

Mediated Reality Bluetooth Device Locator

University of Toronto

Team members	Chris Aimone	aimone@eyetap.org
	Samir Parikh	samir@eyetap.org
	Andrej Marjan	amarjan@eyetap.org
Project mentor	Prof. Steve Mann	mann@eecg.toronto.edu

Abstract

The widespread implementation of Bluetooth technology in everyday life promises to open a Pandora's Box of device management and tracking issues. In an environment filled with Bluetooth-enabled devices, such as printers, scanners, doors, thermostats and other environment control systems, there is a need to easily discover, operate and manage devices remotely. However, it becomes a challenge to physically locate the device one is currently using.

Wearable computers can already be used effectively as a unified personal interface to this environment. However, the wearable computer must also handle situations in which the objects being used need to be located. By implementing the Mediated Reality Bluetooth Device Locator (MRBDL) on a wearable computer, it is possible to modify what the user sees such that nametags are attached to all visible wireless devices. To achieve this, devices must first be located and then tracked relative to the user's field of view.

To illustrate how this might be beneficial, consider the case of an unfamiliar office environment, in which an individual would like to send a document to a nearby printer. This usually means searching through a long list of printing devices, perhaps by trial and error. A user equipped with an MRBDL-enhanced wearable computer system will be able to walk up to the Bluetooth-enabled printer, and immediately send it a document to print.

The MRBDL system visually locates the devices in the area and labels them in the user's field of view. It enables the user to easily select a specific device out of all the available devices that have been discovered.

Contents

1	System Overview	5
1.1	Performance Requirements	7
1.2	Design Methodology	7
1.3	Innovation	8
2	Implementation and engineering considerations	8
2.1	Wearable computing and the EyeTap Principle	8
2.2	VideoOrbits	9
2.3	MRBDL Implementation	13
2.4	Bluetooth Communication Subsystem	13
2.5	Visual Identification Subsystem	16
2.6	Object Tracking System (OTS)	18
2.6.1	Capture Video Frame	19
2.6.2	Convert to Greyscale	19
2.6.3	Undistort Image	20
2.6.4	Capture Gyroscope	20
2.6.5	Gyroscaling: $Y = B(X-A)$	21
2.6.6	Is Rotation Small Enough?	21
2.6.7	Calculate Equivalent Projective Transformation for Gyro Pitch and Yaw	21
2.6.8	VideoOrbits Engine	22
2.6.9	Estimate best fit 3-axis rotation from the PCT	22
2.6.10	Estimate linear model $Y = B(X-A)$ for gyroscope	23

2.6.11	Compose Relative PCT with Absolute PCT	24
2.7	Tracking Performance	25
2.7.1	Performance of Closed Loop Gyro Correction	26
2.8	Future Development	26
2.9	Additional Costs	27
3	Summary	28

List of Figures

1	System Overview	6
2	Wearcomp and EyeTap	9
3	User Interface	14
4	Bluetooth Protocol Overview	15
5	VideoOrbits Process	17
6	Tracking System Block Diagram	19
7	PIC-based gyroscope reader	21
8	Gyro Error Correction	24
9	MRBDL in action: tracking sequence	25

1 System Overview

An environment that is densely populated with Bluetooth devices can become confusing and difficult to navigate and use. One can be presented with an assortment of Bluetooth devices, without knowing the identity of those devices in one's physical environment. Sometimes the location of the device is critical to use it effectively. For example, a disabled person seeking to exit a building may be presented with a menu of all doors in Bluetooth range. MRBDL can be used to pinpoint the identity of the particular door that they wish to use. *Thus the MRBDL system provides a visual relationship between the physical object and its Bluetooth identity.*

MRBDL consists of three parts: the initial discovery phase, the visual identification phase, and the device tracking phase. The discovery phase is an ongoing process that utilizes the Bluetooth hardware extensively. Using the builtin service discovery functionality, it aims to provide the user with a constantly updated list of available services within their immediate vicinity.

The visual identification component of this system consists of two parts that communicate via Bluetooth connections:

1. An intelligent infrared beacon linked to Bluetooth hardware;
2. A beacon identification system on the wearable computer.

This system enables the user to select a specific device and enable its infrared beacon. This beacon, which emits a unique blinking pattern, is visually identified by means of difference images between successive frames of video to determine the beacon's exact coordinates. The Video Orbits algorithm is used to compensate for the user's head motion.

However, relying on this active locating technique to keep track of the beacon's location over

