

Simple, Robust and Accurate Head-Pose Tracking Using a Single Camera

Simon Meers, Koren Ward and Ian Piper

University of Wollongong, Australia.

Introduction

Tracking the position and orientation of the head in real time is finding increasing application in avionics, virtual reality, augmented reality, cinematography, computer games, driver monitoring and user interfaces for the disabled. While developing a computer interface for blind computer users, we encountered the need for a robust head-pose tracking system for accurately monitoring the gaze position of the user on a virtual screen. Although many head-pose tracking systems and techniques have been developed, we found most existing systems either added considerable complexity and cost to our application or were not accurate enough for our requirements. For example, systems described in (Horprasert et al. 1996), (Kaminski et al. 2006) and (Newman et al. 2000) use feature detection and tracking to monitor the position of the eyes, nose and/or other facial features in order to determine the orientation of the head. Unfortunately these systems require considerable processing power, additional hardware or multiple cameras to detect and track the facial features in 3D space. Although monocular systems like (Horprasert et al. 1996), (Kaminski et al. 2006) and (Zhu et al. 2004) can reduce the cost of the system, they generally performed poorly in terms of accuracy when compared with stereo or multi-camera tracking systems (Newman et al. 2000). Furthermore, facial feature tracking methods introduce inaccuracies and the need for calibration or training into the system due to the inherent image processing error margins and diverse range of possible facial characteristics of different users.

To avoid the cost and complexity of facial feature tracking methods a number of head-pose tracking systems have been developed that track LEDs or infrared reflectors mounted on the user's helmet, cap or spectacles (see (NaturalPoint 2006), (Foursa 2004), (Foxlin et al. 2004), and (Hong et al. 2005)). However we found the pointing accuracy of systems utilising reflected infrared light (NaturalPoint 2006) to be insufficient for our application. The other LED-based systems, like (Foursa 2004), (Foxlin et al. 2004), and (Hong et al. 2005), still require multiple cameras for tracking the position of the LEDs in 3D space which adds cost and complexity to the system as well as the need for calibration.

In order to overcome much of the cost and deficiencies in existing head-pose tracking systems we have been developing accurate methods for pinpointing the position of infrared LEDs using an inexpensive USB camera and low-cost algorithms for estimating the 3D coordinates of the LEDs based on known geometry. Our system is comprised of a single low-cost USB camera and a pair of spectacles fitted with three battery-powered LEDs concealed within the spectacle frame. Judging by our results, we believe our system to be the most accurate low-cost head-pose tracking system developed. Furthermore, our system is robust and requires no calibration. Experimental results are provided demonstrating a head-pose tracking accuracy of less than 0.5 degrees when the user is within one meter distance from the camera.

2 Hardware

The prototype of our infrared LED-based head-pose tracking spectacles is shown in Figure 1(a). Figure 1(b) shows our experimental rig for testing the system, which incorporates a laser pointer (mounted below the central LED) for testing the ‘gaze’ accuracy. The baseline distance between the outer LEDs is 147mm; the perpendicular distance of the front LED from the baseline is 42mm.

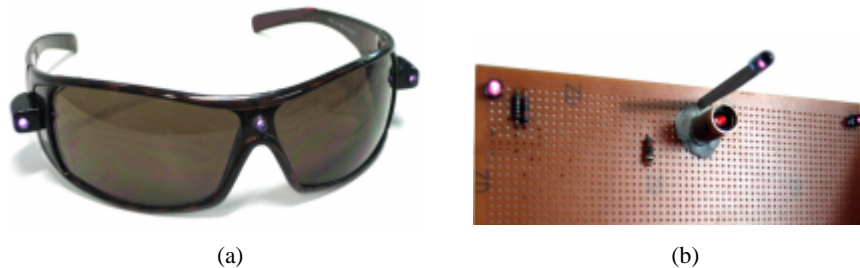


Figure 1. (a) Prototype LED Spectacles (b) LED testing hardware

Although the infrared light cannot be seen with the naked eye, the LEDs appear quite bright to a digital camera. Our experiments were carried out using a low-cost, standard ‘Logitech QuickCam Express’ USB camera (Logitech 2006), providing a maximum resolution of 640x480 pixels with a horizontal lens angle of approximately 35°. The video captured by this camera is quite noisy, compared with more expensive cameras, though this proved useful for testing the robustness of our system. We filtered out most visible light by fitting the lens with a filter comprising several layers of developed, fully-exposed colour photographic negative. We found it unnecessary to remove the camera’s internal infrared filter. The filtering, combined with appropriate adjustments of the brightness, contrast and exposure settings of the camera, allowed the raw video image to be completely

black, with the infrared LEDs appearing as bright white points of light. Consequently the image processing task is simplified considerably.

