A PARALLEL MEDIATED REALITY PLATFORM

Rosco Hill, James Fung and Steve Mann

ECE Department
University of Toronto
10 King's College Road
rosco@cs.toronto.edu, {fungja,mann}@eecg.toronto.edu

ABSTRACT

Realtime image processing provides a general framework for robust mediated reality problems. This paper presents a realtime mediated reality system that is built upon realtime image processing algorithms. It has been shown that the graphics processing unit (GPU) is capable of efficiently performing image processing tasks. The system presented uses a parallel GPU architecture for image processing that enables realtime mediated reality. Our implementation has many benefits; the graphics hardware has high throughput and low latency; the GPU's are not prone to jitter. Additionally, the CPU is kept available for user applications. The system is easily constructed, consisting of readily available commodity hardware.

1. INTRODUCTION

A "mediated reality system" (MRS) is one that can augment, diminish, or otherwise alter the visual perception of reality. By way of explanation, "virtual reality" creates a completely computer-generated environment, "augmented reality" uses the existing real-life environment and adds computer-generated information thereto, "mediated reality" filters and adds to reality to enhance the visual information presented to the user.

A tetherless mediated reality system typically runs on a lower power battery-powered wearable [1, 2] computer system with miniature eyeglass mounted screen and appropriate optics to form the virtual image equivalent to an ordinary computer desktop. However, because the apparatus is tetherless, it travels with the user. Users wearing a MRS could use it for information selection and retrieval, personal safety, enhanced situational awareness and navigation.

This paper presents a parallel GPU architecture that forms the basis of a realtime MRS. There are three problems in mediated reality that are addressed in this paper: world registration, object recognition and scene generation. All three problems are addressed within the context of the parallel graphics architecture and the solutions proposed run on the GPU's. Two proof-of-concept applications are presented here to demonstrate the effectiveness of parallel graphics architecture for realtimemediated reality. In keeping with the parallel spirit of the MRS, the applications leverage the parallel GPU architecture.

The reality window manager (RWM) application is a user interface for MRS. Traditional models for displaying information on a computer display are constrained to a 2D coordinate systems. For example the X11 window system. In some augmented reality systems, windows simply float in front of the user. We present a natural mapping that displays 2D information in a 3D world, similar to the manner in which regular non-electronic information is presented (plastered on signs and billboards). The RWM mediates the contents of planar patches in the real world with information or application windows such as terminals, web browsers, and email. The RWM interface projects information more naturally than the current WIMP standard. Figure 1 shows the author's desktop running the RWM application.

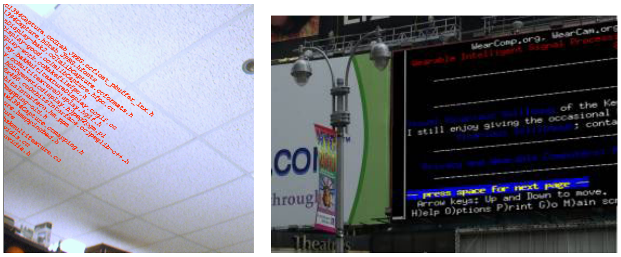
WearComp.org, WearCam.org
I still enjoy giving the occasional
of the Ke
: conta
— press space for next page —
Arrow keys: Up and Down to move.
H)elp O)ptions P)rint G)o M)ain scr

Fig. 1. The GL terminal application and the glynx web browser registered on quadrilaterals in the author's environment.

The virtual architect application superimposes the model of a building on top of the real building, which matches the perspective of the viewer. This enables an architect to envision and communicate the look and feel of the final building. For example the Art Gallery of Ontario recently unveiled the $180-million transformation designed by architect Frank Gehry. A video capture of the model on display is used to overlay the current building so that the viewer can see what the building will look like after construction is finished in 2007.

2. BACKGROUND

Previous work on markerless tracking and registration typically does not run in real time [3, 4]. In this paper we leverage the featureless realtime projective image registration system using parallel graphics processing units that is described in [5].

Other researchers have presented realtime systems that use specialized hardware or markers for regristration. Feiner et al. [6] presented a system with realtime world registration and scene synthesis using 3D trackers so that the real and virtual worlds

Thanks to NSERC and Ontario Graduate Scholarships for support. Thanks to nVIDIA for equipment donations.

could be registered. This research explored three ways to present augmented large windows that the user could explore by looking around: floating windows, windows that remained fixed to a location in the user's display, and world fixed windows. The system presented by the author is much more flexible, it requires no specialized hardware and allows information to be blended onto any planar surface in the environment. Azuma et al. [7] explored a hybrid approach using inertial, optical and compas inputs to track registration. In constrast, our system focuses on running on hardware available at the local computer store.

