A 6-D Mouse for Virtual Environments

Dominic Laberge, Jean-François Lapointe *, Emil M. Petriu †, Pierre Boulanger ‡

Abstract

In this paper, using new algorithms developed in the field of augmented reality and registration, we investigate further means of extracting three dimensional positioning and orientation information from a device created at Bell Laboratories [7]. We also made some modifications to the device in order to adapt it for use in the context of a virtual reality workbench.

1 Introduction

In order to interact with virtual environments (VE), users often need to interactively position and orient objects or camera views. In order to do so, ,a lot of 6 degrees of freedom (DOF) input devices have been realized such as face trackers [6, 9, 13], nose trackers [5], hand trackers [11, 12, 14], magnetic trackers [2, 4] and other devices targeted more at the home user [1, 3]

Most of the 6 DOF input devices used to date are either expensive or cumbersome because they need a physical link such as a wire to be activated. One exception to that is a 6DOF vision based mouse developed by Kumar[7]. This mouse uses a LED pattern (Figure 1.1) tracked by a video camera located over the user. This paper focuses on improvements to the design of the and controlled device and to the pose estimation algorithm.

Kumar used a over-determined four points auto-calibrated system to track his mouse. He then used known properties of the pattern to infer position and orientation. We favored an approach that was more general in the sense that it can be effortlessly adapted to other patterns, and at the same time, integrated, calculating all at once to lower computational cost.

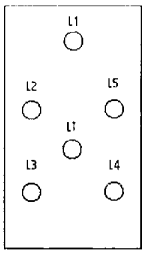

Figure 1.1: Kumar's LED pattern

2 Mouse modifications

We only made a few changes to Kumar's mouse, mainly to adapt it to the setting of a virtual reality workbench. When a user is operating in a virtual world, any visual cues from the real world can contribute to break the state of immersion. That is why we chose to move the light emitting diodes (LEDs) from top to the side of the device. That way, most of the time, the user won't see light in his field of view which would cause distraction from the task at hand.

Also we covered the device with black material to make it harder for human eyes to detect it in a dark environment. It is then convenient for the user and other persons in the room that will not see the device as well.

We kept the same pattern because qualification of targets doesn't pose any problem. We used micro-

*Institute for Information Technology, National Research Council of Canada, Ottawa, Canada

†School of Information Technology and Engineering, University of Ottawa, Ottawa, Canada

‡Department of Computing Science, University of Alberta, Edmonton, Canada

0-7803-7635-8/02/$17.00 © 2002 IEEE

lamps instead of LEDs in this particular implementation because we had them on hand.

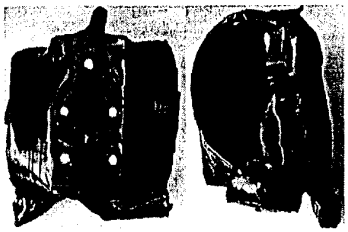

Figure 2.1: **T**ridimensional **A**ctive **P**attern **E**nhanced Mouse

3 Target acquisition

The first step before pose estimation is to detect the targets inside the camera image. To achieve that, a monochrome camera is used and the high contrast in the image is sufficient for target detection. To detect the targets that define the LEDs, we devised a method that makes use of the high contrast and the shape of the LEDs themselves.

The idea is to find the center of all the potential targets by parsing the image only twice, once horizontally and once vertically.

First we threshold the image to find potential target pixels. For that we use a pixel intensity close to the maximum since LEDs directly pointing at the camera are likely to saturate CCD cells. Then we scan horizontally line by line and find the center of all sequences of potential target pixels. Then we do the same scan vertically. The centers identified are all potential target centers because they are located in the middle of a sequence of white pixels. Finding the intersections of the lines then created Using the fact that LEDs are round shaped and that the exposed part is spherical, the assumption that, within the range of operation, an oval or circular chape will be saturated on the CCD is confirmed experimentally. (Figure 3.1).

This method not only gives us the target position, but also the pixelwise centroid thus allowing potential sub-pixel target detection to increase accuracy.

