

Seeing Eye to Eye: a shared mediated reality using EyeTap devices and the VideoOrbits Gyroscopic Head Tracker

Abstract

We present a system which allows wearable computer users to share their views of their current environments with each other. Our system uses an EyeTap: a device which allows the eye of the wearer to function both as a camera and a display. A wearer, by looking around his/her environment, “paints” or “builds” an environment map composed of images from the EyeTap device, along with head-tracking information recording the orientation of each image. The head-tracking algorithm uses a featureless image motion estimation algorithm coupled with a head mounted gyroscope. The environment map is then transmitted to another user, who, through their own head-tracking EyeTap system, browses the first user’s environment solely by head motion, seeing the environment as though it were their own. As a result of browsing the transmitted environment map, the viewer builds and extends his/her own environment map, and thus this is a data-producing head-tracking system. These environment maps can then be shared reciprocally between wearers.

1. Introduction

Wearable mediated reality allows for new interactions between people. In this paper, we propose a mediated reality system that allows users to see the world through each other’s eyes. Each user in this system wears a reality mediator, which augments, diminishes, or otherwise alters their visual perception of their environment. As a user looks around their environment, the reality mediator 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

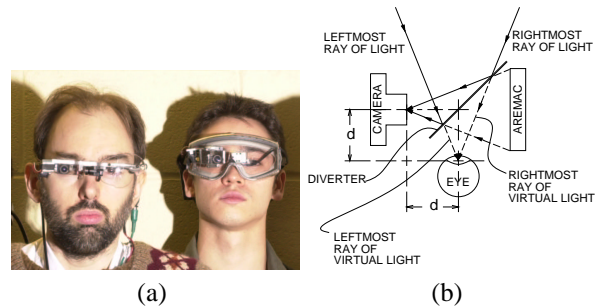


Figure 1: (a) Two examples of EyeTap devices that tap the right eye. Note that both eyes can be tapped if stereo computer mediated reality is desired. (b) EyeTap schematic.

interactions between users that share each other’s reality. In this interaction users see “eye to eye”, to create a shared mediated reality.

2. EyeTap

Our reality mediators incorporate EyeTap devices. The EyeTap (see Figure 1) 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 orthospacially [5] resynthesize 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.)

EyeTap devices have been used for the transmission of live video captured from the user perspective of the surrounding environment. In a previous study, EyeTap reporters accomplish wireless Electronic News Gathering[7] that allow viewers to remotely experience events as if they

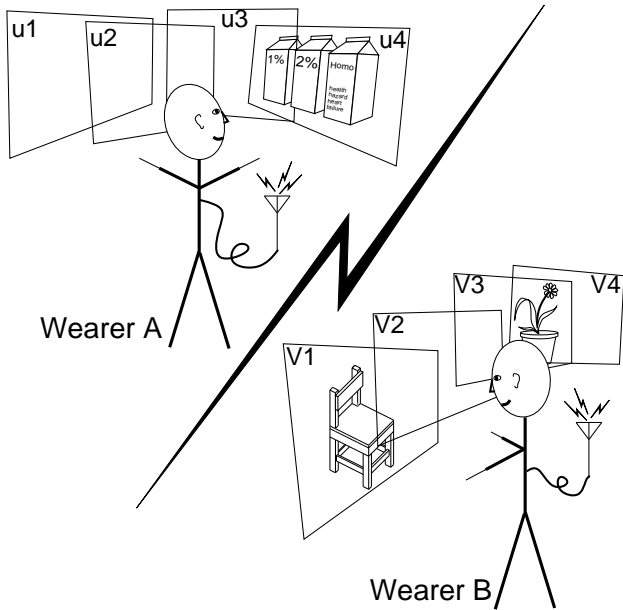


Figure 2: EyeTap wearer A, EyeTap wearer B

were actually present.

In contrast, the system that we now propose, similarly 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 gaze of shaky EyeTap video can be disorienting and nauseating. The way we scan our environment is very strongly tied to the physical motion that we are currently experiencing, together with the vestibular cues. 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.