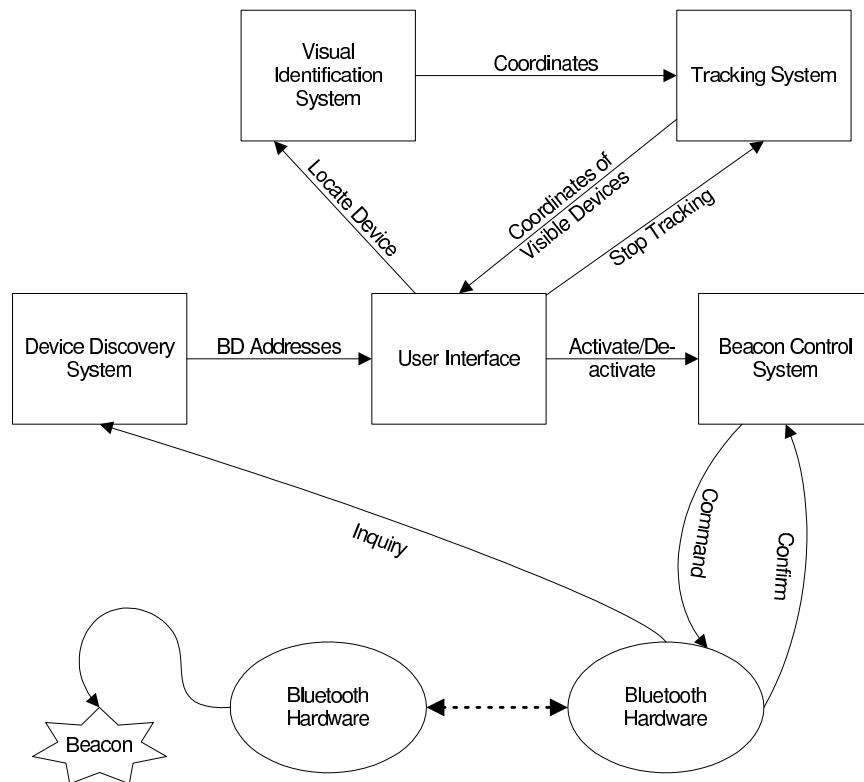


Figure 1: System Overview

time is not appropriate: first, it would require the beacon to flash as long as the device is of interest, which is wasteful and may impede the efforts of others to use the device. Secondly, it would require constant attention on the part of the user.

To allow the user the flexibility of being able to look away from the beacon, the tracking system labels the beacon via the wearable computer’s reality mediation facilities whenever it is in the user’s field of view, and remembers its position relative to the user, when the beacon is not visible. The user’s head movements are measured by a gyroscope–augmented head tracking algorithm and applied to the coordinates of the beacon, so its position relative to the user is known at all times.

In summary, the user first selects a device they wish to locate. Then the visual identification

system engages the device's beacon and identifies it when it first appears in the user's field of view. From this point onwards, the beacon is disengaged and the tracking system continues to update the beacon's position relative to the user, displaying this information whenever the beacon is in the user's field of view.

1.1 Performance Requirements

The MRBDL system can be implemented on any wearable computer (wearcomp) powerful enough to handle the numerical processing requirements of all three components. This implementation of MRBDL is based on the EyeTap class of wearcomps. Because the user is viewing his environment as mediated reality through the EyeTap, the processed information must be presented at a frame rate of at least 15 frames per second, in order to prevent inducing disorientation and nausea in the user.

1.2 Design Methodology

The team adopted a waterfall model to develop the MRBDL system and documented the various stages and processes required to implement the system. During the requirement analysis stage, the system was functionally divided into three major components, in order to facilitate parallel development on the project. Initially the specifications and objectives of each module were defined and a team member was assigned to implement these specifications for each component. After unit testing was successfully completed, the individual parts were integrated into the final system as a group effort. Finally, the completed system was tested to insure functionality in all scenarios. Work continues to further optimize the system.

1.3 Innovation

This project extends the applications of Bluetooth by relating a Bluetooth device's identity to its physical identity. This information may become critical to the user whenever physical interaction between the user and the unlocated device is required. The MRBDL system's use of object identification and tracking algorithms that are further extended by gyroscopic data, is an original approach to mapping the Bluetooth identity of a device with its physical location. In addition these algorithms are supplemented with a robust Bluetooth based protocol so that the issues of latency and concurrent device discovery by multiple users are resolved.

2 Implementation and engineering considerations

2.1 Wearable computing and the EyeTap Principle

The MRBDL system is built around the concept of the "wearcomp" wearable computer as defined by Prof. Steve Mann, University of Toronto (see Figure 2 (a)). The wearcomp is a small body-worn computer system that is always on and accessible, and that mediates the user's experience of the surrounding environment. Often integrated into clothing, this system can be operated through a variety of control devices, from conventional keyboards to specialized portable keyers, and even with brain waves [1]. The wearcomp provides visual feedback through a head-mounted display unit.

The EyeTap (see Figure 2 (b)) is an important part of the wearcomp utilized in the MRBDL system, as it enables the user's vision to be computationally processed and modified in real-time [2]. All light destined for the user's eye is redirected to a camera and digitized for processing on the



(a)



(b)

Figure 2: (a) A late 1990s wearcomp model. (b) A visible EyeTap implemented in safety glasses.

wearcomp. This digitized video maybe modified or replaced and is projected into the user's eye via a head-mounted display element. This act of altering perception through the use of a computer is known as Computer Mediated Reality. An EyeTap must be properly situated in the center of the eye's projection so that the displayed image does not cause a parallax mismatch with what is seen by the untapped eye.

2.2 VideoOrbits

The VideoOrbits [2] algorithm is extensively used in the Visual Identification and Tracking stages to identify and track objects. VideoOrbits is a featureless motion estimation algorithm that computes best fit projective pixel coordinate transformations between two overlapping images (the transformation is applied across the entire image). This group of transformations can describe exactly the relation between two images where all the objects are static in the scene and the camera is free only to rotate and zoom. The motion of a head mounted camera between successive frames

of video can be approximated very well by a pure rotation, since we often scan our environment using head rotation. Thus the application of VideoOrbits to this video can provide extremely good registration between video frames. This characteristic provides the basis for tracking physical head rotation from a real-time video sequence. First, the basics of optical flow and algebraic projective geometry are introduced, then the essentials of the VideoOrbits algorithm are covered.

