EyeTap Video–based Featureless Projective Motion Estimation Assisted by Gyroscopic Tracking for Wearable Computer Mediated Reality

Chris Aimone, James Fung, Steve Mann
University of Toronto
10 King's College Road
Toronto, Canada
{aimone,fungja}@eyetap.org, mann@eecg.utoronto.ca

Abstract

In this paper we present a computationally economical method of recovering the projective motion of head mounted cameras or EyeTap devices, for use in wearable computer mediated reality. The tracking system combines featureless vision and inertial methods in a closed loop system to achieve accurate robust head tracking using inexpensive sensors. The combination of inertial and vision techniques provides the high accuracy visual registration needed for fitting computer graphics onto real images and robustness to large interframe camera motion due to fast head rotations. Operating on a 1.2 GHz Pentium III wearable computer with graphics accelerated hardware, the system is able to register live video images with less than 2 pixels of error (0.3 degrees) at 12 frames per second. Fast image registration is achieved by offloading computer vision computation onto the graphics hardware, which is readily available on many wearable computer systems. As an application of this tracking approach, we present a system which allows wearable computer users to share views of their current environments that have been stabilized to another viewer's head position.

Keywords: Video Head Tracking, Hybrid Tracking, Eyetap, Drift Correction, Augmented Reality, Mediated Reality

1 Introduction

In a computer mediated or augmented reality system, the user visually perceives an environment where real and computer generated objects coexist. The goal in such systems is to augment, diminish, or otherwise modify what the user would normally see. In order to achieve this, the system must run in real time and must align the real and virtual objects with each other. Achieving this registration is a major challenge in augmented and mediated reality. Solving this problem requires accurate tracking of the viewers position and orientation relative to the objects in their environment.

One major approach to achieving registration is vision-based tracking. These techniques involve estimating the user's position and orientation from images captured by a wearable camera. Several systems have been developed that achieve extremely good registration using vision tracking in controlled environments. These systems generally make use of fiducial makers at known positions in the environment. Many of these systems employ hybrid techniques that combine inertial methods with vision, thereby making the tracking more robust and less computationally intensive [10] [2] [11].

In unprepared environments, the registration problem is much more difficult. In this situation tracking must be accomplished without special markers, and a model of the environment must somehow be acquired. You et al [12] presented a system that tracks absolute three–degrees–of–freedom (3DOF) head rotation using a gyroscope and corrects registration by tracking natural features with a vision algorithm. The system achieved high precision, but was also very computationally intensive. On an SGI Onyx2, the system was reported to run at a frame rate of 10 fps. Satoh et al [8] also proposed a hybrid scheme for 3DOF rotation tracking using precision fiber optic gyroscopes with vision based drift compensation. The system relied upon the gyroscope for video rate tracking, and used the slower natural feature tracking algorithm to compensate for drift.

This paper presents a similar system that uses inexpensive vibrating element gyroscopes for high frame rate tracking, and a vision tracking algorithm for low frame rate drift compensation. In contrast to the methods proposed by Satoh or You, this system does not attempt to resolve absolute position or rotation to be used with a known model of the environment. Instead it tracks points in the user's field of view by spatially registering images of what the user sees, thereby eliminating the need for a priori knowledge of the scene's 3D structure for adding virtual content that appears to stay fixed to the real world . There are many types of mediated reality applications where knowledge of actual depth in the scene is unnecessary. An example of this is adding annotations to a scene such that the annotations appear to stay fixed in the environment as the user looks around (see figure 1).

This system tracks the visual motion of a static scene in the users field of view in terms of a projective coordinate transformation (PCT) applied to entire image. The system uses a featureless method for vision tracking called VideoOrbits, and has a closed loop structure that allows the mechanically based inertial sensors to be continuously calibrated during use. This type of tracking works best when the user stands still while looking around, but because tracking by PCT is less constrained, acceptable registration is achieved in many situations where the user moves about (translates) as well.

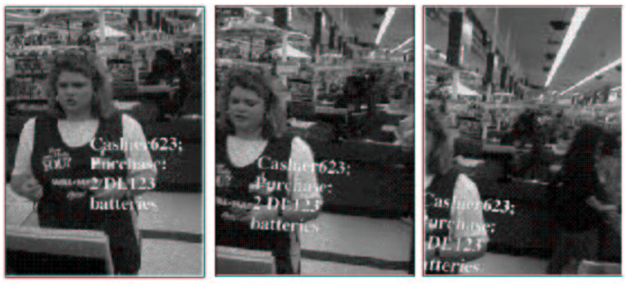
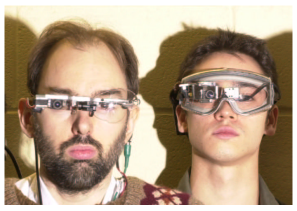
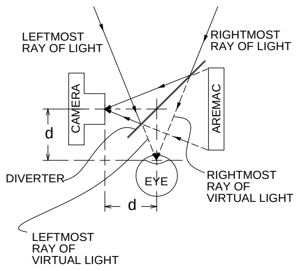


Figure 1: An annotation overlay fixed to the scene using projection motion tracking. By applying the best fit projective coordinate transformation found between frames of video to the overlaid text, the annotation appears as a planer object that is part of the scene. This makes for us to maintain our immersion in the real environment as well as making it easier to interpret the text.

2. EyeTap

In order to capture, process and redisplay what is in the user's natural field of view we use devices called EyeTap. The EyeTap (see Figure 2) is a wearable computer device that enables the user's vision to be computationally processed and modified in real–time [5]. An EyeTap incorporates a camera and a display arranged with appropriate optical geometry such that the camera and eye have equivalent optic centers. This allows the display to orthospatially [5] re-synthesize the incoming light. This arrangement eliminates parallax between what the camera sees what the eye would see (in the absence of the device), allowing for the seamless addition of virtual information, even when the EyeTap device has been manufactured with partial optical transparency. (Optical transparency is often desired for partial mediation, allowing the user to see the majority of their environment with natural light, rather than completely virtual light.) Furthermore, EyeTap devices have been built with fully opaque diverters (see figure 2). Fully opaque diverters allow for EyeTap devices to completely remove or replace portions of the user's field of view [6].


(a)



(b)

Figure 2: (a) Two examples of EyeTap devices that tap the right eye. Note that both eyes can be tapped if stereo computer mediated reality is desired. (b) EyeTap schematic.

3 Projective Head Tracking

The proposed head tracking system combines a featureless vision based tracking algorithm called VideoOrbits[1] [5] with two small, low cost, vibrating element gyroscopic sensors. The system has a closed loop form that enables the calibration of the inertial sensor during use and adds robustness despite independently moving objects (objects whose motion is unrelated to the user's head motion) in the field of view.

VideoOrbits finds best fit Projective Coordinate Transformations (PCTs) to register pairs of overlapping images or scene content (Equation (1)).

$$\mathbf{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \frac{\mathbf{A}[x,y]^T + \mathbf{b}}{\mathbf{c}^T[x,y]^T + 1} = \frac{\mathbf{A}\mathbf{x} + \mathbf{b}}{\mathbf{c}^T\mathbf{x} + 1} \tag{1}$$

where $\mathbf{A}, \mathbf{b}, \mathbf{c}$ are the eight parameters of the projective coordinate transformation (PCT).