3. Shared Mediated Reality

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. In Figure 2, Wearer A, in a grocery store, is building an environment map (u_1, u_2, u_3, u_4) that includes an image of milk cartons. Wearer B is browsing wearer A's environment map at home. The images (u_1, u_2, u_3, u_4) are displayed in the positions (v_1, v_2, v_3, v_4), 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.

People often scan their environment with quick head rotations. As a result, a large percentage of EyeTap video can be described (with a high degree of accuracy) by a camera moving about a fixed center of projection in a static scene. Using this simplification and applying the VideoOrbits algorithm to projectively transform the images, images from arbitrary camera orientations can be synthesized by forming a composite of spatially relevant neighbouring images. The vicariously experienced environment can thus be observed from any angle, allowing the user to navigate the virtual environment by rotating their head as if they were actually there.

3.1 VideoOrbits

Projective Coordinate Transformations (Equation (1)) (PCTs) [10], when applied across an entire image, can describe exactly the relation 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.

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

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

Since the motion of a head mounted camera or EyeTap between successive frames of video can be approximated very well by a pure rotation, a projective coordinate transformation can be used to register a pair of overlapping images with sub-pixel accuracy [5]. Well registered images can then be combined to create image composites of greater spatial extent. By applying warping this composite image with an appropriate PCT and cropping the result, arbitrary camera orientations can be simulated, thus allowing a user to view another's environment without being limited to previously held viewpoints.

While this technique is exact for situations where each user views their world through pure rotation, performance 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 composites to be formed in many cases 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.

The VideoOrbits algorithm estimates the projective coordinate transform (P_o) between the two overlapping images 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:

$$\varepsilon_{flow} = \sum (\mathbf{u}_m^T \mathbf{E}_x + E_t)^2 \quad (2)$$

$$= \sum \left(\left(\frac{\mathbf{A}\mathbf{x} + \mathbf{b}}{\mathbf{c}^T \mathbf{x} + 1} - \mathbf{x} \right)^T \mathbf{E}_x + E_t \right)^2 \quad (3)$$

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 (3) is simplified by weighting by $(\mathbf{c}\mathbf{x} + 1)$, giving:

$$\varepsilon_w = \sum \left((\mathbf{A}\mathbf{x} + \mathbf{b} - (\mathbf{c}^T \mathbf{x} + 1)\mathbf{x})^T \mathbf{E}_x + (\mathbf{c}^T \mathbf{x} + 1)E_t \right)^2 \quad (4)$$

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:

$$\left(\sum \phi \phi^T \right) [a_{11}, a_{12}, b_1, a_{21}, a_{22}, b_2, c_1, c_2]^T \quad (5)$$

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 image 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 \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.

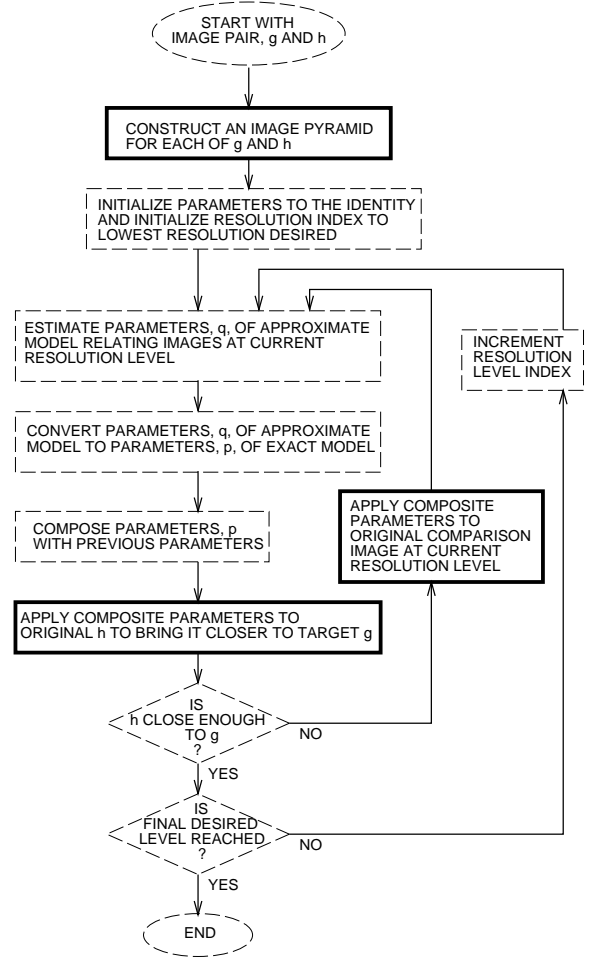


Figure 3: The VideoOrbits algorithm. The steps shown in solid dark lines have been accelerated using graphics hardware available on common 3D accelerated computer graphics chipsets. See section 5.