Optical flow is the canonical method for determining image motion. The brightness constancy constraint equation (BCCE) for 2-D images [3], which gives the flow velocity components in the x and y directions, is:

$$\mathbf{u}_f^T \mathbf{E}_x + E_t \approx 0 \quad (1)$$

where u_f is the 2-D translational velocity vector, $[u_x u_y]$, 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.

Because the 2-D BCCE is under-constrained (that is, it is a single equation with two unknown variables), it is necessary to impose further constraints, usually in the form of a model for the pixel velocities in a region. The simplest model, proposed by Lucas and Kanade, assumes that the velocity is constant over some small region. This provides a velocity vector for each block. This technique is appropriate when such a velocity field is desired, but the constant velocity model perform poorly for describing the motion over a whole image. It is only capable of describing image translation. VideoOrbits uses a rational pixel velocity model known as a projective coordinate transformation (PCT) [4]. This model has eight parameters and can describe the exact pixel motion between two images of a static scene where the camera is free to pan, tilt, and zoom about its optical axis. It also describes exactly the pixel motion of images of a planar surface where the

camera has the added freedom of translation.

$$\mathbf{x}' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \frac{\mathbf{A}[x, y]^T + \mathbf{b}}{\mathbf{c}^T[x, y]^T + 1} = \frac{\mathbf{A}\mathbf{x} + \mathbf{b}}{\mathbf{c}^T\mathbf{x} + 1} \quad (2)$$

VideoOrbits estimates the eight parameters of this model by using the brightness constancy constraint with the PCT to describing the the pixel velocities as:

$$u_m = \frac{ax + b}{cx + 1} - x \quad (3)$$

This method is known as ‘projective–flow’ [5].

Because of effects such as parallax in the images due to motion other than rotation, independently moving objects (IMOs) in the scene, camera automatic gain control (AGC), the PCT cannot perfectly fit 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:

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

The spatial gradient E_x is calculated taking the difference between neighbouring pixels in the horizontal and vertical directions. The temporal derivative E_t is calculated by differencing corresponding pixels in successive video frames.

Minimizing (4) is simplified by weighting by $(\mathbf{c}\mathbf{x} + 1)$ giving:

$$\varepsilon_w = \sum \left((\mathbf{A}\mathbf{x} + \mathbf{b} - (\mathbf{c}^T\mathbf{x} + 1)\mathbf{x})^T \mathbf{E}_x + (\mathbf{c}^T\mathbf{x} + 1)E_t \right)^2 \quad (5)$$

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:

$$\left(\sum \phi\phi^T \right) [a_{11}, a_{12}, b_1, a_{21}, a_{22}, b_2, c_1, c_2]^T \quad (6)$$

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

It can be shown that the set of projective coordinate transformations (PCTs) form a group giving it three main properties: For $a \in P$ where P is the set of all PCTs

(1) Associativity: $a \cdot b \in P \forall a, b \in P$

(2) Identity: $1 \cdot a = a \cdot 1 = a$

(3) Inverse: $aa^{-1} = a^{-1}a = 1$

Since only the first derivative of the spatial gradient is calculated, the modeled change in pixel brightness due to translational velocity is only first order accurate (actually less due to the finite difference approximation of the gradient). This limitation causes the method to be accurate only when the motion between successive images is small. To solve this problem, VideoOrbits exploits the group law of composition to find the optimal PCT between two images using the following iterative technique:

(1) Estimate the projective coordinate transformation **P**

(2) Use this **P** to transform the appropriate original image

(3) Estimate another PCT (call it **P2**) between this warped image and the other original image.

(4) compose an improved **P** by composing **P2** with **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.

2.3 MRBDL Implementation

The MRBDL system was split into three components during the functional design phase. The three components consisting of the Bluetooth communication framework, the Visual Identification Subsystem (VIS) and the device tracking subsystem were implemented separately to reduce the development time required.

2.4 Bluetooth Communication Subsystem

The Bluetooth Communication Subsystem is responsible for all communication between the user and the remote device, which consists of device discovery and connection management. It utilizes the BlueZ Bluetooth drivers [6] for the Linux kernel for communication via Bluetooth hardware. As a result the remote device and the wearcomp contain USB/PCMCIA links and run the Debian Linux Distribution. This component encompasses all Bluetooth related software and hardware on the client side(wearcomp) and the server side (remote device) and follows a client-server model for communication.

The *Client Manager* software consists of utilities to discover Bluetooth devices, communicate with remote devices and update the user interface with a list of currently available devices in the vicinity. The hcitool utility provided with the BlueZ Bluetooth stack was used to perform continuous inquiry of the user environment to determine the addresses of adjacent devices. A wrapper for this utility was written to maintain a database of Bluetooth Device (BD) addresses by comparing the results of the hcitool inquiry with the BD addresses already in the database. A simple GUI was designed that reads this database and presents the user with a list of devices and provides them the option to 'locate and track' (see Figure 3). On selection of a device, the blink



Figure 3: The user interface: through they EyeTap, the user sees one Bluetooth-enabled desk lamp in range, which is being tracked.

utility is passed the address of the device as well as the command to be sent. In this case, it is to switch the infrared beacon to blink mode. The blink utility takes the BD address passed to it and attempts to connect to the remote device. After successful connection , all communication between the client and server takes place via Logical Link Control and Adaptation Protocol (L2CAP) data packets. This utility exits after the command code has been sent to the server and confirmation for it has been received by the client. Now that the beacon is switched on, the GUI activates the VIS and the beacon's location identified. The VIS, which is a separate module, returns and the user interface again utilizes the blink utility to send the command to turn the beacon off. Thereafter, device discovery resumes and the GUI continues updating the list of available devices. The *Client Manager* is run on a wearcomp and was designed with separate components to make its integration with other components easy.

