Fast Multiple Camera Head Pose Tracking

Ravikrishna Ruddarraju†, Antonio Haro‡, Irfan A. Essa†‡,

†School of Electrical and Computer Engineering and ‡GVU Center, College of Computing
Georgia Institute of Technology, Atlanta, Georgia 30332. USA.
ravigtri@cc.gatech.edu, haro@cc.gatech.edu, irfan@cc.gatech.edu

Abstract

This paper presents a multiple camera system to determine the head pose of people in an indoor setting. Our approach extends current eye tracking techniques from a single camera system to a multiple camera system. The head pose of a person is determined by triangulating multiple facial features that are obtained in real-time from eye trackers. Our work is unique in that it allows us to observe user head orientation in real-time using several cameras over a much larger space than covered by a single camera. We demonstrate the viability of this system by experimenting with several people under different lighting conditions performing head movements.

Keywords: Computer vision, eye tracking, multiple camera systems.

1 Introduction

The head pose of a person in a pervasive computing environment is very important for human computer interaction. For example, if the general direction of a user's gaze is known, appliances can determine that eye contact has been established or data can be moved to a display device nearer to the user's line of sight. The goal of our work is to estimate user head pose non-invasively and robustly in real-time. Our system uses a scalable number of cameras mounted in a room to estimate the head pose of users as they move around in the viewing volume of the cameras. Our approach uses three IBM Blue Eyes cameras for input [8], and tracks eyes using the method described by Haro *et al.* [5]. We extended the underlying approach proposed in [5] to perform robust real-time head pose estimation in indoor environments and allow for extension to multiple cameras.

The advantage of this system is that it is robust under decent indoor lighting, is non-invasive and real-time compared to existing commercial systems. The robustness of the eye tracking allows us to use eyes as a very reliable low-level feature to perform higher level processing and tracking. For example, we use the location of the eyes to very reliably find mouth corners in real-time. Such a task would be very hard and prone to error if only edges or intensity values were used as the primary features.

Our approach uses the tracked eye locations and mouth corners as low-level features in estimating head pose. These four facial features from all cameras are used to robustly determine the pose of a person by using a combination of stereo triangulation, an interpolation technique, and an algorithm to switch between subsets of multiple cameras for better tracking.

We obtained results that show that our system gives stable estimates of pose and captures fine variation as a person moves. The results also show the effectiveness of the switching algorithm in terms of the temporal continuity of the tracking.

2 Previous Work

Several researchers have used tracked head pose as part of their research in studying eye movements. Aaltonen *et al.* [1] used eye and head pose tracking in basic PC interface tasks. Goldberg *et al.* [4] used eye movements to infer user intent in real-time interfaces. Both Aaltonen's and Goldberg's systems use wearable hardware for tracking head pose and eye movements. Harville *et al.* [6] used linear depth and brightness constraints along with twist mathematics to obtain 3D head pose. Matsumoto *et al.* [9] used a compact 3D face model to estimate head pose. As in [9], Schödl *et al.* [10], and Cascia *et al.* [2] use more complex polygonal head models for tracking head pose.

Our work significantly differs from the head pose tracking performed in these projects in several ways. First, our work differs from that of Goldberg *et al.* and Aaltonen *et al.* in that our system is completely non-invasive. Unlike the work of Matsumoto *et al.* and Schödl *et al.*, our algorithms are simple enough to run well on consumer-level computers without any

need for special purpose hardware. Our framework also differs from prior work in that tracking is not limited to the viewing range of a single camera alone. We can make use of multiple cameras to estimate head pose in a much larger viewing volume.

Our algorithm also has the advantage in that it works robustly in real-time. The simplicity of the algorithm shields our method from having to use error functions that increase the complexity of other direct-measurement and optimization-based methods. The most unique feature of our approach is being able to track using multiple cameras. This ability to track head pose using multiple cameras is very important for larger settings where the mobility of users is significant.

3 Setup

The overall tracking system consists of three calibrated IBM Blue Eyes infrared lighting cameras [8]. Though we have not determined the scalability limits, we have experimented with up to six cameras. The cameras are placed in one of the rooms at Georgia Tech's Broadband Institute Residential Laboratory, which is prone to natural changes in lighting conditions. All cameras are calibrated individually using the method described by Zhang [12] to obtain the intrinsic and extrinsic matrices. We use one computer with a 600Mhz Intel Celeron processor per camera for feature tracking. A separate computer with a 500Mhz PowerPC G4 processor takes the input from the others, manages cameras, and executes the pose estimation algorithms.