This set of transformations, when applied across the entire image, can describe exactly the coordinate transformation between two images where all the objects in the scene are static and the camera is only free to rotate and zoom. It is also exact for a planar scene imaged from arbitrary locations. The motion of a head mounted camera or EyeTap between successive frames of video can be approximated very well by a pure rotation, since we often scan our environment using head rotations. This allows VideoOrbits to provide extremely good registration between EyeTap video frames [5], and forms the basis for tracking physical head rotations from a real-time video sequence.

While this technique is exact for situations where each user views their world through pure rotation, performance

[1]VideoOrbits available for free at www.wearcam.org/orbits/

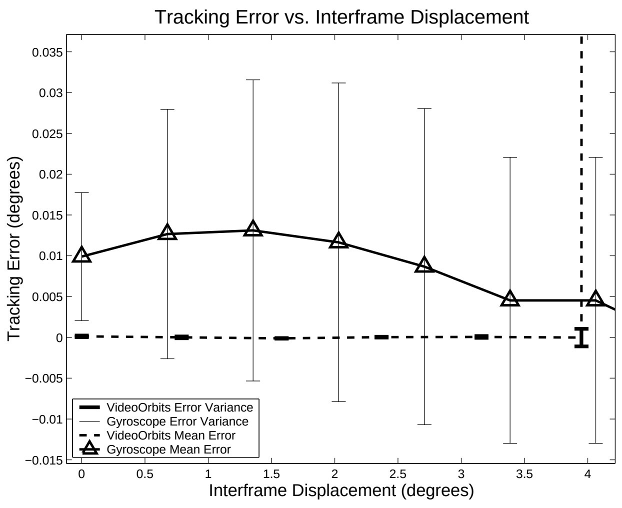


Figure 3: This graph compares the tracking ability of VideoOrbits and the gyroscope used. The error bars represent the variance in the tracking error and are displayed as one tenth the actual size. For this experiment, the execution time for VideoOrbits was limited to 0.3 s per frame. Under these circumstances we can see that if the interframe displacement is less than four degrees, the algorithm converges, resulting in very small error. Beyond four degrees, convergence is not achieved, resulting in unusable PCT estimates. While not shown, the error in the gyroscope remains fairly constant beyond four degrees of interframe displacement.

in the non–ideal case is improved by the extra five degrees of freedom provided by the 8 parameter PCT. The extended freedom allows for visually acceptable registration to be achieved in many cases where the where the pure rotation condition has been clearly violated. This is especially true in outdoor urban environments where many video sequences have dominant planer surfaces. Its effectiveness in these situations has been demonstrated in a mediated reality applications where large planar objects are replaced with other virtual content [5] for arbitrary camera movement.

Like other vision techniques, this method is computationally intensive. Moreover, complexity increases with interframe displacement, thus inhibiting its real–time implementation on a typical wearable computer, unless the camera motion is kept very small. Figure 3 shows the tracking error in VideoOrbits when the execution time of the algorithm is limited. We can see that if the interframe displacement is kept to less than about four degrees, VideoOrbits converges and the tracking error is in the sub–pixel range (with the camera used, 1 pixel $\approx$ 0.15 degrees). The convergence range is dependent on the scene as well as the computation time. The four degree range was found to be typical for an office environment with a 0.2 s VideoOrbits execution time. The four degree range can be dramatically improved by increasing the execution time as well as combining the method with a block matching scheme such as FFT phase correlation.

To make real–time tracking using VideoOrbits computationally feasible, this system uses a small gyroscope to aid VideoOrbits in finding interframe projective coordinate transformations (PCTs). This addition accelerates the algorithm and improves its ability to handle large interframe displacements due to rapid head movements. Figure 3 shows the gyroscope tracking error. With the gyroscope used alone, we can see in that in the zero to four degree range the error is much larger in comparison to VideoOrbits. Beyond four degrees, VideoOrbits does not converge giving extremely large errors while the error in the gyroscope stays fairly constant. By using the gyroscope to estimate the PCT between frames, VideoOrbits is provided with an initial guess at the interframe PCT that allows the algorithm to operate within its convergent range. Thus, the hybrid system is able to find the interframe PCT solutions with sub–pixel error even when there are very large displacements between images.

Although the gyroscope estimate reduces the necessary execution time of VideoOrbits, the algorithm cannot be run at more that a few frames per second on current wearable computers. To achieve higher frame rates, the gyroscope is used to estimate the camera motion between VideoOrbits solutions. This reduces the overall accuracy, but allows the frame rate to be increased to usable level.

The inertial tracking device used is a pair of small, low cost vibrating element gyroscopes made by Gyration Inc. The manufacturer claims a maximum accuracy of 0.15 deg/s with proper calibration and drift correction. Due to the electromechanical nature of the system, the calibration parameters are affected by temperature and power supply fluctuations and the gradual change in gyroscope characteristics over time. The proposed system aims to solve this problem through continuous closed loop calibration.

4 Gyroscopic/VideoOrbits Tracking System

The Gyroscope/VideoOrbits System (GVS) tracks the absolute position of a camera in terms of the projective coordinate transformation (PCT) between the current frame of video and a base frame. In terms of camera rotation, the base frame would exist at $(\theta, \psi, \phi) = (0, 0, 0)$, where θ, ψ and ϕ are the Euler angles describing the direction of the camera's optical axis. If the camera undergoes pure physical 3-D rotation about its optic center, the rotation can be recovered from the PCT provided the camera intrinsic parameters are known. Since in many scenarios, the user's head motion can be approximated very well by pure rotation, it makes it possible to estimate the rotational position of the camera with respect to its visual environment from its PCT position.

The GVS is a closed loop system in which a gyroscope provides an estimate of camera motion to allow VideoOrbits to determine the PCT between video frames even in the presence of large interframe camera movements. The high

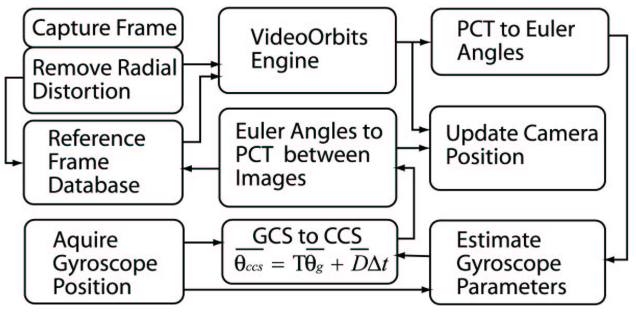


Figure 4: Gyro / VideoOrbits Tracker Block Diagram

accuracy PCTs produced by VideoOrbits are used to build a linear model for the gyroscope in order to correct for gyro drift. A block diagram of the system is shown in Figure 4.