The requirement of the user to wear a special pair of spectacles may appear undesirable when compared to systems which use traditional image processing to detect facial features. However, the advantage of being a robust, accurate and low-cost system which is independent of individual facial variations, plus the elimination of any training or calibration procedures can outweigh any inconvenience caused by wearing special spectacles.

Furthermore, the LEDs and batteries could be mounted on any pair of spectacles, headset, helmet, cap or other head-mounted accessory, provided that the geometry of the LEDs is entered into the system.

3 Processing

The data processing involved in our system comprises two stages: 1) determining the two-dimensional LED image blob coordinates, and 2) the projection of the two-dimensional points into three-dimensional space to derive the real-world locations of the LEDs in relation to the camera.

3.1 Blob Tracking

Figure 2(a) shows an example raw video image of the infrared LEDs which appear as three white blobs on a black background.

The individual blobs are detected by scanning the image for contiguous regions of pixels over an adjustable brightness threshold. Initially, we converted the blobs to coordinates simply by calculating the centre of the bounding-box; however the sensitivity of the three-dimensional transformations to even single-pixel changes proved this method to be unstable and inaccurate. Consequently we adopted a more accurate method — calculating the centroid of the area using the intensity-based weighted average of the pixel coordinates, as illustrated in Figure 2(b). This method provides a surprisingly high level of accuracy even with low-resolution input and distant LEDs.

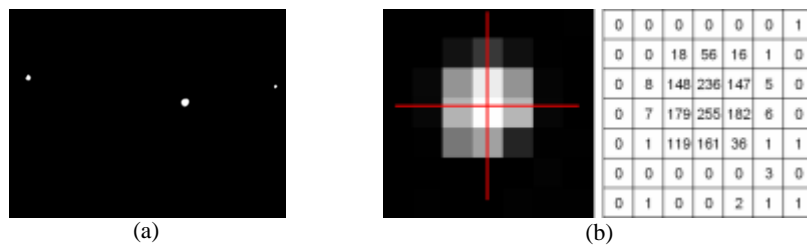


Figure 2. (a) Raw video input (showing the infrared LEDs at close range — 200mm). (b) Example LED blob (with centroid marked) and corresponding intensity data.

3.2 Head-Pose Calculation

Once the two-dimensional blob coordinates have been calculated, the points must be projected back into three-dimensional space in order to recover the original LED positions. This problem is not straightforward. Figure 3 illustrates the configuration of the problem. The camera centre (C) is the origin of the coordinate system, and it is assumed to be facing directly down the z -axis. The ‘gaze’ of the user is projected onto a ‘virtual screen’ which is also centred on the z -axis and perpendicular to it. The dimensions and z -translation of the virtual screen are controllable parameters and do not necessarily have to correspond with a physical computer screen, particularly for blind users and virtual reality applications. In fact, the virtual screen can be easily transformed to any size, shape, position or orientation relative to the camera. Figure 3 also displays the two-dimensional image plane, scaled for greater visibility. The focal length (z) of the camera is required to perform the three-dimensional calculations. The LED points are labelled L , R and F (left, right and front respectively, ordered from the camera’s point of view). Their two-dimensional projections onto the image plane are labelled l , r and f . L , R and F must lie on vectors from the origin through their two-dimensional counterparts.