One of the cameras is chosen as the origin for the camera coordinate system. All pose measurements are interpreted with respect to this origin. Prior information about the relative positions of all cameras is provided to the system before tracking is performed. Our experimental subject pool consisted of five adults, four male and one female. One of our camera setups is shown in Figure 1. The camera in middle, *camera0*, is the origin of the entire system. *camera1* on the left and *camera2* on the right are turned 25 degrees towards the origin of the system, (i.e. *camera0*).

4 Head Pose Estimation

We calculate head pose by triangulating corresponding features from two cameras in a temporally coherent manner. Though our system employs multiple cameras, a set of two cameras is sufficient for determining the head pose. In the simplest sense, two corresponding features of a face from two different cameras can yield the head pose.

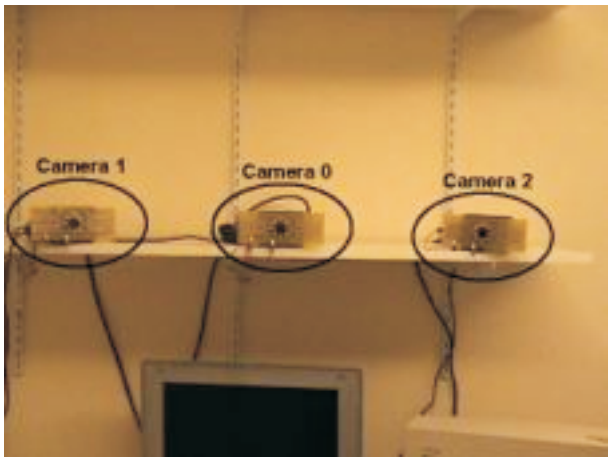

Figure 1: Camera setup. From left to right, *camera1*, *camera0*, and *camera2*.

We assume that the world coordinates of two eyes of a person are given by $P : (X, Y, Z)$ and $P' : (X', Y', Z')$. The pixel coordinates of the two eyes are obtained using the eye tracking sub-system. Then, let $(x_1, y_1), (x_1', y_1')$ be the pixel coordinates of these two points in *camera1* and $(x_2, y_2), (x_2', y_2')$ be the pixel coordinates of the two points in *camera2*. The relationship between the real world coordinates and the pixel coordinates in the two cameras is given by the following equations where f is the focal length of camera and d is the distance between the two cameras:

$$Z = \frac{f * d}{x_1 - x_2} \tag{1}$$

$$X = x_1 * \frac{Z}{f} \tag{2}$$

$$Y = y_1 * \frac{Z}{f} \tag{3}$$

Once both P and P' are calculated, the head pose is obtained from the slope of the line joining these two points.

$$Pose = tan^{-1} \frac{Z - Z'}{X - X'} \tag{4}$$

However, using two feature points alone makes the system highly prone to minor variations in eye movement, resulting in very noisy data. Literature from multi-view stereo [11] also suggests that 3D information about the scene can be better interpreted with a larger set of corresponding points. This is why we track two additional dominant facial features, the mouth corners. We track the corners for two reasons. First, mouth corners have very strong edges, making them easier to locate than non-textured features, such as the tip of the nose. Second, knowledge of the

eye location from the eye tracking sub-system makes it easier to narrow down the search space significantly for locating the mouth corners using standard template matching.

Template matching is sufficient since knowledge of the eye locations allows us to roughly estimate the neighborhoods of the mouth corners. We obtained mouth corner templates from a group of persons and empirically chose the pair that generalized best. Though using correlations is not a robust approach, we found that the choice in templates did not significantly change the head pose estimation results. Moreover, the speed gains obtained from correlation were more significant than using alternative methods such as the probabilistic metric proposed by Moghaddam *et al.*[7]. Figure 2 shows all four of the features being tracked on the face of a subject.

