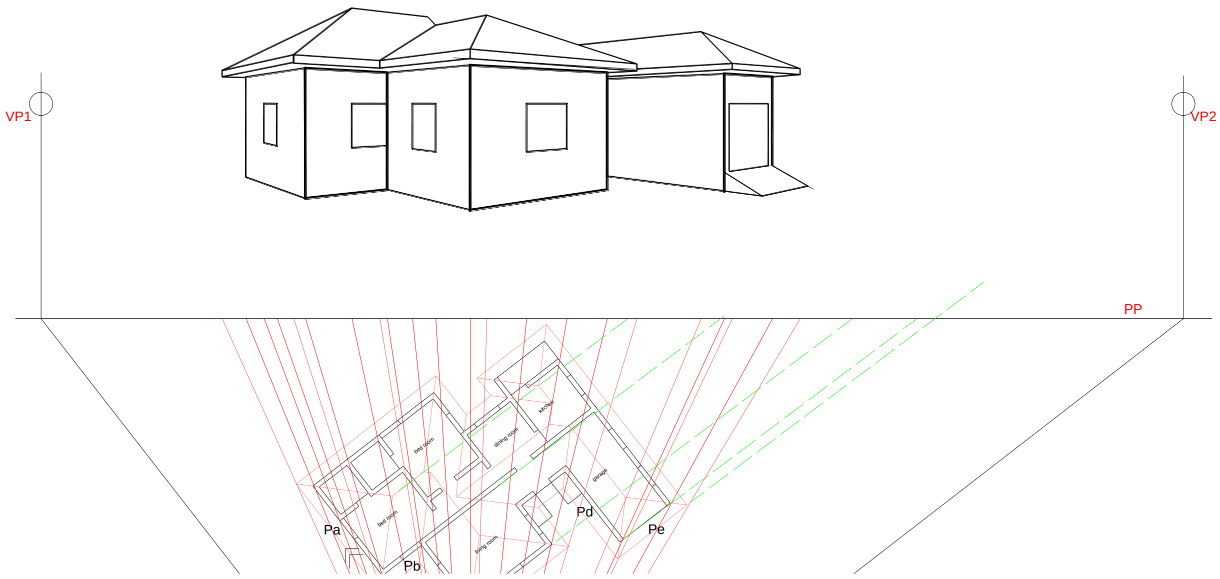


Fig. 10

At this stage, any other desirable details could be projected to make the 3D view more realistic.

3. Reconstructing 3d scene from 2d footage.

The illustrations above show how a camera uses the interaction between light rays and scene objects to projects an image on a screen which we get as photo or video footage. In order to reconstruct a 3D scene from footage, the whole process would be a reverse operation. Meanwhile, it is important to remember that some photos present only one-point perspective and some present three-point perspective once the camera pitches. In any case, the principle basically remains but just a little more issues need to be considered. Below is a good example of a photo footage which the building can be reconstructed in a 3D scene

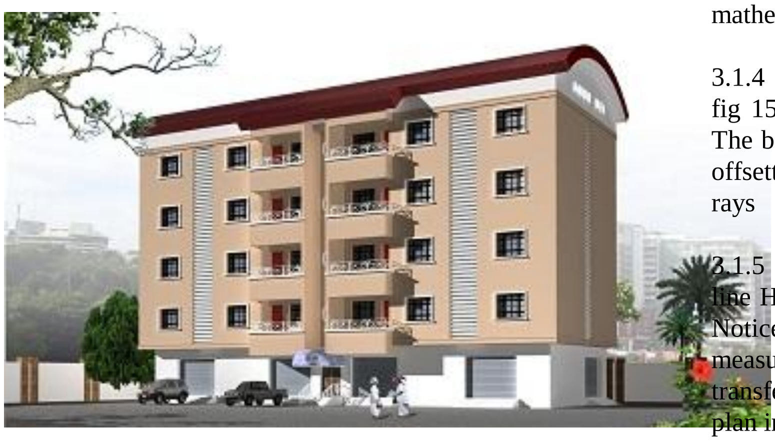


Fig. 11

3.1 Procedure

3.1.1. Trace out the horizontal line to identify the varnishing points VP1 and VP2 and direct line DL that must pass the center of the photo as shown in fig 12.

3.1.2 Draw a paper plane PP line and project vertical lines from the building features to meet PP as shown in fig 13.

3.1.3. Draw a semi circle (C) of diameter PPVP1 and PVP2. Label the intersection of DL and C the view point (VP) being the camera position and draw lines from PPVP1 and PPVP2 to VP. Notice that the angle at VP is a right angle because the angle subtended at the center of circle is mathematically a right angle.

3.1.4 Project lines from the vertical lines to VP as shown in fig 15. These lines represent the light rays from the camera. The basic corners of the building can be identified on plan by offsetting lines PPVP1-VP and PPVP2-VP to meet the light rays

3.1.5 Project the wall plane 'Pa' to meet PP and a vertical line HL from the point of intersection to form the height line. Notice that the relative height of the windows can now be measured from the height line and their values should be transfered to the Z values where the light rays cut the building plan in 3D scene.

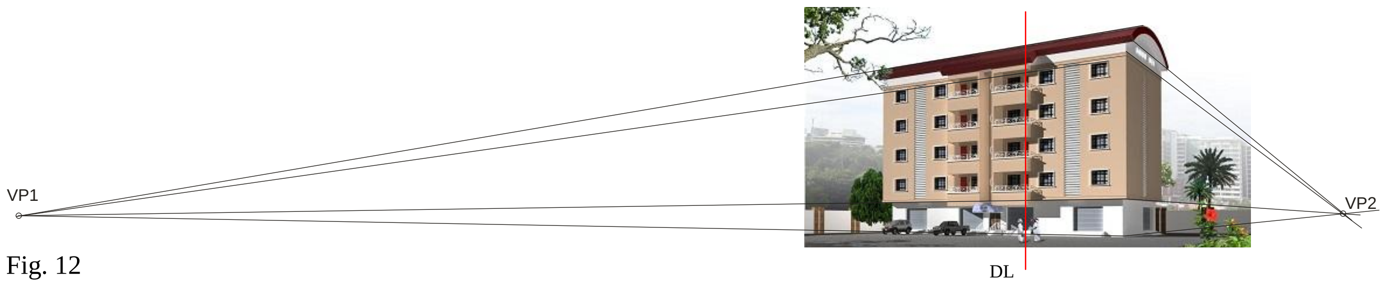


Fig. 12

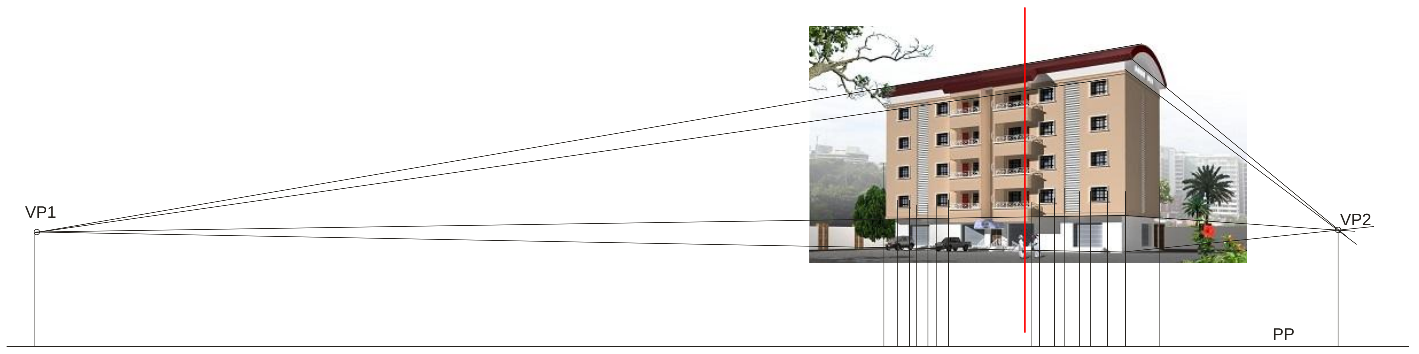


Fig. 13

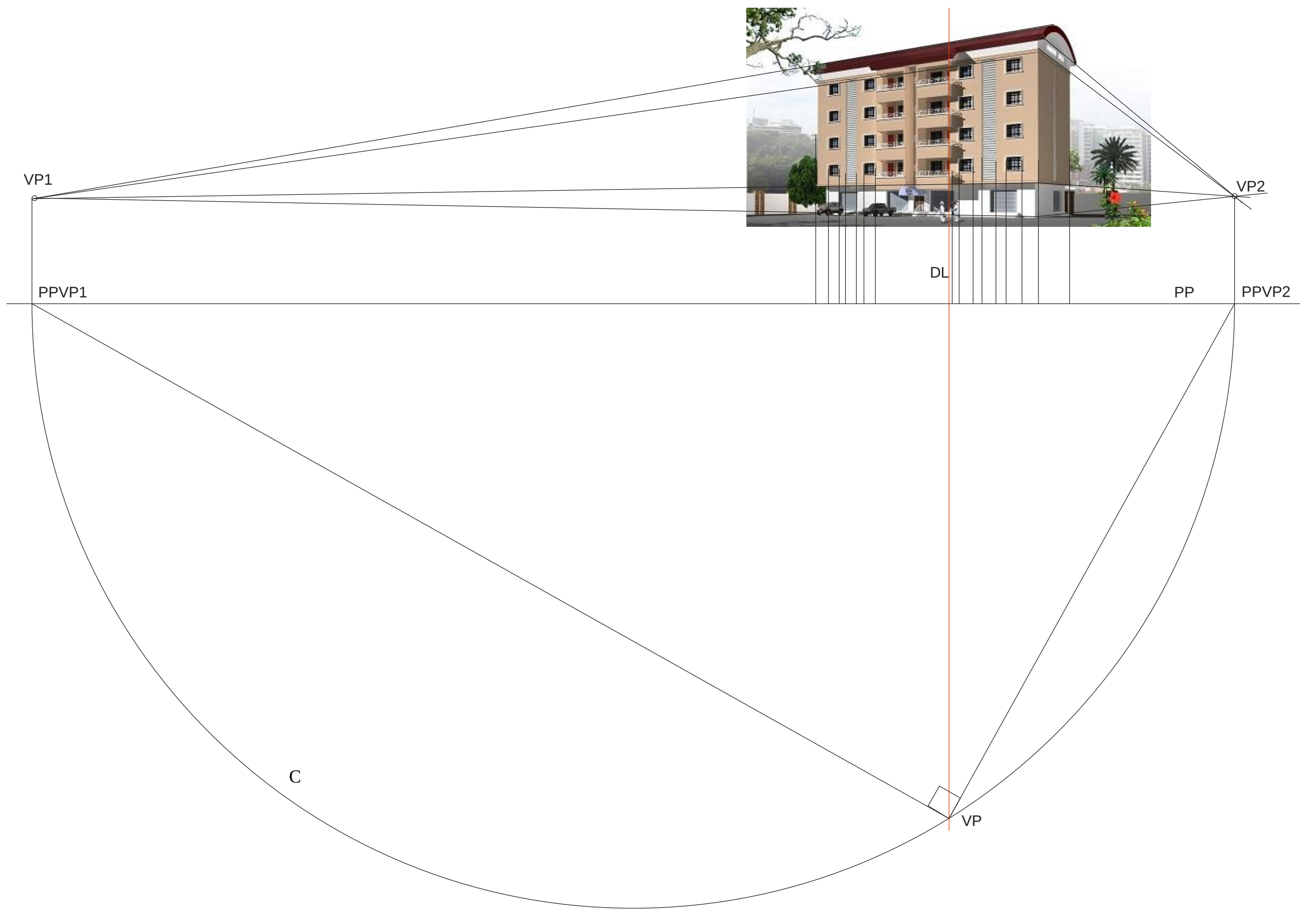


Fig. 14

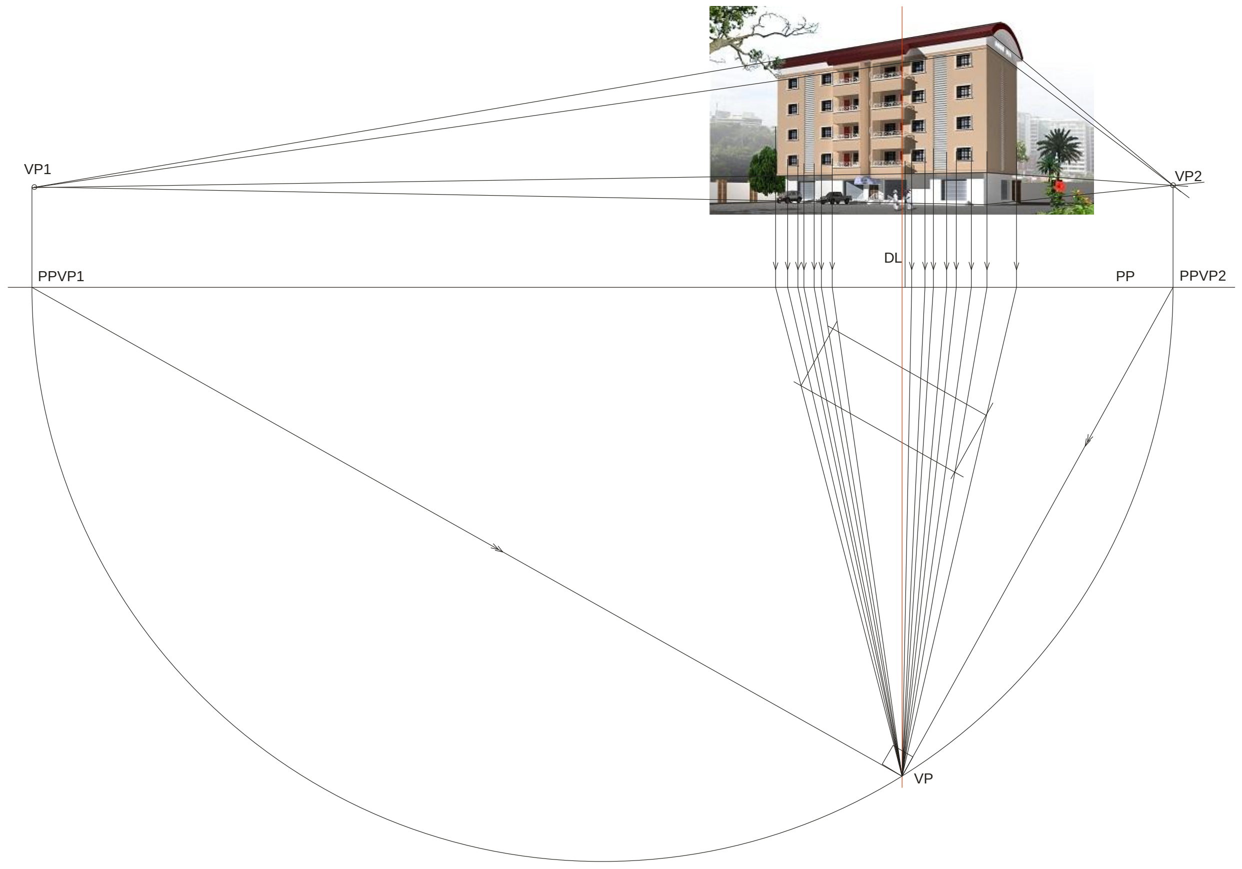
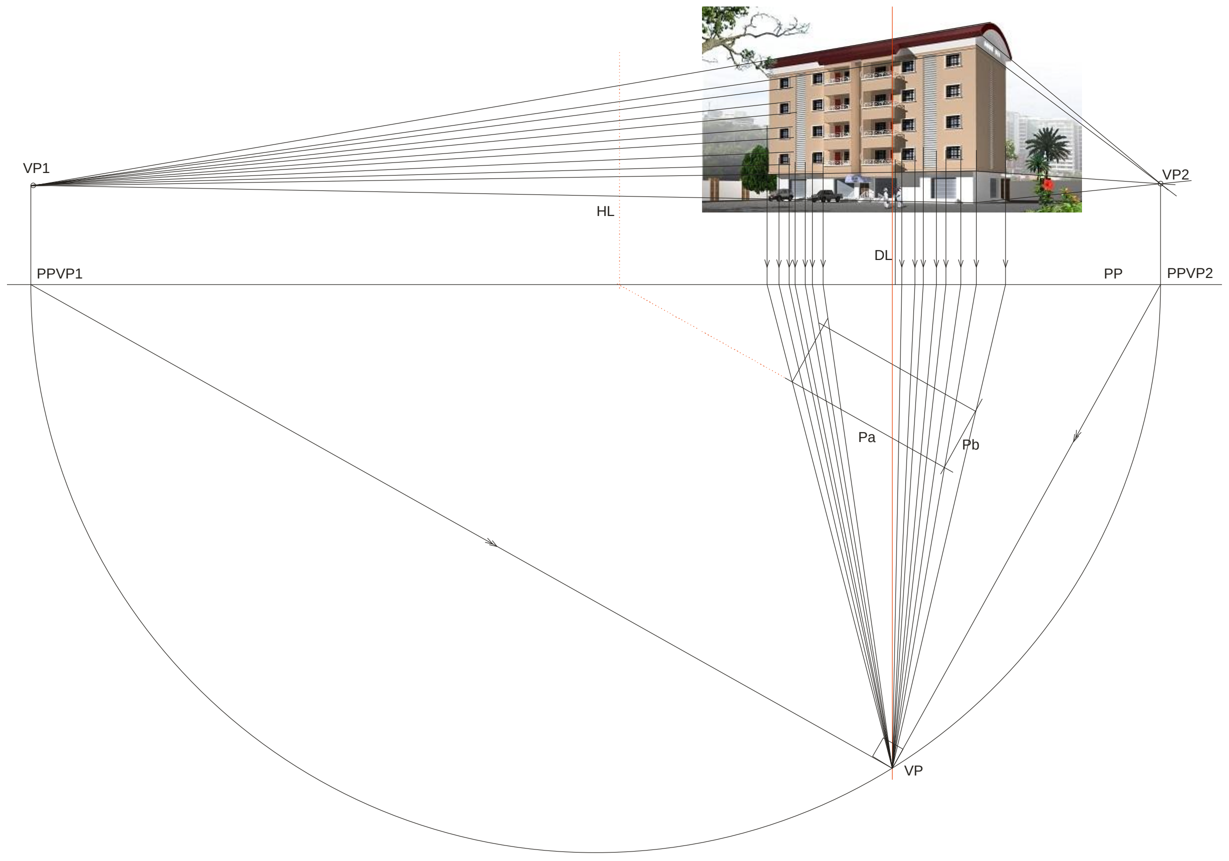

VP1
VP2
DL
PPVP1
PP
PPVP2
VP

Fig.15


VP1
VP2
HL
DL
PPVP1
PP
PPVP2
Pa
Pb
VP

3.2 How to derive the Z values of the building features

It might be necessary to mention that all the above figures and illustrations were viewed directly from top. At this stage, an isometric view is required to appreciate the Z values.

3.2.1 Draw the line 'E' on HL that represent the height of each edge of the building as shown in fig 17.

3.2.2 Rotate it so that it stands upright in 3D scene and place it at it's corresponding positions on plan as represented be E1, E2 and E3 in fig 17.