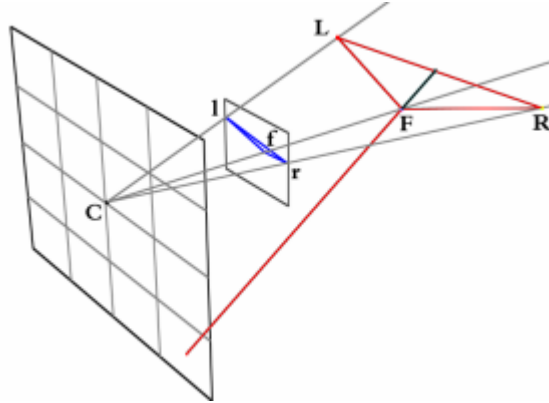


Figure 3. Perspective illustration of the virtual screen (located at the camera centre), the 2D image plane, the 3D LED model and its projected ‘gaze’.

Given our knowledge of the model, we are able to determine exactly where, on the projection rays, the LEDs lie. We know that the front LED is equidistant to the outer LEDs, thus providing Equation 1.

$$d(L, F) = d(R, F) \quad (1)$$

We also know the ratio r between these distances and the baseline distance.

$$d(L, F) = rd(L, R) \quad (2)$$

These constraints are sufficient for determining a single solution orientation for the model. Once the orientation has been calculated, we can also derive the exact physical coordinates of the points, including the depth from the camera, by utilising our model measurements (provided in Section 2).

The distance of the model from the camera is irrelevant for determining the model's orientation, since it can simply be scaled in perspective along the projection vectors. Thus it is feasible to fix one of the points at an arbitrary location along its projection vector, calculate the corresponding coordinates of the other two points, and then scale the solution to its actual size and distance from the camera.

We use parametric equations to solve the problem. Thus the position of point L is expressed as:

$$L_x = tl_x \quad (3a)$$

$$L_y = tl_y \quad (3b)$$

$$L_z = tz \quad (3c)$$

Since z is the focal length, a value of 1 for the parameter t will position L on the image plane.

Thus there are only three unknowns — the three parameters of the LED points on their projection vectors. In fact one of these unknowns is eliminated, since we can fix the location of one of the points — we chose to fix the location of R to be at depth $R_z = z$, thus making its x - and y -coordinates equal to r_x and r_y respectively.

The position of the point F is expressed as:

$$F_z = uz \quad (4c)$$

$$F_x = uf_x \quad (4a)$$

$$F_y = uf_y \quad (4b)$$

Substituting these six parametric coordinate equations for L and F into Equation 1 yields:

$$\sqrt{(tl_x - vi)^2 + (tl_y - vj)^2 + (tz - vz)^2} = \sqrt{(r_x - uf_x)^2 + (r_y - uf_y)^2 + (z - uz)^2} \quad (5)$$

which can be rewritten as:

$$u(t) = \frac{z^2(t^2 - 1) + l_x^2 t^2 + l_y^2 t^2 - r_x^2 - r_y^2}{2(z^2(t - 1) + l_x f_x t + l_y f_y t - r_x f_x - r_y f_y)} \quad (6)$$

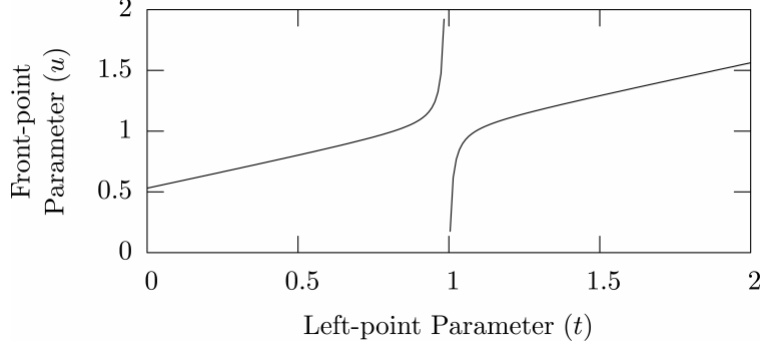


Figure 4. Relationship between parameters t and u

Figure 4 shows a plot of Equation 6. It should be noted that the asymptote is at:

$$t = \frac{r_x f_x + r_y f_y + z^2}{l_x f_x + l_y f_y + z^2} \quad (7)$$

and that the function has a root after the asymptote.