Absolute tracking with respect to the visual environment is measured by composing the relative PCTs between video frames. To eliminate the effects of cumulative error a spherical map of the environment is continuously generated as the user wears the device. The continuous regeneration of this map allows the system to maintain good visual registration even when head motion is not limited to pure rotation. Under these circumstances, the achieved accuracy depends on the amount of translational camera motion and the distance to the objects in the user's field of view.

Video Capture and Undistortion

Digital video is provided by an inexpensive IEEE1394 camera. The IEEE1394 format allows programmatic control of exposure settings for most cameras, which is very desirable since VideoOrbits uses the brightness constancy constraint. Image brightness information is obtained by selecting the luminance component of the YUV color space. Camera barrel distortion was removed from the camera images with an optimized undistortion function in the Intel Open Source Computer Vision Library (OpenCV)[2].

Acquire and process Gyroscope signals

A PIC micro-controller was used to digitize the analog rate outputs from the gyroscope and to compute the interframe integration. Rotation information is provided on demand to the wearable computer over an RS232 communication port. The integrated gyroscope output is transformed by matrix $\mathbf{T}$ to align the gyroscope axis with the camera coordinate system (CCS) and to scale the gyroscope output to match the units used in the system. While gyro drift is inevitable, it remains relatively constant and can be subtracted from the rate output fairly well.

[2]OpenCV is available from http://sourceforge.net/projects/opencvlibrary/

$$\begin{bmatrix} \theta \\ \psi \\ \phi \end{bmatrix}_{CCS} = [\mathbf{T}] \begin{bmatrix} \theta_g \\ \psi_g \\ \phi_g \end{bmatrix} - \begin{bmatrix} d_\theta \\ d_\psi \\ d_\phi \end{bmatrix} \Delta t \qquad (2)$$

where $[\theta\psi\phi]^T$ are the gyroscope rotation angles in the camera coordinate system. $[\theta_g\psi_g\phi_g]^T$ are the raw signals received from the PIC. $[d_\theta d_\psi d_\phi]^T$ are the drift rates in the camera coordinate system.

Unfortunately as with all analog systems, component values drift with temperature, and load changes can affect the power system levels. This tends to cause significant short term variations in the required drift parameters. Thus, the system drift must be continuously re–estimated while the system is operating. The transformation $\mathbf{T}$ changes little so long as the camera and gyroscope are mounted rigidly together, and thus can be computed offline. This calibration is easily accomplished by allowing the system to run in a situation where the vision system performs very well for a few minutes and then using a least squares method to find the transformation parameters from the collected VideoOrbits and gyroscope data. The best situation is a well lit static environment observed with smooth head motion.

Calculate Equivalent Projective Transformation from Euler angles

In order for the gyroscope to provide an initial transformation estimate for VideoOrbits, the interframe rotation experienced by the gyroscope must be converted into a PCT. To calculate this PCT, a rotation matrix is formed using the relative Euler Angles returned by the gyroscope. This matrix is composed with a fixed 3-D rotation matrix to align the gyroscope's coordinate system with the camera coordinate system. VideoOrbits operates on images that have been scaled such that the upper left corner of the image is $(0, 0)$ and the bottom right corner is $(1, 1)$. To create a PCT from the obtained relative rotation matrix an appropriate change of coordinates is applied:

$$\mathbf{WRW}^{-1} = \lambda \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{c}^T & 1 \end{bmatrix} = \lambda P_g \qquad (3)$$

$$\text{where } \mathbf{W} = \begin{bmatrix} \frac{f_x}{S_x} & 0 & \frac{O_x}{S_x} \\ 0 & \frac{f_y}{S_y} & \frac{O_y}{S_y} \\ 0 & 0 & 1 \end{bmatrix}, \qquad (4)$$

$\mathbf{R}$ is the rotation matrix corresponding to the gyroscope rotations in the camera coordinate system, f_x and f_y are the camera focal lengths in the horizontal and vertical directions, S_x and S_y are the dimensions of the video images in the horizontal and vertical directions, (O_x, O_y) is the point of intersection of the optical axis of the camera and the sensor array (in units of pixels), and λ is a scale factor used to force the last entry in the PCT matrix to be one. $\mathbf{A}, \mathbf{b}$ and $\mathbf{c}$ are the PCT parameters defined in in Equation (1).

Reference Frame Database

This system tracks the absolute camera motion by finding the PCT (P_{abs}) relating the current frame of video to a base frame. One method of determining P_{abs} is to compose the relative PCTs, P_{n-1} and P_n, from temporally adjacent frames, numbered $n-1$ and n:

$$P_{abs} = \prod_{n=1}^{N} P_n \tag{5}$$

where P_{abs} and P_n are expressed in matrix form:

$$P = \left[\begin{array}{cc} \mathbf{A} & \mathbf{b} \\ \mathbf{c}^T & 1 \end{array} \right] \tag{6}$$

This method however is quite prone to drift since the errors in the relative PCTs have a cumulative effect on the absolute position. In this system the cumulative error is reduced by storing well registered images as reference frames. Absolute rotation is then calculated by composing the relative PCTs (P_n) relating the current video frame to a reference frame, with the PCT (P_r) relating the reference frame to the base frame. The resulting performance increase can be seen in Figure 5.

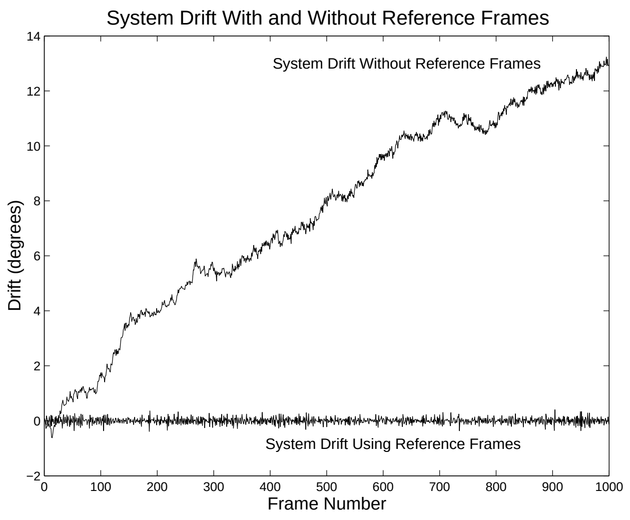


Figure 5: This graph shows the effect of reference frame use on system drift. For this experiment, the camera was held in a fixed position.

The Reference Frame Database is a spherically indexed database of reference images of the environment that are continuously collected and updated through normal system use. Each image in the database includes a time stamp, a PCT (P_r) describing its position with respect to the base frame, and a PCT certainty measure to allow for judicious reference frame selection. Reference frames are selected for use with VideoOrbits based on the current position estimate generated by the gyroscope.

It is important to note that if large head rotations occur, it is possible for consecutive image frames to have no overlap at all. This can prevent VideoOrbits from estimating a useful PCT in the case where the sphere of environment images is poorly populated and no nearby reference images exist. In this case, a new reference image will be stored with a gyro-generated PCT. New reference images with gyroscope generated PCTs can also be created if VideoOrbits cannot find good registration with an existing reference frame.