4 Head Tracking

In the proposed shared mediated reality system, a user may navigate another’s environment by rotating their head to achieve a desired point of view. The user’s head motions are approximated by pure rotations, since the shared environment map does not easily provide depth information. The images that comprise the environment map are stored with the associated rotational position that defines the orientation in which they were captured. This method of browsing the environment map provides the experience of seeing through the other user’s eyes.

To provide the necessary information on rotational orientation, both for browsing and environment creation, this system proposes an innovative head tracking system that combines VideoOrbits [5] with two small, low cost, vibrating element gyroscopic sensors.

By estimating the PCT between consecutive frames of

video, VideoOrbits is able to track the projective motion of the camera. Using this method, VideoOrbits has been applied to create a wearable, tetherless mediated reality [6]. If this method is used on a sequence, where the camera was only free to rotate about its optic center, the motion of the camera can be recovered exactly.

The proposed shared reality system requires the PCT between images as well as rotational orientation of the camera during their capture. As a result, when the camera is not limited to pure rotation, VideoOrbits alone cannot provide necessary information. Also, the feasibility of using VideoOrbits as a tracking device depends on the algorithm running in real-time with sufficiently high frame rates. Unfortunately, VideoOrbits is very computationally intensive. Moreover, complexity increases with interframe displacement, thus inhibiting its implementation a tracking method on a typical wearable computer unless the camera motion is kept very small. Figure 4 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 four degrees, VideoOrbits converges and the tracking error is in the sub-pixel range (with the camera used, 1 pixel \approx 0.15 degrees).

To make real-time tracking using VideoOrbits computationally feasible, the proposed system uses a small gyroscope to aid VideoOrbits in finding interframe projective coordinate transformations (PCTs), accelerating the algorithm and improving its ability to handle large interframe displacements due to rapid head movements. Since the gyroscope provides only rotation information, it can also be used to estimate, the rotational position of the EyeTap camera when its motion with respect to the visual environment cannot be approximated by a pure rotation. Figure 4 shows the gyroscope tracking error. With the gyroscope used alone, we can see 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 able to converge to a PCT solution giving sub-pixel tracking errors 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 than 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 it allows the frame rate to be increased to usable level.

The inertial tracking device used is a pair of small, low cost vibrating element gyroscope made by Gyration Inc. The manufacturer claims a maximum accuracy of 0.15 deg/s with proper calibration and drift correction. Due to

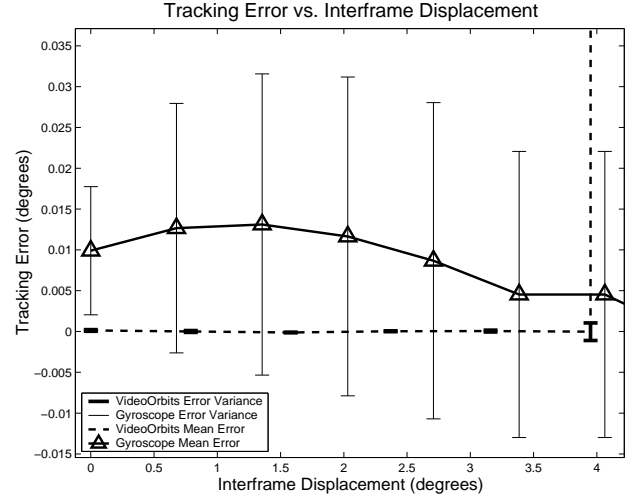


Figure 4: This graph compares the tracking ability of VideoOrbits and the gyroscope used. The error bars represent the variance in the tracking error and are displayed as one tenth the actual size. For this experiment, the execution time for VideoOrbits was limited to 0.3 s per frame. Under these circumstances 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.

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.

