Enhancing 3D Applications Using Stereoscopic 3D and Motion Parallax

Ivan K. Y. Li **Edward M. Peek** **Burkhard C. Wünsche** **Christof Lutteroth**
Graphics Group, Department of Computer Science
University of Auckland
PO Box 2100, Adelaide 5001, South Australia
ili004@aucklanduni.ac.nz, epee004@aucklanduni.ac.nz, burkhard@cs.auckland.ac.nz
lutteroth@cs.auckland.ac.nz

Abstract

The interaction with 3D scenes is an essential requirement of computer applications ranging from engineering and entertainment to architecture and social networks. Traditionally 3D scenes are rendered by projecting them onto a 2-dimensional surface such as a monitor or projector screen. This process results in the loss of several depth cues important for immersion into the scene. An improved 3D perception can be achieved by using immersive Virtual Reality equipment or modern 3D display devices. However, most of these devices are expensive and many 3D applications, such as modelling and animation tools, do not produce the output necessary for these devices. In this paper we explore the use of cheap consumer-level hardware to simulate 3D displays. We present technologies for adding stereoscopic 3D and motion parallax to 3D applications, without having to modify the source code. The developed algorithms work with any program that uses the OpenGL fixed-function pipeline. We have successfully applied the technique to the popular 3D modelling tool Blender. Our user tests show that stereoscopic 3D improves user's perception of depth in a virtual 3D environment more than head coupled perspective. However, the latter is perceived as more comfortable. A combination of both techniques achieves the best 3D perception, and has a similar comfort rating as stereoscopic 3D.

Keywords: stereoscopic 3D, anaglyphic stereo, 3D display, head tracking, head coupled perspective

1 Introduction

In conventional applications, 3D scenes are rendered through a series of matrix transformations, which place objects in a virtual scene and project them towards a view plane. The resulting 2D images look flat and unrealistic because several depth cues are lost during the projection. Two important examples of cues are binocular parallax and motion parallax. These two depth cues are equally relevant when perceiving depth in a 3D environment (Rogers & Graham 1979). Hence it is desirable to re-create them to enhance the realism and presence in 3D scenes.

Binocular parallax is the difference of images seen by each eye when viewing a scene, creating a sense of depth. In electronic media, this can be re-created using stereoscopic 3D (S3D) techniques, where different images are presented to each eye through a filtering mechanism. This is usually accomplished via 3D glasses worn by the user, e.g. when viewing 3D movies in the cinema. In computer applications there are several widely available implementations of stereoscopy, namely NVidia's 3D Vision Kit (NVidia 2011) and customised graphic drivers, such as iZ3D (iZ3D Software 2011) and TriDef (Dynamic Digital Depth 2011).

Motion parallax is the difference in the positions of objects as the viewer moves through the scene. When the viewer moves in a straight line, objects further away move less than those closer by. This effect can be re-created using a technique known as head coupled perspective (HCP). However, there is currently no widely available solution for implementing this enhancement in the consumer market.

Another motivation for implementing HCP is the relative costs of hardware. Typical implementations of S3D require specialised glasses and monitors (Sexton & Surman 1999). For example, the NVidia 3D Vision Kit and the required specialised monitor capable of 120Hz refresh rate costs over $500 (NVidia 2011). With HCP, only a head tracker is required, which can be implemented with a $30 web camera. This is not only more affordable for general users, but web-cams are already widely used for other applications, such as Skype and social media, and are increasingly integrated into display devices. Hence the majority of users would not have to spend any additional money for such a set-up.

This paper presents a 3D display solution using anaglyphic stereo and head coupled perspective using cheap consumer level equipment. We investigate the benefit of HCP and S3D for depth perception, confirming some of the previous results in this area, as well as coming up with new results. In addition methods of integrating HCP with existing rendering engines are presented, which will make this technology available to a wide range of users.

Section 2 reviews previous work investigating the use of S3D and HCP. Section 3 summarises virtual reality and head tracking technologies relevant to the design of our solution. Section 4 and 5 describe how we achieve stereoscopy and HCP for general OpenGL applications.

Section 6 evaluates our solution in terms of improving depth perception and comfort. We draw conclusions in section 7 and give an outlook on future work in section 8.

2 Related Work

The term fish-tank virtual reality (Ware et al. 1993) has been used to describe systems which render stereoscopic images to a monitor with motion parallax using head coupled perspective. The original implementation relied on an armature connected to the user's head which is impractical for widespread adoption. This approach to VR was proposed as an alternative to head mounted displays (HMDs) as it offers several benefits including significantly better picture quality and less of a burden on the user. The authors' user testing found that pure HCP was the most preferred rendering technique, although users performed tasks better when both stereo rendering and HCP were used.

Techniques that use cameras for head tracking to prevent the need for the user to wear head-gear have been also developed (Rekimoto 1995). The use of the vision based tracking system over the physical one for head tracking did not deteriorate the performance of the system, despite the fact that the viewer's distance from the screen was not calculated. Face tracking was performed by subtracting a previously obtained background image and then matching templates to obtain the location of the viewer's face in real-time.

The above techniques rely on generating the appropriate images frame-by-frame depending on the position of the viewer. This makes them inappropriate for presenting live-action media, which must be recorded beforehand. Suenaga et al. (2008) propose a technique that captures images for a range of perspectives and selectively displays the one most appropriate for the viewer's position. This is however infeasible for video as hundreds of images must be captured and stored for each frame in order to support a large number of viewing orientations.