Images whose associated PCTs were generated by VideoOrbits are selected in preference to those whose positions were estimated by the gyroscope alone. More recent images are selected in preference to older images to account for environment changes. Each time a reference frame is used successfully, its timestamp is updated to ensure that it continues to be used instead of its neighbors.

Reference frames are saved when the current video image is offset from the current reference frame by a significant amount. This amount is dependent on the camera's field of view and the system resources of the operating platform. The sphere of reference images is mapped to a two dimensional matrix, indexed by equal increments of azimuth and elevation. In experimentation, four degree increments were used. Pinching at the poles of the sphere is eliminated by dividing the matrix into petals of valid image locations. The remaining entries of the matrix are marked as NULL, as they are redundant. Searching this matrix is simple since the camera position can be directly related to the position in this matrix. If the search includes NULL entries, their nearest neighbors (of greater azimuth and elevation) are used. Near the poles, searches tend to wrap to opposing sides of the location matrix. It is important to note that frames are saved only when the estimated camera velocity at the time of image capture is small enough to produce acceptable amounts of image blurring.

Estimate PCT

The gyroscope estimated PCT is used to aid VideoOrbits in registering the current video frame to a reference frame. Since the PCT produced directly by the gyroscope measurement is relative to the last frame of video, it must be transformed to be relative to the reference frame that VideoOrbits will use: If P_g is the gyroscope estimated PCT, P_l is the PCT describing the position of the last frame of video and P_r is the PCT describing the position of the reference frame, the estimated PCT between current frame of video and the reference frame (P_e) is given by:

$$P_e = \lambda P_r^{-1} P_l P_g \tag{7}$$

where P_e, P_r, P_l, P_g are PCTs in the matrix form. As before, λ is used to force the $(3,3)$ entry of P_e to be 1.

VideoOrbits Engine

VideoOrbits estimates the PCT (P_o) between the current frame of video and the reference frame by substituting $u_m = \frac{Ax+b}{c^T x+1} - x$ into the Horn and Schunk brightness constancy constraint equation (BCCE) [3]. This method is known as 'projective–flow' [4]. Because of effects such as parallax in the images due to motion other than rotation, independently moving objects in the scene, and camera blur, the PCT cannot fit perfectly at all points of the images. Thus the solution we seek is the best estimate of the PCT parameters in a least squares sense across all image points. This is solved by minimizing:

$$\begin{aligned} \varepsilon_{flow} &= \sum \left(\mathbf{u_m}^T \mathbf{E_x} + E_t \right)^2 & (8) \\ &= \sum \left((\frac{\mathbf{Ax}+\mathbf{b}}{\mathbf{c}^T\mathbf{x}+1} - \mathbf{x})^T \mathbf{E_x} + E_t \right)^2 & (9) \end{aligned}$$

where E_x is the 2-D spatial gradient of the image $[d_x, d_y]$ at some point (x, y), and E_t is the temporal gradient, representing the change in brightness at a particular pixel coordinate from one image to the next.

Minimizing (9) is simplified by weighting by $(\mathbf{c^T x} + 1)$, giving:

$$\varepsilon_w = \sum \left((\mathbf{Ax} + \mathbf{b} - (\mathbf{c}^T\mathbf{x} + 1)\mathbf{x})^T \mathbf{E_x} + (\mathbf{c}^T\mathbf{x} + 1) E_t \right)^2 \quad (10)$$

where ε_w denotes the weighted error.

To solve for the minimum we differentiate with respect to the free parameters $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{c}$, and set the result to zero to give a linear solution:

$$\left(\sum \phi\phi^T \right) [a_{11}, a_{12}, b_1, a_{21}, a_{22}, b_2, c_1, c_2]^T \quad (11)$$

where $\phi^T = [E_x(x, y, 1), E_y(x, y, 1), xE_t - x^2E_x - xyE_y, yE_t - xyE_x - y^2E_y]$.

Since only the first derivative of the spatial gradient is used in the BCCE, the modeled change in pixel brightness due to translational velocity is only accurate when the motion between successive images is small. The set of PCTs forms a group, allowing VideoOrbits to utilize the group law of composition in a multiscale iterative technique to find the optimal PCT between two images. The iterations proceed as follows:

(1) Estimate the projective coordinate transformation $\mathbf{P}$

(2) Use this $\mathbf{P}$ to transform the appropriate original image

(3) Estimate another PCT, $\mathbf{P_2}$, between the transformed image and the other original image.

(4) Generate an improved $\mathbf{P}$ by composing $\mathbf{P_2}$ with $\mathbf{P}$

(5) goto (2) until the desired accuracy has been achieved

This scheme allows the final iterations to estimate the PCT between two images that have very little motion between them, thus allowing the first order approximation of pixel brightness change to be sufficiently accurate.

VideoOrbits uses the PCT estimate P_g from the gyroscope as a starting point (step (1)) for the iterative scheme described. VideoOrbits runs for a prescribed number of iterations depending on the computational power of the wearable computer, and returns its best estimate of the PCT parameters. VideoOrbits also returns a Mean Squared Error (MSE) value for the returned PCT, which is used as a measure of registration accuracy. The MSE is calculated by adding the squared difference between the original image and its transformed temporal neighbor. Unfortunately the MSE measure is very sensitive to the nature of the images used, and thus sensitive to the user's local environment. For example, well registered images in a dim environment will have lower MSE values than in a bright environment. To cope with this, local statistics are kept on the MSE levels to provide intelligent adaptation to changing environments.

4.1 Using Graphics Hardware to Accelerate the VideoOrbits Algorithm

In VideoOrbits, the process of image warping (item 2 of the iterative technique) is very computationally intensive, and accurate filtering and interpolation techniques add to the computational complexity of the image warp. Our system is desired to run in real–time, and the time spent re–warping and filtering the images makes up a large portion of the calculations required on each frame. In order to speed–up the algorithm, our system uses the graphics acceleration hardware on board the wearable computer.

Many current graphics chipsets incorporate hardware specifically designed to achieve fast real–time rendering of texture mapped polygons as well as hardware designed for filtering and pixel interpolation to create accurate texture maps. In particular, graphics chipsets are tuned to create perspective projections of planar surfaces. Therefore by texture mapping the image to a planar surface and then applying the desired projection, the image will be projected by the computer graphics hardware rather than doing so in software.

In this way, the computer graphics hardware, which is usually used for image synthesis, is instead being used for the purpose of accelerating a computer vision algorithm (image analysis).

In OpenGL, the most straightforward way of applying the projective coordinate transformation of VideoOrbits is to consider equation 15 to be a transformation to be applied to the projection matrix used in OpenGL.