The *Server Daemon* software runs on the remote device and handles all incoming connections

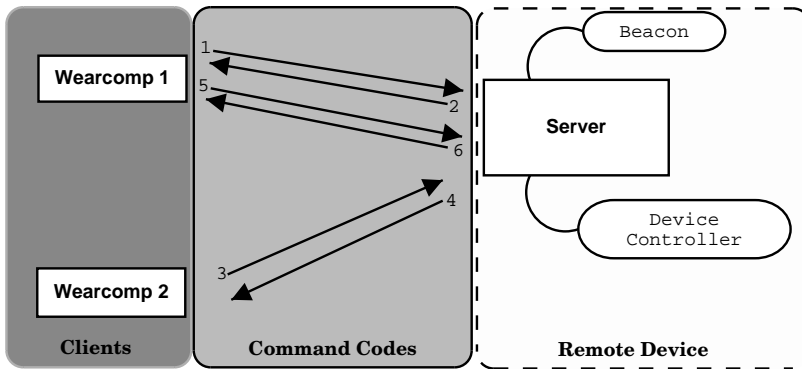


Figure 4: Bluetooth protocol overview: 1) Wearcomp 1 sends the ON command to the remote device.2) Remote device responds with a CONFIRM packet 3) Another Wearcomp sends a command 4) Remote device responds with a BUSY packet 5) Wearcomp 1 sends the OFF command to the remote device 6) Remote device sends the CONFIRM packet and starts listening for connections from any wearcomp

Command	Code
ON:	0x001
OFF:	0x0ff
CONFIRM:	0x002
BUSY:	0x003

Table 1: Bluetooth Packet Codes

and commands. It also controls the infrared beacon according to the commands received from the client. This utility binds itself to an L2CAP socket and listens for incoming connections. Once a connection is established with the client, the server communicates via a protocol designed to work over the L2CAP layer. Refer to Table 1 for the list of data codes for the various commands that can be sent via this protocol. Figure 4 illustrates a scenario where the advantages of this protocol in restricting a single user to a specific device are evident. The server can only receive two types of command packets: *ON* and *OFF*. The *ON* command is a signal for the server to initiate the beacons blinking process and to lock itself to that particular client. After locking itself, the server sends *BUSY* packets to any other client that request connections. This prevents more than one client from attempting to locate the object at the same time. The daemon interfaces with the beacon via the parallel port and controls the beacon's status (blinking/off). The software remains locked to the client as long as a *OFF* command packet is not received. When this happens, it sends

a confirmation and resumes normal operation by listening on the L2CAP socket. This software would be integrated into existing software to control the device so that the server would be able to operate the beacon and manage the device(eg: turn on water fountain). Thus all server software can be integrated into a microcontroller because it is not computationally intensive. However our implementation utilises a standard computer for operation.

2.5 Visual Identification Subsystem

This component of the MRBDL system runs on the user's wearable computer and is activated when the user selects an object to be located. Through the EyeTap, this process continuously analyses the user's field of view, looking for a blinking pattern that is emitted by the infrared beacon on the remote device. The detection of this pattern is based on a series of difference images generated by VideoOrbits algorithm, which operates on successive frames captured from the camera.