Now we can calculate the point on the front-point projection vector which is equidistant to L and R , given a value for t . Of course, not all of these points are valid — the ratio constraint specified in Equation 2 must be satisfied. Thus we need to also calculate the dimensions of the triangle formed by the three points and find the parameter values for which the ratio matches our model.

The baseline distance of the triangle is given by Equation 8 and plotted in Figure 5.

$$b(t) = \sqrt{(r_x - tl_x)^2 + (r_y - tl_y)^2 + (z - tz)^2} \quad (8)$$

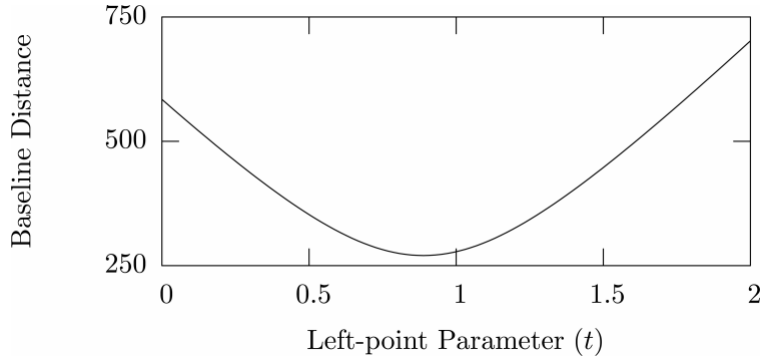


Figure 5. Triangle Baseline Distance

The height of the triangle is given by:

$$h(t) = \sqrt{((u(t)f_x - tl_x)^2 + (u(t)f_y - tl_y)^2 + (u(t)z - tz)^2) - (b(t)/2)^2} \quad (9)$$

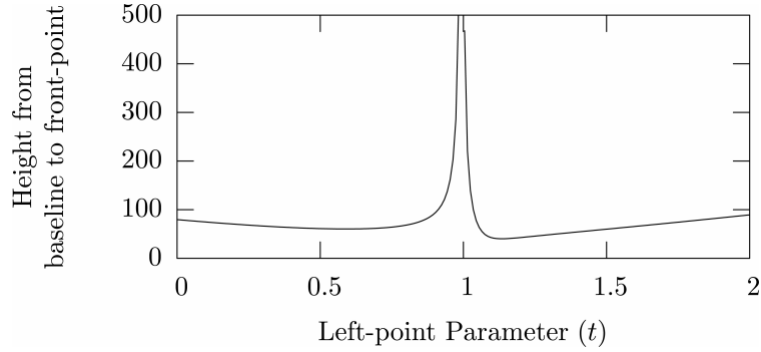


Figure 6. Triangle Height

Figure 6 shows a plot of Equation 9. It should be noted that this function, since it is dependent on $u(t)$, shares the asymptote defined in Equation 7.

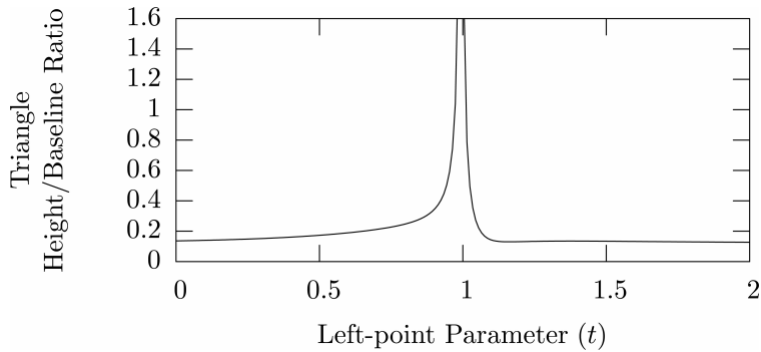


Figure 7. Triangle Height/Baseline Ratio

At this stage we are not interested in the actual baseline distance or height of the triangle — only their relationship. Figure 7 shows a plot of $h(t)/b(t)$. The function has a near-invisible ‘hump’ just after it reaches its minimum value after the asymptote (around $t=1.4$ in this case). This graph holds the key to our solution, and can tell us the value of t for which the triangle has a ratio which matches our model. Unfortunately, it is too complex to be analytically inverted, so we must resort to root-approximation techniques to find the solution. Thankfully, we can reduce the solution range by noting two more constraints inherent in our problem.