It has been shown that graphics processing units are capable of efficiently performing computer vision tasks [8, 9, 5]. The GPU's found in most personal computers and laptops typically exceed, in number of transistors as well as in compute power, the capabilities of the CPU. Fung et al. [5] demonstrated a system that achived realtime video framerate projective image registration using a single GPU. Previous work has discussed implementations which use a single graphics card for realtime computer vision. Work has also been done to apply GPUs to general purpose computing and other specialized tasks beyond computer graphics alone[1]. Our focus is to investigate how graphics cards can function in parallel as the basis for a realtime mediated reality system.

3. ARCHITECTURE

The MRS has three main subsystems: world registration, object recognition and scene generation. Each subsystem runs on the parallel graphics processing architecture.

The parallel graphics processing architecture consists of six commodity graphics cards. Five cards are on the PCI bus and each have a GeForce FX 5200 GPU. The sixth graphics card is on the AGP bus and has a GeForce FX 5900 GPU. The subsystems distribute their workload amongst the graphics cards according to a workload estimation heuristic. Figure 2 illustrates the architecture.

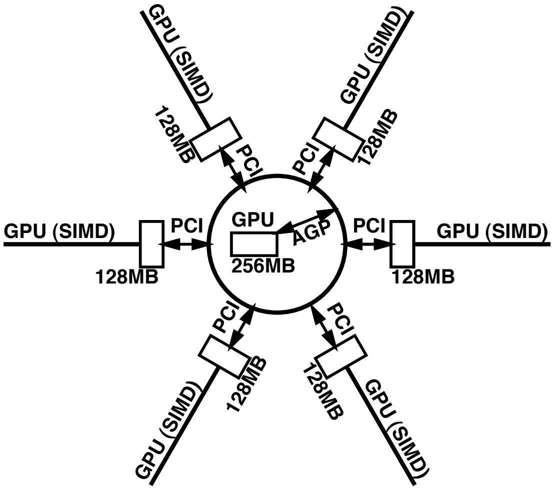

Fig. 2. Parallel GPU Architecture. 6 PCI graphics cards communicate with a faster AGP card via the PCI and AGP buses.

4. PARALLEL WORLD REGISTRATION

To effectively implement mediated reality in arbitrary environments, the world registration must be accurate, fast and reliable. We present afast and reliable image registration algorithm that forms the basis for world registration in our realtime mediated reality system.

4.1. Image Registration

The system presented uses a parallelized version of the projective image registration presented in [5] to achieve realtime robust registration. The mediated reality system incorporates a camera, with a diverter material, such that the lens of the camera captures the image that is going into the eye. The system tracks the relative and absolute position and orientation of the camera in terms of the projective coordinate transformation (PCT) between the current frame of video and the previous frame[2]. This featureless image motion estimation algorithm tracks the ego-motion of the camera so that the virtual world remains aligned. The parallel algorithm solves the Equation 1 to find the projection of an image to register itself with another image.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \frac{\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}}{\begin{bmatrix} c_1 & c_2 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + 1} \tag{1}$$

The number of iterations required for the solution to converge is related to the amount of change between two successive frames. For example, two frames with little relative movement converge in around 10 iterations, however, a sudden jerky movement of the camera a laggy system cause the number of required iterations to increase to as many as 50 (system delay and lag are discussed in Section 8). The workload estimation heuristic distributes the work amongst more cards according to the estimated number of iterations required to converge.

4.2. Registered Mediated Reality

As the user scans his/her environment by looking around, the system tracks the ego-motion of the head by solving successively for the PCT[10]. Images from the current camera orientations can be synthesized by composing the relative PCT with the spatial locations of the virtual objects. In OpenGL this corresponds to applying the appropriate matrix multiplication to the model view matrix. To illustrate this, consider Figure 3. We see a approximate bounding planar patch that is being tracked by the system. As the user looks to the left, the majority of the planar patch is outside of the field of view of the user. The successive compositions of the PCT obtained from the image registration algorithm keep the position of the bounding planar patch registered with the real world.