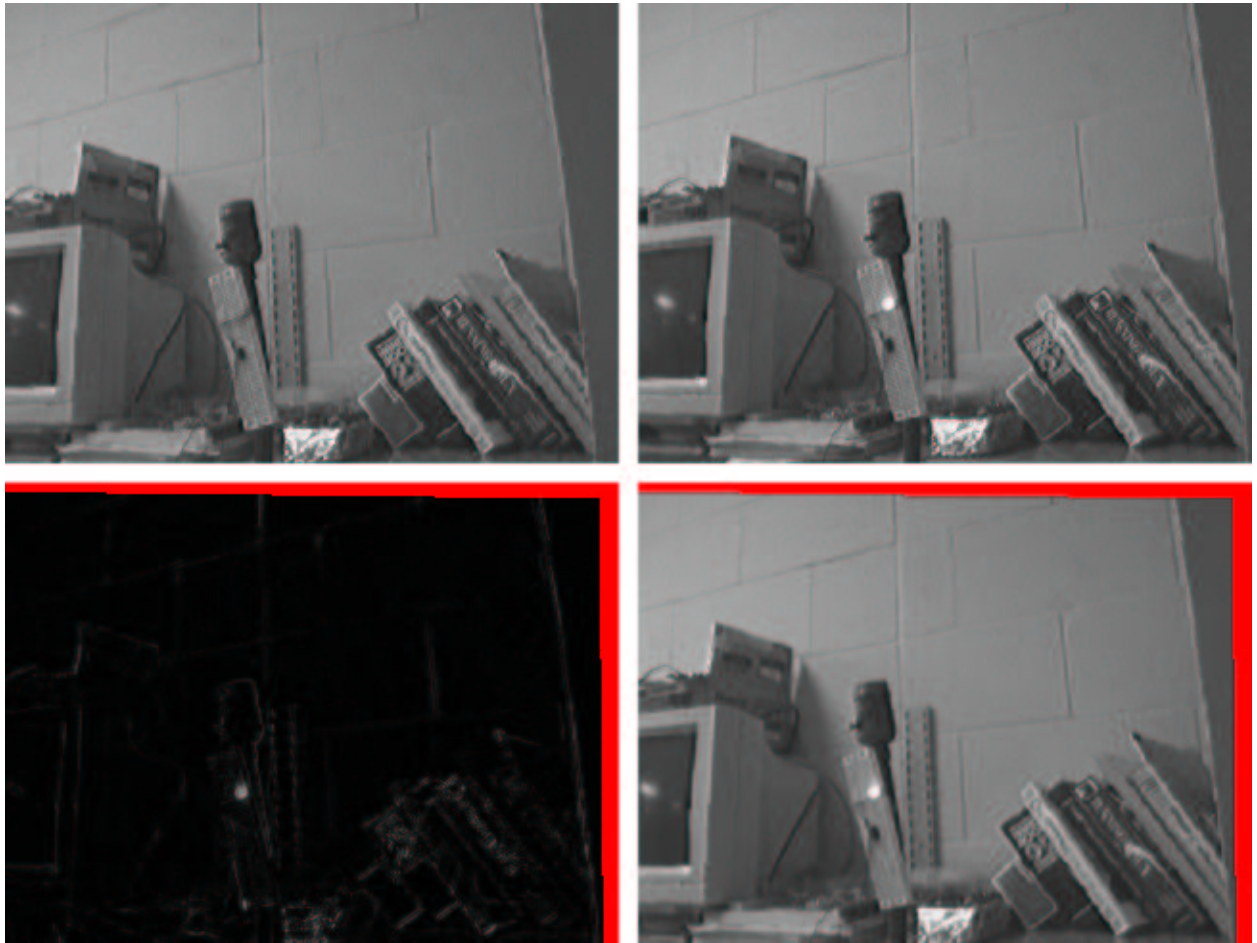


Figure 5: The VideoOrbits algorithm in action. Clockwise from top left are the first image, the second image, the second image after it has been transformed to match the first image, and finally their difference image. Note the presence of the LED light in the second image but not the first image, and the corresponding bright area of the difference image.

Transformations computed by VideoOrbits describe exactly the relation between two images of a planar surface taken from arbitrary camera locations. This allows the algorithm to calculate the second image's projective coordinate transformation compared to the first image. By applying these parameters to the second image, we account for any movement of the head unit after the first image is captured in terms of rotational or translational movement. Since the two images are now registered, a difference image is taken by subtracting the modified second image from the first. Ideally this difference would be black; however, due to the blinking beacon, the first image does not contain the light from the beacon but the second image does (see Figure 5). Thus a difference image will be dark except for a bright spot indicating the location of the infrared beacon. This coordinate is memorized by the identification system and is passed on to the Tracking subsystem to label when the object is visible in the user's field of view. After the location of the beacon has been determined, the Visual Identification Subsystem notifies the Client Manager, which deactivates the remote beacon.

2.6 Object Tracking System (OTS)

The Gyroscope / VideoOrbits system is a closed loop system in which the gyroscope provides an estimate of camera motion to allow VideoOrbits to converge to the desired solution even in the presence of large movement between video frames. The projective coordinate transform (PCT) produced by VideoOrbits (which has good absolute registration) is then used to build a linear model for the gyroscope to correct for gyro drift and scaling problems. A block diagram of the system is shown in Figure 6. Presentation of the system will be done blockwise with respect to this diagram.