Firstly, we know that we are looking for a solution in which the head is facing *toward* the camera. Rearward facing solutions are considered to be invalid as the user’s head would obscure the LEDs. Thus we can add the constraint that:

$$F_z < M_z \quad (10)$$

where M is the midpoint of line LR . This can be restated as:

$$u(t)f_z < (tl_z + z)/2 \quad (11)$$

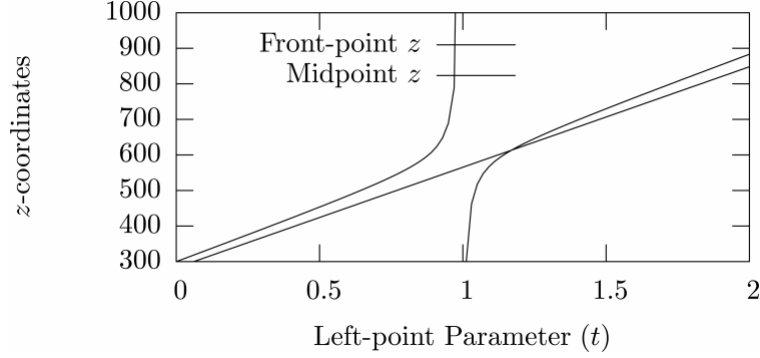


Figure 8. z -coordinates of F and M

Figure 8 shows the behaviour of the z -coordinates of F and M as t varies. It can be seen that Equation 10 holds true only between the asymptote and the intersection of the two functions. Thus these points form the limits of the values for t which are of interest. The lower-limit allows us to ignore all values of t less than the asymptote, while the upper-limit crops the ratio function nicely to avoid problems with its 'hump'. Hence we now have a nicely behaved, continuous piece of curve on which to perform our root approximation.

The domain could be further restricted by noting that not only rearward-facing solutions are invalid, but also solutions beyond the rotational range of the LED configuration; that is, the point at which the front LED would occlude one of the outer LEDs. Our prototype LED configuration allows rotation (panning) of approximately 58° to either side before this occurs.

The upper-limit (intersection between the F_z and M_z functions) can be expressed as:

$$t \leq \frac{-S - \sqrt{-4(-l_x^2 - l_y^2 + l_x f_x + l_y f_y)(r_x^2 + r_y^2 - r_x f_x - r_y f_y) + S^2}}{2(-l_x^2 - l_y^2 + l_x f_x + l_y f_y)} \quad (12)$$

where $S = f_x(l_x - r_x) + f_y(l_y - r_y)$.

Note that this value is undefined if l_x and l_y are both zero (l is at the origin) or one of them is zero and the other is equal to the corresponding f coordinate.

This follows from the degeneracy of the parametric equations which occurs when the projection of one of the control points lies on one or both of the x - and y -axes. Rather than explicitly detecting this problem and solving a simpler equation for the specific case we have chosen instead to jitter all two-dimensional coordinates by a very small amount so that they will never lie on the axes.

We have determined that the lower-limit is bounded by the asymptote; however we can actually further restrict the domain by noting that all parameters should be positive so that the points cannot appear behind the camera. Note that the positive root of Equation 6 (illustrated in Figure 4) is after the asymptote. Since u must be

positive, we can use this root as the new lower-limit for t . Thus the lower-limit is now:

$$t \geq \sqrt{\frac{r_x^2 + r_y^2 + z^2}{l_x^2 + l_y^2 + z^2}} \quad (13)$$

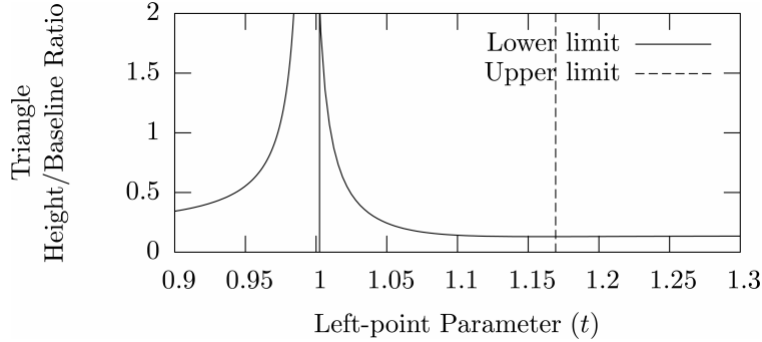


Figure 9. Triangle ratio graph with limits displayed

Figure 9 illustrates the upper and lower limits for root-approximation in finding the value of t for which the triangle ratio matches the model geometry.