5. OBJECT RECOGNITION

Our object recognition subsystem has two methods for detecting objects. The first is quadrilateral contour detection for finding of planar patches in the field of view. Information is typically presented on some 2D surface, and as many applications are constrained to a 2D coordinate system, we find planar patches to isolate the areas where information mediation will take place.

[1] see http://www.gpgpu.org

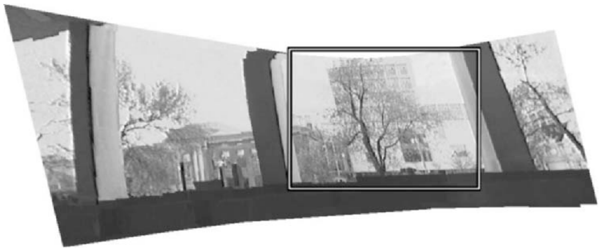

Fig. 3. Head Tracking: The virtual square remains aligned as the user looks around.

Finding candidate quadrilaterals in a scene amounts to finding either dark quadrilaterals on a lighter background or light quadrilaterals on a darker background. The steps for the algorithm are described below. The steps are repetitive in that they are repeated for a number of scales in the image, and for a number of lightness/darkness thresholds.

The video image is pre-blurred with a gaussian filter to remove noise, then a multi-scale pyramid is formed. At each image resolution we threshold the image and compute the approximate quadrilateral bounding box for the detected contours. There is a parallelism inherent in this computer vision algorithm, and each video card runs the quadrilateral detector in parallel at different thresholds. The result is communicated back to the MRS and the best quadrilateral is selected for mediation.

The object recognition also has a template matching search method based on Eigenspace techniques[11]. In advance, many images of an object are taken under various poses. The GPU computes the basis images for the objects, and stores them in a database. Once the database is populated, the template matcher must search for the object under a variety of orientations and scales. The computational requirements for traversing the search space for the template matching algorithm is:

$$[(640 \text{ x } 480) \text{ x 3 colors}] \text{ x 12 orientations x 5 scales} \qquad (2)$$

The template matching algorithms typically search at 5 different scales and each search is independent, hence the search can be performed in parallel on different GPU's. Each GPU performs the search at one scale, and communicates the result back to the program. The program them decides if there is a match and the reality mediator subsystem can diminish or augment information on or around the object.

6. SCENE GENERATION

The MRS generates a scene that contains augmented or diminished reality. The unmediated portions of the field of view are simply re-rendered. The mediated portions are rendered as semi-opaque displays on top of the original view. The RWM application renders a OpenGL based Unix$^{\text{TM}}$terminal window that matches the position and orientation of the planar patch it is attached to. The virtual architect application renders the model of a building on top of the actual building, such that two registration points (i.e. the front doors) remain aligned. It is important to note that the registration points do not need to be in every scene, only in the initial scene. The user is free to look around the world and return his/her gaze to the mediation zone and the mediated objects remain fixed to their respective objects.

All scene generation is performed on the AGP graphics card. The scene consists only of texture mapped quadrilateral, which contain either the video signal or the information for augmenting or diminishing reality. For the RWM application an OpenGL terminal is rendered at a specified position, zoom and orientation and for the virtual architect application a texture mapped quadrilateral is rendered.

7. HUMAN-COMPUTER INTERACTION

Our implemenation allows users to specify a planar patch in their field of view by either selecting 4 corners manually, or allowing the object detection subsystem to automatically select the best planar patch from the field of view. By clicking the moues button, the user attaches an application window to the object. The system tracks the affine transformations from the ego-motion of the camera and applies this transformation to all of the windows displayed on the desktop. The net effect is an application window that remains registered on the planar patch in location, zoom, orientation, keystoning and chirping, as shown in Figure 1. The user is free to move and look around, and the application will remain attached to that object when the user returns.

8. SYSTEM DELAY AND JITTER

Dynamic errors occur in tracking and registration because of system delays. The total delay in an MRS system is the time from moment the system captures an image until the time that the image is re-synthesized to the user. MRS systems can include delays from the tracking subsystem, communication delays, network delays, scene recognition delays, scene generation delays and finally scanout time from the frame buffer.

System delays cause error when motion occurs. Take the case with a static scene with a virtual window attached to a billboard shown in Figure 1. Suppose the user turns their head at time t and the system delay is 100ms, then the user will see the virtual window drawn at the same location in his/her field of view until it is finally updated at time $t + 100$ms. These images are therefore incorrect for the time when they are viewed and the two worlds are poorly registered.