The operation of applying a projective coordinate transformation to an image is isomorphic (isomorphic refers precisely to algebraic isomorphisms, as discussed in [1]) to the process of projecting a texture mapped polygon under perspective projection in OpenGL [5]. Thus, hardware acceleration of VideoOrbits projective transformations can be achieved by defining an isomorphism between the projective space of VideoOrbits and the projective space and homogeneous coordinate system of OpenGL. An isomorphism ϕ is defined by a mapping of VideoOrbits projective transformations G to OpenGL projection matrices M:

$$\phi : G \to M \tag{12}$$

In the VideoOrbits algorithm, the projective coordinate transformation is written as equation 1. Thus, it defines an eight parameter space. The transformation can be re-written as a $R_{3\times3}$ matrix:

$$\begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = G\mathbf{x} \tag{13}$$

where it can be seen that the set of projective coordinate transformation forms a group acting upon a set S of image coordinates.

Thus, what is desired is some isomorphism ϕ mapping the projective coordinate transformation of VideoOrbits to a $R_{4\times4}$ projection matrix in OpenGL.

The desired isomorphism is given by:

$$\phi(G) = \phi\left(\begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix}\right) \tag{14}$$

$$= \begin{bmatrix} 1/a_{22} & -a_{21} & b_2 & 0 \\ -a_{12} & 1/a_{11} & b_1 & 0 \\ c_2 & c_1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{15}$$

This mapping takes into account the different coordinate systems and conventions used by each program. Equation 15 is used as a camera transformation matrix. Thus, it describes the transformation the camera undergoes, such that the plane will appear as required under OpenGL perspective projection.

To perform the image projection in OpenGL, the image is first loaded into the OpenGL program as a texture map. Then the camera is positioned and the perspective projection is applied. The resulting image is read out of the buffer and the image is stored.

To determine the speed–up attained by using hardware acceleration, a program using the hardware acceleration was compared with the software algorithm. A set of projective coordinate transformations was generated according to the equations of [9]. All programs were run on a wearable computer with an Intel i810 graphics chipset, and a 700 MHz Pentium-III processor.

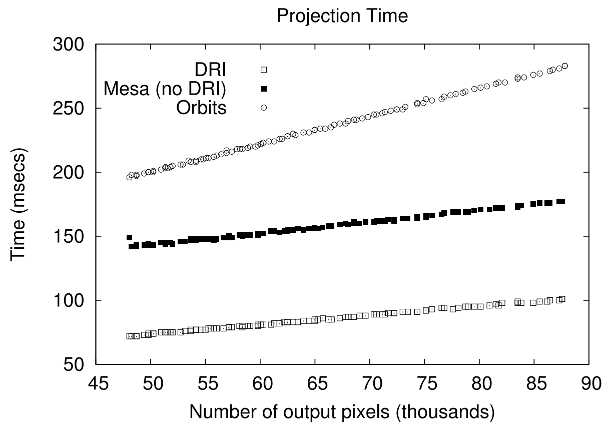


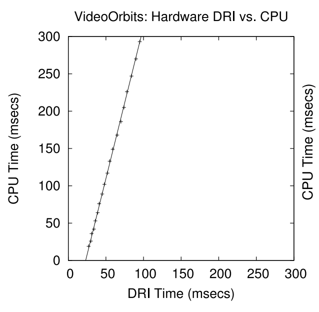
Figure 6: Projection times for VideoOrbits software, Mesa3D (using software rendering) and Mesa3D with DRI hardware rendering enabled.

Figure 6 shows the results of timing a single projection using three methods: 1) a program using Mesa3D and available computer graphics hardware and DRI, 2) a program using Mesa3D, using software algorithms, and 3) a program not using an optimized 3D library.

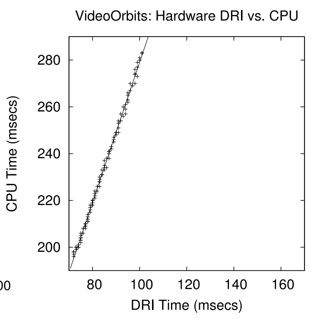
Thus, an additional program is discussed here, which uses the software implementation of Mesa3D (actually the Mesa3D program is the same as the DRI program, with the direct rendering turned off). The software Mesa3D was examined because it is considered to be well optimized code for computer graphics applications. Thus, computer vision algorithms can also benefit from the speed and optimizations used in computer graphics software. So on machines which may not benefit from 3D graphics acceleration, Mesa will still implement an optimized software projection, and additionally this was examined.

For the plot of figure 6, the input image size was set, and different projection parameters were given to the three different programs, and the time taken to project the image was recorded. The projections used were independent rotations about each of the principle axes, with a maximum rotation of 15 degrees about any axis. From the data, the average speedup between VideoOrbits using DRI vs. using the CPU was $2.75\times$. Mesa software rendering was $1.48\times$ faster than not using an optimized 3D library and the DRI implementation improved the performance by another $1.83\times$.

Figure 7 (a) shows the effect of hardware acceleration on input images of different sizes. For this figure, the projection parameters were held constant, and the input image size was varied. In all cases, the hardware accelerated program projected the image faster than one using only the CPU. The smallest image size was 76×58 and the largest input image size was 435×331. The slope of a linear best fit line through the plot is 4.07. Thus, for this range of input image



(a)



(b)

Figure 7: (a) Projection times for VideoOrbits using DRI vs. the CPU given varying image sizes. All image had identical projection parameters. (b) Projection times for VideoOrbits programs, one using DRI and the other the CPU. The algorithm was given a fixed input image and the projection parameters were varied (resulting in larger output images).

sizes, the hardware speedup was $4.07\times$.

Figure 7 (b) shows the effect of different projection parameters on the speed of the projection. For this plot, the input image was held constant, but the projection parameters were varied. The larger projection times shown correspond with increasing numbers of output pixels of the resulting image. Thus, this plot is measuring the effects of increased amounts of pixel interpolation, since large output images required more pixel interpolation since there were more output pixels. The slope of this graph was 2.99. Slope here may be interpreted as how well each of the programs using DRI and the CPU dealt with more interpolation being required. Thus, the graphics hardware was able to handle increased amounts of interpolation $2.99\times$ faster than the CPU.

Estimate Euler Angles from the PCT

In order to close the loop and allow VideoOrbits to correct the changing gyroscope drift parameters, an equivalent 3-axis rotation must be obtained from the PCT. It is known that the PCT has eight free scalar parameters and is thus not limited to pure rotation, so it is necessary to find the best fit 3-D rotation matrix and have some indication of the error to determine if a rotation is valid and can be used in the statistical model for the gyroscope. To find the required rotations, a PCT (P_{rel}) describing the relative rotation since the last successful execution of VideoOrbits is found: $P_{rel} = P_r^{-1} P_{n-1} P_o$, where P_r, P_{n-1}, and P_n are the PCTs relating the reference, previous, and current video frames to the base frame. This relative PCT is then approximated by a pure rotation matrix using a singular value decomposition (SVD) of P_{rel}. The SVD of P_{rel} is given by:

$$P_{rel} = USV^T \tag{16}$$

where S is a diagonal matrix of singular values. The best fit rotation matrix R in terms of the Frobenius norm between itself and P_{rel} is given by:

$$R = UV^T \tag{17}$$

This rotation matrix is then used to compute the corresponding Euler angles $[\theta_o \psi_o \phi_o]^T$. This computation is not difficult, but is lengthy and thus will not be presented here.