4.1 VideoOrbits/Gyroscopic Head Tracking (VOGHT)

The VideoOrbits/Gyroscopic Head Tracking system (VOGHT) tracks absolute position of a camera in terms of the projective coordinate transformation (PCT) between the current frame of video and a base frame. It also estimates the camera's absolute 3-D rotational position. 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. In the proposed shared mediated reality system, the PCT based camera position is used for image registration and the creation of composite views. The rotational position is used to orient the camera views in an environment map as well as for browsing another user's environment.

The VOGHT 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 accuracy PCTs produced by VideoOrbits are used to build a linear model for the gyroscope in order to correct for gyro drift and scaling problems. A block diagram of the system is shown in Figure 5.

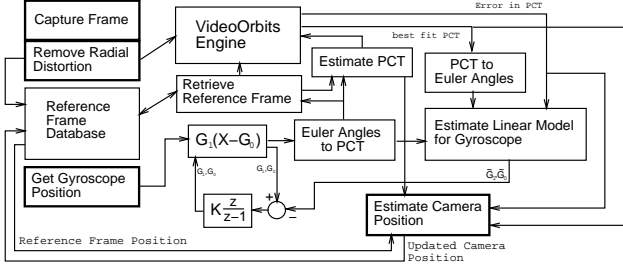


Figure 5: Gyro / VideoOrbits Tracker Block Diagram

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.

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:

$$\mathbf{WRW}^{-1} = \lambda \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{c}^T & 1 \end{bmatrix} = \lambda P_g \quad (6)$$

$$\text{where } \mathbf{W} = \begin{bmatrix} \frac{f_x}{S_x} & 0 & \frac{O_x}{S_x} \\ 0 & \frac{f_y}{S_y} & \frac{O_y}{S_y} \\ 0 & 0 & 1 \end{bmatrix}, \quad (7)$$

\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 :

$$P_{abs} = \prod_{n=1}^N P_n \quad (8)$$

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

$$P = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{c}^T & 1 \end{bmatrix} \quad (9)$$

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 6.

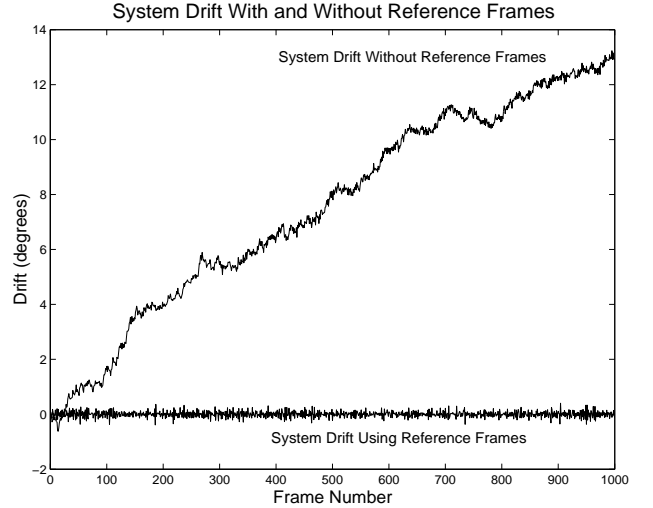


Figure 6: This graph shows the effect of reference frame use on system drift. For this experiment, the camera was held in a fixed position.

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 3-D rotation matrix (R_r) describing the camera orientation, a PCT (P_r) describing its projective 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 describing the position of the reference frame, the estimated PCT between current frame of video and the reference frame (P_e) is given by:

$$P_e = \lambda P_r^{-1} P_l P_g \quad (10)$$

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 uses the PCT estimate P_g from the gyroscope as a starting point (step (1)) for 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.

Estimate Euler Angles from the PCT

In order to close the loop and allow VideoOrbits to correct the gyroscope scaling and 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:

$$P_{rel} = USV^T \quad (11)$$

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:

$$R = UV^T \quad (12)$$

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:

$$\|R - P_{rel}\|_F^2 = trace((R - P_{rel})^T (R - P_{rel})) \quad (13)$$

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 and scaling correction.