Corresponding pixel coordinates from a set of two cameras are used to determine the 3D world coordinates of the four features being tracked. When the cameras are placed in an arbitrary manner, the simple triangulation method described previously cannot be used. 3D points are obtained by using the extrinsic matrices T (translation matrix), R (rotation matrix), and the intrinsic matrix, M, that are the result of the camera calibration process. The standard stereo triangulation method described in [11] is used to obtain these 3D points. Once the world coordinates of the four points are calculated, the relationship between the cross product between any three points can be used to compute the head pose in the x-z plane using points p_1, p_2, p_3:

$$\overrightarrow{v}_{123} = (p_2 - p_1) \times (p_3 - p_1) \quad (5)$$

$$\theta_{123} = tan^{-1} \frac{\overrightarrow{v}_{123}(z)}{\overrightarrow{v}_{123}(x)} \quad (6)$$

In this manner, for four points there are 16 angles $\theta_{123}(x,z), \theta_{234}(x,z), \theta_{341}$.... The average of these angles is computed to obtain the final head pose. Though it is not necessary to compute all 16 angles, all of the angles are used to ensure the robustness of the computed head pose.

Next, the average head pose angle is interpolated in real-time to compensate for noise from the frequent eye motions of the user and camera noise. Second order parabolic interpolation is sufficient for representing local changes in head pose. However, such interpolation cannot represent global variations in the data. Fourth order interpolation is necessary to represent these global changes in pose. Figure 3 shows the head pose calculated when a user turned their head from side to side (-30 degrees to +30 degrees). This interpolation is sufficient for representing one single

Figure 2: Multiple features for camera correspondence.

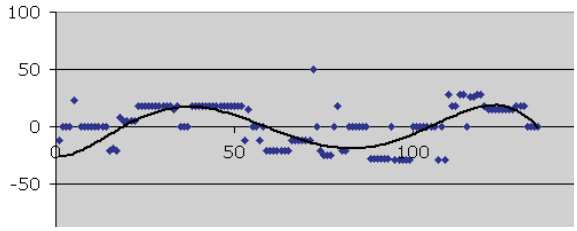

Figure 3: Discrete (dots) and interpolated (curve) averaged angles. Our system uses interpolated angles to estimate head pose. Outliers are rejected at the expense of a smooth temporally coherent estimated head pose.

cycle of head movement. However, concatenated cycles require a higher order of interpolation.

This interpolation not only compensates for data noise, but also significantly improves the quality of head pose calculation by providing the fine variations in pose, which cannot be interpreted by discrete values obtained from the triangulation method alone. Even though the system runs in real time (15 *fps*), our feature tracking is not fast enough to interpret the variation of user attention even when a person moves his or her head at a moderate speed. The smooth variations given by the interpolated curve are necessary for representing these fine-grained changes in user attention. The mean error in all head pose calculations was found to be about $\pm$ 8 degrees with a very low deviation. There are two primary sources for this error. The camera calibration process is one source, but our interpolation mostly nullifies this. The more significant source of error is from the eye tracking sub-system not being able to accurately locate the eyes. As with any tracking system, the eye tracking

sub-system has a modest error rate associated with it. These errors are described in detail in [5] and directly lead to minor errors in angle calculation.

5 Pose with Multiple Cameras

For any practical interactive system, a large viewing volume is desired so that users in it can be tracked as they move around. Multiple cameras afford this ability by providing a large combined viewing volume over which user behavior can be tracked. However, tracking with multiple cameras is a very broad problem that involves issues ranging from system scalability, occlusions, calibration, and camera switching [3]. In our work, since the head pose calculation requires only two cameras, we assume that the pose obtained from any set of two cameras is independent and uncorrelated. Therefore, extending the two-camera head pose tracking to multiple cameras reduces to choosing the best pair of cameras from a group.

Our experimental setup consists of six cameras. All cameras are individually calibrated as mentioned in Section 3, thus adding more cameras does not involve any additional system-wide calibration. Multiple camera systems could possibly coordinate between the cameras to avoid occlusions, but we do not do this because the features we track are very strong and do not have any intra-camera dependence. Moreover, our algorithm performs robustly without considering intra-camera relationships to solve occlusions.

The next most important aspect of the multi-camera version of our system is how to choose the two cameras to compute pose. A feature dependent switching metric that makes the best choice between cameras is an integral part of any multiple camera system [6]. The next section describes our head pose angle based switching metric.