The Frobenius norm is calculated as follows, to give an error measure in the rotation matrix:

$$||R - P_{rel}||_F^2 = Tr((R - P_{rel})^T (R - P_{rel})) \tag{18}$$

This measure is used to help identify non-convergent PCT estimates from VideoOrbits and select Euler angle estimates from P_{rel} that are suitable for use in gyroscope drift correction.

Estimate $[d_\theta d_\psi d_\phi]^T$

In order to estimate the gyroscope drift, a buffer of recent data points correlating the gyroscope Euler angles $[\theta_g \psi_g \phi_g]^T$ to the VideoOrbits Euler angles $[\theta_o \psi_o \phi_o]^T$ is maintained. Data points associated with poor MSE with respect to the local statistics or with high error in the rotation matrix approximation of P_{rel} are rejected.

These data points are used to create a linear model, using linear regression, that relates the gyroscope measured rotation to the rotation estimated by VideoOrbits. For this model, the standard error in the slope and the intercept, as well as the correlation coefficient are calculated. These statistics are used to decide on corrective action for the gyroscope drift. The stringency of the decision tolerances increases as gyroscope calibration improves to ensure convergence to an optimal solution. To allow re–calibration in the event of a large shift in parameter values during operation, the decision tolerances are relaxed if a good linear fit is found with significantly different parameters. Large shifts in parameter values were found to occur when the system was exposed to large temperature changes such as when moving from an indoor to an outdoor environment.

5 Results: System Evaluation

The system presented has shown the ability to correct for variations in gyroscope drift characteristics through normal wear. The length of time required for calibration depends on the user's motions, since VideoOrbits needs to achieve good registration between frames to contribute meaningful data to the gyro model. Figure 8 demonstrates the success of the closed-loop correction.

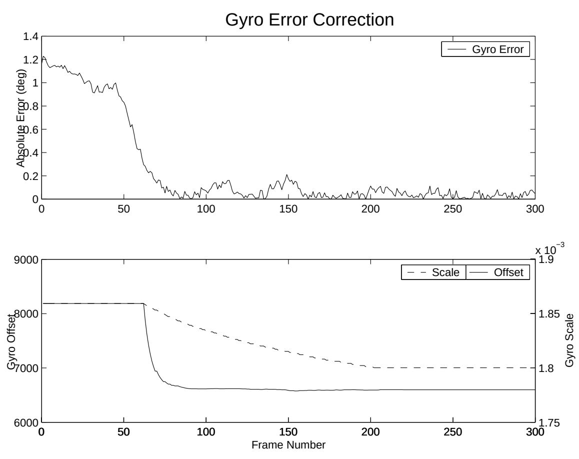


Figure 8: The above data were taken in a scenario where power supply voltage to the gyro had been changed (at frame 0) to simulate power level fluctuations in a battery–driven system. At around frame 65, the error in the gyro model dropped below a threshold value and the system started to correct the calibration parameters. We can see by the reduction of the absolute error that the system has converged to a more desirable state.

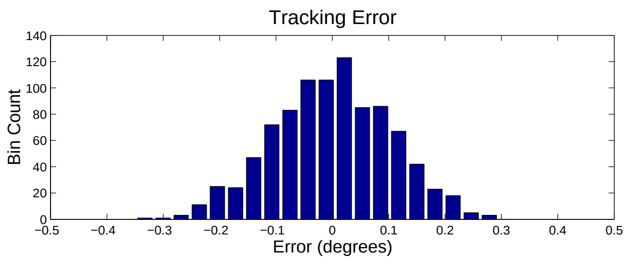


Figure 9: The histogram shows the distribution error in real–time image registration with a well populated reference frame database.

5.1 Tracking Performance

Operating on a 1.2 GHz Pentium III with an i810 graphics chipset (with 512 MByte RAM and running Linux 2.4.18), the system was able to register images with less than 2 pixels of error (0.3 degrees) at 12 frames per second. Figure 9 shows the distribution of registration errors with the system running in a static environment with a well populated reference frame database. These results apply for conditions where the camera motion can be approximated well by 3DOF rotations. The absolute tracking performance of the system however, suffers greatly when it is exposed to large accelerations such as running or jumping. It is also sensitive to low light conditions which cause blurry images through low shutter speeds.

Figure 10 shows the tracking of a point in the scene with significant head motion. This figure also shows how the tracking algorithm is robust in the presence of independently moving objects. VideoOrbits alone is subject to very large errors when an independently moving object is introduced. However, with the gyroscope in the system, the PCT parameters produced by VideoOrbits are rejected due to high associated MSE values.

Figure 11 illustrates real–time registration results. In this example the frame rate has been reduced to 3 frames per second, to produce large interframe displacements and show the system's ability to cope with large head motions.

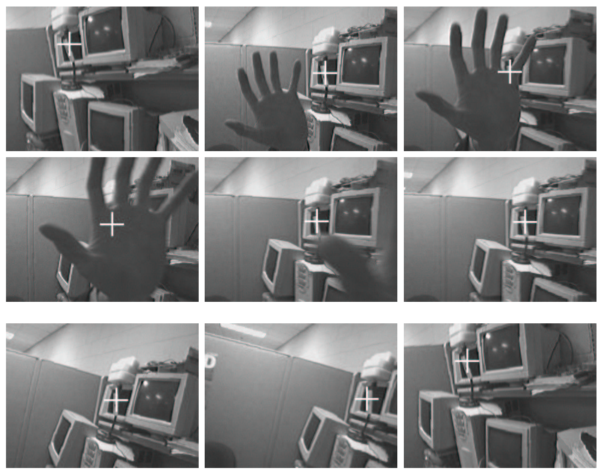


Figure 10: The following images show the GVS tracking a point with significant head rotation in a near environment. The images containing the hand, demonstrate the system's ability to track even when blocked by a large independently moving object.

6. Seeing eye to eye: an application of projective head tracking

Wearable mediated reality allows for new interactions between people. This application of projective head tracking allows users to see the world through each other's eyes. Each user in this system wears an EyeTap reality mediator equipped with the GVS tracking system described in this paper. As a user looks around their environment, the GVS generates an environment map of their surroundings. This environment map is then shared between users. When a user views another's environment, the reality mediator presents it as though it were the user's current environment. By using head–tracking, users can naturally browse another's environment by simply looking around. Users continue to build their environment map even as they browse another's environment, and thus, continually add information to be shared with others. This reciprocity allowed the creation of visual interactions between users that share each other's reality. In this interaction users see "eye to eye", to create a shared mediated reality.