5. Using graphics hardware to do computer vision

In order to view the virtual environments, the images must be registered. This involves projecting the images with the proper perspective of the viewer. The projective coordinate transformation used in VideoOrbits maps straight lines to straight lines at all times, and thus represents a subset of all possible image warps. Thus we refer to image warping in VideoOrbits as projection rather than the more general term of image warping. In OpenGL, this is achieved by the viewing of a plane from different camera angles. In fact, the projective parameters calculated by VideoOrbits can be used as a camera transformation in OpenGL.

The process of image warping is 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.

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) and in the construction of a shared mediated reality.

VideoOrbits is well suited for applying OpenGL video acceleration because VideoOrbits is a repetitive multiscale algorithm. Figure 3 shows the VideoOrbits algorithm. The steps of the algorithm which can be accelerated with graphics hardware are outlined in bold.

5.1 Mapping Projective Coordinate Transformations

In OpenGL, the most straightforward way of applying the projective coordinate transformation of VideoOrbits is to consider equation 17 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 :

$$\phi : G \rightarrow M \quad (14)$$

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

$$\begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = G\mathbf{x} \quad (15)$$

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 $R_{4 \times 4}$ projection matrix in OpenGL.

The desired isomorphism is given by:

$$\phi(G) = \phi \left(\begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \right) \quad (16)$$

$$= \begin{bmatrix} 1/a_{22} & -a_{21} & b_2 & 0 \\ -a_{12} & 1/a_{11} & b_1 & 0 \\ c_2 & c_1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (17)$$

This mapping takes into account the different coordinate systems and conventions used by each program. Equation 17 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.

5.2 Measurements of OpenGL acceleration

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 [10]. All programs were run on a wearable computer with a 700 MHz Pentium-III processor, with

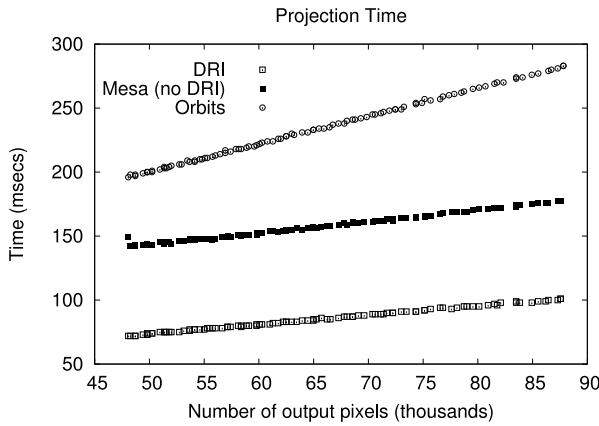


Figure 7: Projection times for VideoOrbits software, Mesa3D (using software rendering) and Mesa3D with DRI hardware rendering enabled.

64 MB of RAM. The wearable computer had an Intel i810 graphics chipset.

Figure 7 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. 3) a program running the equivalent VideoOrbits algorithm.

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 7, 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 axis, 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 $2.75\times$. The average speedup between VideoOrbits and the Mesa software rendering was $1.48\times$ and the average speedup between the Mesa software rendering and the DRI implementation was $1.83\times$. This speedup verifies that our DRI implementation did indeed use the available graphics hardware.

Figure 8 (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

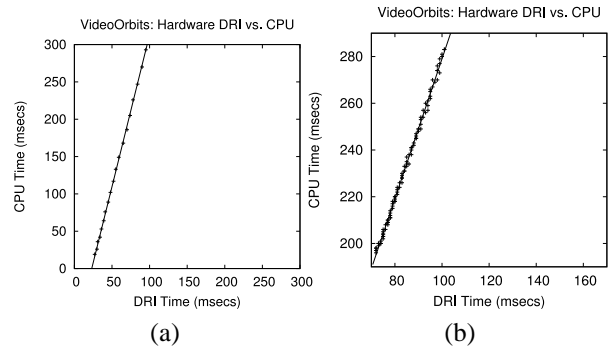


Figure 8: (a) Projection times for VideoOrbits using DRI vs. the CPU given varying image sizes. All image had identical projection parameters. (b) Projection times for VideoOrbits programs, one using DRI and the other the CPU. The algorithm was given a fixed input image and the projection parameters were varied (resulting in larger output images).

was varied. In all cases, the hardware accelerated program projected the image faster than VideoOrbits. 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 2.99. Thus, for this range of input image sizes, the hardware speedup was $2.99\times$.