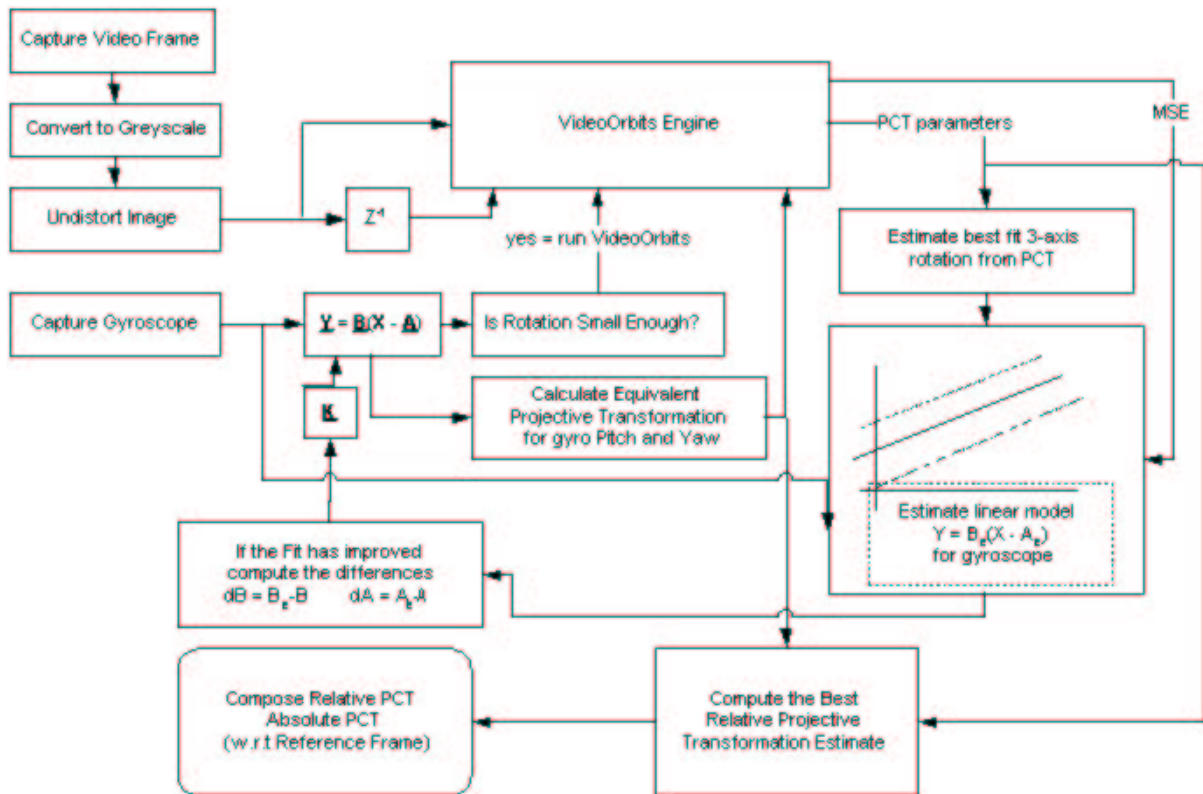


Figure 6: Gyro / VideoOrbits Tracker Block Diagram

2.6.1 Capture Video Frame

The digital video format chosen for this system was IEEE1394, due to the high quality possible with this format as a result of its large bandwidth. It also allows control of the exposure settings on compatible cameras. This is important since VideoOrbits depends on the the brightness constancy constraint.

2.6.2 Convert to Greyscale

VideoOrbits operates on mono images so conversion from a three channel model was necessary. Since the VideoOrbits is based on the brightness constancy constraint, we wish to capture as much of the image brightness information as possible in the mono image. It happens that the YUV color

space yields near optimal energy compaction (most uncorrelated) in most color image with a single luminance component and two chrominance components. Conveniently IEEE1394 cameras can capture in YUV. Conversion to greyscale was then achieved by selecting only the Y component. Conversion to another colourspace such as HSV or HMMD may have been more ideal (since we are looking for perceptual image registration). However any gains would have been very small and would not be worth the computation.

2.6.3 Undistort Image

Wide angle cameras such as the one used exhibit a large amount of radial distortion. Since the PCT assumes an ideal projective camera, such distortion severely impairs the performance of VideoOrbits. To correct this, the camera was calibrated by using a number of images of a known pattern. The camera intrinsic parameters were estimated to first order and then a nonlinear optimization algorithm was used to solve for the radial distortion parameters. Undistortion was performed with the optimized implementation in the Intel Open Source Computer Vision Library (OpenCV) [7].

2.6.4 Capture Gyroscope

The gyroscope used is a product of Gyration Inc. This device gives analog rate outputs for two rotational axes. A PIC microcontroller was used to digitize the signal and to compute the interframe integration on the outputs. The rate outputs were low-pass filtered to reduce signal noise and aliasing due to sampling. Communication between the PIC and the wearcomp was done over an RS232 communication port (see Figure 7).

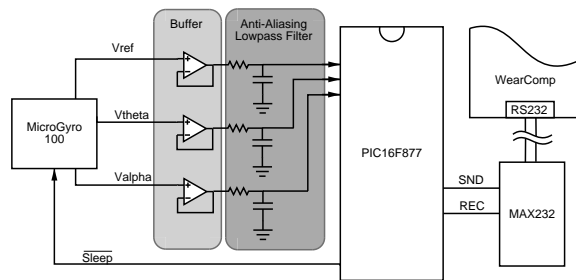


Figure 7: PIC-based gyroscope reader

2.6.5 Gyroscaling: $Y = B(X-A)$

While gyro drift is inescapable, it remains relatively constant and can be subtracted from the rate output fairly well. The gyro output must also be scaled to match the units used in the system. Unfortunately as with all analog systems, component values drift with temperature and load changes can affect the power system levels. Both of these will cause variations in the required drift and scaling parameters. In this system, B and A are continuously modified to adapt to local statistics.

2.6.6 Is Rotation Small Enough?

VideoOrbits needs frame overlap to find the appropriate transformation. This function prevents VideoOrbits from attempting to register two images that are too far apart (in a spatial sense). If the rotation is too large, relative rotation estimation is performed solely by the gyroscope.

2.6.7 Calculate Equivalent Projective Transformation for Gyro Pitch and Yaw

In order for the gyroscope to provide an initial transformation estimate for video orbits, the relative motion in the two gyro axes must be converted into an equivalent representation.

2.6.8 VideoOrbits Engine

Two successive frames of undistorted video are processed, provided that the motion between the frames is not too great. VideoOrbits runs for a prescribed number of iterations (depending on the computational power of the wearcomp), and returns its best estimate of the PCT parameters. Orbits also returns a Mean Squared Error (MSE) value for the returned PCT. The MSE is calculated by adding the squared difference between the original image and its warped temporal neighbor. This MSE value can give an indication of the registration accuracy. This is very important since we have a combined estimation method and there is no guarantee that VideoOrbits will converge towards an actual solution (testing shows that orbits either converges nicely or diverges wildly). Unfortunately the MSE measure is very sensitive to the nature of the images used (and thus sensitive to the EyeTap user's local environment). To cope with this, local statistics are kept on the MSE levels to provide intelligent adaptation changing environments.