EyeTap devices have been used before in the transmission of live video captured from the user perspective of the surrounding environment. In a previous study, EyeTap re-

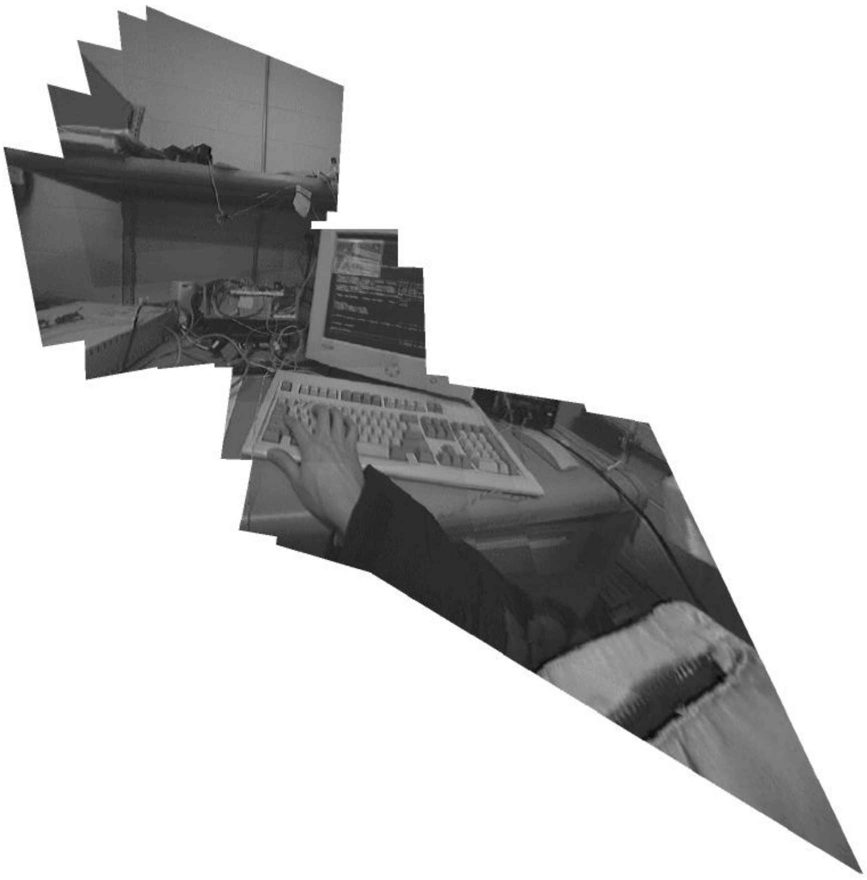


Figure 11: Real–time Image registration in a close environment. In this instance the frame rate has been reduced to 3 frames per second, to produce extra large interframe displacements and put the gyroscope to the test.

porters accomplish wireless Electronic News Gathering[7] that allow viewers to remotely experience events as if they were actually present.

In contrast, the system that we now describe, allows users to see through each other's eyes, yet without the constraint of following exactly the broadcaster's point of view. This development is extremely important, since exactly following another person's shaky EyeTap video can be disorienting and nauseating. Observation of live eye or head mounted video is generally much less stable when viewed vicariously than what we seem to experience in reality since our perceptual system does much of the stabilization based on our physical motions, and vestibular cues.

The requirement for the shared mediated reality system to store, retrieve and spatially register images is satisfied by making use of the spherical reference frame database that is maintained by the GVS for head tracking purposes. The reference frame images are stored along with the rotational orientation of the camera R_r, projective parameters P_r, and indexed by integer values (θ, ϕ), representing the azimuth and elevation of the camera orientation.

Sharing of the mediated reality is accomplished by transferring this reference frame database to wearer B, who can use a second GVS and a browser program to synthesize views of wearer A's environment map from an arbitrary viewing direction. As long as the environment map contains some reference images in the neighborhood of the precise direction desired, the browser program will display a correctly projected view.

Synthesis of views is performed by taking the current estimate of the rotational position (R) of wearer B's GVS and using its equivalent PCT (P) to re–project wearer A's reference images to wearer B's viewpoint. In order to increase the creation speed of the synthesized view, a bounding box is computed and only the reference images that will contribute to the final view are re–projected. In order to minimize composition error due to user translation and independently moving objects, the images are composed in chronological order such that the most recent images are placed on top.

Thus, the head tracking of wearer B is used to provide a perspective with which to view the environment map generated by wearer A. As wearer B rotates his/her head, new perspectives are supplied to the browser program, which

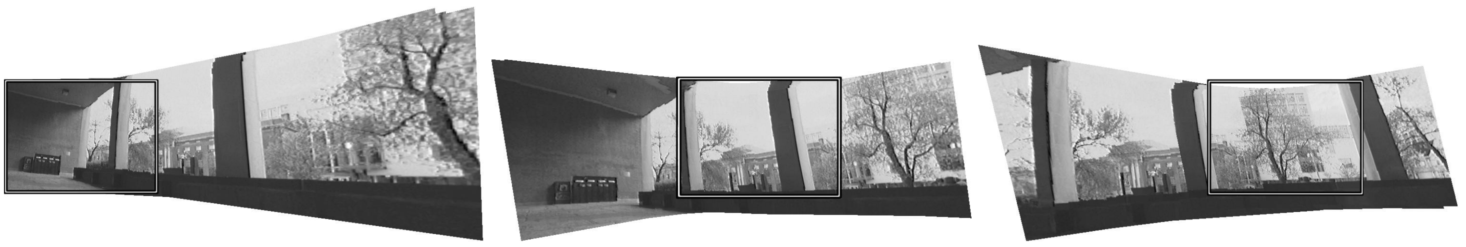


Figure 13: Outdoors: the environment map generated by wearer A, and the views of that map synthesized by wearer B.

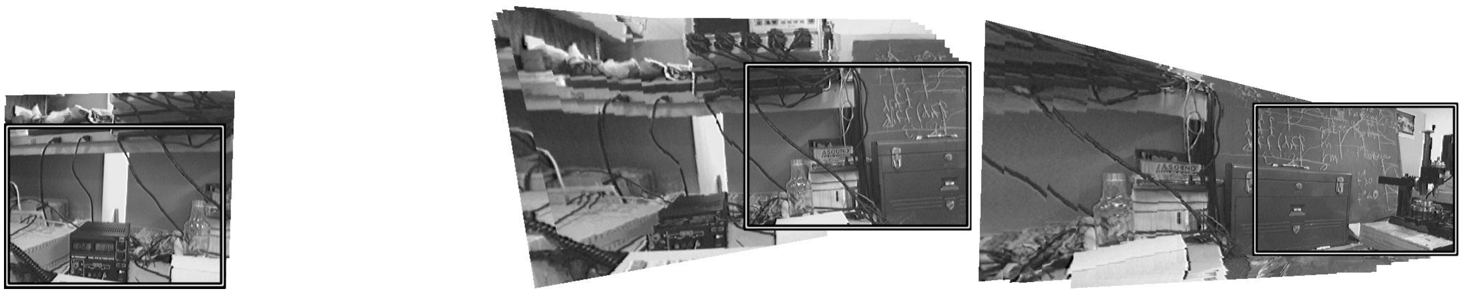


Figure 14: Indoors: the environment map that wearer B generates while browsing wearer A's outdoor map.