The parallel MRS runs end-to-end on the graphics cards. The frame-grabber data is transfered directly to the texture memory on the GPU, and the GPU program runs as soon as the texture memory is updated. Furthermore, the mixing of the physcial world and the virtual world is performed on the video card. For a fixed number of virtual windows, both aspects of the RWM execute in a predictable time constant. This eliminates "jitter" and allows the user to adapt to a constant system delay.

The system delay is extremely short on modern graphics cards. Our prototype system has a NVIDIA NV35 chip, which is the GPU found on the GeForce FX 5900 Ultra board. The NV35 GPU has a transistor count of about 135 million. The system delay is 33 ms for capture, 3ms per registration estimation, and 10ms rendering, plus 16.67ms. Thus our system has an end-to-end lag of 46ms, or roughly 1.5 frames on a 30fps signal. Our system lag is dominated by the slow 30fps frame capture we use and a 60fps capture board would cut this in half (though would likely increase the cost, size and power consumption of the wearable).

It is believed that system delays are not likley to completely disappear anytime soon. The most optimistic prediction made in

[12] is that HMD system with a 10ms lag might become possible in the next few years, but the drastic cut in throughput and the expense required to construct the system would make alternate solutions attractive. The paper also concludes that reducing end-to-end delay to the point where it is no longer a source of registration error is not practical. The system we present has no expensive custom parts. It runs in realtime at 30 frames per second. The signal processing delay is 10ms, plus scan-in and scan-out time, while costing only a few hundred dollars to build. Furthermore our system is robust to registration errors.

9. CONCLUSION

This paper presents the first realtime markerless mediated reality system using low-cost off the shelf computer graphics hardware. The low cost and widespread availability of computer graphics hardware will make hardware accelerated mediated reality possible. Additionally, the algorithms used in mediated reality are inherently parallelizable. A parallel graphics architecture is proposed and the delay and jitter of the system are analyzed. The signal processing delay is under 45 ms and our system is robust to registration errors. Two proof-of-concept mediated reality application demonstrate the viability of the parallel graphics hardware architecture. Both applications run within the parallel graphics architecture and the CPU is kept available for other applications.

10. REFERENCES

[1] S. Mann, "'mediated reality'," TR 260, M.I.T. M.L. vismod, Cambridge, Massachusetts, http://wearcam.org/mr.htm, 1994.

[2] Steve Mann, *Intelligent Image Processing*, John Wiley and Sons, November 2 2001, ISBN: 0-471-40637-6.

[3] A.J. Azarbayejani, B. Horowitz, and A. Pentland, "Recursive estimation of structure and motion using relaitve orientation constraints," in *Proc. CVRP*, 1993, pp. 294–299.

[4] K.N. Kutulakos and J.R. Vallino, "Calibration free augmented reality," in *IEEE Trans. on Visualization and Computer Graphics*, 1998, pp. 1–20.

[5] James Fung and Steve Mann, "Computer vision signal processing on graphics processing units," in *To appear in the Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004)*, Montreal, Quebec, Canada, May 17–21 2004.

[6] Steven Feiner, Blair MacIntyre, Marcus Haupt, and Eliot Solomon, "Windows on the world: 2d windows for 3d augmented reality," in *ACM Symposium on User Interface Software and Technology*, 1993, pp. 145–155.

[7] Ronald Azuma et al., "Tracking in unprepared environments for augmented reality systems," *Computers and Graphics*, vol. 23, no. 6, pp. 787–793, 1999.

[8] James Fung, Felix Tang, and Steve Mann, "Mediated reality using computer graphics hardware for computer vision," in *Proceedings of the International Symposium on Wearable Computing 2002 (ISWC2002)*, Seattle, Washington, USA, Oct. 7 – 10 2002, pp. 83–89.

[9] Ruigang Yang and Pollefeys M., "Multi-resolution real-time stereo on commodity graphics hardware," *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE*, vol. 1, pp. 211–217, 2003.

[10] Felix Tang, Chris Aimone, James Fung, Andrej Marjan, and Steve Mann, "Seeing eye to eye: a shared mediated reality using eyetap devices and the videoorbits gyroscopic head tracker," in *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR2002)*, Darmstadt, Germany, Sep 1 - Oct 1 2002, pp. 267–268.

[11] M.J. Black and A.D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *Proc. 4th European Conf. on Computer Vision*, pp. 329–342, April 1996.

[12] R. Azuma, "A survey of augmented reality," 1995.