Several methods have been developed to enhance the quality of fish-tank VR. An update frequency of at least 40Hz and a ratio of camera-to-head movement of 0.75 have been found to provide the most realistic effect (Runde 2000). A virtual cadre can also been employed to improve the depth perception of objects close to or in front of the display plane, while amplifying head rotations can allow a viewer to see more of a scene which improves immersion (Mulder & van Liere 2000).

Yim et al. (2008) found that head tracking was intuitive when implemented into a bullet dodging game. Users experienced higher levels of enjoyment during game-play. The implementation used the Nintendo Wii-remote setup described by Lee (2008). One downside of the setup is the sensor bar attached to the head, which was cumbersome and received some negative feedback. This highlights the importance of unobtrusive enhancement implementations.

Sko and Gardner (2009) used the fish-tank VR principal to augment Valve's Source game engine. Head coupled perspective along with amplified head rotations were integrated as passive effects, while head movement was also used to perform various in-game tasks such as peering, iron-sighting and spinning. Stereo rendering was however not performed. User tests found that the amplified head rotations "added life to the game and made it more realistic", while the concept of HCP was liked by the users, limitations regarding the latency and accuracy of the head tracking degraded the experience.

These findings suggest that head coupled perspective is an important part of recreating a realistic scene in virtual reality, and that it can improve spatial reasoning and help users perform tasks quicker and more efficiently (Ware et al. 1993). Therefore creating a method that can reliably upgrade 3D computer graphics pipelines to render fish-tank VR could have significant positive impacts on visualizing data, computer modelling and gaming without the need for expensive dedicated VR equipment.

3 Background

3.1 Virtual Reality

Virtual Reality (VR) is a broad term that can be used to identify technologies that improve the user's sense of virtual presence, or immersion in a virtual scene. Complete immersion involves manipulating all the user's senses. Our research focuses on improving visual immersion. Hence technologies such as interaction and haptic and aural feedback will not be investigated.

Current VR display technologies are divided into three main categories: fully immersive, semi-immersive and non-immersive. Fully immersive systems, such as head mounted displays (HMD) and CAVE, are known to improve immersion into 3D environment (Qi et al. 2006). However, they are implemented at high costs and with cumbersome setups. With HMD it is impossible to quickly switch between virtual reality and real life as the user is required to wear some kind of head gear (Rekimoto 1995). These disadvantages prevent widespread adoption of such systems in everyday life.

Non-immersive VR presents the opportunity for adoption in everyday situations because of their unobtrusive design and availability of inexpensive implementations. HCP and S3D techniques are classed as non-immersive techniques because the user views the virtual environment through a small window, usually a desktop monitor (Demiralp et al. 2006). Table 1 summarises some common VR display technologies.

	Type	Immersion	Resolution	Cost
Desktop	Non	Low	High	Low
Fish-tank	Non	Medium	High	Low
Projection	Semi	Medium	Medium	Medium
HMD	Full	High	Low	Medium
Surround (e.g. CAVE)	Full	High	Medium	High

Table 1: Comparison of VR display technologies (Nichols & Patel 2002).

Most VR displays are 2D surfaces. In order to accurately represent a 3D scene, depth cues lost in the 3D to 2D conversion process must be recreated. Several of these cues can be represented in 2D images without

special devices. Examples of depth cues emulated by most modern graphics engines are distance fog, depth-of-field and lighting and shading.

Motion and binocular parallax cannot be recreated passively on standard display systems. However through the use of head coupled perspective and stereoscopy these cues can be artificially created.

3.2 Stereoscopy

Stereoscopy refers to the process of presenting individual images to each of the viewer's eyes. When rendering scenes with slightly different perspectives this process simulates binocular vision. The differences in the perceived images are used by the brain to determine the depth of objects, a process known as stereopsis. The most commonly available methods of displaying stereoscopy are anaglyphs, polarised displays, time multiplexed displays and autostereoscopic displays.

Anaglyphs encode the images for each eye in the red, green and blue channels of the image. The user needs to wear glasses that selectively filter certain channels. There are several combinations of channels in use with the most popular being red/cyan.

Polarised displays work by polarising the individual images in different directions, while the user wears glasses with polarised lenses which block the images with the opposite polarisation. Time multiplexed displays work by displaying the different images alternatively while the glasses alternate which eye receives the image. Autostereoscopic displays work by directing the images from the screen to each eye using a surface covered in tiny lenses or parallax barriers. Table 2 illustrates some of the main differences between the technologies.

	Resolution	Colour	Cost	
			Display	Glasses
Anaglyph	High	Poor	Low	Low
Polarized	Half	Good	High	Low
Shutter	High	Good	Medium	Medium
Autostereo	Half	Good	High	N/A

Table 2: Comparison of stereoscopic display technologies (Fauster 2007).

Implementing stereo rendering is difficult to add externally to the rendering pipeline as it requires draw calls to be duplicated and selectively modified. For this reason it was decided to use an existing program to add stereoscopic rendering. The two programs that were tested are the iZ3D driver (iZ3D Software 2011) and NVidia's 3D Vision driver (NVidia 2011). While this will not allow fine-tuned control over the stereo rendering, it ensures compatibility with the wide range of 3D displays available. Care must be taken to ensure the external stereo functionality does not interfere with the head coupling technique. This will be accomplished by ensuring that the algorithms implementing this functionality have different entry points to the rendering pipeline (Gateau 2009).