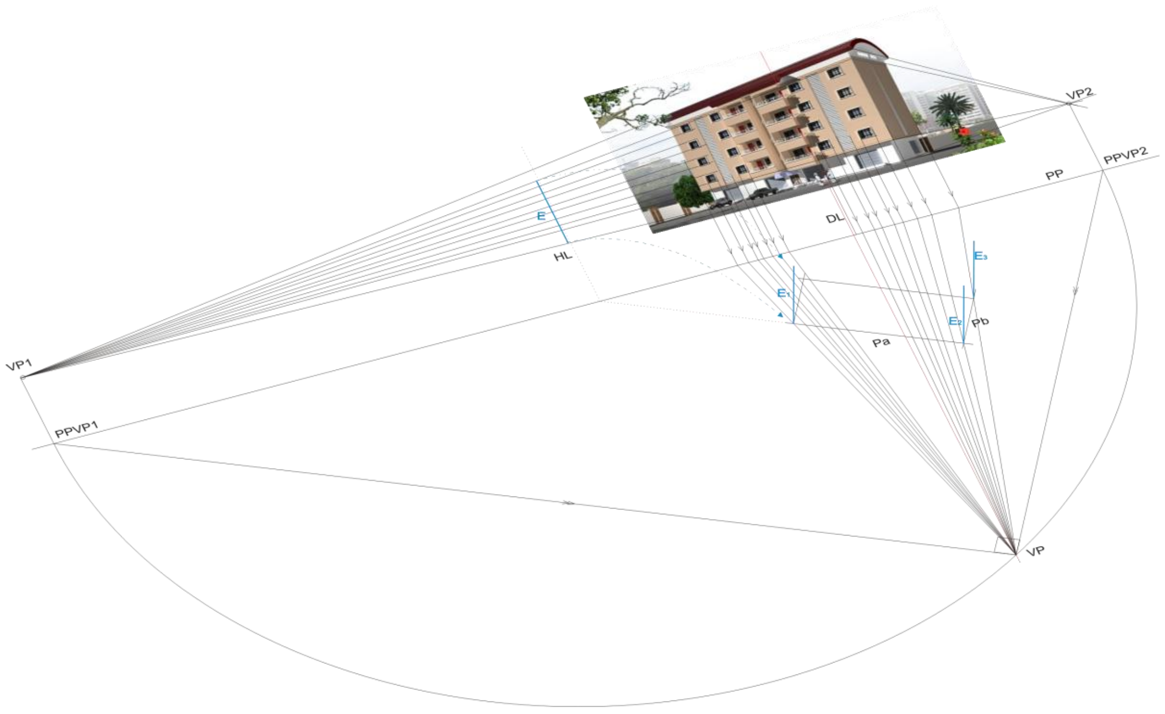


Fig.17

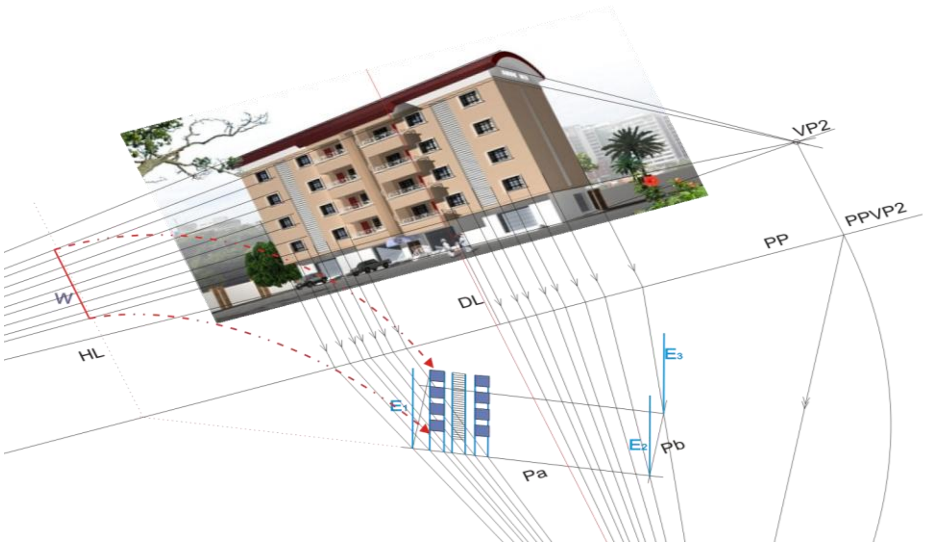


Fig.18

3.2.3 Use the same procedure to locate other features like windows as shown in fig 18, doors, roof etc as shown below.

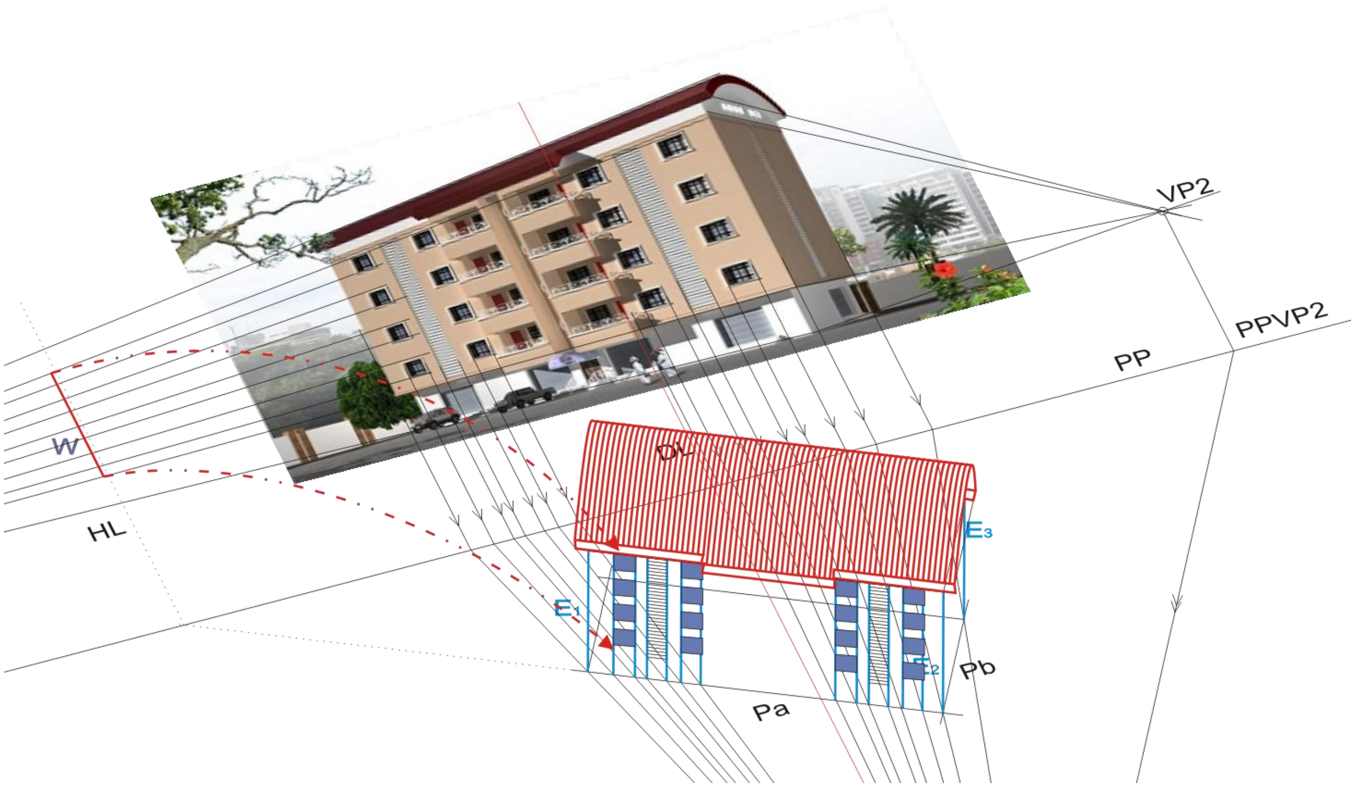


Fig. 19

4 How to scale the scene to match world measurement.

You may have realized the relative positions, scale and proportion of objects were derived without world measurement but the scene often provides a clue that can enable us make a good calculation that would match our scene measurement with the world measurement. For instance, the distance between a building floor to the next floor is often 3 meters. It follows that the distance between a window and the one on the next floor should be 3 meters.
Let 'x' represent scene measurement and 'y' represent world measurement. The formula {x (100)}/y is the scale factor with which the scene would be scaled to match world measurement.

5 Tips

5.1 If objects are scattered in the scene yet they are the best objects to track, create a box around them in such a way that the box relates very well with the perspective. The corresponding positions, scale and proportion of the object can easily be derived in relation to the box.

5.2 While you must use the footage as your scene background in 3D studio Max, set the background image to 'render output'. In the render dialog box, set the output dimension exactly the dimension of the background footage. This operation would prevent distortion which would in-turn falsify certain measurements.

5.3 If the footage provides a noticeable third varnishing point, trace a number of height lines to locate the point. This would provide slanting height lines and height measurements should be made according to their position irrespective of the slanting position.

5.4 Once reconstruction of a footage scene is satisfactorily concluded, it is important to test your accuracy with camera match operation. The position of the automatically generated CG camera in relation with your assumed camera position at 'C' would determine you level of accuracy.

6 Sources of error

6.1 A cropped footage would cause error because the identification of camera position is based on the footage frame.
6.2 An edited footage would cause the projection of embedded elements and after effects can results a falsified scene.

6.3 Avoid projecting object that are far from the camera because grid lines appear to close up and make precise projection very difficult.

6.4 Use the distance between far objects to calculate your scale as short distance would multiply its percentage error.

7 Limitations of this method

7.1 A footage without identifiable varnishing point cannot the reconstructed using this method.

7.2 The method also imposes the objects that would be used for camera tracking since projectable object and point may not be the best object to track.

8 Conclusion

The success of camera tracking in 3D studio max is obviously hinged on world measurement of tracked objects from where footage is shot. The importance of this is not really made in help manuals and online helps but where the footage site is unknown or proves unreachable, a manual operation within the 3D studio max package as explained in this paper can reconstruct 3D scene from the 2D footage without revisiting footage site as claimed compulsory. Although, this paper only offer a manual reconstruction of 3D scene but has tactfully avoided the cost of third party package, rekindled the hope of tracking a footage of a lost site and in addition to accomplishing camera tracking, the investigation of a lost site is also made possible using this method.

Directing a Visualization ala Kubrick

A. Hans-Peter Bischof, and B. Alexander Dong
Center for Computational Relativity and Gravitation, Rochester Institute of Technology, Rochester, NY, USA

Abstract— *Visualization describes the process of converting numbers into a form we can see visually. In order to make the images more comprehensible, it is important to be able to change the visible aspects of an image dynamically. For example, the position of the camera can be moved to focus the attention of the viewer, or the transparency of objects can be changed to expose otherwise hidden features. Visualizations are typically not composed of a single image because they usually contain a time element, meaing the image varies from one time to the next. Movies are the typical result of visualization attempt. This paper discusses a language and visualization environment which allows modifications to the visualization parameters, i.e. transparency and color of the object, camera position, camera focus point, camera up-vector, and surface luminance; in other words every visual attribute is controlled by a program. This language, although it has the ability to control all of these variables, is also extremely simple and can be learned in less than ten minutes. This paper describes the philosophy of our approach.*

Keywords: Visualizations, Directing Languages, Visualization Frameworks

1. Introduction

The input for visualizations is usually a data set based on time or some other scalar value. It is often the case that some parts of the simulation or experiment are of interest while others are repetitive or extraneous. A visualization of the data will most likely fast forward through the unneeded parts and move in slow motion through the parts of interest. In this example, we have two different times: visualization time (vt) moves in constant ticks forward (wall time), while simulation time (st) moves in forward in ticks controlled by the program (experiment time). The simulation time can be described as a function of the visualization time: *st = vt(t)* means that both times are moving forward at the same rate, and *st = 2 * vt(t)* means that the simulation time is twice as fast as the visualization time; an example of fast-forwarding. A more complicated scenario could look like:

```
st  =  vt(t)  for  0 <= t  <   1
    # next line no change in data
st  =  1      for  1 <= t  <   2
st  =  vt(t)  for  2 <= t  <   3
```

This scenario shows that vt(t) is not a function in a strict mathematical sense because *vt(t-1) = vt(t-2)*, but *t-1 != t-2* .

This paper describes a language and a visualization environment which allows the user to direct the visualization, similar to how a movie director casts a movie.

2. Visualization Frameworks

Most visualization systems follow the dataflow framework, which was first described by Foulser [1] The language used to express the data flow differs from system to system, but it is usually expressed as data flowing from one component to another and is occasionally in the form of a mathematical funct ion such as f(g(h(data))) == image Most visualization systems are programmed via a graphical programming language [2] [5]Each component typically has a variety of input and output channels, as well as argument inputs in ord er to fine tune the behavior of a component Individual components are typically triggered by events, and when all required input channels have rec eived the data they need to perform their required operations Many visualization systems like vish[5] NASA's Scientific Visualization Studio[6] VisIt[7] are capable of creating stunning visual images However, they do not have the ability to alter the image creation process with an external script; th ey are not able to change the transparency to make hidden objects visible at a given point in time, or speed up or slow down specific sections of the visualization to skip extraneous parts or focus on other parts. One of our latest visualization projects involved visualizing the merging of a large black hole with a tiny black hole[11]. In the visualization, the black holes were represented as spheres, their respective masses corresponding to the radius of their sphereThe problem was that the tiny black hole was only a single pixel on the screen and, thus, barely visible. One solution could have been to increase the size of the tiny black hole; however, this would have led to a visualization that does not do justice to the scienceWe chose to do the following: the opening scene showed the black holes in their correct proportions, meaning the tiny black hole was near invisibleThe visualization then proceeded to move the camera position towards the tiny black hole, zooming into the new, now visible, tiny black hole A bigger sphere representing the tiny black hole then became visible, and then the camera moved back to its the original position All of the described modification here was driven by a program The rest of the paper describes the programming language and visualization environment used.

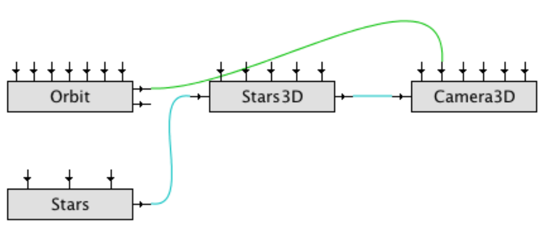


Fig. 1: Hello World

3. KISS Principle

The programs are written via a text-based or graphical programming editor, very similar to the once [5] or [1] The Spiegel visualization framework applies the KISS principle[3]. A program written in this simple programming language is interpreted by the framework, which creates the individual components, connects the input channels with the output channels, and supplies argument values A component will start performing its task when an input channel or an argument changes its value, and when all required values are present A simple Hello World program is shown in Figure 1.

The data flows along the horizontal lines and argument values are supplied via the vertical lines on top of each component The Camera component gets the data from the Stars3D component, while the Orbit component output is con nected to the position argument of the Camera3D component The output of the Orbit component is a series of (x,y,z) points, which move the camera on an orbit aro und (0,0,0) and gives the camera a 360 degree view of the object.

4. Directing

A movie moves forward in time that we consider real time A movie director has the option to fast forward and show us grass growing, slow down to show us how a bullet rips through a pumpkin, or freeze time in a way such that Neo [8] remains still in the air as the camera moves around him.

In order to orchestrate the camera movement for the final scene in Titanic [10], James Cameron needed to define the camera positions and focus points for a few specific moments in time The actual camera maneuvering was then based on these anchor points that he found Functions like *position(t)* and *lookAt(t)* provide information about the camera position and focus point at any and all points in time Nevertheless, there may be more than one function used to calculate each individual shot; although the camera could be moving at a constant speed, it could also be speeding up or slowing down at different points in time

Our directing script works in a similar way The Spiegel component, director, directs the other components based on the viewers time. The input for the director component is time, and its outputs are input for the arguments for the other

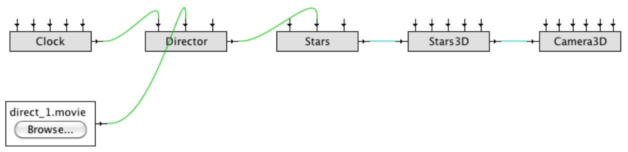


Fig. 2: The First Directed Movie

```
stream time {
    name "time"            # Name of the stream
    type double            # Type of the stream
    interpolator Linear    # Linear interpolation
}                          # between anchor points

time 0 {                   # input time
    time 0.0               # output time
}                          # are equal
time 1 {
    time 1                 # output time
}                          # same delta t for input
                           # and output time
time 2 {
    time 3                 # output time
}                          # rate increase by
                           # factor of 2
time 3 {
        time 3             # output time
}                          # output time
```

Fig. 3: The First Movie Script

Spiegel components The argument values are determined by a program interpreted by the component. This first example shows how time can be directed Figure 2. shows a basic example of the Spiegel program.

The clock component is the viewers time, constantly increasing number The Director component sends out the time as output, which is then used to select the simulation data for the visualization The "director's script" is shown in Figure 3.

The first part defines a stream, giving it a name and a type, and defining what kind of interpolator should be used between anchor points The rest of the file defines anchor points; the interpolator calculates the values in-between the anchor points In this example, if the input time is 0.3, the output time is 0.3, but, if input time 1.5, the output time is 2 Although we did not show it in this example, this allows us to keep the data we visualize unmodified and allows us to move the camera throughout the scenes.

5. Interpolators and Data Types

One important question to ask is what kind of data can actually be manipulated by a directing language The

```
stream <name> {
     name <name>
     type <type>
     interpolator <interpolator>
}
```

Fig. 4: Declaration of the Stream and Example

```
stream position     {
  name cameraPos
  type Point
  interpolator  Linear
}
```

Fig. 5: Declaration of the Stream and Example

```
time 1 {
  cameraPos    1, 0, 1
  outPutTime   1
  color        234, 100, 44
  transparency 0.1
}
time 2 {
  cameraPos    1, 4, 9
  time
   color        255, 0, 0
transparency 0.6
}
```

Fig. 6: Declaration of the Stream and Example

answer to this question is that all argument data types of all components can be manipulated by this directing language, as long as there is at least one interpolator for each data type Our system currently supports the following data types and interpolators: Linear and RxRxR (with Linear and Kochanek-Bartels Cubic Splines Interpolator[12]).

The interpreter is designed in such a way that new types can be easily be added through plug and play technology And because this system is implemented in Java, the system can also easily find new classes and different implementations of the interpreters during run time.

6. Director Language

The director language that we designed only has a couple key words, as well as a very simple structure It follows the “define before use” idea proposed by N. Wirth[4] An example of this in our director language is that the streams must always declared first. Figure 4 shows a formal definition of a stream, Figure 5 shows a use case.

The following blocks declare anchor points The linear position of the anchor point is defined by the input time, and the content of each block defines the values for each input time An example of this is shown in Figure 6.

At input time 1, the position of the camera is at point (1, 0, 1) and moves linearly towards point (1, 4, 9) As shown by this example, many different streams may be defined by the user; through the streams attributes like camera focus point, camera up-vector, and color can be altered in each of these blocks In this example, from time 1 to time 2, the color changes from orange (RGB: 234, 100, 44) to a solid red (RGB: 255, 0, 0) The object also becomes less transparent during this period of time, from an opacity of 0.1 to 0.6 This director script functions in the way shown in Figure 2.

7. Example

The problem was how to create a real 3D movie of a two black hole merger. Both black holes have been represented as spheres, which is obviously trivial to do.The big black hole was 100 times bigger than the smaller black hole.For this simulation, the small back hole was orbiting the larger black hole 3 times, before it merged with the large one. The camera had to be far away (p1) in order to see the complete orbit of the small black hole. From a scientific point of view it was important that the size ratio was represented as is. The final movie can be found at [9].

The small black holes visual representation was one pixel or less as seen from p1, which is a problem from a visualization point of view.The small black hole was barely visible when it was closest to the camera.This would not make a good movie. This is how we solved the problem. The big black hole was centered at (0, 0, 0). The camera was laced at p1 (0, 15, 3) and looked at (0, 0, 0).A grid was place on z = 0. Only the large black hole was visible and the grid at this point in time. Then the camera moved very close to the small black hole, and at the same time the camera focused moved from (0, 0, 0) very close to the small black hole (p2). The grid gave the observer a good visual clue about the movement. After the movement was done. The visual did not change for a little while in order to give the observer a change to understand the scenario. Then a sphere, big enough to be seen from p1, centered identical to the small black hole, went from complete translucent to solid. This image was kept for a moment, than the camera moved back to p1 and focusing at (0, 0, 0).

Neither the small or black hole was moving, which means the simulation time has to be equal to 1 during this process. After the camera reached its final position, the simulation time was moving forward in time.

A movie for the left and the right eye had to be created. The eye distance had to be right in order to achieve the desired 3D effect. The eye distance for p1 and p1 was different by a factor of then. The eye distance was dynamically adjusted during the camera movements.

The code in Figure 7 declares all needed streams. Experiments showed that it was best to use linear interpolator for all streams.

```
stream position {
   name "position"
   type vector
   interpolator Linear
}
stream lookAt {
   name "lookAt"
   type vector
   interpolator Linear
}
stream time {
   name "time"
   type double
   interpolator Linear
}
stream eyeDistance {
   name "eyeDistance"
   type double
   interpolator Linear
}
stream transparency {
   name "transparency"
   type double
   interpolator Linear
}
stream size {
   name "size"
   type double
   interpolator Linear
}
```

Fig. 7: Declaring the Steams

The code in Figure 8 moves the camera and focus point from the original position closer to the small balck hole.

Figure 1 moves the camera back to the original position.

The code in Figure 1 does not move the camera any more, it just moves the simulation time forward.

Figure 11 shows the visual program for the 3D movie. The *Control* componet read the file, created then the necessary output streams which are connected to the arguments of the other components. The whole visualization is driven by the Clock, which starts at 1 and continues ticking to 6623.

8. Conclusion

The director script allows us write relatively simple scripts for much more complicated movies Through it, we are able to control every aspect of the visualization over time Attributes like colors, transparency, object, position, camera focus point, and camera up-vector can be used or altered Although this script allows us much control over the visualization, like any movie script, it also requires a careful crafting of the script The script that we developed is

```
time 1.0 {
   transparency 1
   position (0, 15, 3)  # p1
   lookAt   (0, 0, 0)
   eyeDistance 0.08
   time 1
}
time 576.0 {
   transparency 1
   size 0
   time 1
   eyeDistance 0.008
   position (4.952562636, 0, 0.5) # p2
   lookAt   (4.952562636, 0, 0)
}
time 800.0 {
   transparency 1
   eyeDistance 0.008
   size 0
   time 1
   position (4.952562636, 0, 0.5)
   lookAt   (4.952562636, 0, 0)
}
```

Fig. 8: Close Up

```
time 2000.0 {
   transparency 0
   eyeDistance 0.08
   position (0, 15, 3)
   lookAt   (0, 0, 0)
   time 1
}
```

Fig. 9: Back to the Original Position

excellent for creating movies that expose and reveal many different attributes of the visualization in a perfect manner
Future Work

9. Future Work

After designing and writing many scripts, we realized that our language is a bit too simple Functionalities like of the speeding-up of time is cannot be described in an elegant way Our next goal and future work will be to streamline the language and giving it more functionality.

Acknowledgements

We would like to express our thanks to the members of the Center for Computational Relativity and Gravitation at RIT, especially Manuela Campanelli, Carlos Lousto and Yosef Zlochower They provided the scientific data for the visualizations, as well as interpretations of the visual representations.

```
time 6623.0 {
   transparency 0
   eyeDistance 0.08
   time 4623
   position (0, 15, 3)
}
```

Fig. 10: Simulation moves Forward in Time

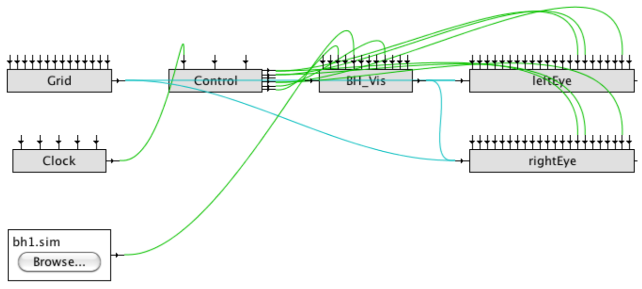


Fig. 11: The 3D Movie Program

10. References

References

[1] David Foulser. “Iris explorer: a framework for investigation” in *SIGGRAPH Comput. Graph.*, 1995, p. 13, 16
[2] Hans-Peter Bischof, Edward Dale, and Tim Peterson. “Spiegel - a Visualization Framework for large and small scale Systems’ in *Proc. MSV* 2006 paper, p. 199, 205
[3] Hans-Peter Bischof and Swathi Annamalai, “The kiss principle applied to dataflow language paradigms for visualization frameworks”, in *Proc. MSV*, 2009, paper, pages 48, 55
[4] Niklaus Wirth, “The programming language pascal” *Acta Inf.*, 1971, p. 35,63.
[5] Vish. (2011) The vish project page. [Online]. Available: http://vish.origo.ethz.ch/wiki/development/
[6] svs. (2011) The scientific visulization studio home page. [Online]. Available: http://svs.gsfc.nasa.gov/
[7] visIt. (2011) The VisIt home page. [Online]. Available: http://wci.llnl.gov/codes/visitnt/
[8] The Matrix. (1999) The Matrix imdb web page. [Online]. Available: http://www.imdb.com/title/tt0133093/
[9] 100:1. (2011) The 100:1 page. [Online]. Available: http://ccrg.rit.edu/movies/
[10] Titanic. (1997) The Titanic imdb page. [Online]. Available: http://www.imdb.com/title/tt0120338//
[11] Manuela Campanelli, Carlos O Lousto, Bruno C Mundim, Hiroyuki Nakano, Yosef Zlochower, and Hans-Peter Bischof. “Advances in simulations of generic black-hole binaries”, *Classical and Quantum Gravity*, vol. 28, number 8.
[12] David Eberly. (1999) Kochanek-Bartels Cubic Splines (TCB Splines). [Online]. Available: http://www.geometrictools.com/Documentation/KBSplines.pdf/

SESSION

AUGMENTED REALITY + WEB TV + IMAGE MORPHING + VIRTUAL REALITY

Chair(s)

Prof. Hamid R. Arabnia

ARTransform: Visualization of Three Dimensional Geometric Transformations in Augmented Reality Environment

Kah Pin Ng, Guat Yew Tan

School of Mathematical Sciences, Universiti Sains Malaysia, Penang, Malaysia

Abstract - *Three dimensional (3D) geometric transformations are basic and important skills when working in computer graphics related areas. Traditional ways in learning the skills include forcing the 3D results to be demonstrated on 2D output media thus challenge the imaginations of the viewers. This paper presents ARTransform – an augmented reality (AR) application for 3D geometric transformations analysis. ARTransform animates 3D geometric transformations in real environment, which enables the viewers to move around the transforming objects and make closer observations. Graphical user interface control panels are developed to provide user-specified geometric transformations and animations. ARTransform adopts the concept of keeping the learning curve gentle to enable the viewers to spend more time in transformation analysis, thus, user interaction is achieved by conventional input devices via keyboard and mouse. ARTransform is still at its infancy and much work remains to be done. User-test on the application will be carried out when more features are added to the application.*

Keywords: Augmented Reality, Geometric Transformation

1 Motivation and Related Work

Geometric transformation is an important component in computer graphics related areas, such as animation, computer vision, image processing, geometric modeling, pattern recognition, etc. It deals with changing the geometric descriptions of an object in terms of position, orientation or size, and adopts the nature of affine geometry where the object's properties remain unchanged after the transformation [1]. The traditional way to view the geometric transformation is paper-based, i.e. using pen and paper to draw and animate the transformed object with the imagination of the viewer. Nowadays, more common and sophisticated ways are using virtual reality (VR) and computer animation to create motion parallax especially while transforming a three-dimensional (3D) object, to achieve the depth illusion. Geometric transformation functions are the most basic functions available in almost all well-known graphics packages, e.g. Mathematica, MATLAB, AutoCAD, 3D Studio Max, Maya, Lightwave 3D, etc. The OpenGL tutorial developed by Nate Robin animates OpenGL's geometric transformation functions by rolling the mouse over the functions parameters on the screen to change the position or shape of the object's local axes [2]. However, examples mentioned above display their 3D outputs on a two-dimensional (2D) screen. One major limitation is that, only the front face of the object can be seen, the back of the object can be analyzed visually only by "turning" the object with a keyboard/mouse interactively. An alternative is to display the object in wireframe view on the computer screen, but the wireframe creates too many lines which will confuse the viewer eventually. In consideration of the drawbacks to represent 3D objects on 2D devices, researchers have been actively working towards a solution for 3D object representation in 3D environment, and this is where Augmented Reality (AR) becomes the overwhelming research focus.