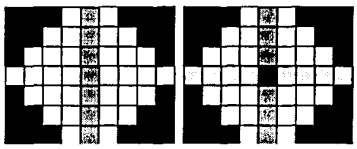

Figure 3.1: Target centroid detection.

The approach followed to identify the targets consists in combining the cross-ratio method proposed in [7] and a sort by cross-products. First, LED L_1 is identified by analyzing cross-ratio of angles. The other targets are then sorted in counter clockwise order.

4 Pose estimation

Once the targets are identified, comes the pose estimation. When using video cameras, two methods are generally used: stereo vision and single camera auto-calibration. We chose the later because having two video inputs from two video cameras imply two things. One, accuracy can be improved by using disparity between two cameras images rather than noisy focal length calculation that comes with auto-calibration. But two, we found that some of that improvement is sheared of by doubling the lens distortion factor. A non-negligible factor in lower end cameras. We also have to take into account that calculations will have to be done on twice as much pixel data, hindering the overall performance.

Another way to evaluate the pose of the mouse is to use auto-calibrated registration as proposed in [8]. In this method we only need four targets, we'll use $L_2..L_4$. L_1 is only used to identify other targets.

4.1 Finding the homography

Once we know the position of $L_2..L_4$ (in pixels) of each target on the camera plane, we can build a set of linear equations that can be solved to find the homography. The homography is a $3X3$

matrix, that we call H, that mathematically defines the projection of each target onto the camera plane. Thus, we have to find the 9 parameters $\{h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33}\}$.

In [8] it is shown that for every point we get two equations. In matrix form, $\forall i$:

$$\begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & x'_i x_i & x'_i y_i & x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & y'_i x_i & y'_i y_i & y'_i \end{pmatrix} H = 0 \tag{4.1}$$

Where x_i and y_i are the coordinates of the i^{th} target in pattern space and x'_i and y'_i its coordinates in pixel on the image.

Using only four targets out of five yields a situation where there are 9 unknowns but 8 equations. But, we can safely assume that the scaling parameter h_{33} is 1. This gives us a 9th equation in matrix form:

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} H = 1 \tag{4.2}$$

With this we have a fully defined set of 9 linear equations with 9 unknowns that we can solve with a standard method such as *LU Decomposition* (LUD) or *Singular Value Decomposition*(SVD) from [10]. LUD was used in the implementation.

4.2 Position estimation

Now that we have matrix H we can consider finding position and orientation of the mouse pattern in 3D space. Following [8], we can derive the rotation matrix and translation vector from the homography. For that we'll need focal lengths along both optical axis (f_u and f_v) and a scaling parameter λ. They can be derived from the homography matrix as follows:

$$f_u = \sqrt{\frac{h_{11}h_{12}(h_{21}^2 - h_{22}^2) - h_{21}h_{22}(h_{11}^2 - h_{12}^2)}{-h_{31}h_{32}(h_{21}^2 - h_{22}^2) + h_{21}h_{22}(h_{31}^2 - h_{32}^2)}} \tag{4.3}$$

$$f_v = \sqrt{\frac{h_{11}h_{12}(h_{21}^2 - h_{22}^2) - h_{21}h_{22}(h_{11}^2 - h_{12}^2)}{-h_{31}h_{32}(h_{11}^2 - h_{12}^2) + h_{11}h_{12}(h_{31}^2 - h_{32}^2)}} \tag{4.4}$$

$$\lambda = \frac{1}{\sqrt{h_{11}^2/f_u^2 + h_{21}^2/f_u^2 + h_{31}^2}} \tag{4.5}$$

Translations along X (t_x), $Y(t_y)$ and $Z(t_z)$ can then be computed directly:

$$t_x = \frac{\lambda h_{13}}{f_u} \tag{4.6}$$

$$t_y = \frac{\lambda h_{23}}{f_v} \tag{4.7}$$

$$t_x = \lambda h_{33} \tag{4.8}$$

4.3 Orientation estimation

Here two options were available. Calculating position in X, Y and Z for each target would enable us to find orientation with knowledge of some properties in the pattern like what's done in [7]. Or we could go a step further than in [8] and get individual angles where only a general rotation matrix is provided. We chose the later option for the sake of saving a couple operations.