3.3 Head Coupled Perspective

Head coupled perspective (HCP) is a technique used to emulate the effect of motion parallax on a 2D display. HCP is implemented for 3D rendering applications by projecting virtual objects' vertices through the screen plane to the location of the viewer. The point on the screen plane that intersects the line between the object and the viewer is where the object is drawn on the display. This projection is typically performed through a series of matrix multiplications with the object's vertices. Normally the view point is a virtual camera inside the scene that corresponds to a static point in front of the display. This however does not take into account the motion of the user's head, and so the projection becomes incorrect when the user's actual viewing position is different to the assumed position. Figure 1 shows how motion parallax causes an object to appear in a different location on the screen when viewed from a different position. HCP works by coupling the position of the user's head to the virtual camera such that the users head movements in front of the display cause proportional movements of the virtual camera in the scene. The ratio of head-to-camera movement is referred to as the gain of motion parallax and the value that gives the most realistic effect varies from person to person (Runde 2000).

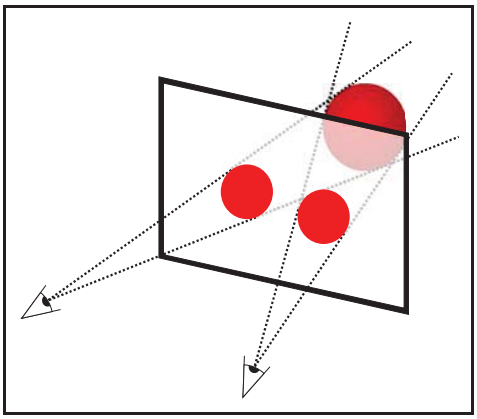

Figure 1: Diagram illustrating how the correct projection of a virtual object to a surface changes depending on the viewing position.

3.4 Head Tracking

An adequate head tracker is needed for an effective implementation of HCP. We therefore evaluated the temporal error, spatial error and latency of head trackers.

Visual head trackers are most suitable for our research because of their low hardware costs and unobtrusive nature. A NZ$ 45 Logitech C500 web camera was used with computer vision techniques, which extract the position of the user's head. The web camera operates at VGA resolution of 640 by 480 pixels with 30 fps. Implementing face detection and tracking from scratch is very complicated if accurate and reliable tracking is desired. Therefore, we evaluated the tracking performance with and without anaglyph glasses of freely available APIs, which can be integrated directly with as little modification as possible.

3.4.1 FaceAPI

The FaceAPI library (Seeing Machines 2010) was first evaluated due to the fast response and excellent accuracy seen in Sko's demonstration videos (Sko 2008). When tested without stereoscopic glasses, the FaceAPI was able to track up to 1.5m in range. It could also handle very fast head movements and the latency was unnoticeable.

However, it encountered some difficulty when tracking users wearing anaglyph glasses. In some rare instances, the user's face could not be detected at all. For most of the time, the position of the eyes was shifted to the lower edge of the glasses. This is shown in Figure 2, where the yellow outline represents the predicted positions of facial features. As the tracking with anaglyph glasses is inadequate, the ARToolkit library was investigated.

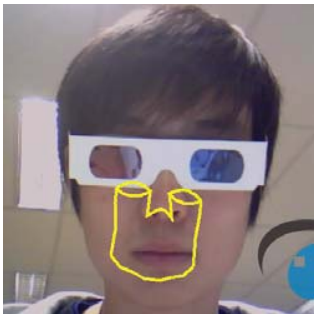

Figure 2: Inaccurate tracking with FaceAPI when anaglyph glasses are worn. The predicted face positions are indicated by the yellow outline.

3.4.2 ARToolkit

Fiducial marker tracking was found to be the most suitable alternative because a paper marker is sufficient to track 6 degrees of freedom. The marker can be attached to the anaglyph glasses without affecting the user. ARToolkit is a open source library designed to track fiducial markers, such as ones shown in Figure 3 (ARToolworks 2011).

Figure 3: Example fiducial markers used with ARToolkit.

The library was able to detect fast motions and had little latency. The main drawback was the restricted distance range and marker size. Relatively large markers are required for a good tracking range. With the limited space on the anaglyph glasses, the maximum marker size without additional installations was 3.5 cm by 3.5 cm. We found that a 3.5 cm marker allowed reliable tracking up to a distance of 60 cm. If the user moved further away the tracking became jittery and unstable.

The cause of the limitation was identified as the template matching stage of the tracking algorithm (ARToolworks 2011b). At long ranges, the inside of the marker became too small to be successfully matched.

3.4.3 OpenCV

An attempt was made to develop a simple marker tracker with OpenCV (Willow Garage 2011). The tracker would not need rigorous template matching because exact marker identification is not necessary for head tracking. Hence, detection of smaller markers is possible and the range can be extended to fit the requirements.

The pre-processing stage performs image conditioning and filtering operations. A bilateral filter was applied to smooth the image while retaining the sharpness of the edges. Histogram equalisation was used to increase the contrast of the image.

Contours were extracted from an edge image created using the Canny edge detector method. The thresholds for the detector were changed dynamically to keep the number of detected contours within reasonable range. This was necessary to keep the processing time for one frame roughly constant.

Squares were detected and stored with polygon approximation technique. They are then normalised and template matched with an empty pattern inside to determine the likelihood of it being a valid marker. Two of the most likely markers are used for pose estimation.

Some optimisation was performed to reduce the computation time required. The region of interest of the image was limited once the marker has been detected. This is done assuming the marker does not disappear instantly.

This algorithm was able to reliably detect stationary markers up to a distance of 1.2 m. However, motion blur and processing time were the two major problems which caused faulty detections for moving markers.

The amount of motion blur from the webcam caused the contours to break whenever the marker moved. Even with the shortest exposure setting, the black edges of the markers were smeared by the white regions surrounding it. Different markers were tested but without any success.

Performance was another issue which prevented further development of the marker tracker. Even with a limited region of interest the total processing time for one frame was 60 ms (see Table 3), which did not allow smooth tracking with 30 frames per second.