The ability of AR to bring the 3D objects into real environment has made it a useful tool in spatial skill training [3]. Construct3D developed in Vienna University of Technology focuses on the education in mathematical and geometry construction process to high school students [4], University of Washington uses AR to teach their third year students earth-sun relationships and found significant overall improvements in students' understandings after the AR exercises [5].

In this paper, we propose an application called ARTransform to visualize the spatial geometric transformation by taking the advantage of viewer's mobility offered by AR to animate the object transformations in the real world. By moving freely in the room, the viewer can see and analyze step-by-step changes in orientation, shape, size and position of the transforming object. The project was inspired while teaching Computer Graphics course to our undergraduate students by using OpenGL's geometric transformation functions. Confusions were always caused by the 3D geometric transformations especially when involving a series of scaling and rotation. By displaying the animated transformation in AR, we believe that the viewer's mobility will help him/her achieve the best learning results.

2 Geometric Transformation Overview

In general, geometric transformation can be classified into two categories, viewing and modeling transformations which deal with changing of the camera/viewer's positions and changing of the object's position respectively [6]. The first version of our project focuses on modeling transformation. The modeling transformation can be expressed as

$$\mathbf{P}' = \mathbf{M} \cdot \mathbf{P} . \qquad (1)$$

where **P**' denotes the transformed coordinates after composite transformation **M** applied to the original coordinates, **P**. The composite transformation is formed by,

$$\mathbf{M} = \mathbf{M}_n \cdot \mathbf{M}_{n-1} \cdot \mathbf{M}_{n-2} \cdot \ldots \cdot \mathbf{M}_3 \cdot \mathbf{M}_2 \cdot \mathbf{M}_1 . \qquad (2)$$

where $\mathbf{M}_k$, $k \in \{1, 2, \ldots, n\}$, denotes individual geometric transformation. The transformation sequence is written in reverse order, i.e. from right to left, being $\mathbf{M}_1$ and $\mathbf{M}_n$ the first and the last transformations respectively, to be applied to an object. Thus, if an object is rotated $\theta°$ anti-clockwise about a space line parallel to the z-axis, with equations $x=x_r$ and $y=y_r$; then it is scaled to (s_x, s_y, s_z) of its original size, by referring to a fixed point (x_f, y_f, z_f), finally translated (t_x, t_y, t_z) units; the transformation sequence is

$$\mathbf{T}(t_x, t_y, t_z) \cdot \mathbf{T}(x_f, y_f, z_f) \cdot \mathbf{S}(s_x, s_y, s_z) \cdot \mathbf{T}(-x_f, -y_f, -z_f) \cdot \mathbf{T}(x_r, y_r, 0) \cdot \mathbf{R}(0, 0, \theta°) \cdot \mathbf{T}(-x_r, -y_r, 0) . \qquad (3)$$

In OpenGL, the above transformation statements are written as,

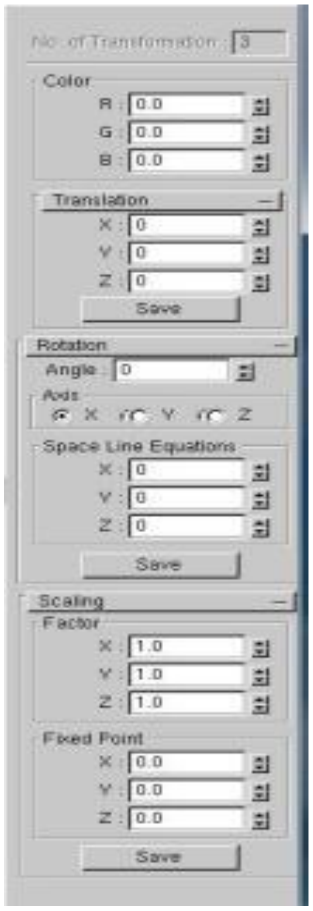
```
glTranslatef(tx, ty, tz);
glTranslatef(xf, yf, zf);
glScalef(sx, sy, sz);
glTranslatef(-xf, -yf, -zf);
glTranslatef(xr, yr, 0);
glRotatef(theta, 0, 0, 1);
glTranslatef(-xr, -yr, 0);
```

Note that the statements in OpenGL require the last transformation to be executed first. Traditionally, we learnt that when an object undergoes a series of geometric transformations, the coordinates of the object are changed depending on types of transformations applied to, and causing the changes in shape and size of the transformed object. This type of transformation is called fixed coordinate system transformation. However, anecdotal evidence showed that this concept is confusing while following the execution sequence by using OpenGL statements, as the intermediate transformed object obtained in OpenGL's execution sequence is the exact reverse of the fixed coordinate system transformation.

An alternate and better way to solve the conceptual confusion is to use local coordinate system transformation. In this case, the local coordinate system of the object is transformed while maintaining the coordinates of the object, instead of applying the transformation operations on the object coordinates as mentioned in fixed coordinate system transformation. The local coordinate system transformations work well by following the sequence of the transformation statements in OpenGL. Each intermediate transformed object is displayed based on the execution sequence as noted. Though the sequence of writing the transformation and the final result are the same, the concept is changed to view the transformation from a different perspective.

3 Application design

Conceptually, ARTransform is designed based on local coordinate system transformation to view in detail the geometric transformations step-by-step. It is built on ARToolkit for its freely available source codes, after analyzing the set of AR tools available [7]. The created AR objects and their transformations are displayed on the ARToolkit marker. Though a normal PC camera is sufficed, we opt to use the 3DVisor's Z800 Pro AR head mounted display to view the transformed objects, mainly to take advantage of the viewer's six degree of freedom offered by AR.



No. of Transformation : 3

Color
R : 0.0
G : 0.0
B : 0.0

Translation
X : 0
Y : 0
Z : 0
Save

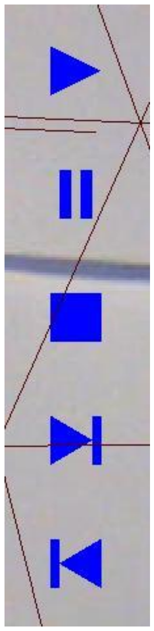
Rotation
Angle : 0
Axis
X [✓] Y Z

Space Line Equations
X : 0
Y : 0
Z : 0
Save

Scaling
Factor
X : 1.0
Y : 1.0
Z : 1.0

Fixed Point
X : 0.0
Y : 0.0
Z : 0.0
Save

(a)



(b)

Fig. 1. Graphical user interface control panels: (a) TRANSCP; (b) ANIMCP.

The aim of ARTransform is to enable the viewer to be focus and grasp the concept of 3D geometric transformation easily and thus we keep manipulation of ARTransform simple during our design and development stages. Further, in order to minimize the abrupt change in user interaction habit, the viewer is guided to adapt to AR environment in stages by using the familiar classic input methods via keyboard and mouse in the first version of ARTransform. For logical input, we have developed two types of easy-to-understand graphical user interface control panels, i.e. transformation control panel (TRANSCP) and animation control panel (ANIMCP). TRANSCP contains options to enable the users to specify geometric transformation desired, ANIMCP contains buttons to animate the transformation edited in TRANSCP. Fig. 1 shows the TRANSCP for keyboard input and ANIMCP which can be displayed in the real environment for mouse input.

After transformation operations have been specified by the user in TRANSCP, each transformation is stored as a single entry in a table named `transData`. Space line rotation and fixed point scaling are decomposed into 3 basic geometric transformations, i.e. translation, rotation/scaling, and translation again; before they are stored as linked-lists of three entries in table `transData`. The desired color of the transformed object is also specified via TRANSCP. Fig. 2(a) shows the attributes for an entry of the table. In Fig. 2(b), index 0 shows a simple translation; index 1 shows decomposition of a composite transformation – fixed point scaling at (50.0, 0.0, –50.0), into three entries; index 2 shows decomposition of a rotation about a space line parallel to *z*-axis, with equation *x*=50 and *y*=50, into three other entries.

The initial version of ARTransform offers a set of basic functions on the ANIMCP to animate the transformations, including start, stop, pause, continue, forward, backward/undo and step-by-step view of the transformations. The design of the animation control buttons adopts the design concept of the buttons on a movie player. The viewer can analyze the same transformation over and over again until he/she has completely understood.

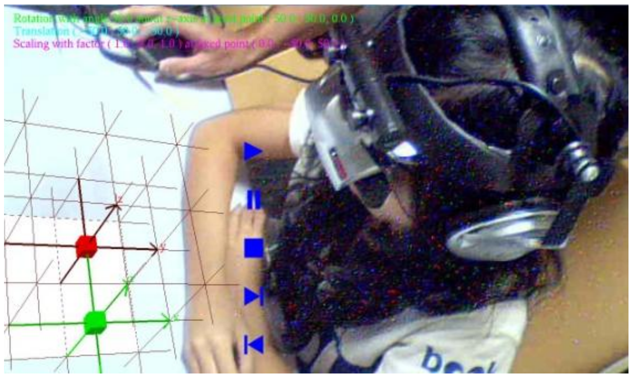


Fig. 3. A student with HMD is observing the transformations animations on ARTransform.

Fig. 3 shows a student with head mounted device observing the transformation animation. Each transformation to be observed is displayed on the top left-hand-corner position in the real environment.

4 Example

Based on the anecdotal evidence collected from the students' feedback in the Computer Graphics course, the main confusion in geometric transformation is caused by applying a series of scaling and rotation immediately one after another. In this section, we demonstrate an example involving rotation about a space line parallel to one of the coordinate axes, followed by fixed point scaling, which was mentioned in sections 2 and 3. The series of transformations are defined as: rotation of 90° anti-clockwise about the space line parallel to *z*-axis with equations *x*=50 and *y*=50; scaling of (1, 2, 1) by referring to a fixed point (50, 0, –50), and finally translation of (–50, –50, 50) units. Each transformation is given a different color to distinguish from the previous transformations.

The OpenGL statements in transformation sequence is given as follows,

```
glTranslatef (-50.0, -50.0, 50.0);
glTranslatef (50.0, 0.0, -50.0);
glScalef (1.0, 2.0, 1.0);
glTranslatef (-50.0, 0.0, 50.0);
glTranslatef (50.0, 50.0, 0.0);
glRotatef (90.0, 0.0, 0.0, 1.0);
glTranslatef (-50.0, -50.0, 0.0);
```

The output given by ARTransform application is displayed in Fig. 4. Fig. 4(a) – (d) show step-by-step transformations. Fig. 4(a) shows the original object in red. Fig. 4(b) shows the translation of (–50.0, –50.0, 50.0) units in each *x*-, *y*- and *z*-directions resulting in green object. Fig. 4(c) shows previously transformed green object is scaled to (1, 2, 1) of its original size at a fixed point (50, 0, –50) and resulting in yellow object. Fig. 4(d) shows previously transformed yellow object is rotated anti-clockwise about a space line parallel to *z*-axis with equations *x*=50 and *y*=50, resulting in magenta object. In this example, the original object in red is displayed as a consistent reference to the transformed objects. The *x*-, *y*- and *z*-grid lines of the original object are turned on, each grid space denotes 100 units and thus the transformed object's location and size can be analyzed when comparing with the original object.

5 Conclusions and Future Work

The hypothesis of seeing the 3D geometric transformations in the real environment can improve the viewer's understanding of geometric transformation was supported by the anecdotal evidence from our students' feedback at the end of the Computer Graphics course each year. In the feedback, the students expressed that the whiteboard teaching method of geometric transformation challenged their imagination and concentration, and that they were always lost in the 2D whiteboard space. Computer aided teaching in virtual reality is easier to understand compared to whiteboard, however, there was a strong urge to pull the object and its transformations out of the 2D display for closer analysis, especially when the original and the transformed objects were blocking each other.

At the moment, ARTransform is still at its infancy and we have not yet used it widely for students' learning purpose. Much works remain to be carried out. In our next version, we plan to include animated graphical explanations on 3D viewing attributes and characteristics, for example, eye/camera position, projection reference point, look-at point, perspective and orthographic transformations with frustum and parallelogram respectively. All animations will be carried out in real environment. The new version will also enhance the user interaction by adding a 3D mouse to interact directly with the AR object. For the new enhancements, the original aim of maintaining the learning curve gentle is kept in mind during the implementation.

6 Acknowledgement

This work is jointly sponsored by Universiti Sains Malaysia's Fellowship and Short Term Grant under reference number FPP 2010/0860 (P3427).

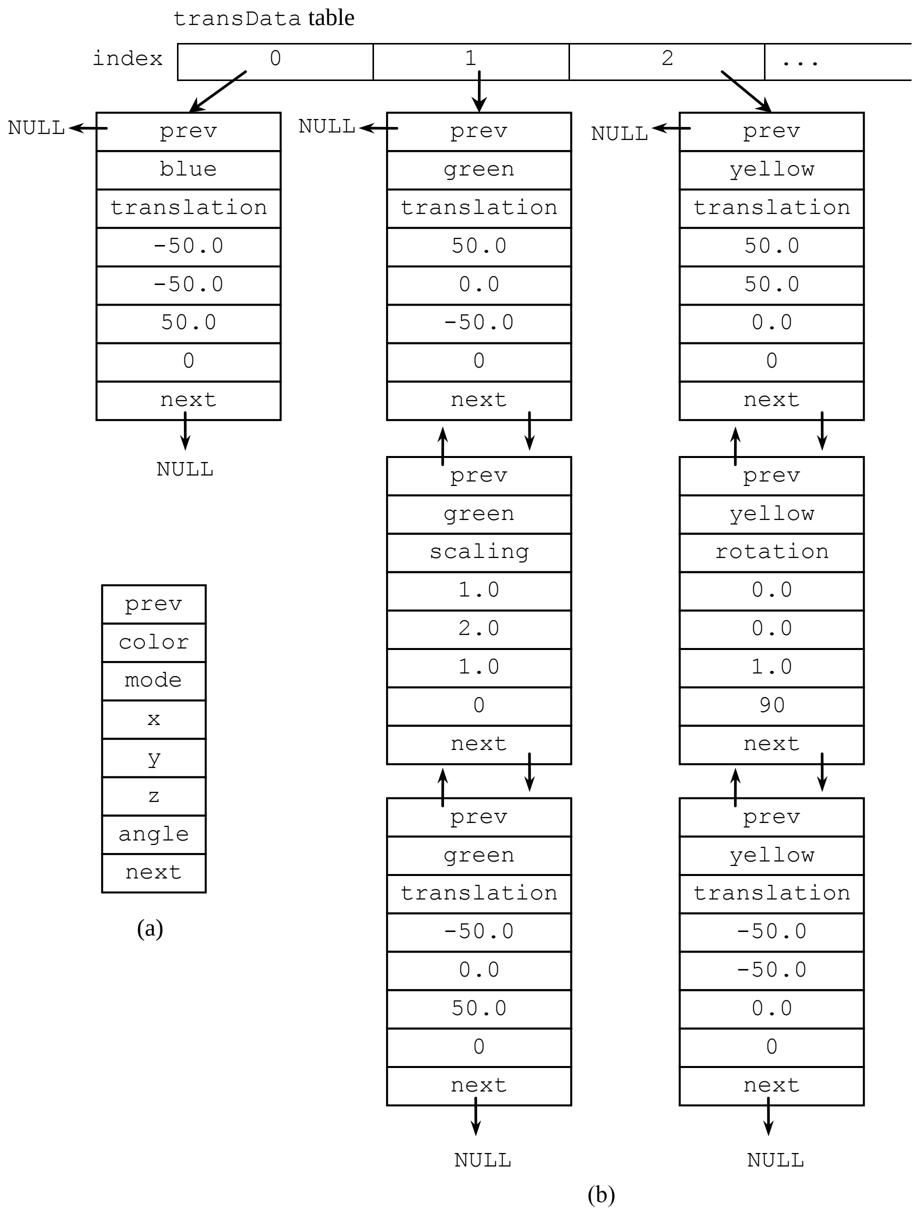
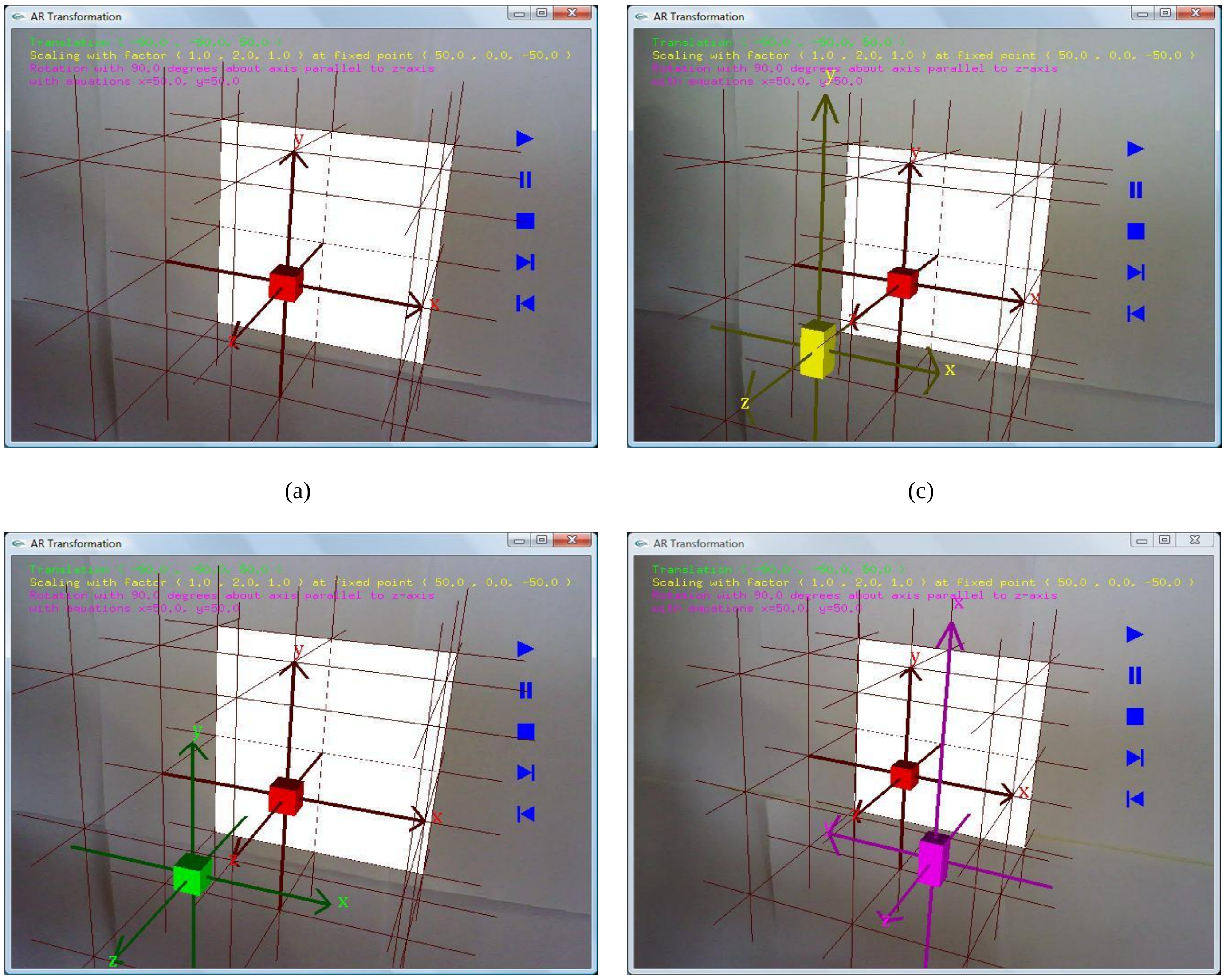


Fig. 2. Structures of `transData` Table: (a) Attributes of one entry in `transData`; (b) Index 0 shows simple transformation, indices 1 and 2 show decomposition of composite transformations to linked-lists.



```
Translation ( -50.0 , -50.0, 50.0 )
Scaling with factor ( 1.0 , 2.0, 1.0 ) at fixed point ( 50.0 , 0.0, -50.0 )
Rotation with 90.0 degrees about axis parallel to z-axis
with equations x=50.0, y=50.0
```

Fig. 4. Step-by-step transformations: (a) Original object; (b) Translation of (–50, –50, 50); (c) Scaling of (1, 2, 1) at fixed point (50, 0, –50); (d) Rotation of 90° anti-clockwise about space parallel to *z*-axis with equations *x*=50 and *y*=50.

7 References

[1] Hearn, D., Baker, P.: Computer Graphics with OpenGL, 3rd Ed. Pearson Prentice Hall (2004)

[2] Nate Robin's OpenGL Tutor, http://www.xmission.com/~nate/tutors.html

[3] Dünser, A., Steinbügl, K., Kaufmann, H., Glück, J.: Virtual and augmented reality as spatial ability training tools. In: 7th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: design centered HCI. ACM Press, New Zealand (2006)

[4] Kaufmann, H.: Construct3D: An Augmented Reality Application for Mathematics and Geometry Education. In: 10th International Conference on Multimedia, pp. 656 – 657. ACM Press, France (2002)

[5] Shelton, B. E., Hedley, N. R.: Using Augmented Reality for Teaching Earth-Sun Relationships to Undergraduate Geography Students. In: 1st IEEE international Augmented Reality Toolkit Workshop, Germany (2002)

[6] Shreiner, D., The Khronos OpenGL ARB Working Group: OpenGL Programming Guide 7th Ed.: The Official Guide to Learning OpenGL, Ver 3.0 and 3.1. Addison-Wesley Professional (2010)

[7] Ng, K.P., Tan, G.Y., Iman, L.Y.: Overview of Augmented Reality Tools. In: 18th National Symposium in Mathematical Science (SKSM), Malaysia (2010)

Live TV-Set with mobile Augmented Reality

Thiemo Kastel[1]
[1]Institute of Media Productions, St. Pölten University of Applied Sciences, St. Pölten, Lower Austria, Austria

Abstract - *To interact as audience to a live television production is today a common procedure. New is to use Augmented Reality support to reach the audience at home. Also new is the situation to have many Television Sets at private houses connected with the internet. Both are offering lots of possibilities for interactive live television productions. The author shows current productions, a by himself produced case study with Augmented Reality at the OpenCampus 2011 and is going into discussion.*

Keywords: Audience Interaction, Broadcast, Streaming, WebTV, Augmented Reality, Virtual Reality

1 Introduction

Nowadays Live - Television Shows often include a way of audience interaction. The idea is to reach the viewer to make him watching the show as longs as possible. One reason for that is often selling advertising time (commercial spots) to finance the station. Another is the concept of the show itself, social participation. Many technologies are in use to interact with the audience. A new way is to use Augmented Reality and SmartTV for interactive experiences, where the technology itself can also be used for different applications. Both are based on the latest technology. Augmented Reality, based on a mobile device App, needs a Wifi or UMTS internet connection to work. SmartTVs are also based on an internet connection. These two technologies are getting more and more common. Telecommunication Companies started to realize the "Last Mile" to the customer with LTE and fiber optics to support higher data rates. The feedback channel to the broadcast center and streaming server doesn't need these high bandwidths. Only video streaming needs nowadays such bandwidths.