Let $\theta_1, \theta_2\theta_3$ be respectively the angle of rotation around the axes X_c, Y_c and Z_c. Rotations along each axis are defined with the following $3X3$ matrices:

$$R_X = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_1 & -\sin\theta_1 \\ 0 & \sin\theta_1 & \cos\theta_1 \end{pmatrix}$$

$$R_Y = \begin{pmatrix} \cos\theta_2 & 0 & -\sin\theta_2 \\ 0 & 1 & 0 \\ \sin\theta_2 & 0 & \cos\theta_2 \end{pmatrix}$$

$$R_Z = \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & 0 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In order to simplify writing, let:

$$\begin{array}{ll} a = \sin\theta_1 & b = \cos\theta_1 \\ c = \sin\theta_2 & d = \cos\theta_2 \\ e = \sin\theta_3 & f = \cos\theta_3 \end{array}$$

The order in which rotations occur are irrelevant because the center of rotation remains the same. Therefore we can safely say that the final rotation matrix R is formed as following:

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & b & -a \\ 0 & a & b \end{pmatrix} \cdot \begin{pmatrix} d & 0 & -c \\ 0 & 1 & 0 \\ c & 0 & d \end{pmatrix} \cdot \begin{pmatrix} f & -e & 0 \\ e & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.9}$$

Solving equation 4.9 we obtain:

$$\begin{pmatrix} df & -de & -c \\ -acf + eb & eac + fb & -ad \\ bcf + ae & -bce + af & bd \end{pmatrix} \tag{4.10}$$

From there we see that:

$$r_{13} = -c = -\sin\theta_2 \tag{4.11}$$

thus

$$\theta_2 = \arcsin(-r_{13}) \tag{4.12}$$

From there we can derive θ_1 from r_{23} and θ_3 from r_{12} :

$$\theta_1 = \arcsin\left(\frac{-r_{23}}{\cos\theta_2}\right) \tag{4.13}$$

$$\theta_3 = \arcsin\left(\frac{-r_{12}}{\cos\theta_2}\right) \tag{4.14}$$

Next, from [8] we know that

$$R = \begin{pmatrix} \lambda h_{11}/f_u & \lambda h_{12}/f_u & r_{21}r_{32} - r_{31}r_{22} \\ \lambda h_{21}/f_v & \lambda h_{22}/f_v & r_{31}r_{12} - r_{11}r_{32} \\ \lambda h_{31} & \lambda h_{32} & r_{11}r_{22} - r_{21}r_{12} \end{pmatrix} \tag{4.15}$$

then we can derive equations that define rotation from the homography matrix. First using equation 4.15 in equation 4.13:

$$\theta_1 = \arcsin\left(-\frac{\lambda^2}{f_u\cos\theta_2}(h_{31}h_{12} - h_{11}h_{32})\right) \tag{4.16}$$

then using equation 4.15 in equation 4.12:

$$\theta_2 = \arcsin\left(-\frac{\lambda^2}{f_v}(h_{21}h_{32} - h_{31}h_{22})\right) \tag{4.17}$$

finally using equation 4.15 in equation 4.14:

$$\theta_3 = \arcsin\left(-\frac{\lambda^2 h_{12}}{f_u\cos\theta_2}\right) \tag{4.18}$$

5 Experimental results

In order to measure the accuracy, we created an experimental table. For two different depth coordinates, we took measures given by the algorithm and compared them to reality. We obtained the following accuracy results:

Parameter	Z=30cm	Z=70cm
Working area ($W \times H$)	$20cm \times 10cm$	$50cm \times 35cm$
X accuracy	$\pm 0.7cm$	$\pm 1.4cm$
Y accuracy	$\pm 0.3cm$	$\pm 1.3cm$
Z accuracy	$\pm 6.0cm$	$\pm 2.2cm$
θ_1 accuracy	$\pm 23°$	$\pm 9°$
θ_2 accuracy	$\pm 11°$	$\pm 9°$
θ_3 accuracy	$\pm 3°$	$\pm 7°$

Table 1: Experimental accuracy results

6 Discussion

In this paper we presented an innovative way of combining a vision-based low cost input device with an auto-calibrated pose estimation algorithm.