5.1 Switching Metric

When tracking head pose using a single set of cameras, there exists an angular limit of tracked head pose which depends on the features being tracked and the viewing range of the cameras. Since we use four facial features to obtain the pose, the angular limit would be the angle beyond which one or more of these facial features cannot be tracked. Figure 4 shows one such scenario where 3 of the 4 features are not being tracked by one of the cameras. When using multiple cameras, it is possible for more than two cameras to see the same features. In such a case, even if one of the cameras has reached the limiting angle, another camera in the group can possibly still view the features.

Moreover, as the pose angle gets closer to the limiting angle, it is likely to be less accurate. The accuracy of the pose angle also decreases as the user moves further away from the cameras. In such a case, if there is another set of cameras that has a better view of the face, the system should use those cameras to obtain the pose angle. This switching from one camera set to another set is performed using a decision metric. We experimented with two such metrics: one based on dot products, and one based on Gaussian functions.

Figure 4: Only one out of four features is visible from this camera.

5.1.1 Dot Product Metric

We found standard dot products to be a good switching metric. The pose angle and the user's distance obtained from triangulation described in section 3 can be used to define a pose vector. Let $\hat{p}$ be the unit vector defining the head pose, and Z be the magnitude of the person's distance from any camera. Both $\hat{p}$ and Z are defined in the common world coordinate system. The scaled pose vector $\bar{P}$ is then:

$$\bar{P} = Z\hat{p} \tag{7}$$

Similarly, let $\hat{c}$ be the unit vector defining the direction of a camera in the world coordinate system. For example, *camera0* will have $\hat{c} = (1, 0°)$, and *camera1* (turned $+25°$) will have $\hat{c} = (1, 25°)$ etc. Then, the dot product of the pose vector $\bar{P}$ and camera vector $\hat{c}$, will result in a scalar R as given below. Only the magnitude of the dot product matters, as its sign always remains the same in this system.

$$R = |\bar{P} \cdot \hat{c}| \tag{8}$$

For example, let us assume that R_0, R_1, and R_2 are the three scalars resulting from the dot product of the pose vector with the camera vectors of *camera0*, *camera1*, and *camera2*, respectively. To get the best switching, we choose the cameras that give the highest scores.

5.1.2 Gaussian Metric

The Gaussian metric is based on the intuition that a camera cannot accurately track all features as the

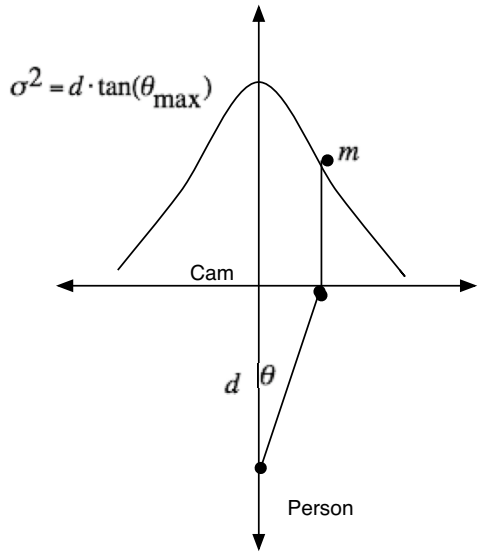

Figure 5: Gaussian when the person is closer to camera, higher score.

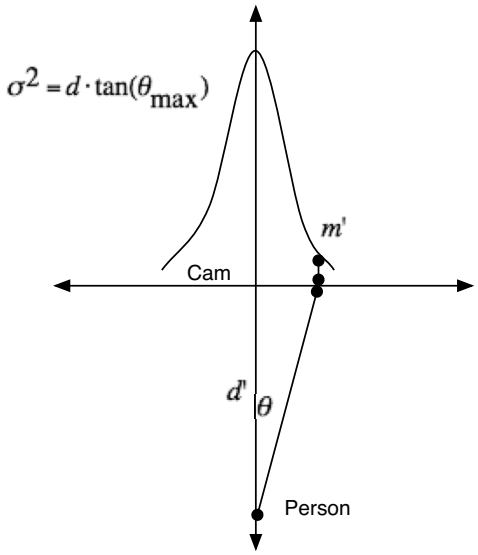

Figure 6: Gaussian when the person is far off from the camera, lower score.

pose gets closer to the limiting angle. The Gaussian metric uses the variance to represent the head distance and angle measurements from each camera.