2 Interactive television today

In the past there were many good and bad examples of interactive television shows. Planning and realizing a TV format is very extensive and expensive. A few projects showed that interactive television can be very profitable. The challenge is to find a creative idea and to use an easy way of technology for customers. Hollitzky describes, that nowadays there are several business plans to create a successful interactive television project [2]:

Indirect profit

Television stations can create interactive applications to increase customer loyalty. This is very important for PayTV stations. With new concepts of interactivity they acquire new customers and hold regular ones.

Selling content

Television broadcasters sell their videos or applications like PayPerView, PayPerTime or PayPer Download. Sky Austria for example provides additional channels, where viewer can pay for new movies.

Selling interactivity

This means that people can participate in contests by sending SMS or call in. Viewers get the feeling of interactivity and the TV station earns money by its premium-rate telephone service.

Selling products

Interactive television applications offer products while the show is running. For example merchandising or product placements. The viewer has a quick chance to get the product.

Advertising specials

Specialized interactive television systems can personalize the advertising, depending on what a viewer is watching. Advertising specials can also be interactive advertising shows.

Today there are a few interactive television concepts, which show the different ways to attract viewers. Nowadays a good example is icueTV. […] Viewers can interact with programs by voting, requesting additional information or purchasing items related to their viewing experience. The ETS platform handles the back-office functionality required for the fulfillment of interactive events. This allows viewers to select and purchase products offered by the content provider or advertiser. With a few simple clicks on their existing remote control, the TV viewer can purchase a DVD of the show they are watching, buy their favorite actress dress or even download a ringtone of the latest hit song [10]. This means, that icueTV provides the technical solution with an IPTV backchannel for a fluently process. Television broadcasters have to provide interactivity.

ActiveVideo is another interactive television concept of on-demand television in America. The main concept of ActiveVideo is CloudTV. This means, that viewers get their content served as they enter their preferences. Beside normal television shows viewers can also choose mobile services, web videos or social platforms. Content developers are able to store and process video content in the network cloud of ActiveVideo. While watching a television show people can chat with each other on the TV screen. Customers need a set-top box and a broadband internet connection or a cable connection. Founded in 2008, ActiveVideo acquired more than 5 million customers and proofs how successful their concept is [8].

YouView, formerly known as Project Canvas, is a television platform with an internet-connection in the United Kingdom. Project Canvas was founded by the four broadcasters BBC, Channel 4, Channel 5 and ITV plc [11]. The basic idea of YouView is that viewers can watch whatever they want. YouView needs a set-top box, a broadband internet connection and a satellite connection. Viewer can choose from a big variety of television shows. YouView also offers plugins, similar to smartphone apps nowadays. The fact that customers just have to pay for the set-top box shows, that it could be a big success, even if it will be released in 2011 (cf. YouView Questions 2010, no page).

3 Augmented Reality in live performances and TV productions

The simplest form of AR is optically-based, where the system only draws in the AR detail with the rest of the display remaining black for the physical world to be viewed in the HMD [Head Mounted Display] [5]. In order to improve accuracy and achieve higher quality images, my implementations have switched from using optical overlay to video overlay […]. All of the pictures […] were captured using video AR. Performing video overlay is more complicated than in the optical case, and the computer needs to capture the video from a head mounted camera, render this to the display, and then overlay the 3D and 2D graphics on top to produce the final output for the HMD. The implementation still contains the ability to switch to optical overlay, but this is rarely used. Wearable systems are often vaguely described as electronics somehow integrated with clothing. We argue that while this is the way many systems are implemented, the definition of a wearable system is much more broad [4]. Wearable systems are better defined through their functionality as systems that are usable always and everywhere. It is only from such functional definition that viable architectures and concepts for the integration of electronics with the users outfit can be developed. For consumer markets, today mainly smartphones with an AR Browser like the Layar Reality Browser or Metaio’s Junaio App are used for AR.

Marking the first step for true interactive television, German TV show Galileo has partnered with Metaio to beta test out Junaio, the interactive broadcast for iPhone or Android phone users [13]. Viewers could participate in an interactive quiz, get feedback on the pooling results and compare the answer to other viewers. All a user will have to do is select the Galileo channel on his/her smartphone through Junaio application, point the phone camera at the TV screen at close enough distance, and Junaio application will capture the screen image using digital image recognition and start connecting the TV station’s server via internet to kick start the two-way transmission. User could send their response by clicking the available answer on their smartphones touch screen, and could view the answer immediately to check for correct answer.

Zambo is a cross media platform for kids located in Switzerland. A part of this platform is broadcasted by the Swiss television (SF) as the interactive live show “Zambooster” [12]. The interactive approach by this Swiss television show is to interact in a real TV set with virtual Avartars of the kids. The children can answer questions at an interactive website and their avatars will be divided into winner and loser. In the end, there is only one winning kid who is immediately invited to the live show to talk with the host and plays a final game to win a price. While the host of the show talks via phone to the real child, the virtual avatar is mapped through augmented reality markers in the studio and overlays the camera image of the live show [3] and [7].

Woolard et. al. [14] build up a study on the use of an augmented reality application in a BBC news TV studio. In their solution they use a 3D landscape, tanks and missiles, tracked with the help of pattern markers, to present current war scenes interactively. For implementation they use the “ARToolKit” [15] software library with several adaptations to handle Chroma keying, zooming cam lenses and interlaced images.

The Cyber-Illusionist Marco Tempest uses augmented reality for his stage performance “Augmented Reality Magic 1.0”. “I'm using a camera eye, which shows what happens on the table in front of me. Then the computer overlays the image with virtual objects. The cards will start to move or grow things out of them. The state of the development of Augmented Reality is currently known from the advertising. On a real object, such as a magazine, a specific pattern is printed. You can hold this pattern in front of a webcam and the real object and virtual images or enriched animations appear on the screen. With this card trick, I've tried to make it a story, to tell a story. The cards serve as pattern, the computer will recognize these and overlays with digital information” [1]. Tempest encourages the industries to use augmented reality especially for games. He sees the technology only as good enough for games, because the optical see-through glasses, which are relevant for the most augmented reality solutions, are not yet available for the mass [6].

4 Live-TV Productions at the Open Campus 2011

The annual OpenCampus event on 18th + 19th March 2011 at the St. Poelten University of Applied Sciences was used to produce two live TV-Productions with Augmented Reality support. Both Productions were based on a Virtual Studio Production. The Studio Set was a Sports studio and the guests of the two talk shows were on the first day the Soccer Player Andreas Gradinger and a day later the Basketball Coach Hubert Schreiner and one of his Players.

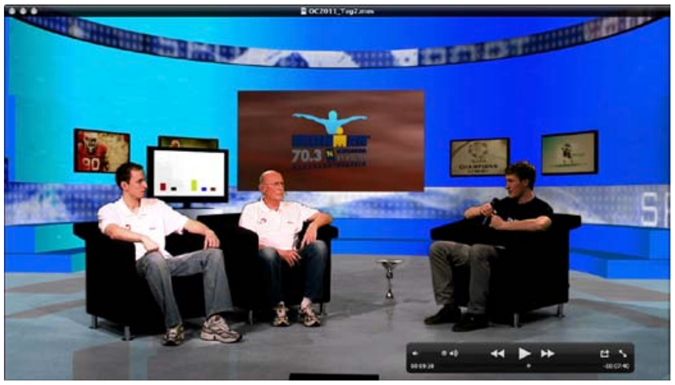


figure 1. Live – TV-Production OpenCampus 2011 – Virtual Studio 19/3/2011

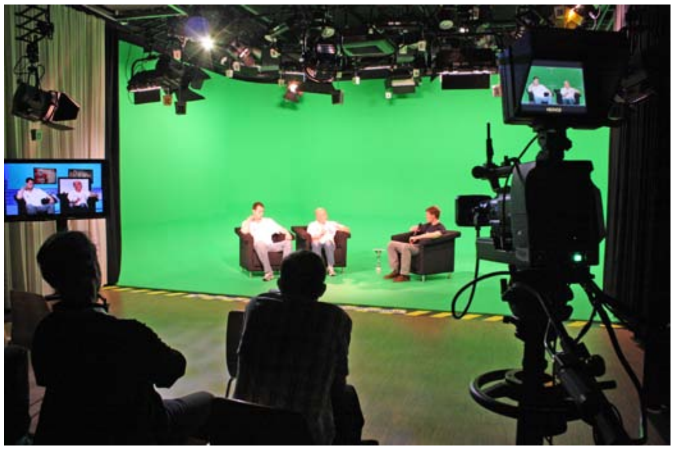


figure 2. Live – TV-Production OpenCampus 2011 – Greenscreen 19/3/2011

5 Results

The interaction by the audience was realized with the Augmented Reality App “Junaio” for Android and iOS smartphones. The Universities logo was used as marker which had to be scanned with the smartphones camera. The feedback after the image recognition was an augmented start screen. This start screen was the basis for all further selections. It was possible to start the live video stream in the Media player on iOS for http-streaming and for Android Handhelds; a website started with rtmp-streaming (Adobe Flash Player).

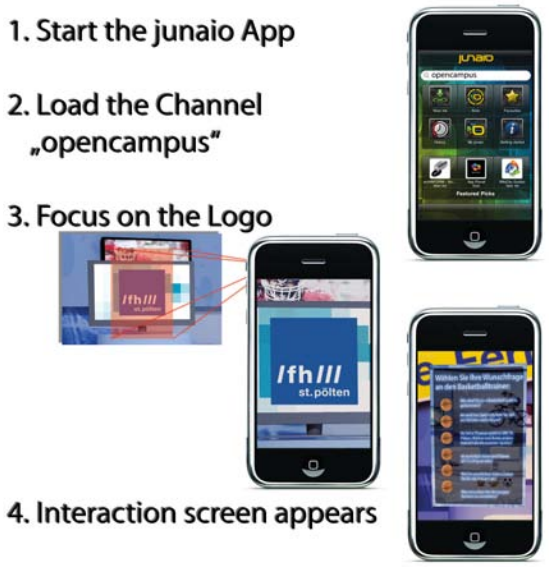
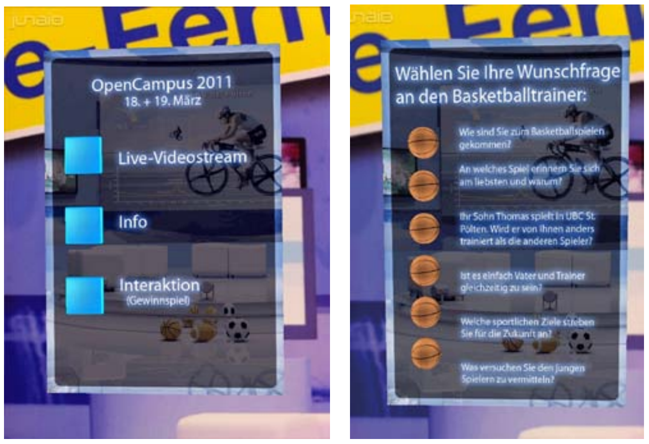

1. Start the junaio App
2. Load the Channel „opencampus“
3. Focus on the Logo
4. Interaction screen appears

figure 3. AR Browser Junaio

A website displayed some general information about the project. Finally, the audience interaction feedback to the live set was accessible and switched the screen. In the next step, a selection screen appeared and the audience could select the interaction. For each one of the three studio guests, one question could be chosen by the user. In addition, the background color of the virtual studio set could be chosen (white, blue or yellow). All buttons on these screens were animated 3D-Objects with custom textures (e.g. soccer ball, basketball). The results of the selections by the audience about the questions to the studio guests have been shown on a Flatscreen in the virtual environment of the Ventuz real-time 3D-Graphics which was in use for the Virtual Set. Finally, the interviewer asked the guests the selected questions.



Wählen Sie Ihre Wunschfrage an den Basketballtrainer:

Wie sind Sie zum Basketballspielen gekommen?

An welches Spiel erinnern Sie sich am liebsten und warum?

Ihr Sohn Thomas spielt in UBC St. Pölten. Wird er von Ihnen anders trainiert als die anderen Spieler?

Ist es einfach Vater und Trainer gleichzeitig zu sein?

Welche sportlichen Ziele streben Sie für die Zukunft an?

Was versuchen Sie den jungen Spielern zu vermitteln?

figure 4-5. Interaction Screens

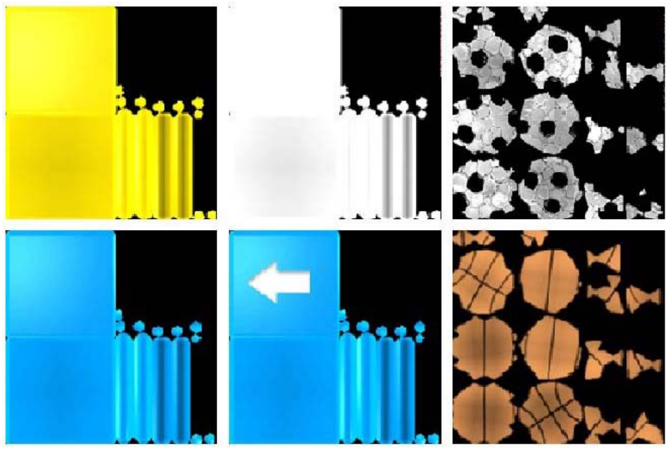


figure 6. Animated Button Textures

5.1 Website statistic

762 Hits on Friday and 624 hits on Saturday were recorded from our apache webserver during the OpenCampus 2011 event. The website visitors' source operating system count for Windows: 303; IPhone: 141; IPAD: 1; MAC OS: 76; UNKOWN: 241 hits on Friday and on Saturday for Windows: 224; IPhone: 78; IPAD: 1; MAC: 50; UNKOWN: 271 hits.

23 visitors' decided to join our lottery. They have had the possibility to use IPhones provided by our faculty or their own. Because of this reuse the unique IP address which reached our webserver count only 18. 26 only played with the OpenCampus junaio App and decided not to join our lottery.

The live video stream was played/stopped 75 times on Friday and 130 times on Saturday by the users. To connect to our Wowza Media Server, 21 unique IP sources use a windows operating system; 12 a Mac OS; 9 an IPhone; 5 an IPad and one an Android like shown in figure 4. In figure 5 the client locations are shown, departed in Austrian districts and other countries.

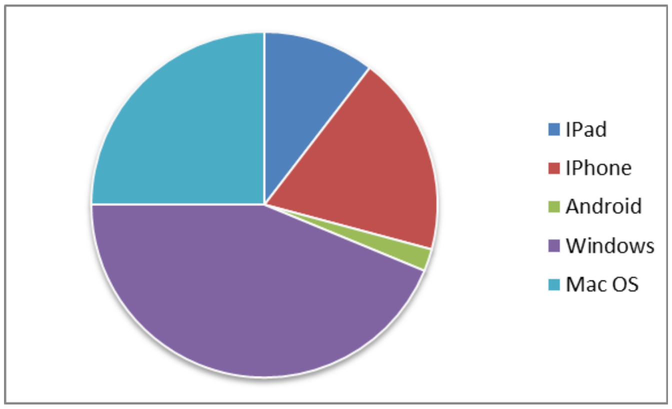


figure 7. Live Stream Visitors Operating Systems

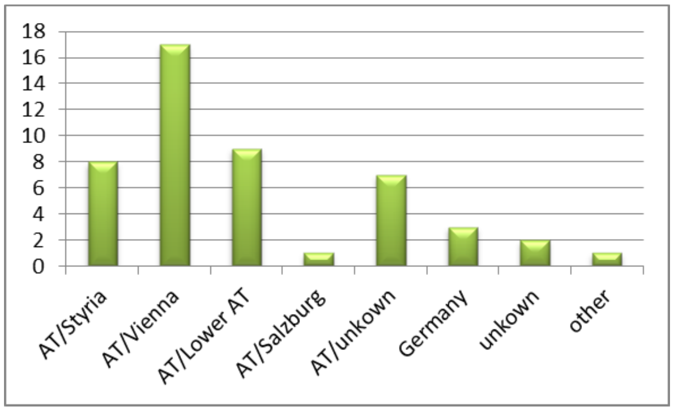


figure 8. Live Stream Visitors Location

5.2 Video On Demand

The two Video productions are available on YouTube: http://www.youtube.com/playlist?list=PL14E2A6799BA1970D

6 Discussion

In this work the author described how Augmented Reality could be used for audience interaction. In the case of the usage with smartphones, a live TV-Production was realized. It was possible to develop audience interactions with iOS and Android smartphones of the latest generation, with the Junaio App and a colored marker. The image recognition, based on Metaio's Junaio App, worked well under stable light conditions, printed markers and also computer screen based markers. Reflections on the markers and too pixilated images haven't been useful. The user had to be aware of these issues to get a stable augmentation of the interaction screens. The usability of the interaction Screens must be very well thought. The Internet connection is another important issue. There must be always a 3G or free accessible Wifi network available to use this kind of interaction. In our case, some of the Phones we were using had a slow internet connection, while we were using them inside of the Videostudio. So it is better to use a Wifi network at home, if there is also limited 3G access. The visitors responded good experiences with this way of interacting with the production crew and looking forward to further broadcast productions.

Finally, the cases study shows that interactive television will get more important by using new technologies like smartphones instead of traditional remote controls of TV Sets. In combination with internet connected TVs, lots of opportunities for interactivity are possible.

7 Acknowledgements

The Author thanks the visitors of the OpenCampus2011 event, the studio guests and the host of the show Dipl.-Ing. Dr. Christian Fabian. In addition Ing. Daniel Holzer, Krzysztof Mikolajczyk and the entire production crew of the OpenCampus 2011 Live Broadcast.

Figure 9. Live – TV-Production OpenCampus 2011 – direction 19/3/2011

8 References

[1] Gross, S. (2010). Marco Tempest: Virtuelle Magie als YouTube-Hit, Vienna: Die Presse: 2010-12-10

[2] Hollitzky, R. (2010). Interactive Television for TV Productions. Bachelor Thesis, University of Applied Sciences St. Poelten

[3] La Tendresse, C. (2011). Realtime 3D - Avatare im Kinderstudio. Speech. Media Future Day <Virtual & Beyond>, Zurich/CH: tcp – technology and production center Switzerland ag – Studio 1: 2011-01-27

[4] Lukowicsz, P. (2007). Bodynet architectures: the meaning of wearability. Proceedings of the ICST 2nd international conference on Body area networks -BodyNets '07

[5] Piekarski, W. (2004). Interactive 3d modelling in outdoor augmented reality worlds. PhD Research Thesis. University of South Australia

[6] Tempest, M. (2011). Cyber Illusion - Is Seeing Believing? Speech. World Economic Forum. Davos: 2011-01-28

[7] WEAVE Magazine (2010), Article: AR-Kinder-TV für den Schweizer Rundfunk, Ulm: Ebner. Volume 05.2010

[8] ActiveVideo Networks (2010) [http://www.activevideo.com/] (2010-09-16)

[9] BBC.co.uk (2010) TV Interactive – Using the red button. [http://www.bbc.co.uk/digital/tv/tv_interactive.shtml] (2010-09-14)

[10] icueTV (2010)
[http://www.icuetv.com/home] (2010-08-18)

[11] Projectcanvas.info (2010)
[http://www.projectcanvas.info/] (2010-09-09)

[12] SF (2011). Schweizer Fernsehen – Zambooster Kids TV.
[http://www.zambo.ch/Sendungen/Zambooster] (2011-02-08)

[13] Xu, E. (2011). Junaio: Two Way Interactive TV Transmission – For iPhone And Android Smartphone Users. [http://thecoolgadgets.com/junaio-two-way-interactive-tv-transmission-for-iphone-and-android-smartphone-users/#ixzz1DxQ3wUQn] (2011-01-28)

[14] Woolard, A. / Vali Lalioti, V. / Hedley, N. / Carrigan, N. / Hammond, M. / Julien, J. (2003). Case Studies in Application of Augmented Reality in Future Media Production. Proceedings of the Second IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR '03)

[15] ARToolKit (2011)
[http://www.hitl.washington.edu/artoolkit/] (2011-07-15)

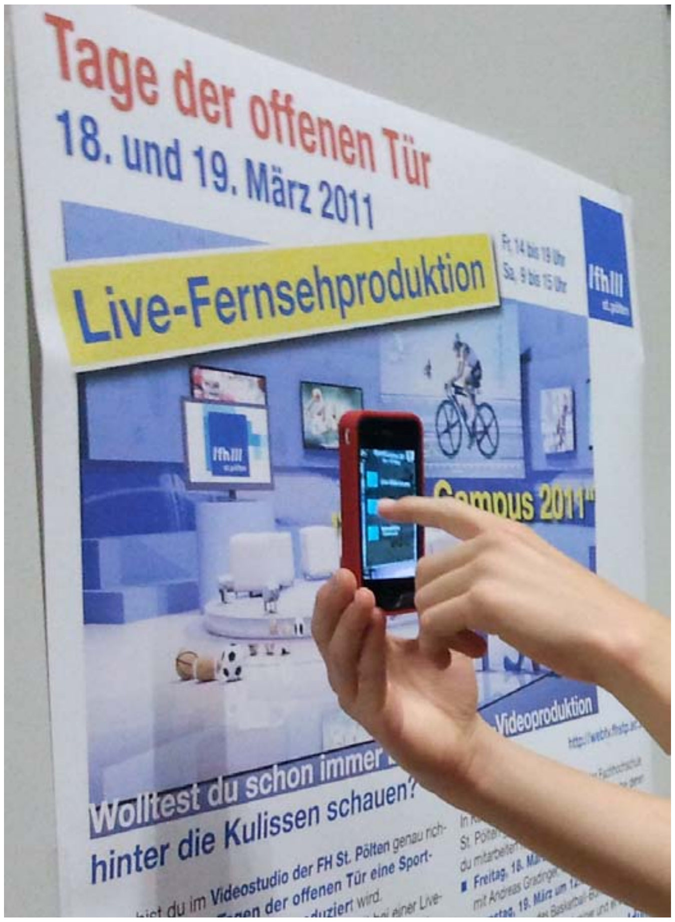


Figure 10. AR OpenCampus Channel in Use

Image Morphing Using Mass-Spring System

D.W. Choi[1] **and C.J. Hwang**[2]
[1]Agency for Defense Development, Daejeon, Korea
[2]Chungnam National University, Daejeon, Korea

Abstract - *A fast and efficient deformation model is essential for realistic simulation of image morphing. In order to create models that are fast to simulation, we use a mass-spring system, which is applied widely in the real time animation of deformable objects. One-to-one deformation is one of the important issues in image morphing. We propose a method of detecting and removing overlaps in the process of deformation. After detecting any overlaps on a node, the external forces are set automatically on its four neighboring nodes until there are no overlaps.*

Keywords: Image morphing, physically based modeling, mass-spring system, preventing overlaps

1 Introduction

Image morphing is a widespread technique in entertainment and animation that generates a smooth transformation from a source image into a target image. Given the source and target images, an animator specifies the corresponding features with points, polylines, and curves on the two images. Feature correspondence is then used to compute warps to interpolate the positions of the features through the in-between sequence. Finally, the animator uses ordinary colour interpolation (i.e. cross-dissolve) to generate the in-between images [1].