2.6.9 Estimate best fit 3-axis rotation 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 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 when we have a valid rotation to be used in the statistical model for the gyroscope. To find the required rotations, the PCT is first approximated by a pure rotation matrix (up to a multiplicative factor). This is done as follows: if P is the PCT and the Singular Value Decomposition P is:

$$P = USV^T \quad (7)$$

Where S is a diagonal matrix of singular values. The best fit rotation matrix in terms of the Frobenious norm between it and P is given by:

$$R = UV^T \quad (8)$$

This rotation matrix is then used to compute the corresponding Euler angles. Computation of these angles will not be presented here.

2.6.10 Estimate linear model $Y = B(X-A)$ for gyroscope

In order to estimate the gyro drift and scaling parameters, data points are collected based on their associated MSE with respect to the local statistics and the error in the rotation matrix estimated from the PCT. Two different model estimates are computed here:

1. A constant model that is used only for drift correction. This model does not depend on the PCT parameters, and is intended to correct for a wildly drifting gyroscope. The goodness of fit for this model is described by the variance of the gyro rate.
2. A linear model that describes both the drift and scaling parameter of the gyroscope. For this model, the error statistics are computed as the standard error in the slope and the intercept, as well as the correlation coefficient.

The obtained error statistics are then used to decide whether or not to correct the gyro model. In this decision the tolerances become continuously more stringent to converge toward a desirable solution. However, there is a condition that checks for a good linear fit as well as significantly different parameters to allow the system to accommodate a significant change in parameters, which may occur in events such as running into a wall or operating a power tool off the computer power supply.

2.6.11 Compose Relative PCT with Absolute PCT

To track absolute motion, this process, chooses the best estimate of PCT between successive frames, based on MSE returned by VideoOrbits and the locale MSE statistics.

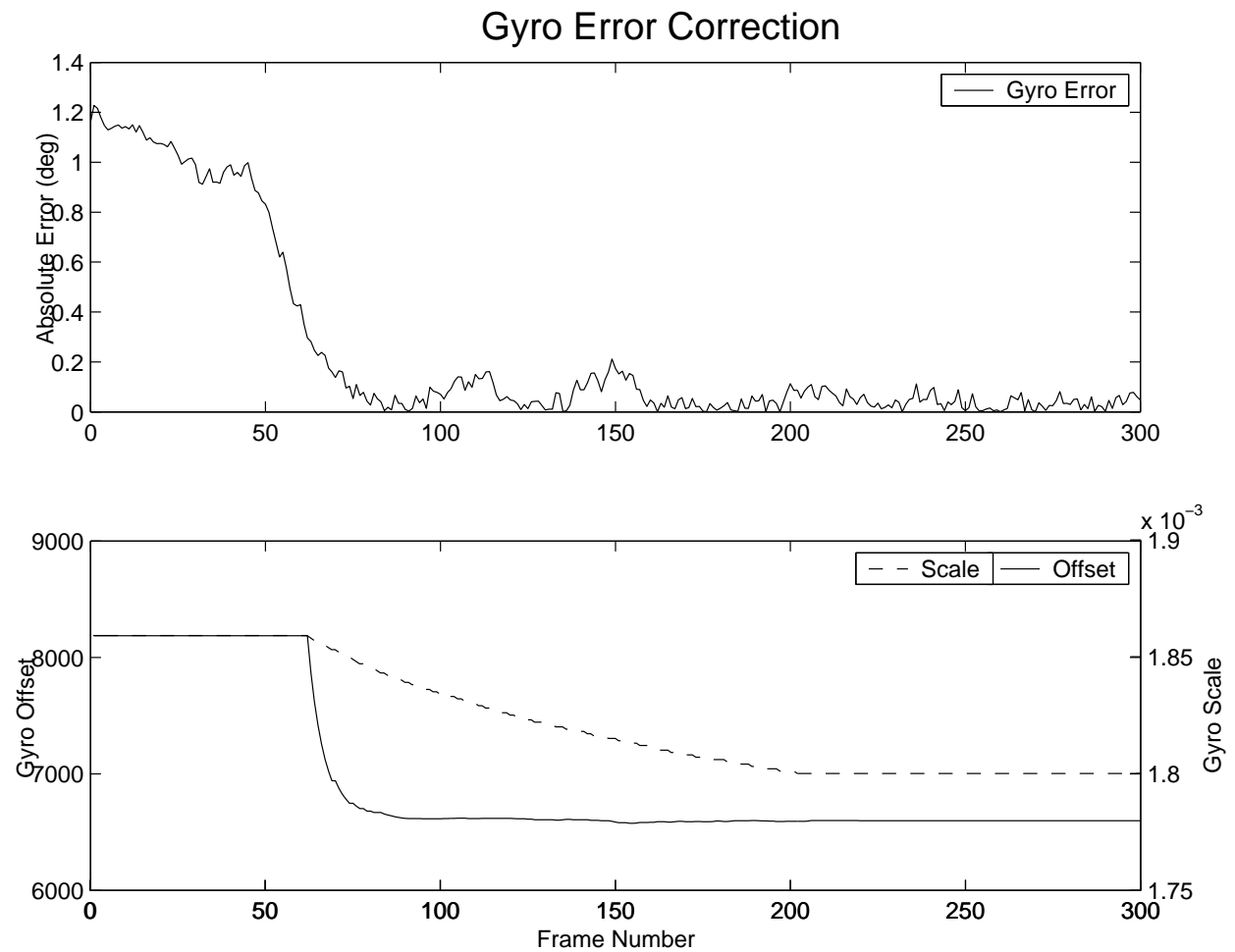


Figure 8: The above data were taken in a scenario where power supply voltage to the gyro had been changed to simulate power level fluctuations in a battery-driven system. We can see by the reduction of the absolute error that the system has converged to a more desirable state.