Figure 8 (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 4.07. 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 hardware was able to handle increased amounts of interpolation $4.07\times$ faster than the VideoOrbits software.

6. Seeing Eye to Eye: a shared mediated reality

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 VOGHT 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 VOGHT and a browser program to synthesize views of wearer A's environment map from an arbitrary

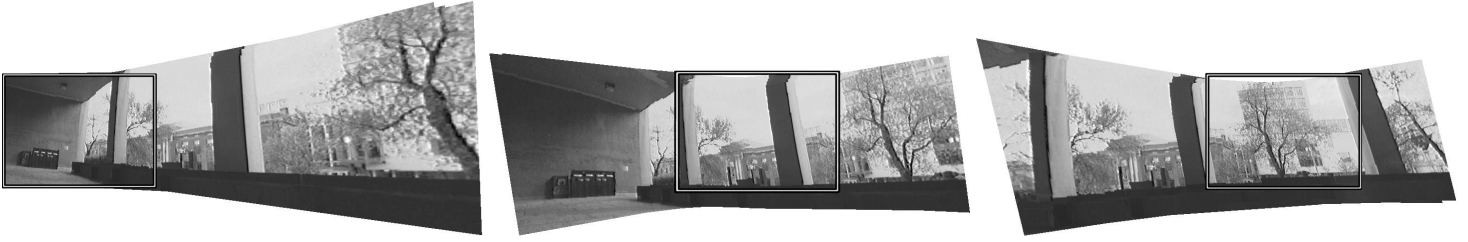


Figure 9: Outdoors: the environment map generated by wearer A, and the views of that map synthesized by wearer B.

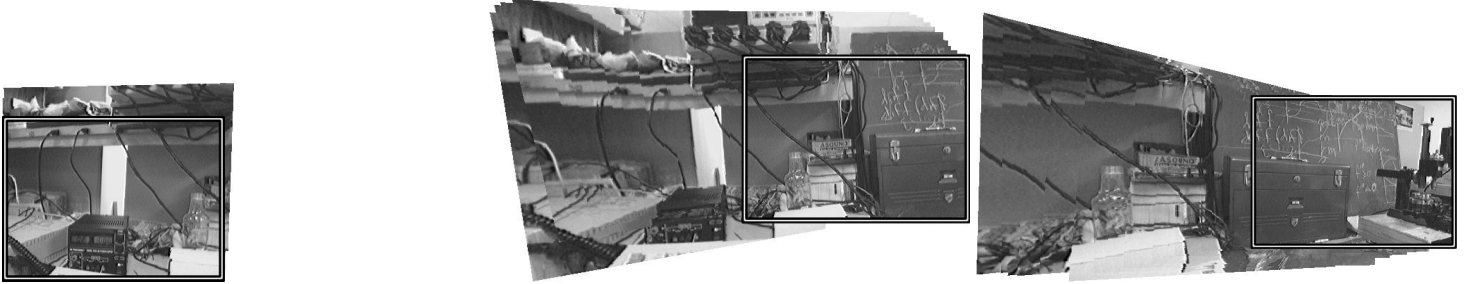


Figure 10: Indoors: the environment map that wearer B generates while browsing wearer A's outdoor map.

viewing direction. As long as the environment map contains some reference images in the neighbourhood 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 VOGHT 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 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 computation of the bounding box can be adjusted to the available computational resources, balancing final image quality against execution time. On a resource constrained wearable computing system, the projective coordinate transform may be applied only to the single closest and newest image.

The resulting image will be quite complete if the image from the map is near the desired viewing perspective. If the image from the map is distant from the viewing perspective then the generated image will not be complete due to the fact that there is no information available.

If the search area is large then multiple images from the map can be projectively transformed and registered (see figures 9 and 10). This registered image allows for the creation of a complete image for the desired perspective even if there is no image in the map at that exact perspective.