Many algorithms have been proposed for image morphing. Some of the most popular approaches are mesh warping, field warping, and energy-based warping. In mesh warping [2], non-uniform meshes are used to specify the image features. These techniques are fast and show good distortion behaviour. However, they have a critical drawback in specifying features. Field morphing [3] uses a pair of lines as feature primitives to simplify the user interface. A pair of lines in the source and target images defines a coordinate mapping between the two images. The mapping of points in the vicinity of the line can be determined by their distance from the line. In the case of multiple line pairs, the field morphing technique calculates the warping of a given point using a weighted sum of the mapping of all line pairs. This technique suffers from unexpected distortions, referred to as 'ghosts'. Energy minimisation methods [4-5] use points, polylines, and curves as the feature specification primitives to reduce the burden of feature specification on animators. These methods derive a C^1- or C^2-continuous and one-to-one warp from a set of feature point pairs. The one-to-one property ensures that the distorted image does not fold back upon itself, but the performance is hampered by its high computational cost.

In this paper, in order to create models that are fast to simulate, we use a mass-spring system, which is applied widely in the real-time animation of deformable objects. Examples include cloth animation [6-7], face animation [8], and soft tissue behaviour in surgery training systems [9-11]. For image morphing, the mass-spring system consists of a mesh of m × n virtual masses. Each mass is connected to its neighbouring masses by massless springs and dampers. To have stable deformation results, two thin plate mass-spring systems are used internally. This internal structure is similar to the thin plate spline (TPS), which is a conventional and widely used method for interpolating surfaces over scattered data [12-13].

Overlaps on the mesh may occur for rapid and large deformations. Overlaps cannot meet the one-to-one property, which is an important issue in image morphing. Several methods have been proposed to prevent overlaps in a mass-spring system. One such method is edge repulsion force [14]. Non-linear edge repulsion springs are added between each node and its opposite edge. These springs exert a repulsion force based on the perpendicular distance of the node from the opposite edge. A penalty force [15-16] is added to moving nodes to prevent overlaps. When a node is too close to its opposite edge, the penalty function is applied to its position. These methods set the edge repulsion or penalty force on the nodes detecting overlaps. However, one spring or penalty force that has been added to a moving node to prevent overlaps cannot exert enough force on its neighbouring nodes in a mass-spring system in the case of large deformations.

In this paper, to prevent overlaps, we propose a new method of detecting and removing them in the deformation. Jacobian determinant is a good method to detect any overlaps. We explain how to calculate the Jacobian determinant directly from the thin plate mass-spring system. After detecting any overlaps on a node, we remove them by setting the external forces on the four neighbouring nodes. The magnitudes of the external forces are automatically computed proportional to the computed differences of displacements, and they are applied repeatedly until there are no overlaps.

2 The Structure of the mass-spring system

A mass-spring system used for the image morphing consists of a mesh of m × n virtual masses. Each mass,

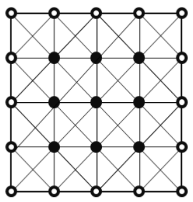


Figure 1. The structure of mass-spring system

hereafter referred to as node, is connected to its eight neighbours by massless springs and dampers. Since the source, target, and in-between images that are generated from two given images for image morphing are equal in size, we regard the outer nodes of the mass spring system as fixed points.

For the mass-spring system, the governing Lagrange's equation is

$$m_i\ddot{x}_i + \gamma_i \dot{x}_i + k_i x_i = f_i + g_i \quad (1)$$

where x_i, $\dot{x}_i$, and $\ddot{x}_i$ are the position, the velocity, and the acceleration vectors of node i, respectively. The variables m_i, γ_i and k_i are scalar values that represent the nodal mass, the damping, and stiffness coefficient at node i. The variable g_i is the spring force vector to be applied on node i through the springs connecting node i, and f_i is the external force vector on node i [17].

To calculate the spring force, g_i, we first compute the spring length at the resting stage, r, from the initial positions of the nodes. The spring forces between a pair of nodes at position x_1 and x_2 with velocity v_1 and v_2 are

$$\begin{aligned} g_1 &= -\left[k_1(|\Delta x| - r) + \gamma_1\left(\frac{\Delta v \cdot \Delta x}{|\Delta x|}\right)\right]\frac{\Delta x}{|\Delta x|} \\ g_2 &= -g_1 \end{aligned} \quad (2)$$

where $\Delta x = x_1 - x_2$ is the actual distance between the two nodes, and $\Delta v = v_1 - v_2$ is the difference between the two nodes' velocities. In addition, Δx and Δv are vector components. The operation $\cdot$ represents an inner product of two vectors.

The explicit Euler scheme is widely used for the simulation of various fields using a mass-spring system. Explicit Euler integration has the benefits of simplicity and computation efficiency [18]. The position $x_i(t + \Delta t)$ and velocity $v_i(t + \Delta t)$ of the mass m_i at time $t + \Delta t$ are computed as

$$v_i(t + \Delta t) = v_i(t) + a_i(t)\Delta t \quad (3)$$
$$x_i(t + \Delta t) = x_i(t) + v_i(t + \Delta t)\Delta t \quad (4)$$

where a_i is the acceleration vector of node i, and Δt is the time step of the evolution.

To stabilize the deformation on a mass-spring system, we use two thin plate mass-spring systems. In this case, we compute the displacements of the *x*- and *y*-components of all the nodes separately when the control nodes (selected by animator) move to new positions. The structure of thin plate mass-spring systems is the same as the normal mass-spring system in Fig. 1. However, the movement of control nodes and external forces are now limited to the direction perpendicular to the surface of the thin plate mass-spring system. Fig. 2 shows the deformation results using thin plate mass-spring systems is globally smoother and more stable than those using normal mass-spring system.

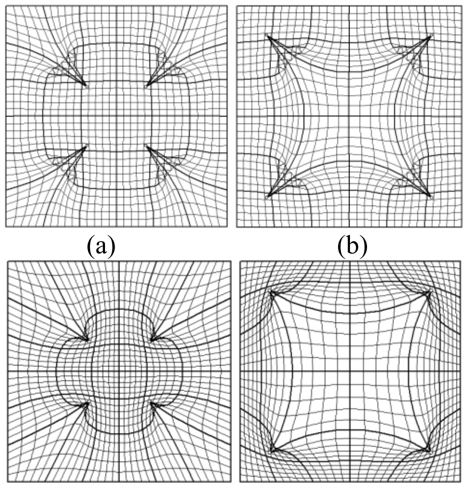


Figure 2. Comparison of the deformation results: (a) and (b) using normal mass-spring system; (c) and (d) using two thin plate mass-spring systems.

3 Preventing Overlaps

A one-to-one deformation cannot be satisfied when overlaps occur on the mesh. The Jacobian determinant is one of the good methods of detecting overlaps on the mesh. The mapping will overlap if the Jacobian determinant J passes through zero (becoming negative) [4],[19-20].

$$J = \frac{\partial(u,v)}{\partial(x,y)} = \frac{\partial u}{\partial x}\frac{\partial v}{\partial y} - \frac{\partial u}{\partial y}\frac{\partial v}{\partial x} \quad (5)$$
$$J = \frac{\partial(u,v)}{\partial(x,y)} = \frac{\partial(x+a)}{\partial x}\frac{\partial(y+b)}{\partial y} - \frac{\partial(x+a)}{\partial y}\frac{\partial(x+b)}{\partial x} \quad (6)$$
$$J = \left(1 + \frac{\partial a}{\partial x}\right)\left(1 + \frac{\partial b}{\partial y}\right) - \frac{\partial a}{\partial y}\frac{\partial b}{\partial x} \quad (7)$$

where *a* and *b* are the displacements for the *x*- and *y*-components respectively. The four terms ($\frac{\partial a}{\partial x}$, $\frac{\partial b}{\partial y}$, $\frac{\partial a}{\partial y}$, and $\frac{\partial b}{\partial x}$) of (7), related to the difference of displacements on the thin plate mass-spring system, can be calculated directly from thin plate mass-spring system. It is clear that the closer the value of each of the four terms is to 0, the closer the Jacobian determinant is to 1. If J of any node is a negative, we set the external forces in the *z*-direction for the four neighbouring nodes to reduce the values of these four terms for removing overlaps. The magnitudes of the external forces are automatically computed proportional to the computed differences of displacements as follows:

$$f_j = \beta \Delta d\, k_s \quad (8)$$

where k_s is a spring constant. The value Δd is the value of the difference between the two nodes' displacements. The parameter β controls the external force. The external force, f_j, is limited to the direction perpendicular to the surface of the mass-spring system.

The procedure for detecting and removing overlaps is as follows:

1. Initialize thin plate mass-spring systems and set the maximum number of iterations(max_iterations). Set the external forces for each node to 0. Set the interval of time(j_step) for detecting and removing overlaps.
2. Calculate the displacements a_0 and b_0 of the x- and y-components between the initial positions and the target positions of the control nodes.
3. Move the control nodes to a_0 and b_0 in the z-direction on the thin plate mass-spring systems for the x- and y-components respectively.
4. For (i=1 ; i <max_iterations ; i= i +1){

```
    Use (3) to compute velocity of each node.
    Use (4) to compute new position of each node.
    If ( i mod j_step) = 0 ){
        Use (7) to compute $J$ of each node.
        Use (8) to add the external force on its four
          neighbouring nodes if $J$ of any node is negative.
    }
}
```

5. Compute J of each node and if J of any node is negative, then go to 4 else go to 6.
6. Determine the new positions for each node on the mesh by adding the displacements of the x- and y-components to their initial positions.

Since the initial values of the external forces for each node are all 0, the external forces for the neighbouring nodes connected to the nodes in the list of overlaps are increased through the iterations until the Jacobian determinant becomes positive. Fig. 3(a) shows the result of a deformation when four nodes move to the center. Overlaps occur at the control nodes and the near nodes. Fig. 3(b) and (c) are two internal thin plate mass-spring systems for the x- and y-components respectively. The surface near the four control nodes is very sharp. Fig. 3(d) shows that overlaps are removed after our method of detecting and removing overlaps is applied. In addition, Fig. 3(e) and (f) show the thin plate mass-spring systems for the x- and y-components respectively. The pointed surface becomes smooth, because the external forces decrease the differences in the displacements between the node and its neighbours.

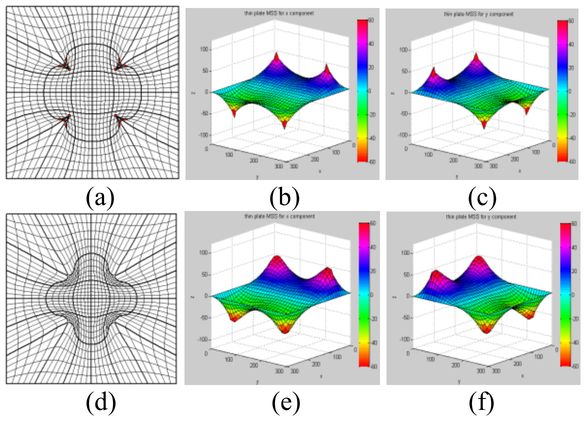


Figure 3. An example of detecting and removing overlaps: (a)~(c) deformed mesh and mass-spring systems for x- and y-components after detecting overlaps; (d)~(f) after removing overlaps.

4 Implementation and experimental results

We implemented image morphing using thin plate mass-spring systems. After loading the source and target images, an animator specified the control points on the source image and the corresponding control points on target image. Points, line segments and curves were used as feature specifications. Line segments and curves were internally sampled as the adequate gap. Fig. 4(a) and (b) show the result images of the feature specification on the 51 × 51 mesh of source and target images respectively. Fig. 4(c) and (d) were the results of the source and target image warping, after applying the method for detecting and removing overlaps. We could see that the control points of the source image moved to the corresponding control points of target image, and the control points of the target image moved to the corresponding control points of source image. In Fig. 5, the first row shows the result images of source image warping from left to right, and the third row shows the result images of target image warping from right to left. The middle row is the result images of cross-dissolve between the result images of the source and target image warping. The middle row was produced by applying the weights [1.0, 0.8, 0.6, 0.4, 0.2, 0.0] and [0.0, 0.2, 0.4, 0.6, 0.8, 1.0] to the six images in the first and third rows respectively, and adding the two sets together. We use personal computer (Intel Core(TM2) Quad CPU 2.5GHz) to generate examples. The deformed mesh in Fig. 3(a) and (b) is 31 x 31 and they take 0.7 and 2.8 sec respectively. In Fig. 3(d), overlaps are removed after our method of detecting and removing overlaps is applied 4 times. It takes 58 sec to produce 10 deformed meshes for source and target image warping in Fig. 5.

5 Conclusions

In this paper, we presented a fast and stable approach for image morphing using a mass-spring system. To stabilize deformation, we used two internal thin plate mass-spring systems. The deformation results using thin plate mass-spring systems were globally smoother and more stable than those using normal mass-spring systems. To meet the one-to-one property, we proposed a method of detecting and removing overlaps. For detecting overlaps, the Jacobian determinant was calculated directly from the thin plate mass-spring systems. After detecting an overlap on a node, we set the external forces on the neighbouring nodes to remove the

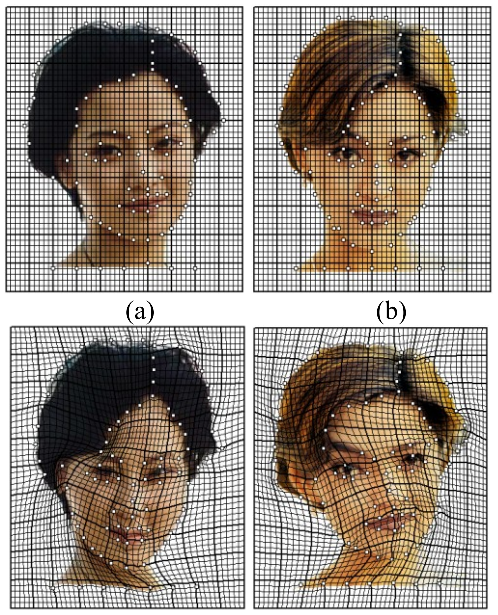


Figure 4. Feature specification: (a) and (b) source and target image; Image warping: (c) and (d) source and target image.

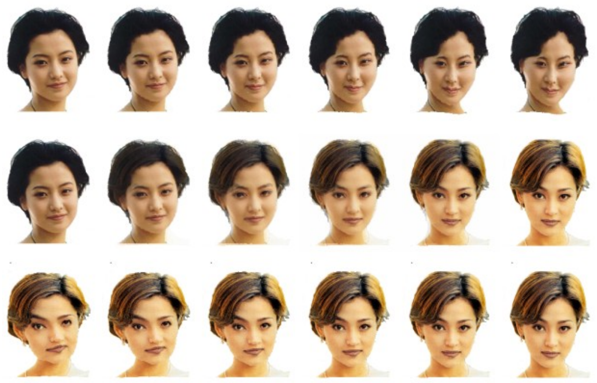


Figure 5 . An example of image morphing

overlaps. The external forces were automatically determined by calculating the difference in displacements on the thin plate mass-spring systems, and applied the forces repeatedly until there were no overlaps. We showed the test results of detecting and removing overlaps in many cases.

6 References

[1] G. Wolberg, “Recent advances in image morphing,” *IEEE in Proc. Int. Conf. Comput. Graph.*, pp.64-71, 1996.

[2] G. Wolberg, *Digital Image Warping*. IEEE Computer Society Press: Los Alamitos, CA, 1990.

[3] T. Beier, and S. Neely, “Feature-based image metamorphosis,” *in Proc. SIGGRAPH*, vol. 26: pp.35-42, , 1992.

[4] SY. Lee, KY. Chwa, J. Hahn, and SY. Shin, “Image morphing using deformation techniques,” *J. Vis. Comput. Anim.*, vol 7, no.1: pp.3-23, 1996.

[5] SY. Lee, KY. Chwa, SY. Shin, “Image Metamorphosis Using Snakes and Free-Form Deformations,” *Proc. SIGGRAPH ’95*, pp. 439-448, 1995.

[6] D. Baraff, and A. Witkin, “Large Steps in Cloth Simulation,” Proc. 25th Ann. Conf. Computer Graphics and Interactive Techniques(SIGGRAPH ’98), pp.43-54, 1998.

[7] X. Provot, “Deformation Constraints in a Spring-Mass Model to describe rigid cloth behavior,” *In Graphics Interface,* pp.147-154, 1995.

[8] K. Kähler, J. Haber, and HP. Seidelm, “Geometry-Based Muscle Modeling for Facial Animation,” *Proc. Graphics Interface Conf.*, pp. 37-46, 2001.

[9] W. Mollemans, and F. Schutyser, Cleynenbreugel JV, Suetens P. “Fast Soft Tissue Deformation with Tetrahedral Mass Spring Model for Maxillofacial Surgery Planning Ststems,” *Proc. Mdeical Image Computing and Computer-Assisted Intervention (MICCAI ’04)*, pp. 371-379, 2004.

[10] S. Zhang, L. Gul, P. Huang, and J. Xu, “Real-Time Simulation of Deformable Soft Tissue Based on Mass-Spring and Medial Representation,” *Proc. First Int’l Workshop Computer Vision for Biomedical Image Applications (CVBIA’05)*, pp.419-426, 2005.

[11] J. Brown, S. Sorkin, JC. Latombe, K. Montgomery, and M. Stephanides, “Algorithmic Tools for Real-Time Microsurgery Simulation,” *Medical Image Analysis,* vol. 6, no.3: pp.289-300, 2002.

[12] FL. Bookstein, “Principal Warps: Thin-Plate Splines and the Decomposition of Deformations,” *IEEE Transactions On Pattern Analysis and Intelligence,* vol. 11, no. 6: pp.567-585, 1989.

[13] N. Arad, N. Dyn, D. Reisfeld, and Y. Yeshurun, “Image Warping By Radial Basis Functions: Application To Facial Expressions,” *CVGIP:Graphical Models and Image Processing.*, 56(2): pp.161-172, 1994.

[14] L. Cooper and S. Maddock, “Preventing Collapse Within Mass-Spring Damper Models of Deformable Objcets,” *The 5th Int. Conf. in Central Europe on Comput. Graphics and Vis.,* 1997.

[15] CCL. Wang, SSF. Smith, and MMF. Yuen, “Surface flattening based on energy model,” *Computer-Aided Design*, 34: pp.823-833, 2002.

[16] J. Li, D. Zhang, G. Lu, Y. Deng, X. Wen, and Y. Sakaguti, “Flattening triangulated surfaces using a mass-spring model,” *Int. J Adv Manuf. Technol.*, 25: pp.108-117, 2005

[17] YJ. Choi, M. Hong, MH. Choi, and MH. Kim, “Adaptive surface-deformable model with shape-preserving spring,” *Computer Animation and Virtual Worlds,* 16: pp.69-83, 2005.

[18] Y. Bhasin, A. Liu. “Bounds for Damping that Guarantee Stability in Mass-Spring Systems,” *Medicine Meets Virtual Reality,* 2006.

[19] GH. Meisters and C. Olech, “Locally one-to-one mappings and a classical theorem on Schlicht functions,” *Duke Mathematics Journal*, 30: pp.63-80, 1988.

[20] B. Tiddeman, N. Duffy, and G. Rabey, “A General Method for Overlap Control in Image Warping,” *Computers and Graphics,* vol. 25, no.1: pp.59-66, 2001.

A General Design Pattern for Programs of Scene Graph and its Application in a Simulation Instance

Youyi Bi[1], **Carlos Domínguez**[2], **and Houcine Hassan**[2]
[1]School of Mechanical Engineering and Automation, Beihang University, Beijing, P. R. China
[2]School of Design Engineering, Polytechnic University of Valencia, Valencia, Spain

Abstract - *The prevalence and significance of programs of Scene Graph, which mainly concern with the simulations of 3D scenes (hereinafter referred to as graphic programs), are more and more noticeable in many realms, such as Virtual Reality and physical simulators. This paper presents a general design pattern, Data-Processing-Interaction (DPI), for such programs' development. This pattern explicitly divides the development into several necessary structures, functions and corresponding solutions, which accelerates the programming process as well as provides clear application architecture. Developers can attain rich alternatives of solutions and concise decision-making process under the DPI pattern. And an instance of a simulation of motion in 3D scene will be followed. This instance clearly demonstrates the convenient features of this pattern for the preliminary preparation, the lower cost and high-efficiency of the development process.*

Keywords: Scene Graph, Design Pattern, Virtual Reality

1 Introduction

One of the most impressive achievements brought by the information technology is that scientists and engineers are able to implement their experiments and designs in computer. Especially following the rapid development of Computer Graphics and high-efficiency graphic chips, people are attempting to simulate all kinds of processes in computer, including computer aided design, robot motion, virtual manufacturing and assembling, visualization in scientific computing, 3D geological information system, medical examination and of course the most direct ones—3D games. This tendency of the wide application mainly derives from two aspects of simulation programs. First, the features of immersion and interaction greatly improve the user's sensation and intimacy with machines, especially for the fields of education, exposition, and entertainment. Another important aspect reflects on the much lower cost and higher efficiency than real physical simulation in scientific and engineering realms. Mechanical designers can detect the fatal flaws of their designs by executing interference and kinetic examinations on computer avoiding being inspected as physical entities with higher cost.

In many circumstances, a simulation program is just a simple graphic program, for which there is no necessity to build those complicated design documents under the normal regulations of Software Engineering. A scene graph [1] is a general data structure that arranges the logical and often spatial representation of a graphical scene. It is commonly used by vector-based graphics editing applications and modern computer games. The most direct example of the application of Scene Graph is virtual tour, which enables people to visit museums or palaces on computer freely and immersively.

Thus, the importance of graphic programs is calling for an easy and swift design pattern for developers. The first problem for those beginners in this filed usually is "which step should I start from?" Some researchers have already proposed their design patterns in the researches regarding graphic simulation. Closely related, C. Marcu [2] put forward a reasonable structure for his simulation program of a Fanuc M-6iB/2HS articulated robot. This structure consists of OpenGL Scene, Motion control thread, Mathematical models and the Main interface, which distinguishes the program into different functions orderly. Fu Qiang [3] also raised a MVC (Model-View-Control) design pattern in his Software for Quality Evaluation of Seismic Acquisition Data. Martin Eigner [4] presented an application to investigate and manipulate JT data, a neutral data format for products, due to the better efficiency and convenience of managing and communicating product information and data in enterprise networks. In this application, the author put forward a two-layer underlying framework, including the toolkit layer (Qt toolkit, OSG toolkit and JT toolkit) and the application layer (Extensible viewer).

However, none of them illustrated their design patterns specifically. They didn't present the details for how to prepare and implement the development process corresponding to their design patterns. Moreover, their design patterns are lack in the necessary simplicity and comprehensiveness. For example, interface between the machine and user should include other interactions like haptics and force feedback functions. Developers need a more efficient and straight design pattern.

This paper firstly focuses on the concept of the DPI design pattern in Section 2, presenting a three-layer structure and the connections between each layer. With detailed information regarding the tools used in graphic programs, the DPI pattern provides developers with concise developing principles and plentiful alternatives of solutions. Section 3 then depicts a developing instance to show the typical application of the DPI pattern and several noticeable details in the actual developing process, before Section 4 wraps up by giving a conclusion and outlook in terms of future work.