Once t has been approximated, u can be easily derived using Equation 6, and these parameter values substituted into the parametric coordinate equations for L and F . Thus the orientation has been derived. Now we can simply scale the solution to the appropriate size using the dimensions of our model. This provides accurate three-dimensional coordinates for the model in relation to the camera. Thus the user's 'gaze' (based on head-orientation) can be projected onto a 'virtual screen' positioned relative to the camera.

4 Experimental Results

Even using as crude a method of root-approximation as the bisection method, our prototype system implemented in C++ on a 1.3GHz Pentium processor took less than a microsecond to perform the entire three-dimensional transformation, from two-dimensional coordinates to three-dimensional head-pose coordinates. The t parameter was approximated to ten decimal place precision, in approximately thirty bisection approximation iterations.

To test the accuracy of the system, the camera was mounted in the centre of a piece of board measuring 800mm x 600mm. A laser-pointer was mounted just below the centre LED position to indicate the 'gaze' position on the board. The system was tested over a number of different distances, orientations and video resolutions. The accuracy was monitored over many frames in order to measure the system's response to noise introduced by the dynamic camera image. Table 1 and Figure 10 report the variation in calculated 'gaze' x - and y -coordinates when the

position of the spectacles remained static. Note that this variation increases as the LEDs are moved further from the camera, because the resolution effectively drops as the blobs become smaller (see Table 2). This problem could be avoided by using a camera with optical zoom capability providing the varying focal length could be determined.

Resolution	320x240 pixels				640x480 pixels			
	500	1000	1500	2000	500	1000	1500	2000
<i>Avg. x-error</i>	0.09°	0.29°	0.36°	1.33°	0.08°	0.23°	0.31°	0.98°
<i>Max. x-error</i>	0.13°	0.40°	0.57°	2.15°	0.12°	0.34°	0.46°	1.43°
<i>Avg. y-error</i>	0.14°	0.32°	0.46°	2.01°	0.10°	0.20°	0.38°	1.46°
<i>Max. y-error</i>	0.22°	0.46°	0.69°	2.86°	0.15°	0.29°	0.54°	2.15°

Table 1. Horizontal and vertical ‘gaze’ angle (degrees) resolution

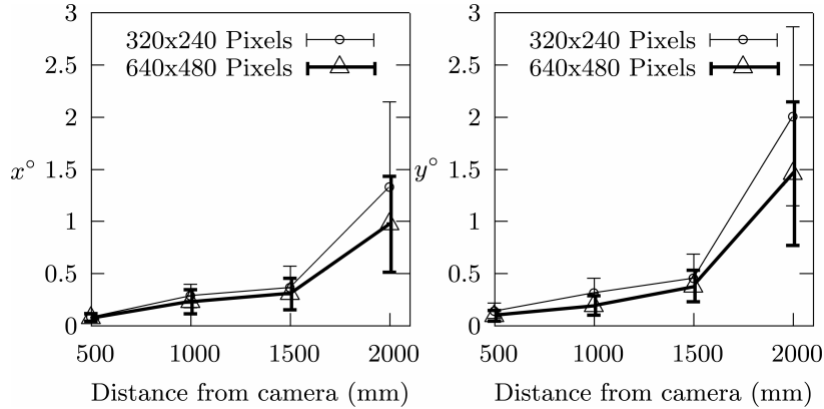


Figure 10. Horizontal and vertical ‘gaze’ angle (degrees) resolution graphs

	500mm	1000mm	1500mm	2000mm
640x480 pixels	20	13	10	8
320x240 pixels	7	5	4	3

Table 2. LED ‘blob’ diameters (pixels) at different resolutions and camera distances