We therefore found that the free version of the FaceAPI is the most suitable software for implementing HCP.

Stage	Time Taken
Grab Image	1 ms
Pre-Processing	43 ms
Contour extraction	9 ms
Marker and pose extraction	3 ms
Total Execution	60 ms

Table 3: Execution times for each stage of the OpenCV marker tracking algorithm.

4 Stereoscopic 3D for OpenGL Applications

Stereoscopic 3D was implemented with the anaglyph technique. This relies on colour channels to selectively filter the image presented to each eye. The advantage of anaglyph 3D is the cheap cost of hardware – no special monitor is required and the coloured glasses costs approximately NZ$1 per pair.

The OpenGL library has natural support for rendering anaglyph 3D images with the `glColorMask()` functions. The scene is rendered twice, once from the correct perspective of each eye, to replicate binocular parallax. The perspective corrections are performed identically to HCP described in section 5. The difference is that the scene is rendered once from each eye on different colour channels and blended together.

Different colour combinations were tested to determine the pair which gives the least amount of ghosting on the screen. The ghosting occurs depending on the saturation and hue of colour output with the monitor. Since this is a hardware limitation, it cannot be fixed by making adjustments on the screen or program.

The colour pairs tested were: red-cyan, red-blue, and red-green. The red-blue gave the least amount of ghosting but caused a shimmering effect because of the high contrast between the two eyes. Red-cyan had the best colour but also the most ghosting. Red-green was chosen as it had only minor ghosting with minor shimmering.

Since the scene is rendered twice in every frame, care has to be taken that the scene is not too complex and an acceptable frame rate is achieved. The time delay between the head movement and image update has a significant effect on the user's depth perception when the delay is 265 ms or greater (Yuan et al. 2000). Most graphics applications are designed to have a frame rate of at least 30 frames per second. Hence this problem is unlikely to occur in practice.

5 Head Coupled Perspective for OpenGL Applications

In order to make HCP available to a wide range of users it must be integrated into existing applications. Figure 4 shows the general layers of a 3D computer graphics application. Modifying the source code of an application or rendering engine or developing plug-ins is not an option, since this solution is not general enough, adds a high level of complexity, and requires suitable access mechanisms. Since there are only two graphics libraries commonly used on desktop computers, OpenGL and Direct3D, it was decided to perform the integration at the graphics library level.

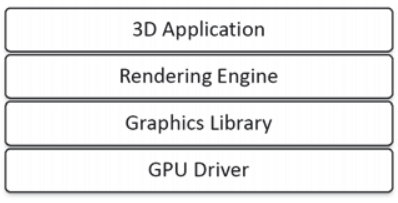

Figure 4: Hierarchy of program libraries for a normal 3D application.

5.1 Hooking

Since the integration is done at the library level where source code is not available a technique known as *hooking* was employed. This term refers to techniques that intercept function calls made by another program. There are two different ways of hooking, either statically by physically modifying the program's executable file before it executes or by dynamically modifying it at runtime. The second approach was chosen as graphics libraries are frequently updated. This would be problematic for static hooking as the library would need to be modified every time an update occurs.

With dynamic hooking the hooking program consists of three sections: the injection, interception and application specific code. The injection part of the program is responsible for getting the hooking program to run in the target's address space. The interception code reroutes function calls within the program to the application specific code. The last section is the code specific to the application, in this case the head coupling algorithm.

For the injection section CBT-style hooking is used (Microsoft 2011). This type of hooking uses native Windows functions to inject a dynamic-link library (DLL) into the address space of processes which receive window events. When the DLL is injected the operating system invokes the DLLMain function which starts the function interception.

Because the target OpenGL library is a DLL the functions are intercepted by modifying the import descriptor table (IDT) using the APIHijack library (Brainerd 2000). The IDT maps the names of the functions exported by the DLL to the address of their code. Whenever a program tries to call a function from the DLL it will first find the address of the function by looking it up in the IDT. By changing the addresses in the IDT to point to the modified functions, calls to the original function can be efficiently redirected with almost no overhead.

An alternative method for function hooking exists and is called *trampolining*. This technique is more flexible than modifying the IDT as it works for functions not in a DLL, however there is more overhead as several redirections are needed (Hunt & Brubacher 1999).

5.2 OpenGL Library Modification

Section 3.3 explained that the implementation of head coupled perspective relies on modifying the perspective transformation matrix. With OpenGL there are two different rendering pipelines used, a fixed-function pipeline and a programmable pipeline. Each of these approaches uses a different method to load transform matrices: in the fixed-function pipeline functions load the matrices individually, while with the programmable pipeline the matrices are combined by the program and passed to OpenGL as a single transformation matrix. Because of this modifying the projection transformation in the programmable pipeline is very difficult. For this reason only the fixed function pipeline was modified to support head coupled perspective.

The functions used to load the perspective projection matrix in the fixed-function pipeline are the

`glLoadMatrix` functions and the helper functions `glFrustum` and `gluPerspective`. With the `glLoadMatrix` functions different types of matrices can be loaded, not only projection ones. To ensure that the head coupling algorithm is only performed on projection transformations, the matrices loaded via `glLoadMatrix` are checked to see if they match the template shown in figure 5.

$$\begin{pmatrix} \frac{\cot\left(\frac{y}{2}\right)}{r} & 0 & 0 & 0 \\ 0 & \cot\left(\frac{y}{2}\right) & 0 & 0 \\ 0 & 0 & \frac{f}{n-f} & -1 \\ 0 & 0 & \frac{nf}{n-f} & 0 \end{pmatrix}$$

Figure 5: Generic perspective projection matrix shown in row-major format where *y* is the vertical field of view, *r* is the aspect ratio, *n* is the distance of the near clip plane and *f* is the distance of the far clip plane.