2 The DPI design pattern

The design pattern of DPI (Data-Processing-Interaction) is able to provide developers with a more explicit programming process. Most required functions of the program can be allocated into the three layers of the DPI pattern. Each layer deals with certain demands and connects with other layers smoothly. Figure 1 illustrates the basic structure of the DPI pattern.

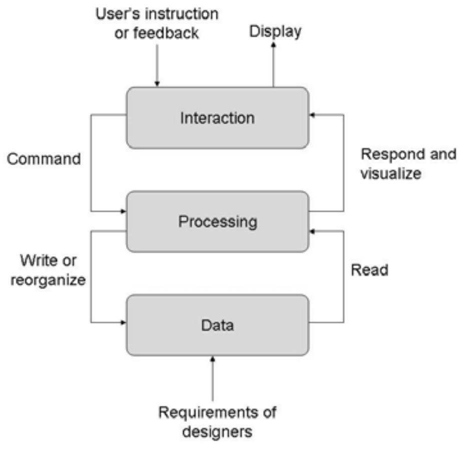


Figure 1. The basic structure of the DPI pattern

2.1 The Data Layer

Data files are the most fundamental elements to comprise a scene graph. Normally, there are five kinds of data files, and table 1 indicates their types and descriptions.

Table 1. Common data types in graphic programs

Data type	Common formats	Roles
Model	OBJ, FLT, 3DS, DAE	For rendering 3D model objects
Video	AVI, MPG, RM, WMA	For rendering dynamic texture, and multimedia
Image	BMP, GIF,JPG, PNG	For rendering texture data
Font	TTF, TTC, FON,PFB	For displaying of words and multilingual support
Others	WMV, MP3, PDF	For auxiliary effects and designers' special requirements

The most important data used for graphic programs is model, which is mainly built by various kinds of CAD software or by drawing functions from those graphic libraries like OpenGL directly in some cases. One kernel problem to be solved in the data layer is to choose reasonable modeling software. For those projects which need high mathematical or physical accuracy or high visual effect like 3D games, large commercial software is indispensable. Otherwise, those free and open-source ones are sufficient. Table 2 presents the corresponding features of the major modeling software. Developers should select the most suitable modeling software according to the actual requirements and financial conditions.

Table 2. Major modeling software's features

Name	Cost	Platform	Applications
Autodesk Inventor	$7,995	MS Windows	Creating 3D digital prototypes, visualization and simulation of products
Pro/Engineer	$4,995	MS Windows HP-UX, Unix	Solid modeling, assembly modeling and drafting, finite element analysis, and NC and tooling functionality
Solidworks	$3,995	MS Windows	CAD of industrial products
NX(UG)	$8,700	MS Windows Mac OS X	Computer-aided mechanical design, manufacturing and engineering
CATIA	$15,000	MS Windows, IBMAIX, HP-UX, Solaris	Industrial equipment, defense aerospace, automotive, consumer packaged goods
3DS Max	$3,495	MS Windows	Animation, Modeling, Visual 3D Effects
Blender	Free	MS Windows, Solaris, Linux, Mac OS X	Animation, Modeling, Visual 3D Effects. It can be even run in Pocket PC,

Another important problem is to select appropriate file formats for the models. And the selection should follow three factors. Accuracy is the primary consideration. In graphic programs, those essential information regarding the models, including geometric dimensions, colour and light, and textures, should be kept completely and consistently. Another factor is portability. There is no guarantee that a model will be used in only one application. Those neutral file formats can bring much convenience for further development. Finally, the compressibility of the model data should be concerned. Once the total volume of the model data exceeds the normal conditions of computer's hardware, developers will be busy with upgrading their computers. Table 3 indicates the features of major graphic model formats.

Table 3. The features of major general 3D model formats

Format	Origin	Descriptions and Applications
3DS	3DS Max	Open model data format of 3D studio software
AC	AC3D	Stored by the text format, used in FlightGear for scenery objects and aircraft models
DAE	COLLADA	A general open 3D model data exchange standard, using XML syntax, supported by the COLLADA library
DXF	AutoCAD	Drawing Exchange Format is developed for enabling data interoperability between AutoCAD and other programs.

IGES	Graphic standards	Widely-used format, composed of 80-character ASCII records
OBJ	Wavefront	Open and universally accepted. Including the position of each vertex, the UV position of each texture coordinate vertex, normal.
STL	Stereo-lithography CAD	Describing raw triangulated surface by the unit normal and vertices of the triangle, widely used for rapid prototyping and computer-aided manufacturing.

To sum up, choosing a suitable modeling software and file format suggests a well beginning of the development.

2.2 The Processing Layer

The processing layer manages the model data and communicates with the Interaction layer. The main functions of this layer can be classified into three aspects—data structure, visualization and specific algorithms. Figure 2 shows the structure of the Processing layer.

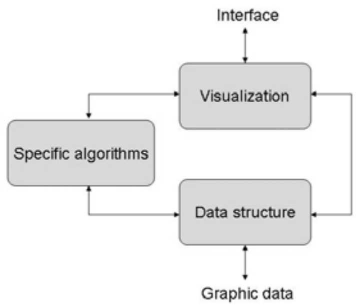


Figure 2. The structure of the Processing layer

Obviously, the primary task for the Processing layer is to organize the model data from the former layer. A well organized data structure benefits both the programmers and model builders. Actually, a scene graph is a collection of nodes in a graph or tree structure. A node may have many children but often only a single parent, with the effect of a parent applied to all its child nodes; an operation performed on a group automatically propagates its effect to all of its members. A common feature, for instance, is the ability to group related objects into a compound object that can then be moved, transformed, selected, as easily as a single object.

Furthermore, some software development kits (SDK) even encapsulate the tree structure functions into their own classes, like the OpenSceneGraph (OSG). In OSG [5], graphic model data can be managed as a leaf node, which is the most basic unit. Larger scene or more sophisticated models can comprise these basic nodes and they can be organized by superior nodes or so-called groups and so on. In this case, those algorithms from data structure, familiar to developers, like the depth-first traversal algorithm, can be applied directly.

Visualization is the central function of the Processing layer. Normally, 2D graphics can be drew and displayed straightly by the interior drawing classes of the programming language, such as the *CDC* class in Microsoft Foundation Class (MFC) and the *java.awt.Graphics2D* class in Java. For 3D graphics, OpenGL and DirectX are the most fundamental tools, which can create almost all kinds of graphs theoretically. However, OpenGL [6] is a low level library that takes lists of simple polygons and renders them as quickly as possible. To do something more practical, the programmer must break down the object into a series of simple OpenGL instructions and send them into the engine for rendering. For simple programs a tremendous amount of programming has to be done just to get started. Hence those Three-dimensional rendering engines based on such industrial standards are more popular for their integration of many common functions. An evident example is reading 3D models in graphic programs. It would be more strenuous if using OpenGL functions. Table 4 presents the characteristics of major graphics rending engines (Running platforms are inside the brackets).

Table 4. The characteristics of major graphics rendering engines

Rendering engine	Description and Application
Open Inventor (MS Windows, Linux, Unix)	A C++ object-oriented retained mode 3D graphics API designed by SGI to provide a higher layer of programming for OpenGL
Unreal (MS Windows, Linux, Mac OS Xbox, PS2 , PS3)	The most famous commercial 3D game engine, designed by Epic Games, based on C++
OpenSG (MS Windows, Linux，Solaris, Mac OS X)	A scene graph system to create real-time graphics programs, Open Source, and can be used freely, based on OpenGL, Well performed in clustering support and advanced multithreading.
OERG 3D (MS Windows, Linux，Solaris, Mac OS X)	A scene-oriented, flexible 3D rendering engine, supported by OpenGL or DirectX, realizing the scene octree, BSP tree, CLOD and paging mechanism
OpenSceneGraph (MS Windows UNIX/Linux, Mac OS X, Solaris)	An open source 3D graphics API, written in standard C++ using OpenGL, used in fields such as visual simulation, computer games, virtual reality, scientific visualization and modelling.

Additionally, specific algorithms are necessary for engineering requirements and better realistic sense. Collision detection [7] algorithm is a widely-applied example. Collision detection typically refers to the computational problem of detecting the intersection of two or more objects, which is indispensable in the simulation of robots, first-player shooter games and other physical simulations.

2.3 The Interaction Layer

The central function for the Interaction layer is to respond all kinds of commands and signals from users and transmit them to the Processing layer. The Interaction layer normally consists of the Graphic User Interface (GUI) and functions dealing with other physical reactions, including the digital gloves, the haptics and force feedback devices or sensors if necessary. The separation of the Processing layer and the Interaction layer reflects the idea of encapsulation and modularity in the Object-Oriented design. Such separation brings better maintainability, i.e. the Processing layer can avoid modification even the GUI needs to change.

Traditional interactive devices are mainly the keyboard and mouse, while other more realistic equipments are being introduced fast. The flight simulator to help training pilots is the pioneer one. Pilots are able to grasp basic controlling skills by operating joysticks while watching the virtual flight scene. Such simulation systems have made great success in the 3D game markets, no matter people wish to shoot, drive, or even play tennis in the virtual world.

The mainstream interfaces are optional abundantly, free or commercial, cross-platform or exclusive. As the great tendency of portable smart terminals (smart phone, tablet PC, etc…) and embedded devices, free and cross-platform developing environment will be more suitable for small-scale projects for lower cost and better portability.

2.4 The connections between each Layer

The connections between each layer in the micro level are reflected in the interfaces of functions. These interfaces play the same roles as those transfer stations and conveyors in factories, which help with visualizing 3D scenes and responding to users' instructions. Specifically, the connections between the Data layer and the Processing layer mainly deal with the input and output of data, i.e. reading and writing data files.

Large projects usually require data of numerous types, which is a big challenge for low-level graphic libraries. Fortunately, customized plugin technology has been applied widely in high-level graphic rendering engines. For example in OSG, it defines a file format extension mechanism that allows users to write their own specific data import and export plug-ins. Once a file format plugin is completed, people can spread it freely between different developers and use it directly, without having to recompile the source code of engineering systems. This mechanism greatly improves the universality of graphic programs.

The connection between the Processing layer and the Interaction layer is a typical trigger-respond relation. The most common examples are the "message-response" mechanism in MFC, the "event-listener" mechanism in Java and the "signal-slot" mechanism in Qt, which are similar theoretically. Figure 3 presents the connections between each layer in the details.

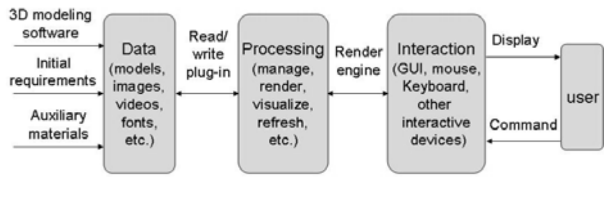

3D modeling software
Initial requirements
Auxiliary materials
Data (models, images, videos, fonts, etc.)
Read/ write plug-in
Processing (manage, render, visualize, refresh, etc.)
Render engine
Interaction (GUI, mouse, Keyboard, other interactive devices)
Display
Command
user

Figure 3. The connections between each layer

3 A developing instance

This instance comes from a project of Polytechnic University of Valencia: "the simulation of physical worlds to test software control agents". The actual developing of a simulation program makes the DPI pattern more convincing. The specific developing steps will be presented in this section.

3.1 Primary requirements

The main goal of this instance is to realize roaming in virtual scene. User can finally ramble in the virtual environment of a University teaching building. Through the keyboard, user can choose four directions (forward, backward, left and right) to move, rotate right or left, and make the view go up and down.

Since this project serves for the university research, another important property of this project is low cost, which means to avoid using commercial software for modeling and programming as much as possible. At the same time, because of this project being the initial part of a simulation of robot control and motion and the restrictions of computer conditions, the demand for the geometric accuracy and visual effect of scenes is moderate.

3.2 The pre-programming work

Following the above primary requirements, to determine the related work of each layer is the first step. According to the DPI pattern, the whole project is divided into several key problems in each layer. Table 5 shows the preliminary decomposition of this task. There is no necessity to make the items of this table specifically at first, for the solutions of each layer may interplay to some extent. And developers don't have to rush for starting modeling and programming immediately unless having made suitable decisions about the model format and render engine. Thus, work①③⑤ should be completed firstly in general cases.

Table 5. The decomposition of this task

The Data Layer	①To choose the modeling software and file format
	②To model objects
The Processing Layer	③To choose the graphic rendering engine
	④To program
The Interaction Layer	⑤To design graphic user interface
	⑥To program and test

Actually, the most common software for modeling buildings is 3DS Max. It's frequently used by video game developers, TV commercial studios, architectural visualization studios, movie effects and movie pre-visualization. The strong modeling function, fluent and high-performance of visual effect and the system of rich plug-ins contribute to its wide application around the world. But it seems to cost unnecessarily to use commercial software according to the actual situations of this project. Therefore, Blender, a free and open source one is adopted.

Blender can be used for modeling, UV unwrapping, texturing, rigging, water and smoke simulations, skinning, animating, rendering, particle and other simulations, non-linear editing, compositing, and creating interactive 3D applications, including video games, animated film, or visual effects. The most amazing point of Blender is its incredibly small volume, for the application package of Blender 2.49 is only less than 32MB after installed. Its feature of cross platform even enables designers to do 3D design on mobile phones. Furthermore, Blender also provides various plug-ins for the import or export of special model file formats. These impressive features make Blender suitable modeling software for this project. Figure 4 shows the modeling process in Blender.

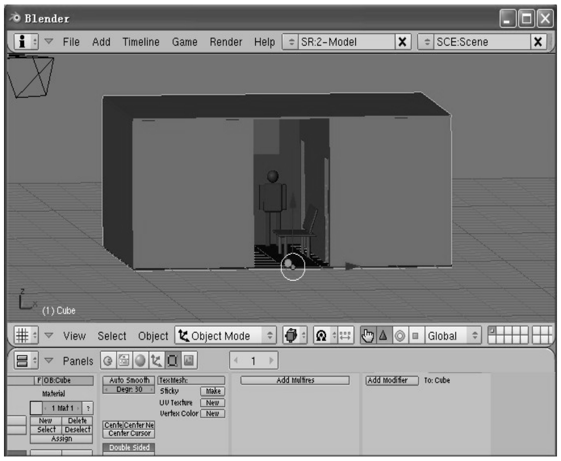


Figure 4. The modeling process in Blender

In the modeling process, especially for those large-scale projects, it is important to unify the plan and management of objects to be modeled before the formal formation. Designers should make clear of the significant properties of objects and even using the ideas of standardization and Group Technology. Usually the mechanical products could be classified into three kinds: special parts, similar parts and standard parts, and their respective proportion of the total parts is 10%, 70% and 20% approximately. Therefore, parts with similar geometric or other physical properties occupy the main aspects in the mechanical industry. Similarly, in this modeling process, similar objects can be built jointly to achieve high efficiency. In this project, the objects to be built are mainly the stuff in a teaching building, like offices, tables, benches, chairs, decorates and persons. Static objects, such as an office and inside facilities could be jointed as one node manipulated in scene graph.

And in selecting the file format for the model, the STL format was considered firstly for its feature of lightweight. STL is developed by 3D System Company and is known as the standard format for the description of parts in the Rapid Prototype field. In STL, geometric information is stored as the coordinates of the three vertexes of the triangle facet and its normal vector. An obvious advantage of the STL is the relatively low amount of data when storing models. However, when confronting the selection of the graphic rendering engine, the STL format was abandoned for its defect in the integration of color and texture information. Hence the selection of file formats and graphic engine is interplayed and it is reasonable to consider the characteristics of the rendering engine and the format of models comprehensively.

In the selection of graphic engines, OpenGL was the first consideration for its affluent development references on Internet and its wide application. However, the complexity of reading 3D model files and controlling view cameras leads to another option--- OSG, which is free and open source and very suitable for small-scale projects. As of 2006, the OSG-user mailing list exceeded 1,500 subscribers. OSG [8] is a high-level programming interface for 3D computer graphics, used in simulation, animation and visualization applications. It’s built on OpenGL, which ensures that OSG is both cross platform and high performance. But it goes beyond OpenGL to provide functionality common to many 3D applications, such as 2D and 3D file loaders, texture mapped font support, level of detail control, threaded database paging and others. OSG has become widely accepted by both academia and industry for its rich feature set and liberal open source license.

3.3 The application design and result

After the decision of rendering engine, the design of interface and application architecture can be started. Figure 5 shows the main classes in this program. Among these classes, the Core_OSG class is responsible for calling the read/write mechanism of OSG library to read model data. The View_control class is in charge of calling the view control functions from OSG library. The Application class is served as the basic structure of this application. The interface deals with user’s commands and the Dialog class provides users with extra control parameters.

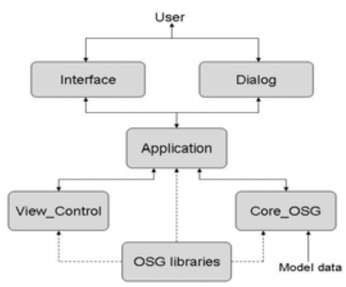
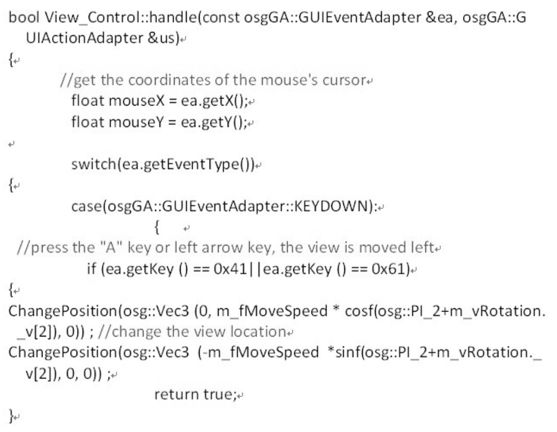


Figure 5. The main classes in the program.

Fortunately, Blender offers a special plug-in for OSG, which can export the Blender model file as the OSG format. As mentioned above, since the OSG encapsulates many advanced functions, like reading model files and controlling cameras directly, the programming process is simplified greatly. Figure 6 is part of essential codes for controlling camera views of scene graph.



```
bool View_Control::handle(const osgGA::GUIEventAdapter &ea, osgGA::G
   UIActionAdapter &us)
{
          //get the coordinates of the mouse's cursor
            float mouseX = ea.getX();
            float mouseY = ea.getY();

            switch(ea.getEventType())
{
            case(osgGA::GUIEventAdapter::KEYDOWN):
                              {
  //press the "A" key or left arrow key, the view is moved left
               if (ea.getKey () == 0x41||ea.getKey () == 0x61)
{
ChangePosition(osg::Vec3 (0, m_fMoveSpeed * cosf(osg::PI_2+m_vRotation.
  _v[2]), 0)) ; //change the view location
ChangePosition(osg::Vec3  (-m_fMoveSpeed  *sinf(osg::PI_2+m_vRotation._
   v[2]), 0, 0)) ;
                              return true;
}
```

Figure 6. Essential codes for controlling camera views.

And the developing environment was chose as Qt finally. The most distinguished advantages of Qt are the portability, compatibility and flexibility. Using the Qt programming environment we can obtain a portable simulation program running in different operating systems, and even in the embedded devices. Qt framework is used for GUI, providing with all the functionality needed to develop advanced GUI applications on desktop and embedded platforms. Additionally, since Qt is a C++ based environment, a number of C++ libraries can be included for better development, such as OSG, which provides more advanced functions of rendering and view control based on OpenGL. After the decision of the three-layer‘s design, modeling and coding can start. Figure 7 shows the effect of main interface. The scene graph is presented in the view area. In the menu bar, user could read different files of scene graph to be simulated from the “File” menu. In the “Simulate” menu, user could enter related parameters to control the quantity and location of specific objects in scene graph.

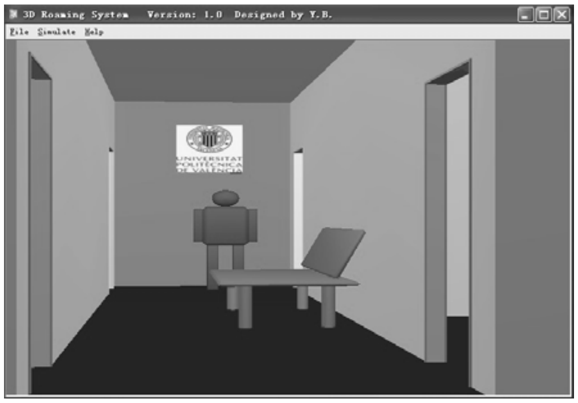

3D Roaming System Version: 1.0 Designed by Y.B.
File Simulate Help
UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Figure 7. The effect of main interface

4 Conclusions

The DPI provides developers with a simple and high-efficiency design pattern for graphic programs. The explicit classification of a task is the decisive point. Developers could quickly make clear of their thoughts and choose the most suitable solutions among those alternative options. As a matter of fact, the heavy work on writing complete developing document is simplified as only to solve several kernel problems. Once they are solved, the modeling and programming work can be executed almost simultaneously if the human resource is adequate, which will shorten the developing time further. The further work of DPI will be focused on establishing more elaborate and general connections between each layer and the possibility of auto-programming of graphic programs.

5 Acknowledgements

This work has been partially funded by Escuela Técnica Superior de Ingeniería del Diseño, Universidad Politécnica de Valencia.

6 References

[1] Avi Bar-Zeev. Scene graphs: Past, Present and Future ， http://www.realityprime.com/articles/scenegraphs-past-present-and-future, 2007

[2] C. Marcu, Gh. Lazea, S. Herle, R. Robotin, L. Tamas. 3D Graphical Simulation of an Articulated Serial Manipulator based on Kinematic Models. 19th International Workshop on Robotics in Alpe-Adria-Danube Region, 2010.

[3] FU Qiang, XIAO Yunshi, YUE Jiguang. QESADSYS V1.0: A New Cross-Platform Software for Quality Evaluation of Seismic Acquisition Data. World Congress on Software Engineering, 2009.

[4] Martin Eigner, Florian Gerhardt. Extensible JT Open Tool for Prototypical Process Support, based on OpenSceneGraph and QT. 13th International Conference on Computer Supported Cooperative Work in Design, 2009.

[5] Paul Martz. OpenSceneGraph Quick Start guide http://www.openscenegraph.org/osgwiki/pmwiki.php/ Documentatio n/ QuickStartGuide

[6] Dave Shreiner. OpenGL programming guide, Addison-Wesley Publishing Company, 2009.

[7] Yang Shixing, Cao Mingliang. Step into OpenSceneGra ph. Virtual Reality Lab of Zhengzhou University, 2008.

[8] Wang Rui, Qian Xuelei. The design and practice of OSG rendering engine, Tsinghua University Press, 2009.

An Inexpensive Personal Virtual Reality Videolaparoscopy Platform

Alessandro Brawerman, James Skinovsky, Diego Roberto de Souza & Rodrigo Daron Wang

Computer Engineering and Medicine Departments –Positivo University - Curitiba –Brazil
brawerman@up.edu.br , {skinovsky, rodrigodaron, dstringg}@gmail.com

Abstract— **Virtual reality flight simulators have being used for years to train pilots around the world on a safety, cheaper end easy way. Using this premise a videolaparoscopy simulator should help medical students to learn the basic movements of this kind of surgery. Videolaparoscopy is a minimally invasive surgery that uses a clamp and a camera, so the image of the surgery is shown on a TV. A surgeon must have a 3D perspective on a 2D screen. This project presents the development of a platform that accounts the hardware pieces, working as joysticks, and the software part, simulating a training environment. The objective of this videolaparoscopy platform is to train the doctor to understand the idea of depth perspective in a 2D environment and simplify the learning curve on a videolaparoscopy surgery. A simple prototype is presented to investigate feasibility of this project.**

***Keywords-* Virtual reality, simulator procedures, videolaparoscopy, training platform.**

I. Introduction

THE concept of virtual reality is to bring something from the real world to a computer simulator environment. Virtual reality based training system has been proofed a promising technology in many fields including pilot training. In recent years, virtual reality technique found its application in several minimally invasive surgeries.