Figures 5 and 6 show how the Gaussian varies with respect to the subject's distance to the cameras. The variance σ^2 and score m for any pose angle θ is given by the following equation:

$$\sigma^2 = d \cdot tan(\theta_{max}) \tag{9}$$

$$m = \frac{1}{\sigma\sqrt{2\pi}} \cdot exp\frac{(d \cdot tan(\theta))^2}{2\sigma^2} \tag{10}$$

where θ_{max} is the limiting angle, which is chosen to be 45 degrees. So, for the same pose angle θ, if $d^{'} > d$, we obtain $m > m^{'}$. This means that when the person is standing far off, cameras are less likely to track the features accurately.

The effectiveness of both of the metrics was tested based on how often they switched camera pairs for tracking. A good metric is one in which the pairs are switched such that the pose remains temporally coherent as the system switches between cameras. We found that both of the metrics gave similar results. In order to keep the final system as simple as possible, we use the dot product as the camera switching metric.

6 Results

The multiple camera system was tested on five adult subjects, four males and one female using 3 cameras. All of the subjects were instructed to move their head

Figure 7: Experimental setup.

in front of the cameras. Two points were marked on the wall at approximately +30 degrees and −30 degrees in the *x*-*z* plane to indicate specific locations to look at. The angles obtained by the system when users viewed these locations were compared with the real angles to obtain the error. The results show that the system is insensitive to eye size, skin tone, facial hair, and clothing.

Figure 7 shows an example of the real pose of a subject. The subject is looking towards *camera2*; her real pose with respect to the origin (*camera0*) is around 30 degrees. All the cameras track her four facial features. Figure 8 is the estimated head pose obtained after the subject rotated her head about

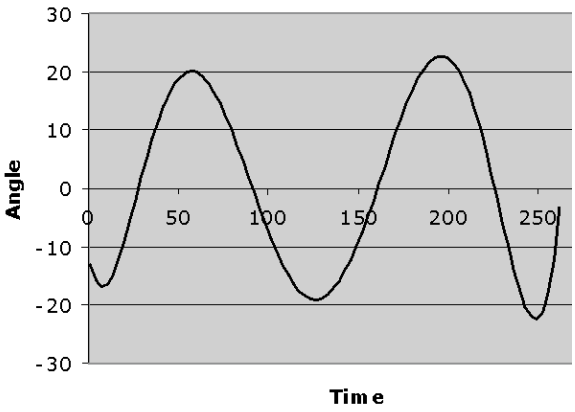

Figure 8: Estimated head motions.

Ground truth	Mean error in degrees
0 degrees	5.125
30 degrees	9.96
-30 degrees	8.545
For all angles	7.87

Table 1: Statistics of pose error for all subjects.

three times. The curve is very smooth because the estimated head pose is based on a large number of measurements and interpolated to estimate accurate and temporally coherent head pose (as discussed in Section 4).

Table 1 shows the statistics of calculated and real pose error for all of the subjects. These values are for the multiple camera version of our system. The average pose error between our subjects is about ± 8 degrees.

The main limitation of the system is varying lighting conditions due to bright sunlight. This is an inherent limitation of all systems using infrared illumination in the presence of ambient infrared light. Significant changes in lighting conditions requires changing

Figure 9: Subject 1: Male with Glasses, evening.

Ground truth	Mean error in degrees
0 degrees	4.3
30 degrees	10.56
-30 degrees	8.36
For all angles	7.74

Table 2: Statistics of pose error for subject 1.

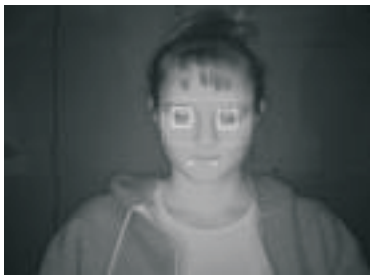

Figure 10: Subject 2: Female, evening.

Ground truth	Mean error in degrees
0 degrees	4.57
30 degrees	9.5
-30 degrees	7.83
For all angles	7.3

Table 3: Statistics of pose error for subject 2.

Figure 11: Subject 3: Male, morning.