Experiments show that at 30 cm from the camera, the workable area is roughly similar in size to a standard mouse pad. This size is fine if we introduce mouse *clutching* with the trigger button but it is small for having a static coordinate to action ratio.

The error in angle can be attributed to noise in the input at close range. One pixel offset does not mean much in terms of absolute position because it is averaged out, but when calculating angle, every pixel counts. When the active pattern moves away from the camera, targets are smaller and the distance between the edge of the blob and the center is smaller enabling a tighter centroid detection yielding tighter angle detection at the same time. On the other hand, on average, one pixel offset will mean more in absolute position, shown here by a slight loss in position accuracy.

We also noticed that shadows of people moving, even in a semi-dark room can hinder detection.

7 Future work

With the options of auto-calibration and stereo vision opened, extending initial idea and work by Kumar, we will further test precision and accuracy of this system on three different algorithms. We will also experiment on the limitations of the device in terms of distance form the camera.

One big problem with this mouse is sensitivity to light. The next step in that direction is to test new ways of detecting targets, or perhaps infrared targets or polarized light with filters to narrow the possibility of targets and reduce false detections.

We will also attach the device to a glove to track finger position as well .

References

[1] 3d connection web site, http://www.3dconnexion.com/.

[2] Ascension technology corporation web site, http://www.ascension-tech.com/.

[3] Cymouse website http://www.cymouse.com/.

[4] Polhemus web site, http://www.polhemus.com.

[5] D. Gorodnichy, S. Malik, and G. Roth. Nouse 'use your nose as a mouse' - a new technology for hands-free games and interfaces. In *Proceedings of the 15th International Conference on Vision Interface*, Calgary, Canada, May 2002. CD-ROM Only.

[6] T. Jebara and A. Pentland. Parametrized structure from motion for 3d adaptive feedback tracking of faces. In *Proceedings of Computer Vision and Pattern Recognition*, 1997.

[7] S. Kumar. 6d-mouse: A vision-based computer input device. Technical report, Bell Laboratories, 2000. http://www.bell-labs.com/user/senthil.

[8] S. Malik. Robust registration of virtual objects for real-time augmented reality. Master's thesis, Carleton University, 2002.

[9] N. Oliver, A. Pentland, and F. Berard. Lafter: Lips and face real time tracker. In *Proc. Computer Vision and Patt. Recog.*, 1997.

[10] W. H. Press, W. T. Vettering, S. A. Teukolsky, , and B. P. Flannery. *Numerical Recipes in C++*. Cambridge University Press, Cambridge, 2002.

[11] J. M. Rehg and T. Kanade. DigitEyes: Vision-Based Human Hand Tracking. Technical Report CMU-CS-93-220, December 1993.

[12] Y. Sato, M. Saito, and H. Koike. Real-time input of 3d pose and gestures of a user's hand and its applications for hci. In *2000 IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000)*, pages 462–467, March 2000.

[13] K. Toyama. Look, ma — no hands!' hands-free cursor control with real-time 3d face tracking. In *Proc. Workshop on Perceptual User Interfaces (PUI'98)*, pages pp. 49–54, San Francisco, CA, November 1998.

[14] D. Wei and L. Hua. Vision based gesture recognition using just one camera. In *Proceedings of ICSP 2000*, volume 2, pages 1351–1357, Beijing, China, 2000.

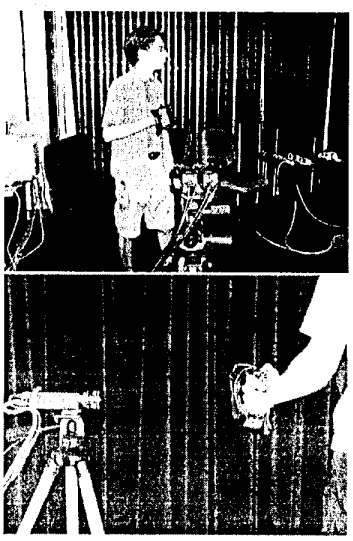

Figure 7.1: Stereo and mono experiments.