2.7 Tracking Performance

Operating on a desktop class computer, the OTS was able to 3-d rotation with reasonable accuracy in real-time. As we can see in Figure 9, accuracy to within a few pixels is achieved with significant head movement, so long as the motion is near pure rotation. We can also see from the image sequence that the tracking algorithm is robust in the presence of independently moving objects. VideoOrbits, when operating alone, is subject to very large errors when an IMO is introduced. With the gyro in the system, IMO-tainted PCT parameters are rejected due to the relatively high MSE values. The system also performed very well when exposed to large, rapid rotations. This is important because people regularly move their heads rapidly when scanning their environments.



Figure 9: MRBDL in action: the first sequence of four images shows VIS locating the beacon attached to the monitor. Notice that the beacon toggles on and off in subsequent images. The next three images show the OTS correctly positioning the crosshairs on the disengaged beacon's location in spite of significant head movement. The final five images illustrate the gyro-assisted OTS's ability to track devices even in the presence of independently moving objects.

2.7.1 Performance of Closed Loop Gyro Correction

The system presented, has shown the ability to correct for the gyro scaling and drift parameters through normal wear. The length of time for the calibration to occur 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 of gyro drift and scaling.

2.8 Future Development

During the design and implementation phases of this project, several limitations were realised. While some of them have already been resolved, others are still being worked on.

The initial design of the Bluetooth subsystem did not consider the case of several users searching for several devices in the same space. As a result, the protocol was limited to permit only one user to locate a specific device at a time. In addition, the VIS has the similar limitation of allowing only one user to locate an object at a time because the blinking pattern for the infrared beacon is the same for any two adjacent devices. Thus the VIS would not know which of the two devices the user wishes to locate. Lastly, the VideoOrbits algorithm which forms the basis of the VIS and the OTS is limited to head motion such as panning and rotation. It cannot handle fast motion because of the requirement for overlap between successive images.

To resolve these limitations, several solutions are proposed. Firstly, the VideoOrbits algorithm was augmented with feedback from a gyroscope to remove the limitation of fast head motion.

To track multiple objects, a unique blinking pattern system was proposed whereby each device is assigned a unique pattern when it is manufactured, just as Network Interface Cards are assigned

unique MAC addresses. The protocol would be modified to allow the device to communicate its blinking pattern to the client. Furthermore, due to the uniqueness of the blinking pattern, multiple clients can locate a single device concurrently. This is made possible by making the device the master in a piconet with the clients acting as slaves.

2.9 Additional Costs

Equipment	Cost
Gyrations MicroGyro 100 Miniature Gyroscope	\$100
Orange Micro iBOT IEEE1394 Web Cam	\$125
PIC16F877 Microcontroller	\$10
Miscellaneous Electronic Parts	\$15
Total	\$250

3 Summary

The Mediated Reality Bluetooth Device Locator was designed to help users cope with the overwhelming number of Bluetooth-enabled devices that will find their way into the environment. The system achieves this goal by providing a visual relation between a device and its identity in the Bluetooth realm. Through the innovative application and augmentation of the VideoOrbits algorithm, MRDBL achieves this goal effectively. Not only can it locate any device of interest to the user, but it also maintains a memory of the device's location relative to the user.

While several scenarios have been presented, another possible commercial application is the location of inventory in a warehouse. A worker who needs to locate a specific item can use the Bluetooth devices discovery mechanism, to select the section of the aisle containing the item. MRBDL will visually identify, track, and label the aisle and then the section. Thus a new worker's efficiency will be increased, because he is not limited by his ignorance of where items are stacked.

As the popularity of wearable computing rises, wearcomps are becoming more compact and more affordable. Work is currently underway to create a hardware implementation of VideoOrbits, which would provide an inexpensive and resource-efficient means of implementing VIS and OTS at the full 30 frames per second required for maximum user comfort.

However, the system is practical today, because current generation wearcomps are sufficiently powerful to run the current all-software MRBDL implementation. Also, the remote beacon control system can be implemented entirely in a microcontroller, which is feasible for mass production and inclusion in a wide variety of devices.

References

- [1] Steve Mann, Daniel Chen, and Saman Sadeghi. Hi-cam: Intelligent biofeedback signal processing. In *IEEE ISWC-01*, pages 178–179, Switzerland, October 8-9 2001.
- [2] Steve Mann. *Intelligent Image Processing*. John Wiley and Sons, November 2 2001. ISBN: 0-471-40637-6.
- [3] B. Horn and B. Schunk. Determining Optical Flow. *Artificial Intelligence*, 17:185–203, 1981.
- [4] 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.
- [5] Steve 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>.
- [6] *Bluez Linux protocol stack*. <http://bluez.sourceforge.net>.
- [7] Intel Corp., <http://www.intel.com/research/mrl/research/opencv/>. *Intel Open Source Computer Vision Library*.