To ascertain the overall accuracy of the system’s ‘gaze’ calculation, the LEDs were aimed at fixed points around the test board using the laser pointer, and the calculated gaze coordinates were compared over a number of repetitions. The test unit’s base position, roll, pitch and yaw were modified slightly between readings to ensure that whilst the laser gaze position was the same between readings, the positions of the LEDs were not. The averages and standard deviations of the coordinate differences were calculated, and found to be no greater than the variations caused by noise reported in Table 1 and Figure 10 at the same distances and resolutions. Consequently it can be deduced that the repeatability accuracy of the sys-

tem is approximately equal to, and limited by, the noise introduced by the sensing device.

As an additional accuracy measure, the system's depth resolution was measured at a range of distances from the camera. As with the 'gaze' resolution, the depth resolution was limited by the video noise. In each case, the spectacles faced directly toward the camera. These results are tabulated in Table 3.

Distance from Camera	500mm	1000mm	1500mm	2000mm
<i>Accuracy at 320x240 pixels</i>	±0.3mm	±2mm	±5mm	±15mm
<i>Accuracy at 640x480 pixels</i>	±0.15mm	±1.5mm	±3mm	±10mm

Table 3. Distance from Camera Calculation Resolution

5 Conclusion

The experimental results demonstrate that the proposed LED-based head-pose tracking system is very accurate considering the quality of the camera used for the experiments. At typical computer operating distances the accuracy is within 0.5 degrees using an inexpensive USB camera. If longer range or higher accuracy is required a higher quality camera could be employed. The computational cost is also extremely low, at less than one microsecond processing time per frame on an average personal computer for the entire three-dimensional calculation. The system can therefore easily keep up with whatever frame rate the video camera is able to deliver. The system is independent of the varying facial features of different users, needs no calibration and is immune to changes in illumination. It even works in complete darkness. This is particularly useful for human-computer interface applications involving blind users as they have little need to turn on the room lights. Other applications include scroll control of head mounted virtual reality displays or any application where the head position and orientation is to be monitored.

6 Acknowledgements

Equations 6, 7, 12 and 13 were derived with the assistance of the Mathematica (Wolfram 2006) software package.

7 References

- Foursa, M. (2004) Real-time infrared tracking system for virtual environments. In Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry, pages 427–430, New York, USA. ACM Press.

- Foxlin, E., Altshuler, Y., Naimark, L. and Harrington, M. (2004) FlightTracker: A novel optical/inertial tracker for cockpit enhanced vision. In ISMAR '04: Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'04), pages 212–221, Washington, DC, USA. IEEE Computer Society.
- Hong, S.K. and Park, C.G. (2005) A 3d motion and structure estimation algorithm for optical head tracker system. In American Institute of Aeronautics and Astronautics: Guidance, Navigation, and Control Conference and Exhibit.
- Horprasert, T., Yacoob, Y. and Davis, L.S. (1996) Computing 3-d head orientation from a monocular image sequence. In Proceedings of the Second International Conference on Automatic Face and Gesture Recognition, pages 242–247.
- Kaminski, J.Y., Teicher, M., Knaan, D., and Shavit, A. (2006) Head orientation and gaze detection from a single image. In Proceedings of International Conference Of Computer Vision Theory And Applications.
- Logitech (2006) QuickCam Express. <http://www.logitech.com>.
- NaturalPoint Inc. (2006) TrackIR. <http://www.naturalpoint.com/trackir>.
- Newman, R., Matsumoto, Y., Rougeaux, S. and Zelinsky, A. (2000) Real-time stereo tracking for head pose and gaze estimation. In Proceedings. Fourth IEEE International Conference on Automatic Face and Gesture Recognition, pages 122–128.
- Wolfram Research Inc. (2006) Mathematica 5.2. <http://www.wolfram.com>.
- Zhu, Z. and Ji, Q. (2004) Real time 3d face pose tracking from an uncalibrated camera. In First IEEE Workshop on Face Processing in Video, in conjunction with IEEE International Conference on Computer Vision and Pattern Recognition, Washington DC.