Projection matrices are also used for other applications such as shadow mapping. In this case the projection matrix must not be modified. In order to check the current use of the projection matrix we assume that the main camera projection is the only one that uses a non-square texture buffer. This is based-on the assumption that the application runs in full-screen mode, which usually results in an aspect ratio of 4:3 to 16:9. Thus any projection matrix with an aspect ratio of 1 bypasses the head coupling algorithm.

Conventional perspective transforms use a virtual camera position and camera field-of-view (FOV) and aspect-ratio to determine how the scene is projected. With head coupled perspective the projection is determined by the head position and the position and size of a virtual window. While the head position is determined automatically, some method is needed to convert the virtual camera specified by the application to a virtual window. As the virtual camera corresponds to an assumed head position a simple mapping is done where the virtual monitor is mapped at the same distance and size from the virtual camera as the real monitor is from the normal viewing position. Figure 6 illustrates this relationship.

This approach however has some disadvantages, one being that this does not always produce good results as the scene can be at an arbitrary scale. For this reason the mapping parameters can be changed at runtime by the user to make the mapping more realistic. Another disadvantage is that zooming does not work in the application as the virtual camera's FOV is ignored. Also applications tend to have a large FOV so the user can see a large portion of the virtual world, but this process significantly reduces the effective FOV giving the illusion of tunnel vision. These are inherent disadvantages with using a correct perspective projection. One potential way to get rid of them would be to use a hybrid approach that uses an approximation of head coupled perspective with conventional virtual camera projection.

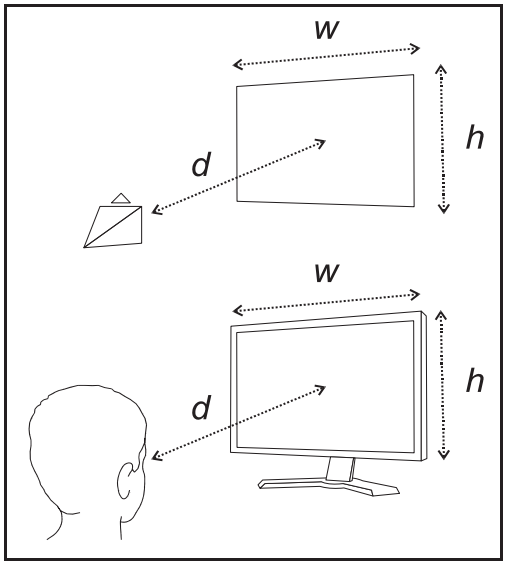

Figure 6: Diagram illustrating the initial mapping between the virtual camera and window compared to the physical viewer and monitor

The above described mapping process is performed whenever the loading of a valid projection matrix is detected. Using the calculated virtual monitor we create a new projection matrix, which is loaded instead of the original one.

6 Results

A user study was performed to determine the effectiveness of the implemented HCP and S3D enhancements. Previous work using a customized set-up reported significant improvements in speed and accuracy when performing a tree tracing task with enhancement (Arthur et al. 1993). In that work a head tracking armature was used, while shutter glasses (with a significant amount of cross-talk) were used for S3D. This is significantly different from the vision-based head tracker and anaglyph 3D used in our evaluation.

There has been no recent study comparing the effectiveness of HCP and S3D enhancements directly. Hence, it is worthwhile to investigate whether the enhancements have different effects on users with our newer, cheaper, and less obtrusive technology.

6.1 User Study Design

An OpenGL test application was written for testing and recording depth perception in a virtual 3D environment. The scene was adapted from Marks (2011), who tested HCP for use in a virtual surgery simulation system. The test scene consists of 4 square plates inside a box as illustrated in Figure 7. The plates were about 9 units wide and 1 unit thick. Participants had to determine the plate closest to them using the available depth cues. This was

repeated 50 times using 4 different set-ups: no enhancement, HCP, S3P, and HCP & S3P. For each set-up the difficulty was progressively increased by linearly decreasing the maximum difference of depth between the plates from 10 units to 3.3 units.

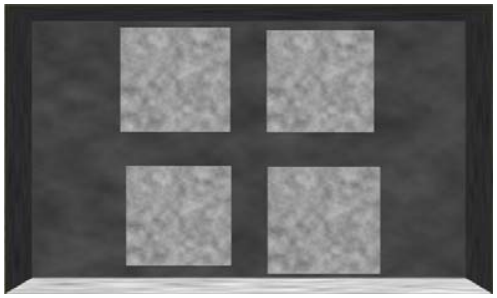

Figure 7: Screenshot of the user study application with the "no enhancement" set-up.

Selection of the closest plate was made by using keys '1', '2', '4' and '5' on the numerical keypad, which corresponded to the layout of the plates on screen. Users were allowed to provide a "don't know" answer by pressing the '0' key.

The application recorded the participant's choice and the ordering of the tiles to determine the accuracy of the user's response. The reaction time was determined by recording the time elapsed between the display and selection of plates. In addition the length of time that head tracking was lost during each test was recorded in order to prevent distortion of the results.

Different depth cues were available in each set-up as shown in Table 4. Note that in order to isolate the measurement of the effect of binocular and motion parallax, most depth cues normally present in a 3D scene had been intentionally removed. The scene shown in Figure 7 uses size as the only depth cue for the "no enhancement" set-up.