Although the main idea of virtual reality platforms is to provide a training environment that leads the student to gain experience in a short period of time, such commercial platforms are normally expensive for single users to afford and usually too complex to train. They regularly are prepared with high detailed surgery scenarios taking into account expert users, while the basic steps of such procedures are not featured.

The objective of this project is to present an inexpensive videolaparoscopy training platform, which can be afforded by single users, focusing in the initial learning steps of videolaparoscopy surgery. The platform aims to provide a rapid initial learning curve by training depth perspective notion, since a videosurgery gives a 2D perspective in a 3D environment.

All aspects of the platform, the development of the hardware pieces, which will work as the clamps of a surgeon, and the development of software that will simulate the videosurgery view, were planned taking into account cost and usability.

The hardware pieces consist in surgery clamp based joysticks that are connected to a computer using USB ports. They are viewed by the underline operating system as regular joysticks facilitating the software development. Inside the hardware part is located the mechanical part, which is a system comprised of precision potentiometers and developed to capture the clamp movements in the most possible realistic way.

The software is a 3D program that simulates a simple environment to train depth perspective. The software screen presents initially the end of the clamps and a ball or another object to interact. It also uses a simple physical engine to become a little bit more realistic. Keep in mind that it is not the objective of this project to generate high quality complex surgery scenarios, but to train the 2D perspective and to be afforded by single users.

The rest of this paper is described as follows, section II presents some of the related work, section III brings the methods used to build up the training platform, section IV shows some of the practical experiments to demonstrate feasibility, and finally, section V concludes the work.

II. Related Work

This section presents some of the related work. They deal with different ways to bring together computer engineering, virtual reality and medicine in order to simulate a videolaparoscopy training platform.

In [1], Immersion presents a commercial product called LapVR Surgical Simulator. It presents a very complex system, which recreates a full surgery scenario in a virtual system. The whole system is made over a hardware that fully integrates all the parts, including the clamp joysticks and the screen. The product itself is a little too heavy and large, but presents a tutorial that gives the correct position to use the clamps providing the student a very close sense of the environment. Nonetheless, is a very expensive product and not suitable to be afforded by single users.

The second work is the SPRING framework, presented in [2]. It is a real-time soft-tissue simulation platform maintained by the Stanford University. It is designed to be used with haptic devices, which are sold separately. One well-known commercial simulator that uses the SPRING framework is the VRMSS[3] (Virtual-Reality Motor-Skills Simulator), which is designed to teach baseline fine-motor skills. It presents the user a score based on his/her performance, including motor skills (speed, accuracy, efficiency of motion) and cognitive skills.

Another interesting solution is presented in [4]. In this paper, a virtual reality based arthroscopic surgery simulator is developed. Mechanical design, kinematics, dexterity measure and control loop of the haptic device are investigated. Organ mesh generation, tissue deformation simulation and collision detection are also discussed.

Other related research papers can be found in [5], which presents a platform for colonoscopy simulation, in [6], which proposes a virtual reality simulator for scoliosis surgery training, and in [7], which develops a haptic device for hysteroscopy simulation.

III. Specification

This project presents an inexpensive videolaparoscopy training platform that provides a virtual environment to quickly gain expertise in the first steps of a videolaparoscopy surgeon.

The platform is divided in two parts, the development of hardware pieces, which work as the clamps of a surgeon; and the development of software that will simulate the videosurgery view.

The hardware pieces are viewed by the underline operating system as joysticks; however they are physically similar to laparoscopic clamps. USB (Universal Serial Bus) ports are employed to connect the clamps with the computer. The advantages of such ports are the ease to connect, they are well-known by users and they can be found in any computer sold nowadays.

Inside the hardware part is the mechanical part, which is a system to capture the movements in the most realistic possible way. There are five movements: rotational, open/close clamp and the x, y, z axis. Those movements are caught by precision potentiometers to avoid fluctuation on the signal.

The software is a 3D program that simulates a simple environment to train depth perspective. The software screen presents initially the end of the clamps and a ball or another object to interact. It was developed using a free IDE that is easy to use and easy to create 3D objects. It also uses a simple physical engine to become a little bit more realistic.

An overview of the platform is depicted in Figure 1. The connection between the joystick and the computer uses the HID (Human Interface Device) communication protocol, and passes through a microcontroller. In the computer, after acquiring the data movements from the joystick, the 3D graphic is built as the movements are performed by the user.

The difficult level of the exercises is not high as it is just for basic training; however the hardware system has the capability to be upgraded. Thus, in the future it can be used in a high level program, as it presents all the movements needed for a surgery.

A. Hardware Specification

The joystick clamps are using a microcontroller from microchip, the PIC18F4550, which includes a USB interface and Analog/Digital ports [8]. This last feature is very useful as it is needed to get the data from the potentiometers.

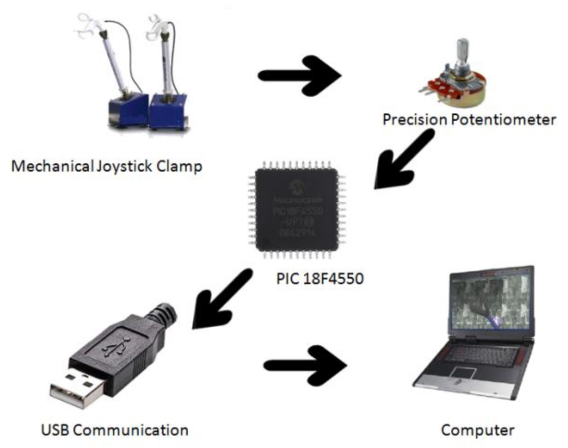


Figure 1 – Platform overview.

By using the potentiometers and the microcontroller shown in the Figure 1 an extremely simple and cheap solution was achieved. Besides those components, the hardware is designed to only need some extra capacitors, a crystal, and the USB connector. Figure 2 depicts the hardware schematic diagram.

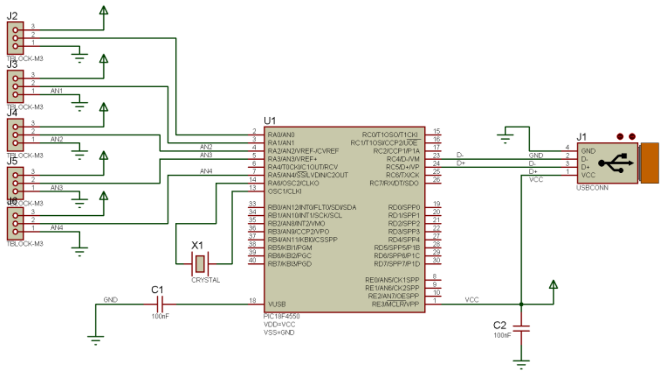


Figure 2 – Hardware Schematic Diagram.

The microcontroller firmware was created using the MPLab program from microchip and the C language. It uses the Human Interface Device (HID) protocol making the underline operating system to recognize the hardware as a common joystick. Furthermore, the firmware performs all the conversions needed to transform the signal from the potentiometers to the HID protocol

B. Communication

The whole project has several levels of communication, since the data acquisition up to the clamp movements represented by 3D graphics in the computer. A communication diagram is depicted in Figure 3.

The communication between the hardware and the operating system did not need a specific driver; instead it was employed the HID solution, which is a standard created to the USB port communication.

The HID descriptor is a hard coded array of bytes that describes the device's data packets. This includes how many packets the device supports, how big are the packets and the purpose of each byte and bit in the packet. For instance, the joystick's firmware with its five analog movements (Throttle, X, Y, Rx, and Ry) binds the converted analog values, each one to a position in a five sized integer array, and sends the packet to the HID descriptor. The HID descriptor is stored in the microcontroller's ROM and does not need to intrinsically understand or parse the HID descriptor [9].

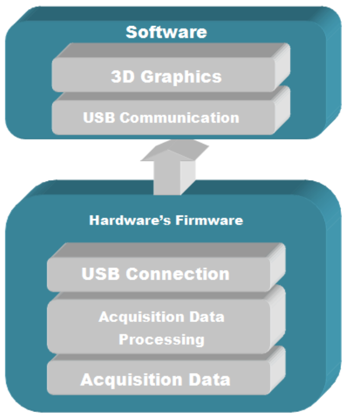


Figure 3 – Communication Data Diagram.

C. Software

The software was developed using the Lite-C IDE [10], which is a very easy IDE to design 3D graphics. This IDE is divided into two small programs, the Script Editor (SED) and the Model Editor (MED). The first one is used to program in C, the 3D objects are loaded in this program, and the movements and physics have to be implemented. The second program is used to develop the 3D objects in a simple way, with a format that the SED can read without major problems.

The program recreates the simple training environment as shown on the Figure 4. Note that it initially presents the clamp's tip and some objects to interact with.

Lite-C is a very simple way to program graphics in 3D as it uses C language regular commands to move and interact with the objects in a world with physics; however this ease comes with a drawback, as the software does not presents a high detailed scenario. Nevertheless, the use of a free IDE also makes it possible to build an inexpensive quality product.

D. Future Upgrades

As the Lite-C IDE has some limitations, in the future the authors plan to use a more complex scenario building IDE, based on XNA. Also, it is necessary to implement force feedback in the joysticks, giving different sensations to the user as he/she interacts with different objects.

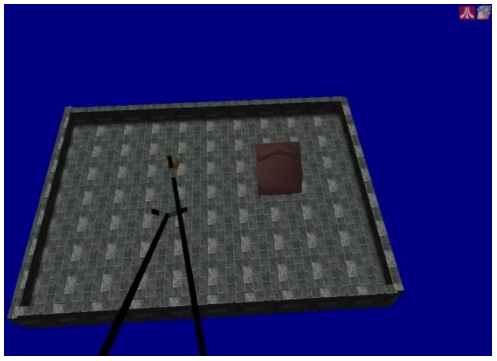


Figure 04 – Screenshot of the 3D software.

IV. Experimental Results

The experiments executed were performed to test processor use in different screen resolutions and performance in terms of response time.

The tests were performed on a netbook computer, used by several users nowadays. The configuration of the netbook is as follows: an Atom 1.6GHz processor, 2GB of RAM, an Intel 950gme with 256MB graphical interface and the Windows 7 32-bits operating system.

A. Processor use

The goal in this experiment is to show feasibility even for a simple computer configuration. The experiment measures processor use in different screen resolutions.

Table 1 presents the results. Note that, even with this simple configuration, the processor use when running the simulator did not go over 50%, proving that the platform can be adopted by single users without fast expensive machines.

B. Response Speed

The goal in this experiment is to measure the time it takes to capture a movement made by the user and execute the action in the screen.

Results showed that the use of the USB communication, to transmit data from the clamps to the computer, once again proved to be an excellent choice. It did not present any noticeable delay to carry out the information and it worked as a power source for the joysticks. The consumption of energy is very low and can be compared to a Generic Joystick.

Table 1 – Computer performance tests.

Mode	Resolution	Clamps	Memory	CPU(%)
full screen	1024x768	1	30 MB	25
full screen	1024x768	2	36 MB	30
window mode	1024x768	1	36 MB	49
window mode	1024x768	2	36 MB	50
full screen	640x480	1	36 MB	22
full screen	640x480	2	36 MB	25
window mode	640x480	1	36 MB	50
window mode	640x480	2	36 MB	50
full screen	1600x1200	1	36 MB	25
full screen	1600x1200	2	36 MB	25
window mode	1600x1200	1	36 MB	50
window mode	1600x1200	2	36 MB	50

In terms of total delay, the results shown that the platform is highly responsive. 50µs were necessary to convert each A/D input. Since the project has five inputs, it took a total of 250µs to convert all the inputs. Adding to that the time to send the data packet to the computer and the time to execute the action gives a total of about 465µs, since the movement was performed by the user and executed on the screen.

V. LIMITATIONS

The software presented a simple environment due to the use of a free simple IDE. The main limitations are lack of high detailed surgery scenario and 3D graphics that are very simple. Although these are real limitations, the authors think a cost effective solution was designed and stressed that the objective is to train the first steps of a non-experienced surgeon.

As showed on Figure 5 the clamp joysticks are on the first prototype. They presented some problems of reliability. The potentiometers used shown to work irregularly with time and started to fail after a long period of use without system initialization. Better potentiometers can easily solve this problem and increase precision.

VI. CONCLUSION

This project presented the development of an inexpensive videolaparoscopy training platform, which can be afforded by single users, focusing in the initial learning steps of videolaparoscopy surgery. The platform proposed aims to provide a rapid initial learning curve by training depth perspective, since a videosurgery gives a 2D perspective in a 3D environment.

The development description started with the creation of hardware and software taking into account the wish to easily upgrade them in the future. The communication between hardware and software was also developed as generic and simple as possible.

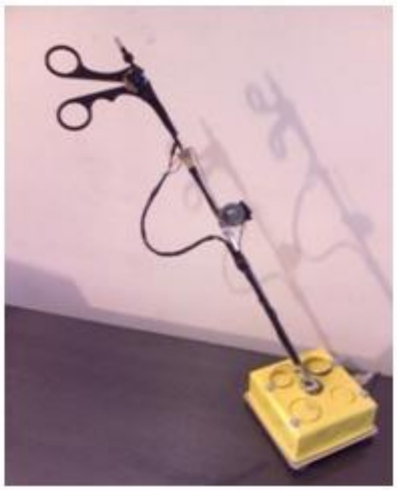


Figure 5 – First prototype of the Joystick Clamp.

The platform is very intuitive offering great usability. From the moment the users start to interact with it, visualizing the 3D graphics animation, becomes obvious which action is needed to be performed at a time. Several users provided a great feedback approving the depth sense presented by the 3D software together with its hardware interaction. The project exploited simple technologies to create a platform that can help medical students in their initial training steps.

Results shown that a powerful computer is not necessary to execute the simulator and that the system is highly responsive, taking about 465µs to perform an action. Future work includes developing more training scenarios and improving the hardware precision.

REFERENCES

[1] Immersion , "Surgical Simulator: The LaparoscopyVR Virtual-Reality System".http://www.immersion.com/markets/medical/products/laparoscopy/index.html . Last viewed 02/14/2010.

[2] SPRING, "SPRING Surgical Simulator". http://spring.stanford.edu/ Last viewed: 02/06/2010.

[3] VRMSS, "Virtual Reality Motor-Skills Simulator". http://www.tri.jabsom.hawaii.edu/surgicalsimweb/vrms.jsf. Last viewed: 03/10/2010.

[4] Guicai Z., Shushang Z., Yangsheng Xu. "A virtual reality based arthroscopic surgery simulator". IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, 2003.

[5] Samur E., et all. "A Haptic Interface with Motor/Brake System for Colonoscopy Simulation". Symposium on Haptic interfaces for virtual environment and teleoperator systems, 2008.

[6] Cote M., et all. “Virtual reality simulator for scoliosis surgery training: Transatlantic collaborative tests”. IEEE International Workshop on Haptic Audio visual Environments and Games, 2008.

[7] Spaelter U., et all. “A 4-dof haptic device for hysteroscopy simulation”. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004.

[8] Microchip, “PIC18FXX5X - Enhanced Flash, USB Microcontrollers with nanoWatt Technology” Datasheet, 2007. http://ww1.microchip.com/downloads/en/DeviceDoc/39632D.pdf. Last viewed:03/10/2010.

[9] HID, “Human Interface Device”. Available at: http://en.wikipedia.org/wiki/Human_interface_device. Last viewed: 03/01/2010.

[10] Lite-c, “Lightweight virtual reality language”. http://www.conitec.com/english/gstudio/litec.htm. Last viewed: 03/10/2009.

[11] USB, “Universal Serial Bus”. http://www.usb.org/developers/docs/. Last viewed: 02/22/2009.

Neo Instant City Construction

Raj Shankar

R & D Division, Panacea Infotech & Consultancy Pvt Ltd(Old) / (YOU-ME & TRIUMPH (Private Individual Enterprise Company), Kathmandu, Baghmati, Nepal)

Neo Instant City Construction – Pyramid Shaped Houses for porous soil/land.
Hell city: Let’s be honest to us. In last 5000 years, we haven’t learnt to construct our cities in a few hours. We spend almost a year to construct one single dwelling house.
City 2201: Several hours, and we have a city. Of course, we have to plan for the city now, today this time.
The scientist has come up with this research work of a miniature 1600 Sq Ft city in which a few identical single dwelling houses can be constructed very quickly. City can be on hills, on planes or where we have porous/soil land.
The research work focuses mainly on planes and planes with porous soil.
What does it entail?

1. Water Resource
2. Concrete Resource
3. Sand Resource
4. Wood Resource
5. Neo Automatic City Builder (Computer, Electronic Devices, Mechanic Devices, Energy)
6. Heavy machines

While more complex cities can be built, the solution focuses mainly on building cities using the following model houses except that concrete will be used instead of bamboo houses.

Keywords:
Artificial Intelligence: Flip flops to replace the hippocampus, pituitary gland, neurons (grey cells).
Virtual Reality: Real but not real. Artificially world and objects in that artificial world look real they way you experience it.
Hippocampus: Human memory
Pitutary gland: The master gland – part of the brain.

1 Introduction

Imagine living in Amazon/Kango basin or for that matter in any other part of the world where the surface of the land is supposed to go down a few inches every few years.
Let’s assume that initial height for the right angled triangle
◿ ABC is:
AB= 50 ft
AD= 30 ft
BD= 40 ft
$AB^2 = AC^2 + BC^2$
$2500 – 900 = BC^2$
$1600 = BC^2$
40= BC

Let’s assume that that reduction in the height of AB per year = AB –x
BC-y
Let’s assume that $x \neq y$
Make the bamboo house’s height uniform:
Solution: Automatic
An electronic device to measure change in height from ground.
Circumference of external Bamboo
Circumference of internal Bamboo
An electronic switch synchronized with height measurement device that calculates the external mass/pressure and internal bamboo is lifted accordingly.

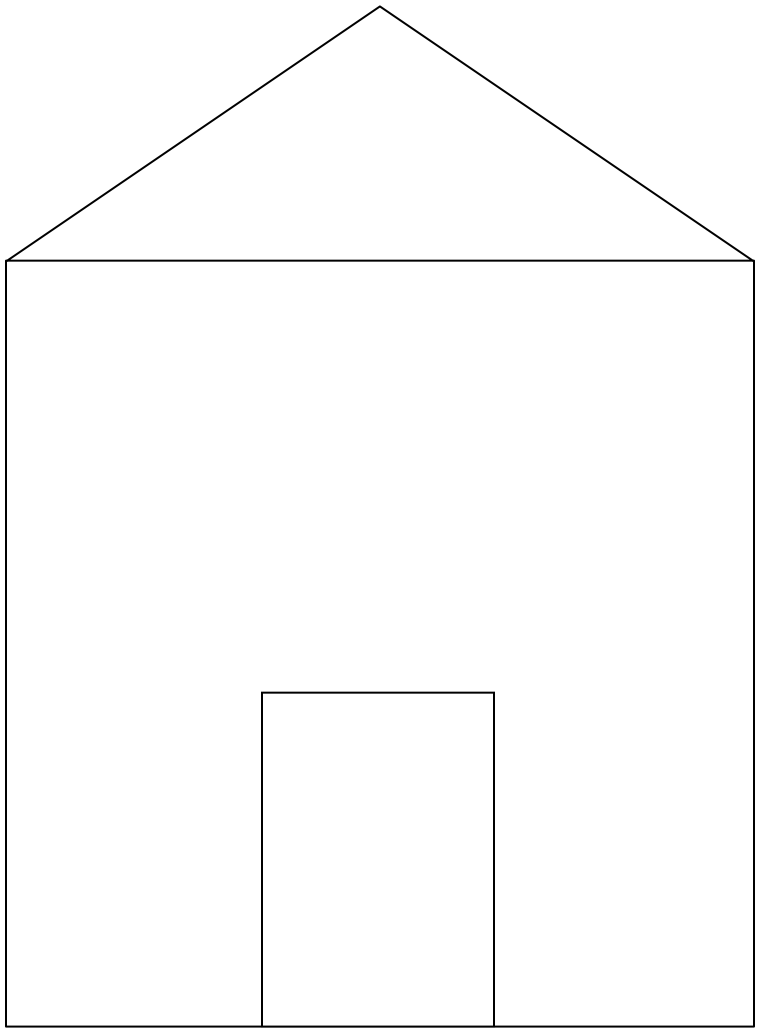


Figure 1 - Pyramid Shaped Houses for porous soil/land
B

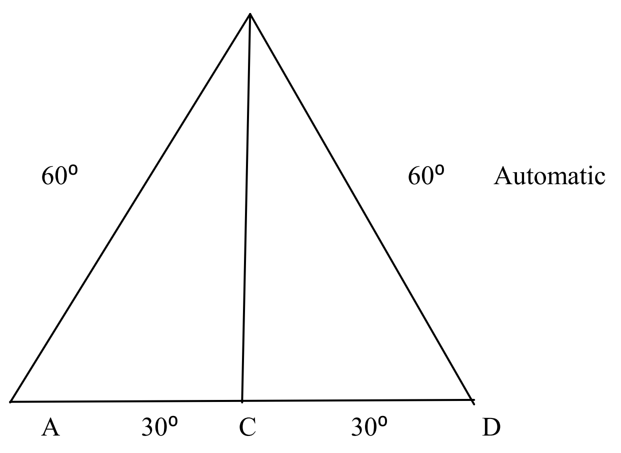


Figure 2 – House Internal architecture

The device also ensures that the half of the house remains a right angled triangle. The analozy can be extended to incorporate the area/of ABD
and area of area/perimeter of ◺ BCD. It also means that you can have two such triangles in your house and this apex is joined by a straight line (bamboo). Any change in the height of the one side can result in introduction of major issues to your house.

Mechanical Solution:
A first class lever device will take care of it. Force required as per the load will be computed.

Imagine going to Hollywood to enjoy "Back to the Future" show. Virtual reality at its best. You actually feel that you are travelling through time and through objects around you. Consider, if we can do it in real life. The solution does not talk about virtual reality but taking the real life project and letting you experience it in virtual reality show will be fun.

The solution works on a square field. The area is 1600 sq feet. Four houses of 10' X 10' followed by empty 10'X 10' space followed by 3 houses of 10' X 10'. An empty space of 10' X 80' .
The solution has x number of iron rods across the square shaped plane ABCD. The solution has 2x number of Iron poles on which each of these iron rods fixed. There are robotic hands taking concrete + cement + sand in the ratio of 2:1:1 mixing them in water (Generally this is not recommended. Recommended mainly in very specific situation. The ratio can be changed by pushing a button on the computer. It is a configurable parameter. The robotic hand transfers the mixture to a bucket that slides on the iron rod and is controlled by a computer. The mixture is put down at a configured place. Multiple buckets can slide on the same rod for multitasking. A computer hand uses another tool to dig about 10' down around the perimeter of 4 (1' X 10') . Once again it is configurable parameter and it can be changed to save the cost or provide more reliability. The robotic hands puts the mixture inside the base and up to 10' above the base. The area above the base is supported by rectangular solid woods covered by plastic towards its interior so that the mixture does not overflow outside the perimeter boundary. Inside the square a machine automatically joins square shaped woods 1' X 1' covered on top by plastic. The external robotic hands put all the mixture on top of the square shaped woods so that it is joined with the external perimeter. 4 very powerful fans from four sides and on top do the drying job. The process is repeated in parallel for all the houses. As far as top pyramid shape is concerned similar arrangement can be made. If we don't need the flat roof, the internal machines will have triangular support. In other words for vaulted ceilings the shape of the internal objected will be different. However, if we do need best of both the worlds to allow attic we probably are going to have to use a very strong triangle shape object starting from the external perimeter to the middle of the house from two sides. Rest of the area will be covered appropriately using the same construction material (mixture which may include small size/large size iron rods as appropriate).