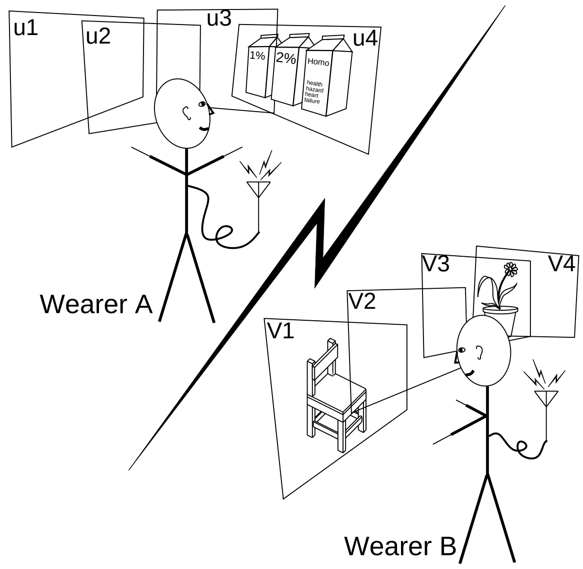


Figure 12: The shared mediated reality system allows users wearing EyeTap devices to share their visual experience. As each user looks around, they create a visual/temporal map of their surrounding environment. By sharing this visual history, users can see through the eyes of another. They are able to look about without being constrained to a single point of view, rather being able to generate new projective views from the visual/temporal environment maps. Wearer A, in a grocery store, is building an environment map $(u1, u2, u3, u4)$ that includes an image of milk cartons. Wearer B is browsing wearer A's environment map at home. The images $(u1, u2, u3, u4)$ are displayed in the positions $(v1, v2, v3, v4)$, replacing wearer B's view of the chair and plant. At the same time, wearer B is incidentally constructing an environment map of the home, which would allow wearer A to view the chair and plant from Wearer B's position.

cause the projection and synthesis of new portions of the environment map. All these projections are relative to wearer B's current head position, thus all navigation is performed solely with head movement.

The the number of images selected within the bounding box to contribute to the final image composite that wearer B would see, is adjusted to fit the available computational resources. With a powerful computer, multiple images from the map can be projectively transformed and registered (see figures 13 and 14). This allows for the creation of images that are spatially complete for the desired perspective even if there is no images in the map for that exact viewing perspective. It also ensures that the most recent images are used, giving the most up to date estimate of the other user's environment. On a resource constrained wearable computing system, Only a few images may be used, so the user must decide on the relative importance of spatial extent and temporal accuracy.

6.1 Accounting for the Camera Intrinsic Parameters

The EyeTap devices are custom made prototypes and as such contain different cameras. Because of this, for one user to view the other's environment as if they were seeing it through their own EyeTap, the intrinsic parameters of both cameras must be accounted for. In the case where wearer B is viewing wearer A's environment, the appropriate projection to transform the images in wearer A's environment map to wearer B's viewpoint is:

$$P_{view} = R_B^{-1} M_{int_B} M_{int_A}^{-1} P_n, \tag{19}$$

where R_B is the rotation matrix dscribing the viewing orientation of wearer B. P_n is the PCT associated an image in wearer A's environment map describing its visual location. M_{int_B} and M_{int_A} are the matricies of intrinsic parameters for the EyeTap cameras of wearer A and wearer B respectively. The definition of the intrinsic parameter matrix is given below:

$$M_{int} = \begin{bmatrix} -f_x & 0 & O_x \\ 0 & -f_y & O_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{20}$$

where f_x, f_y are the focal length and O_x, O_y is the location of the image center in pixel coordinates.

6.2 Demonstration

The system may be seen in action in figures 13 and 14, demonstrating results of wearer B browsing an environment map created by wearer A. The first image in figure 13 shows the environment map that wearer A has generated out of doors on a veranda. The bounding box indicates the portion that wearer B is currently viewing. The first image of figure 14 shows the corresponding view of B's environment inside a laboratory, with the bounding box described by the EyeTap. The middle images show the result of looking to the right: in figure 13 the view has moved to the middle of the environment map, while in figure 14, we see that wearer B is simultaneously mapping out their own environment. The final pair of images carries this process further, and also reveals how the system handles the case of insufficient reference frames to fill a view completely: there is a narrow band of white at the top of the bounding box in figure 13 indicating an area with insufficient information in the environment map.

7 Conclusion

The hybrid Gyro / VideoOrbits System (GVS) operates in real–time, with reasonable frame rates on currently available wearable computers. GVS uses an image registration technique based on projective coordinate transformations as well as a gyroscope, and is suitable for operation in unprepared environments. The computer graphics hardware on board the wearable computer was used to provide hardware acceleration for the image registration algorithm using OpenGL. The system achieves real–time head tracking with sufficient accuracy to be used in many augmented and mediated reality applications where the motion of an eyetap device or head–mounted camera can be approximated by 3DOF rotation.

A shared mediated reality application that utilized the GVS was presented that allowed users to exchange the visual environments viewed through their EyeTap reality mediators. By utilizing both the projective and 3DOF rotation tracking capabilities of the system to stabilize the head mounted video from one wearable computer user to the viewing orientation of another, the tracking capabilities of the GVS were demonstrated in a very tangible way.

References

[1] M. Artin. *Algebra*. Prentice Hall, 1995.

[2] G. W. et al. High-performance wide-area optical tracking–the hiball tracking system. *Presence: Teleoperators and Virtual Environments*, 10(1):1–21, February 2001.

[3] B. Horn and B. Schunk. Determining Optical Flow. *Artificial Intelligence*, 17:185–203, 1981.

[4] S. Mann. Humanistic intelligence/humanistic computing: 'wearcomp' as a new framework for intelligent signal processing. *Proceedings of the IEEE*, 86(11):2123–2151+cover, Nov 1998. http://wearcam.org/procieee.htm.

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

[6] S. Mann and J. Fung. Eye tap devices for augmented, deliberately diminished or otherwise altered visual perception of rigid planar patches of real world scenes. *To appear in PRESENCE: Teleoperators and Virtual Environments*.

[7] S. Mann, J. Fung, and E. Moncrieff. Eyetap technology for wireless electronic news gathering. *Mobile Computing and Communications Review*, 3(4):19–26, 1999.

[8] K. Satoh et al. Townwear: An outdoor wearable mr system with high precision registration. In *Proceedings of Internation Symposium for Mixed Reality (ISMR2001)*, pages 210–211, 2001.

[9] R. Y. Tsai and T. S. Huang. Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch I. *IEEE Trans. Accoust., Speech, and Sig. Proc.*, ASSP(29):1147–1152, December 1981.

[10] Y. Yokokohji, Y. Sugawara, and T. Yoshikawa. Accurate image overlay on video see-through hmds using vision and accelerometers. In *Proceedings of IEEE Virtual Reality 2000*, pages 247–254, 2000.

[11] S. You and U. Neumann. Fusion of vision and gyro tracking for robust augmented reality registration. In *VR*, pages 71–78, 2001.

[12] S. You, U. Neumann, and R. Azuma. Hybrid inertial and vision tracking for augmented reality registration. In *Proceedings of IEEE Virtual Reality 1999*, pages 260–267, 1999.