6.1 Demonstration

The system may be seen in action in figures 9 and 10, demonstrating results of wearer B browsing an environment map created by wearer A. The first image in figure 9 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 10 shows the corresponding view of B's environment inside a laboratory, with the bounding box described by the Eye-Tap. The middle images show the result of looking to the right: in figure 9 the view has moved to the middle of the environment map, while in figure 10, 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 9 indicat-

ing an area with insufficient information in the environment map.

6.2 Accounting for the Camera Intrinsic Parameters

The EyeTap devices are custom made prototypes and as such contain different cameras. In a shared mediated reality, the remote environment maps will be generated by the remote EyeTap with camera intrinsic parameters $M_{int_{rem}}$. The browser will have an EyeTap with camera intrinsic parameters $M_{int_{loc}}$. Therefore to create the view of the remote environment through the EyeTap of the local user, the intrinsic parameters of both cameras have to be accounted for in equation 18. This allows not only the correct projective transformation but accounts for the differences in each EyeTap's camera intrinsic parameters.

$$P_{view} = P_r^{-1} M_{int_{loc}} M_{int_{rem}}^{-1} P_n, \quad (18)$$

$$M_{int} = \begin{bmatrix} -f_x & 0 & O_x \\ 0 & -f_y & O_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (19)$$

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

The images that are projected are selected by the absolute position (theta and alpha) of the VOGHT. This estimate is used to calculate a region in which to apply the projective coordinate transform. Then using the projected images an image composite is formed. From this composite the correct view is cropped. A range of images is used in order to use a sufficient amount of data from which to generate the new frame.

The absolute gyro position of the image does not affect the calculation of the projection of the images, it is used only to select the region the scene will be constructed from.

The search size of the area is a tunable parameter, a small parameter is useful on a resource constrained wearable computing system. This will apply the projective coordinate transform to the closest and newest image. The result if the image is near the view will be quite good, however if there is no image nearby then no data can be PCT.

7. Conclusion

The Eye to Eye shared mediated reality system allowed users to exchange their current environments through the EyeTap reality mediator. The views were generated and controlled through the users' head motion using the VOGHT. The VideoOrbits algorithm was integral in the system and used to calculate the projective coordinate transforms between images as well as in image registration for the creation of image composites.

Due to the constrained computing power of the wearable computer and the limited bandwidth of the wireless connection, it is not feasible to transmit every captured frame. Through the use of a head tracker, the frames captured and transmitted are optimized to span the entire sphere around the user in an efficient manner. The VideoOrbits algorithm was optimized through the use of common 3D graphics chipsets available on wearable computers providing a $2.75 \times$ speed improvement.

This was demonstrated through the use of the system between two wearable computer users, one outdoors and one indoors. The views generated had the correct perspective and allowed each viewer to see the other's environment as if it were their own. This resulted in the users seeing Eye to Eye in a projectively stabilized shared mediated reality.

References

- [1] M. Artin. *Algebra*. Prentice Hall, 1995.
- [2] E. Foxlin, M. Harrington, and G. P. Pfeiferi. Constellation: A wide-range wireless motion-tracking system for augmented reality and virtual set applications. *Computer Graphics Proceedings, Annual Conference Series*, pages 371–378, 1998.
- [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. Videorbits on eye tap devices for deliberately diminished reality or altering the visual perception of rigid planar patches of a real world scene. In *Proceedings of International Symposium on Mixed Reality (ISMR2001)*, pages 48–55, March 14-15 2001.
- [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] S. Mann and R. W. Picard. Video orbits of the projective group; a simple approach to featureless estimation of parameters. TR 338, Massachusetts Institute of Technology, Cambridge, Massachusetts, See <http://hi.eecg.toronto.edu/tip.html> 1995. Also appears in *IEEE Trans. Image Proc.*, Sept 1997, Vol. 6 No. 9, p. 1281–1295.
- [9] K. Sawada, M. Okihara, and S. Nakamura. A wearable attitude measurement system using a fiber optic gyroscope. In *Proceedings of International Symposium on Mixed Reality (ISMR2001)*, pages 35–39, March 14-15 2001.
- [10] R. Y. Tsai and T. S. Huang. Estimating Three-Dimensional Motion Parameters of a Rigid Planar Patch I. *IEEE Trans. Acoust., Speech, and Sig. Proc.*, ASSP(29):1147–1152, December 1981.