Enhancements	Depth Cue
No Enhancement	Size
HCP	Size & motion parallax
S3D	Size & binocular parallax
HCP & S3D	Size, motion parallax & binocular parallax

Table 4: Depth cues available in each set-up of the user test.

A set of pilot tests were performed with 5 participants and several problems were found with the initial test scene. Shading of the plates affected the subjects' depth perception. For some configurations the chosen lighting options resulted in the lower edges of the plates and the background having very similar in colour, which made it difficult to judge size. All of these problems were fixed before beginning the user study.

6.2 Set-Up and Methodology

The user study was performed with a Dell 2009W monitor and Logitech C500 webcam in a shared computer laboratory. Users were required to sit down while using the application.

Before beginning the test, each participant was given a briefing of the experiment. A pre-test questionnaire was completed to determine the amount of prior experience with the HCP and S3D enhancements.

Each participant had to do use the application with the four different set-ups in random orders. A training scene at the beginning of each phase enabled users to become familiar with the controls and enhancement. During training users were given feedback on their selections (i.e. whether their choice was correct). The recording phase began when the participants felt competent at completing the task at a relatively fast speed.

After completing a task participants had to answer a questionnaire. For each task the amount of discomfort, realism of the technique and perceived ease and accuracy of performing the task were assessed with 5-point Likert scale. Open ended questions assessed the depth cues and users were allowed to give general comments regarding the test. After completing the tests for all four set-ups, the comfort, preference, and perceived effectiveness and ease-of-use were ranked for each enhancement.

6.3 User Study Results and Discussion

The user study had 13 participants aged 18 to 24 years old. All of them were university students. The majority of participants had previous experience with conventional 3D applications, most commonly with Blender, CAD tools and/or computer games. 10 users had experienced S3D at least once from either 3D movies or comic books. None of the subjects had prior experience with head coupled perspective systems.

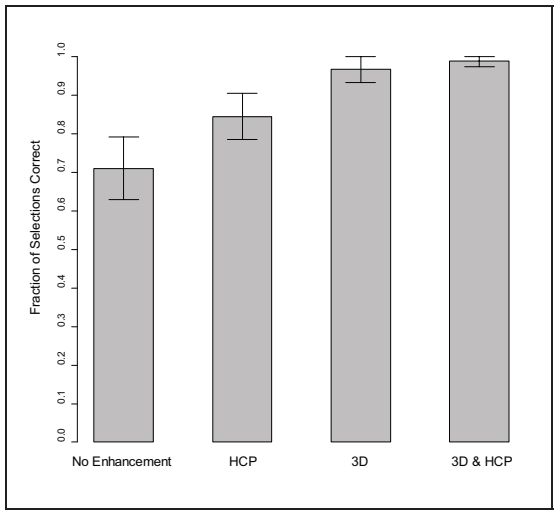

Figure 8: Bar plot of the percentage of times where the closest plate was correctly identified. The overlayed interval represents the 95% confidence interval.

The results of using the four set-ups for testing the accuracy of depth perception are shown in Figure 8. All

enhancements provided improvement in the accuracy of depth perception. Combining HCP and S3D resulted in the highest number of correct answers (98.8%). For S3D, HCP and no enhancement the number of correct answers was 93.3%, 78.4%, and 62.9%, respectively. In the combined enhancement test, subjects reported that they found it easy to use S3D to determine depth when the difference between plates is large, while HCP was most useful when the difficulty increased.

When compared to Arthur et al. (1993), HCP and S3D still provided a general improvement of depth perception accuracy. In our case the S3D result is significantly better than for HCP, which is the opposite of the findings reported by Arthur et al. We hypothesise that the following factors could have led to this difference:

- Our vision based tracker has a higher latency and less sensitivity than the armature tracker used by Arthur et al.
- The effectiveness of S3D and HCP depends on the chosen application.
- Our anaglyph S3D has less crosstalk, which makes it more beneficial than the old shutter technology.

Unfortunately we did not have access to the equipment used by Arthur et al. and to their software. This prevented us from performing more research into the reasons for the disparity between the results. An important conclusion we can draw, however, is that the benefits of S3D and HCP depend on the chosen implementation and use case.

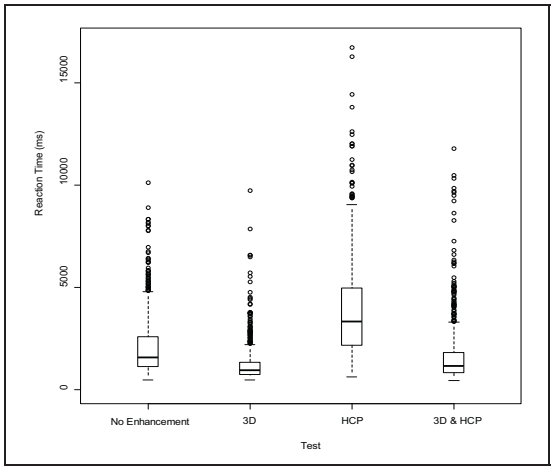

Figure 9: Box plot of the task completion times for each set-up.

The task completion time for each set-up is shown in Figure 9. The median task completion time when using HCP is approximately twice of the time when using no enhancement. This can be attributed to the physical movement required for HCP. Conversely, S3D was fastest because no movement was required. When using both HCP and S3D the majority of participants used S3D at the beginning, and then switched to HCP when the task got more difficult, i.e. depth differences decreased. Hence the recorded time is only slightly longer than for S3D.