Ground truth	Mean error in degrees
0 degrees	6.16
30 degrees	9.43
-30 degrees	8.39
For all angles	7.99

Table 4: Statistics of pose error for subject 3.

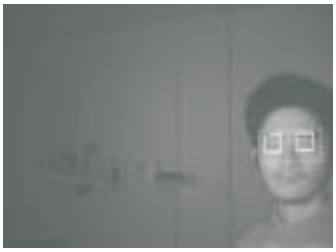

Figure 12: Subject 4: Male, morning.

Ground truth	Mean error in degrees
0 degrees	5.47
30 degrees	10.35
-30 degrees	9.6
For all angles	8.47

Table 5: Statistics of pose error for subject 4.

the brightness gain of the cameras, which could be done adaptively as the day progresses.

Another limitation is caused by the interpolation process described in section 4. One of the standard issues with interpolation is the danger of data overfitting. Our interpolated head pose cannot represent subtle changes in pose that can be obtained from the raw data. However, our system is intended for use in indoor settings, where the ability to decide whether a person has looked at a general object is more important than the minor variations in head pose.

7 Conclusions and Future work

In this paper, we have demonstrated the ability of our system to reliably track head pose using multiple cameras non-invasively and in real-time. We are able to perform robust tracking in a realistic indoor residential setting, over a number of users with a wider coverage area than afforded by a single camera. In addition, our framework does not perform any error function minimization as in some previous head pose approaches [6][9], so we avoid getting stuck in local minima/maxima, yet are able to provide good head pose estimates in real time. The robustness of the eye tracking subsystem combined with the multiple camera framework described in this paper makes our system highly practical.

We plan on making use of the head pose information by combining it with physiological properties of human attention to track the actual focus of attention of users in an interactive environment.

Acknowledgments: We would like to thank Myron Flickner and David Koons from IBM Research Almaden for providing us with the Blue Eyes cameras and related basic software. Funding was provided by the Aware Home Research Initiative, Georgia Tech's Broadband Institute, Intel Corporation, and Georgia Tech President's Undergraduate Research Award Program.

References

[1] A. Aaltonen, A. Hyrskykari, and K. Raiha. 101 spots on how do users read menus? In *Human Factors in Computing Systems: CHI 98.*, pages 132-139, New York: ACM Press, 1998.

[2] La Cascia, M. Sclaroff, S. Athitso. Fast reliable head tracking under varying illumination: An ap-proach based on robust registration of texture mapped 3D models. In *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 2000

[3] X. Chen. Design of many-camera tracking sys-tems for scalability and efficient resource allocation. *Ph.D Dissertation, Stanford University,* 2002.

[4] J.H. Goldberg, and J.C. Schryver. Eye-gaze determination of user intent at computer interface. In *Eye Movement Research: Mechanisms, Processes, and Applications,* pages 491-503, New York: Elsevier Science Publishing, 1995.

[5] A. Haro, M. Flickner, and I. Essa. Detecting and tracking eyes by using their physiological prop-erties, dynamics, and appearance. In *IEEE Conference on Computer Vision and Pattern Recognition,* 2000.

[6] M. Harville, A. Rahimi, T. Darrell, G. Gordon, and J. Woodfill, 3D pose tracking with linear depth and brightness constraints, In *International Conference on Computer Vision,* 1999.

[7] B. Moghaddam and A. Pentland. Probabilistic visual learning for object detection. In *International Conference on Computer Vision.* 1995.

[8] C.H. Morimoto, D. Koons, A. Amir, and M. Flickner. Pupil detection and tracking using multiple light sources. *Technical Report RJ-10117, IBM Almaden Research Center,* 1998.

[9] Y. Matsumoto, T. Ogasawara, and A. Zelinsky. Behavior recognition based on head pose and gaze direction measurement. In *IEEE International Conference on Intelligent Robots and Systems,* 2000.

[10] A. Schödl, A. Haro, I. Essa. Head tracking using a textured polygonal model. In *Proceedings Workshop on Perceptual User Interfaces,* 1998

[11] Trucco and Verri. Stereopsis. In *Introductory techniques for 3-D computer vision,* pages 139-175, Prentice Hall. 1998

[12] Z. Zhang. A flexible new technique for camera calibration. In *IEEE transactions on Pattern Analysis and Machine Intelligence,* pages 1330-1334, 2000.