Let's look at other factors. Leveling the plane means adding extra soil. The measurement is done by "Measurement Object". It is a computer software that use Flying Scanner object with scanner to determine the distance from the plane. When it moves through the coordinate xy to x1y1 it knows the exact distance of every square inch on the plane. "Plane It" object does the rest which allows the Construction Controller object to launch the road roller for leveling the ground. Yet another scan from the Flying Scanner helps matches the configurable parameters compared by the "Construction Controller".

The solution does require the need of some heavy equipments. However, they are invoked and controlled remotely. Human involvement is to be reduced to 0% is target goal of this solution. The problem is, « Where will you get the raw materials required for building the house ? » In phase 2, the solution proposes to electronically and remotely control wood cutting machines, blasting small hills or mountains for concrete and cement where applicable. Transporting sand, concrete, water etc in huge containers through private/public railway line to the construction site with 0% human involvement. The scanners again, to know where to plant the bombs and set the timers. Infra-red scanners for scanning living object around to try and save lives. Undoubtedly, there will be conventional security measures such as announcements for the blast will be part of the process. If water source such as river is not around two sites will be needed. Similarly, x number of sites might be required to get the job done. The solution will not delve into too much detail as far as phase2 is concerned. However, to make it interesting the virtual reality program (proof of concept, model, computer based city model experience) will be very useful. It will be a rewarding challenging to turn this virtual reality program into real life construction of city.

The design has been kept simple. A few stepper motors, a few sensors, a few DACs (digital to analog devices), a few analog to digital devices, a computer with graphics card that supports 3D and we are pretty much done as far as the construction controller object is done. RF devices ranging several miles do not require government involvement and they are easy to build and use during transportation. Building a train without presence of a man is not such a big task if you have played with those miniature robot toys that circles around a perimeter and their angular velocity, direction and momentum is controlled by chips that run on battery.

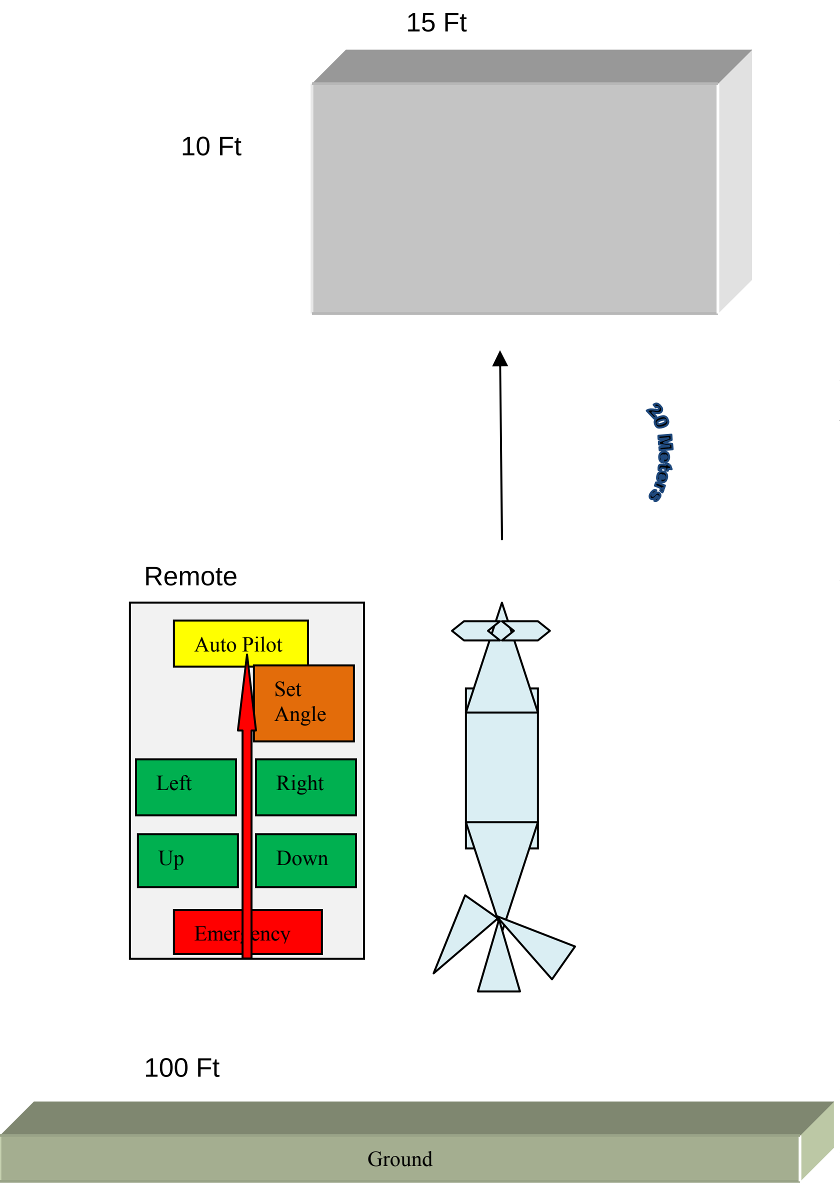


Figure 3 - Neo Autopilot Toy Plane (Scanner Object)

A plane surface for a runway, emergency control buttons for collisions situations i.e., if there is an object in your way and do not have much time to react to it, what will be your action. Will there be a button that puts the aero plane in auto pilot mode or simply detects an object in its way automatically, dives down at 0° angle, has to be shown pictorially in the user manual. If the object in the way of the aero plane is huge then the angle must not change and the speed at which it has to go down further has to be mentioned. The care has to be taken that the plane does not hit the ground and hence the sensors should look for the size of the objects in its way and change the angle appropriately to move in forward direction. If these details are not explained with a diagram, there can be some confusion. The plane has to rotate around a plane surface where it has to land to ensure that the friction level is right and consistent for it stops properly. If it has to land in the even of an emergency, then it must activate a lock system that prevents the parts not to get broken. For instance, there can be a steel box, which surrounds the entire engine at the time of the emergency landing when it has to land at inconsistently plane surface. If the engine fails, a fan at the bottom of the plane should emerge to displace the air down below. This will ensure that the plane will land at a slow speed. If it were a customer aero plane a mechanical fan would have be devised in case of total engine failure. The plane as sensors and electronic devices to read and find the distance from the plane as implemented in NeoVehicle1971X **Vehicle** (2010 ESV).

Neo Time Bomb: The time bomb is use of an explosive such as TNT integrated with a timer. Rather than someone putting manual fire it will be activated using a timer chip. You set the timer chip to a specific time, use an electronic switch that causes a short circuit and you are good to go. Due to the government regulations you have to have license for it. For raw bombs you can use sulphur with glass in an appropriate proportion.

If you look at the design of the infrastructure of the city in following page you would realize that it uses a few existing technologies, integrate them and uses it in a new innovative way. In other words, the existing technologies and processes have been improved. Having said that, there is something very new. We have allowed 360° for a new house before you buy. We allow models on computer or paper before you get your house or city built. This city takes the builders and the homeowners both to a virtual reality experience where they are allowed to see things being constructed as and when it is build. Since the parameters are configurable a homeowner or community may choose to have the polymorphic behavior for a house object. This allows more personalized construction that connects a builder to the homeowner. This allows the residents of the city to know the plan and walk through it.

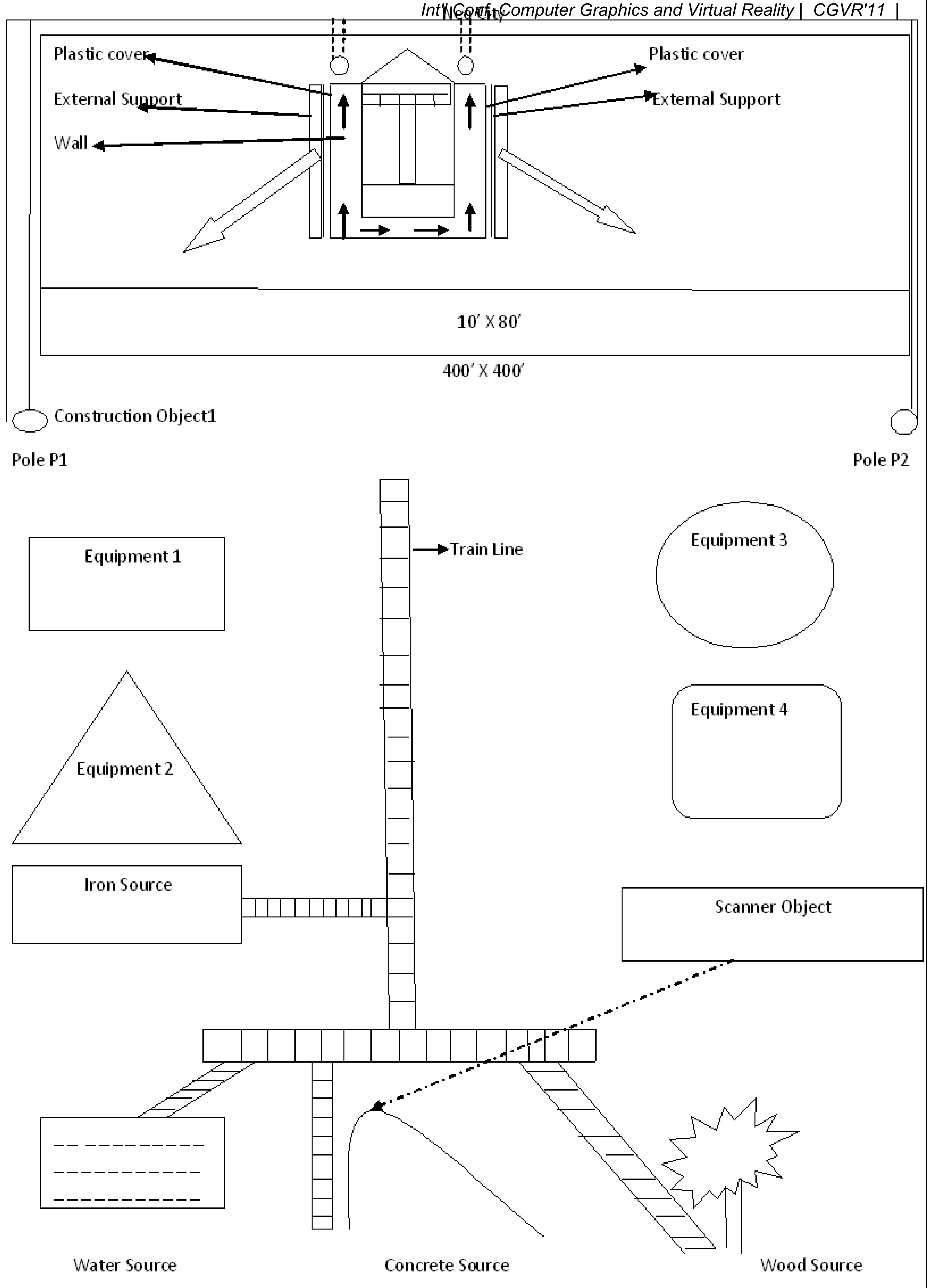


Figure 4 – Neo Instant City Architecture

This allows the residents of the city to know the plan and walk through it.

Construction Controller: First thing first, scan and compute the level of the area under construction. Use marker for digging the trenches so that when the robotic hands dig they know where to dig. This object is responsible for driving the entire solution. 0% human involvement is the target. It relies heavily on the AI object to accomplish the tasks related to use of intelligence. It's AI object depends on the Configuration Object to configure a house attribute or city attribute. There are defaults in the Configuration Object that allows you to construct a city without any additional input.

Deta 1.0: An intelligent bot that scans images through its eyes (cameras) processes it (compares it with the matrix of pixels) to match the objects and takes action. It takes the help of an AI (Artificial Intelligent) object.

Deta 2.0: It operates the Construction Controller, plays with the Configuration Object to accept external input and drive the entire solution.

Virtual Reality Object: Deta 2.0 has speech recognition ability. You use a microphone that has a built in ability to change the amplitude of the voice and does voice sampling. It supports G723 codec. It also accepts messages typed on the keyboard using a wireless interface. Deta 2.0 accepts 10 commands without super user's administrative authorization.

The entire process can be experience by you. The process lets you experience the construction of the city in real time. You will be made to sit before a computer. Your image will be taken by the computer and you will see you moving on the screen. It is actually a deception. There will be a robot behind the scenes that will be moving but the images of the bot will be overlayed by your image. When you will see Deta 1.0 cutting the tree you will be able to see it real time. The camera in Deta 1.0 will pass the image immediately to the computer processor on which you are able to see things moving. As long as it sends 25 images per second you will not lose nothing. The processing of the speed at which the images are taken is done internally in Deta1.0's brain. It is pretty simple. If you are able to take more images in 1 seconds deta simply delays the speed at which you receive and for you things are in slow motion.

Deta 1.0 has built in voice, video and data processing ability up to some extent. When Deta 1.0 receives an external trigger such as heat or pressure it sends it to a device that is connected to your body. All you need is to exert pressure to the external layer of the skin tissues that passes it to the attached dendrites through which it goes to the brain via axons and you feel what Deta 1.0 feels. The data passed on by the external trigger is matched with what is stored in the hippocampus which is processed by the brain as heat, pain, pressure or any other relevant feeling. The pressure machine exerts pressure for you. The heat machine passes on the heat to you. It is only in the future version of Deta that it will be able to send triggers that will traverse through the peripheral nerves to the brain. Different kind of data is passed through the peripheral and central nervous system. This is a bridge between synthetic and organic feelings and the way two different entities interpret the triggers and take action on them. What if deta sends a signal of heat temperature of 30^{o} C and your body somehow receives 5^{o} C. You will be feeling cold in place of heat. The accuracy of the signals is required. To make the experience more fun the signals can be amplified or reduced.

Besides, in future versions Deta can allow you to send signals to the brain without causing external trigger to the skin. In other words, dendrites can be activated to send the signal irrespective of the fact whether or not they have received a trigger. Deta 2.0 will start from the very beginning until you experience the houses being built and will take you inside every room of your house and let you feel every part of it. While this is not in the scope of this solution, Deta 2.0 will allow you interior decoration and send you his feelings with every object added to the room. For instance the color of your couch, the experience of sitting on the couch, playing with T.V. remote etc. You will feel that you are inside your home not inside a house. It is all about playing with the different part of the nervous system.

The special organs to feel anxiety, pain etc are sent appropriate signals through the neurons. Eventually, it is the part of the brains that are responsible for deciphering the information passed onto the body.

In true virtual reality sense the entire process can be experienced by a user even if nothing is happening. An old video will play and activate all the feelings except the image of the viewer will be replaced on line.

Complexities: A sharp hand controlled by another robotic hand to dig the soil requires a mechanism to transfer the soil to the nearest container that get transferred to the railway track. Deta 1.0 drives through a wood, stops before a tree when it receives a signal from its embedded Distance Finder Object that it is the right place to stop. It scans the metals in front of eyes and gets a green color button matrix of pixels. The AI object ask the stepper motor to move at x^{o} angle so that it can apply pressure on the green button (apply x Newton of force using the stepper motor driven robotic hand). The button activates stretches two sword shaped wood cutting tool that cuts the tree. Off course the AI module tells how much deep on what side of the tree so that the tree falls on a particular side. The tree gets cut and the smaller pieces are loaded to the nearest timber godown where the woods are cut in the sizes of 1, 2, …, n sq inch.

One of the issues is to ensure that the plastic used to wrap the mixture right behind the external support should be able to be removed without leaving any mark on the wall. Other things remaining the same, it will work as expected. However, the fans need to dry the walls real quick and they would ideally like to have cottons which is an option that need to be kept in mind. It becomes easier if you have tiles pasted externals. They will be pressed inside the wall up to 0.1 inch. Let's focus on paints now. Cottons can be used using thermal reverse print. Instead of fans, the heat will be used in such cases.

If you are to build thousands of houses, the Construction Controller object simply controls special open pickup trucks capable of automatically getting loaded with equipments and raw materials and move to distant locations using co-ordinate on the earth or degrees in a particular distance. Indeed the Distance Finder does the job for these vehicles. These vehicles can have special purpose intelligent bots capable of sliding through a slope on these vehicles that touches the ground and are able to move around to do the job. They contact the Construction Controller in case they need help. They have their hands with buttons that remotely move heavy equipments.

Virtual Reality object is a bit complicated. While you have to have a special program to move a part of the body at a particular angle, you need the same program to calculate the angle at which the rest of the body should be moved. It requires the knowledge of all the joints in the body and their flixibility (generally 15^{o} for bones). You need to know the flexibility of muscles etc. What about the memory ? You need to have enough memory in the brain of Deta to hold video images. The transfer of video images to the Construction Controller is processor intensive, bandwidth hog (The solution will use « Solving Port Density Problem – Raj Shankar, ISC 2005 « and requires guarenteed delivery of packets over wireless network. The firewire interfaces are required for manual download of the data. There is nothing that stops having a wired transfer of the data to Deta 2.0. However, it will impose restriction on the movement of Deta 1.0.

Image processing will be a daunting task due to humongous volume of images and will require an automated solution. Application of geomatory and trigonomatory will be required. It will require the application of Determinants and Matrix besides progression. Undoubtedly, recursion will be used.

Theta (ø) will ensure that the images do not go outside a particuluar range of co-ordinantes. Besides, their position can be reset. For instance if the vehicle is moving and takes a turn, the entire image can be reset to a default position.
Lets talk about image processing for a bit. The matrix of pixels can be put in a proprietory format through raw data or to a well known format. The real time images may not be required to be converted to three dimensions. However, any two dimension image will be converted to three dimension using a computed formula. The images can be moved at 360^{o}.

The solution will require thorough testing to make the process repeatable. The AI object will be responsible for calculating the time taken by the end to end solution. It will also be responsible for finding areas for improvement.This is all about setting default paraemters and comparing with the configured to values to see if the results are optimum and the best all the time.

The solution requires knowldege and skills from architecutre, transportation, computer science (AI, Graphics, Virtual Reality, 3D, Animation, Image processing, C++, OO, Networking, Embedded Systems, Chips etc).

Since both Deta 1.0 and Deta 2.0 are capable of speaking, they have to have intelligence to answer questions. They are initially configured with default to answer 200 questions. However, they have to be able to compute the values and then answer. They will look for words like (Sum, Total, Plus, Minus, distance, degree, mass, water, gallon, mixture, ton etc) in any combination. The intelligence in Deta will be able to do the arithmtical and logical calculation. For instance, if the question is asked whether or not it is raining today, they should be able to answer after looking at the humidity in the air, pressure on the body etc. This will help them decide whether or not they want to initiate the construction process. It will use Speech recognition (Provisional Patent – Raj Shankar) . This ensures that the SU doesn't have to tell him what his rights are. It will provide QOS (Quality of Service).

The solution integrates dozens of hardwares, softwares and protocols and warrants special integration testing to ensure compatibility, scalibility and interoperability where applicable.

Since every vehicle, tool is controlled remotely they are an Object in true OO sense. Their attributes and behaviors are known to the Construction Controller.

Test Matrix /Integration Variables

Objects	24
Hardware	12
Software	12
Protocols	4
Time Computation	Breakdown
Reliability	
Usability	
Configuration Variables	
Computer Simulation for exptected test results	

The usability part is important since the construction solution arguably is to be used very frequently. Deta is pivot to all operations and the Construction Controller is being driven by it. There will be 100s of test cases to complete the entire system plan. Every time configuration changes a regression test based on static data can be run. Let's talk about configuraton variables for a minute. Houses can have different dimension, roads can have different dimensions. The houses can have different interior and exterior and color. There is nothing that stops the solution to color these houses.

Imagine the number of variables the Construction Controller has to play with. The intelligence is driven by the AI object. In other words, you have to pick the brain of the best in the construction industry to get your heuristic knowledge database.

The part of the construction require blowing part of hills for which you need experts knowledge. The same holds true for the identification of the quality of the tree. The scanner only passed the data to the Construction Controller it is the AI object that has to process the image and look at the data to make sense. If the external visual appearance looks okay it is go for cutting. However, it is the pressure applied on the tree that tells you the quality of the wood.

The same hold true for sand and concrete. However, a final sample of a dry mixture (sand, water, cement, concrete and iron rod [optional)) will decide the quality of the raw material used in the construction work. The breaking point is stored in the database for future. Any issues can be then investigated and traced back to the root. The solution also look at the quality of paint before and after using.

The precision of location of vehicles, tools and equipments is of immense value and is calculated beforehand by the scanner object.

Number of rooms in the house decides the differnt dimension of small trenches inside the perimiter of the house. The process is repeatable and hence any rack can be built using the support infrastructure discussed as above. The water pipes can be put at appropriate places during the time the mixture is being put. In other words, the entire internal and external can be fully automated and it requires computer simulation for evaluating the test result.

Running the computer simulation for a city is one thing and manually verifying the results is quite another. It can take months to verify the results manually and hence the results from the automated regression test on static data.

Deta acquires the knowledge to verify the test results and regression test results will be looked for pass/fail by him and Deta will be able to qualify and call a go with the help AI object and Construction Controller objects.

Apparently, it is a very complex solution that requires patience, diligence and is tedious and rigours. It is job of the Construction Controller to monitor the progress of the entire construction and check the progress against learned data.

One of the challenges is that you can't predict the dimension of the house or the external support as shown in figure 4 and hence the external supports is also assembled on the construction site. Deta has a lot of job to do through out the process. Besides, these huge supports has to be planted in ground one step at a time.

Deta has to compute the height and angle at which the support has to be erected. Overall there are several processes and sub-process that make this solution complex.

Impact: The solution have a significant impact on the construction industry. It will significantly reduce the time to build houses. If the infrastructure is ready beforehand you can get the houses built in one day. It will have significant impact on the way we build and use bots today. Historically we have been dissuaded to use bots due to the energy requirement for moving these heavy robotic hands. However, there is more benefit to it that covers for the cost involved in setting the infrastructure. The buyers of the houses will be willing to pay premium for houses available to them in such short span of time. In case of emergencies a new city can be built and the entire population can move to another area. It will be a new experience for the builders and the residents of the city to walk through the entire city as and when the city is build through virtual reality.

It is touching virtual reality solution in a way it makes your experience look so real. It will have an impact on artificial intelligence as well since the solution can be made commercially viable. The concept can be used by armed forces as well. For the programs in the national interest a steel plant might be required to reduce the cost of transportation. This will make the entire solution independent.

Conclusion : The complexity involved is worth its results once you get an opportunity to look at the city with your naked eyes. It is satisfying to see the use of artificial intelligence that can take the construction and/or other vertical to a level you can call more simplified. Press a button, watch and experience.

References:

LifeSaverNeo1971X Vehicle, ESV, WorldComp 2010 – Raj Shankar
Perception – Raj Shankar
Go Selling _ 1, Raj Shankar
Solving port Density problem, ISC 2005- Raj Shankar
Speech Recognition – Provisional Patent, USPTO - Raj Shankar