Table 5 shows a pairwise comparison of how comfortable participants perceived the different set-ups. Using no enhancements received the highest comfort rating, whereas S3D received the lowest comfort rating. This can be attributed to the discomfort of wearing a physical device. More importantly, most users complained about colour fatigue after performing the S3D tests. Interestingly HCP was perceived as more comfortable than no enhancement. One reason might be that users only had to use head motions when displayed configurations were ambiguous, whereas in simple cases size was sufficient to give the correct answer.

Enhancements	**1**	**2**	**3 4**	
1 (None)		42%	67%	67%
2 (HCP)	50%		42%	50%
3 (S3D)	25%	42%		25%
4 (HCP & S3D)	25%	42%	50%	

Table 5: Pairwise comparison of comfort ratings. The values indicate the proportion of users who found the row enhancement was more comfortable than the column enhancement.

Table 6 shows a pairwise comparison of participants' preference for completing the given task using different set-ups for depth perception. Very few users preferred the no enhancement option. HCP and S3D rated about equally well, and HCP & S3D combined received the highest ratings and was preferred over all other options by the majority of participants.

Enhancements	**1**	**2 3 4**		
1 (None)		8.3%	8.3%	16.7%
2 (HCP)	83.3%		41.7%	25%
3 (S3D)	83.3%	50%		8.3%
4 (HCP & S3D)	75%	66.7%	66.7%	

Table 6: Pairwise comparison of preference of enhancements. The values indicate the proportion of users who found the row enhancement to be more preferable than the column enhancement.

In summary S3D provides the best depth perception. Both accuracy and task completion time was better than for HCP. In terms of user comfort HCP is favoured over S3D. Colour fatigue is a major drawback of anaglyph S3D and usually occurred after only around 10 minutes. Most real world 3D applications require considerable longer interaction times. Overall the combination of HCP and S3P was preferred, mostly because of its superior depth perception. Although HCP had a lower performance than S3D, it is still able to offer a considerably improved depth perception with no negative effect on user comfort. HCP is hence the most viable solution for applications requiring improved depth perception during protracted tasks.

6.4 Integration of HCP into Blender

We added HCP to the popular 3D modelling and animation tool "Blender". An example of the thus

achieved effects is illustrated in Figure 10. The addition of HCP dramatically improves depth perception and perceived realism. Several limitations exist and we made the following observations:

- The modifications described in section 5 currently only affect the display routine. Interaction with objects, such as selecting vertices, does not work correctly when the head position changes.
- Blender only updates the view when the displayed scene changes. Head movements are not detected by Blender itself and hence redisplay must be initiated manually.
- HCP will be rendered in any perspective view (but not orthographic view). Hence the traditional 4-view layout works as expected with the addition of HCP for the perspective view.
- If a display window is not full-screen and not centred, then the view projection is incorrect since we assume that the user is seated in front of the centre of the display window. This is, however, barely noticeable when using only one display window.
- When using more than one active perspective view they are all rendered with the same head offset. Ideally we would like to adjust the head offset depending on the user's position relative to the display window's position on the screen.

Figure 10: The effects achieved by integrating HCP into the modelling and animation tool "Blender".

7 Conclusion

Head coupled perspective is a viable alternative to stereoscopy for enhancing the realism of 3D computer applications. Both head coupled perspective and stereoscopy improve a user's perception of depth in a static environment. Our testing showed that head coupled perspective is slightly less effective than stereoscopy. However, we believe that HCP can become more popular in future due to its simple implementation and high comfort rating, especially for time-consuming tasks. A key requirement will be the development of technologies for adding HCP to existing applications and media, without necessitating modifications.

Integration with the OpenGL library has been accomplished using hooking. We demonstrated the concept for the popular modelling tool Blender. The application worked well for exploration tasks in full-screen mode. However, problems exist when using smaller windows and when interacting with the scene, such as selecting objects. In addition the possible field-of-view is constrained. These shortcomings need to be overcome before the technique can be used in a wider range of applications.

8 Future Work

Future work will improve the integration of our technology into the programmable rendering pipelines of both Direct3D and OpenGL. This would allow for head coupled perspective to be used in a much larger range of applications. To do this more sophisticated ways of isolating the projection matrix would need to be developed as the transformation matrices are typically pre-multiplied inside the application. The current integration method also breaks mouse input, as the mouse picking no longer uses the same projection as what is used to render the scene. Further research is needed to determine if a solution to this is possible with the current integration approach.

We also want to develop better algorithms for mapping from the application's virtual camera to a virtual window. This would greatly improve the usability of applications that require a large field-of-view, such as first-person shooters, and also require less calibration by the end-user to get a realistic effect.

Further testing needs to be done to determine how the performance benefits from stereoscopy and head coupled perspective change depending on the type and difficulty of the task being evaluated. It would also be interesting to determine how user preferences change when taking into account cost and rendering performance penalties. In addition testing needs to be performed using static and dynamic environments, and a direct comparison with other technologies is required.

Another major area for future research is adapting the head coupling algorithm so that it works with pre-recorded media such as film and television, not just 3D computer applications. Limited 3D information suitable for this can be extracted automatically from 2D frames using the algorithm by Hoiem et al. (2005).

9 References

Arthur, K. W., Booth, K. S., Ware, C. (1993), Evaluating 3D task performance for fish tank virtual worlds. *ACM Transactions on Information Systems* **11** (3), pp. 239-265.

ARToolworks (2011), ARToolKit Home Page, http://www.hitl.washington.edu/artoolkit (Last accessed: 2011, August 26).

ARToolworks (2011b), How does ARToolkit work? http://www.hitl.washington.edu/artoolkit/documentation/userarwork.htm (Last accessed: 2011, August 26).

Brainerd, W. (2000), APIHijack - A Library for easy DLL function hooking, http://www.codeproject.com/KB/DLL/apihijack.aspx (Last accessed: 2011, August 26).

Demiralp, C., Jackson, C. D., Karelitz, D. B., Zhang, S., Laidlaw, D. H. (2006), CAVE and Fishtank Virtual-Reality Displays: A Qualitative and Quantitative Comparison, *IEEE Transactions on Visualization and Computer Graphics* **12** (3), pp. 323-330.

Dynamic Digital Depth (2011), TriDef - Stereoscopic 3D Software, http://www.tridef.com/home.html (Last accessed: 2011, August 26).

Fauster, L. (2007), Stereoscopic Techniques in Computer Graphics, Project report, Technische Universität Wien,

Austria, http://www.cg.tuwien.ac.at/research/publications/2006/Fauster-06-st/Fauster-06-st-.pdf.

Gateau, S. (2009), The In and Out: Making Games Play Right with Stereoscopic 3D Technologies, *Game Developers' Conference*, http://developer.download.nvidia.com/presentations/2009/GDC/GDC09-3DVision-The_In_and_Out.pdf.

Hoiem, D., Efros, A., Hebert, M. (2005), Automatic photo pop-up, in *ACM Transactions on Graphics* **24** (3), pp. 577-584.

Hunt, G., Brubacher, D. (1999), Detours: Binary Interception of Win32 Functions, in *Third USENIX Windows NT Symposium*, USENIX, http://research.microsoft.com/apps/pubs/default.aspx?id=68568 (Last accessed: 2011, August 26).

iZ3D Software (2011), iZ3D Drivers download page, http://www.iz3d.com/driver (Last accessed: 2011, August 26).

Lee, J. C. (2008), Head Tracking for Desktop VR Displays using the Wii Remote, http://johnnylee.net/projects/wii/ (Last accessed: 2011, August 26).

Marks, S. (2011), A Virtual Environment for Medical Teamwork Training With Support for Non-Verbal Communication Using Consumer-Level Hardware and Software, PhD Thesis, Dept. of Computer Science, University of Auckland.

Microsoft (2011), MSDN Windows Hooks, http://msdn.microsoft.com/en-us/library/ms632589(VS.85).aspx (Last accessed: 2011, August 26).

Mulder, J. D., van Liere, R. (2000), Enhancing Fish Tank VR, in *Proceedings of the IEEE Virtual Reality 2000 Conference* (VR '00). IEEE Computer Society, Washington, DC, USA, pp. 91-98.

Nichols, S., Patel, H. (2002), Health and safety implications of virtual reality: a review of empirical evidence, *Applied Ergonomics* **33** (3), pp. 251-271.

NVidia (2011), NVidia 3D Vision, http://www.nvidia.co.uk/object/GeForce_3D_Vision_Main_uk.html (Last accessed: 2011, August 26).

Qi, W., Taylor, R. M., Healey, C. G., Martens, J.-B. (2006), A comparison of immersive HMD, fish tank VR and fish tank with haptics displays for volume visualization, in *Proceedings of the 3rd symposium on Applied perception in graphics and visualization* (APGV '06). ACM, New York, NY, USA, pp. 51-58.

Rekimoto, J. (1995), A vision-based head tracker for fish tank virtual reality-VR without head gear, in *Proceedings of the Virtual Reality Annual International Symposium (VRAIS '95)*. IEEE Computer Society, Washington, DC, USA, pp. 94-100.

Rogers, B., Graham, M. (1979), Motion parallax as an independent cue for depth perception, Perception **8**, pp. 125-134.

Runde, D. (2000), How to realize a natural image reproduction using stereoscopic displays with motion parallax, *IEEE Transactions on Circuits and Systems for Video Technology* **10** (3), pp. 376-386.

Seeing Machines (2010), FaceAPI Homepage, http://www.seeingmachines.com/product/faceapi/ (Last accessed: 2011, August 26).

Sexton, I., Surman, P. (1999), Stereoscopic and autostereoscopic display systems, IEEE Signal Processing Magazine **16** (3), pp. 85-99.

Sko, T. (2008) Using Head Gestures in PC Games, http://www.youtube.com/watch?v=qWkpdtFZoBE (Last accessed: 2011, August 26).

Sko, T., Gardner, H. J. (2009), Head Tracking in First-Person Games: Interaction Using a Web-Camera, in *Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction: Part I* (INTERACT '09), Springer-Verlag, Berlin, Heidelberg, pp. 342-355.

Suenaga, T., Tamai, Y., Kurita, Y., Matsumoto, Y., Ogasawara, T. (2008), Poster: Image-Based 3D Display with Motion Parallax using Face Tracking, in *Proceedings of the 2008 IEEE Symposium on 3D User Interfaces* (3DUI '08). IEEE Computer Society, Washington, DC, USA, 161-162.

Ware, C., Arthur, K., Booth, K. S. (1993), Fish tank virtual reality, in *Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems* (CHI '93). ACM, New York, NY, USA, 37-42.

Willow Garage (2011), OpenCV Wiki, http://opencv.willowgarage.com/wiki (Last accessed: 2011, August 26).

Yim, J., Qiu, E., Graham, T. C. N. (2008), Experience in the design and development of a game based on head-tracking input, in *Proceedings of the 2008 Conference on Future Play: Research, Play, Share* (Future Play '08). ACM, New York, NY, USA, pp. 236-239.

Yuan, H., Sachtler, W. L., Durlach, N., Shinn-Cunningham, B. (2000), Effects of Time Delay on Depth Perception via Head-Motion Parallax in Virtual Environment Systems *Presence: Teleoperators and Virtual Environments* **9** (6), pp. 638-647.