Interactive Skills Using Active Gaze Tracking

Rowel Atienza and Alexander Zelinsky
Research School of Information Sciences and Engineering, The Australian National University
Canberra, ACT 0200 Australia
{rowel,alex}@syseng.anu.edu.au

ABSTRACT

We have incorporated interactive skills into an active gaze tracking system. Our active gaze tracking system can identify an object in a cluttered scene that a person is looking at. By following the user's 3-D gaze direction together with a zero-disparity filter, we can determine the object's position. Our active vision system also directs attention to a user by tracking anything with both motion and skin color. A Particle Filter fuses skin color and motion from optical flow techniques together to locate a hand or a face in an image. The active vision then uses stereo camera geometry, Kalman Filtering and position and velocity controllers to track the feature in real-time. These skills are integrated together such that they cooperate with each other in order to track the user's face and gaze at all times. Results and video demos provide interesting insights on how active gaze tracking can be utilized and improved to make human-friendly user interfaces.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Interaction Styles*; I.4.8 [**Computing Methodologies**]: Image Processing and Computer Vision, Scene Analysis—*Tracking*

General Terms

Algorithm, Human Factors, Measurement

Keywords

active gaze tracking, active face tracking, selecting an object in 3-D space using gaze

1. INTRODUCTION

Gaze indicates where a person is looking and what is the focus of attention (Figure 1). It is an effective means of conveying information about objects a user is interested in. In this paper, we show how we utilize gaze information to direct an active vision system to search and identify the object a user is looking at. A typical scenario is shown in Figure 2 where a person is looking steadily at an object. When a steady gaze is detected, the active vision traces the gaze line to find the point of minimum stereo disparity where the object is most likely positioned (Figure 3). The system is easy to use and versatile since: 1) the user does not need to wear any special device in order to measure eye gaze, head position and head translation, 2) the user and the object with focus of attention can be anywhere in free space since all computations are done through a pair of calibrated zooming cameras mounted on the active head and 3) no priori knowledge about the properties of the object is required other than it is small enough to fit in the field of view of the predefined region of the fovea.

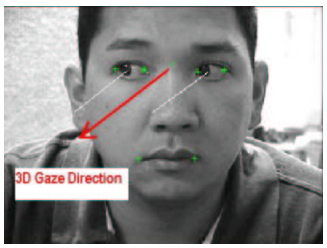

Figure 1: Gaze direction can be modeled by a 3-D vector originating from the user's head

The level of interaction with the active vision increases when the ability to direct attention to a potential user is added. By combining motion and skin color, a person can bootstrap the active vision to begin tracking using normal human activities such as by waving a hand (as shown in Figure 4), walking about or sitting in front of the system.

Although gaze tracking applications can already be found in areas like attentive user-interfaces [9], machine assisted driving [5], game consoles, etc. [14], our objective here is to develop vision-based interfaces that can make human-machine interaction more effective and human-friendly. One experiment we envision to accomplish is directing a robot to pick an object in space by simply looking steadily at it. Furthermore, since children at early ages are known to develop their social skills by observing faces and by tracking gaze directions [13], the basic skills presented here will also find useful in building humanoid robots or machines that learn how people do their tasks.

The rest of the paper describes the theory and results on the first reported implementation of these new skills.

2. POINTING TO AN OBJECT USING GAZE

In order to understand how to control the active vision to do search in 3-D space, we first present a kinematic model of its mech-

ICMI'03, November 5–7, 2003, Vancouver, British Columbia, Canada.

Figure 2: A user looks steadily at an object (white cup) while the active vision searches for it along the gaze line using a zero-disparity filter (upper left inset)

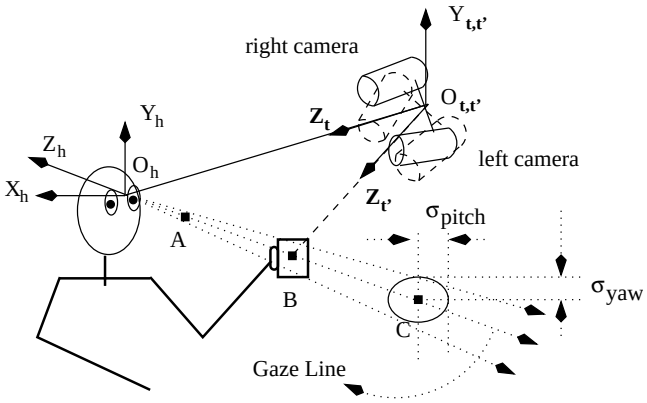

Figure 3: The active vision traces the gaze line from gaze point A to C to find the object (the cup at gaze point B) using a zero-disparity filter. σ_{yaw} and σ_{pitch} are the standard deviations of gaze yaw and pitch respectively that are used to detect steady gaze.

anism on the special case where the camera rotation angles are equal. A model of steady gaze is also formulated using statistical techniques. Steady gaze detects when a user is looking steadily at a certain object in his/her environment. Once a steady gaze is noticed, the active vision can use its kinematics and zero-disparity filtering to follow the 3-D gaze line originating from the user's head and identify the object. Gaze following mimics our natural tendency to look for a certain object another person is looking at by searching it through the gaze direction. The dimensions of the object being viewed are usually small compared to its depth from the camera origin that an affine projection is assumed.

An important assumption in the rest of this paper is that in order to do the above tasks, we already have a working active gaze tracking system that generates the following measurements: 1) 3-D eye gaze direction (${}^{w}G$), 3-D head pose (${}^{w}R_h$) and translation (${}^{w}H$) or 2) 3-D head pose and translation only while gaze is estimated from ${}^{h}G = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T$. Both measurements when available come in real-time (>30Hz) with a corresponding confidence value, p_g, and expressed with respect to a fixed world coordinate system, O_w. All measurements are made possible through real-time readjustment of stereo camera parameters in the gaze tracking algorithm of faceLab [14]. Since the head translation is known, the zoom level is also adjusted automatically to maintain high image resolution of the face at all times. For a more detailed discussion on active gaze tracking, please refer to our earlier paper [1].

2.1 Kinematic Model of the Active Vision

The chief advantage of active vision systems over static stereo configurations in tracking is their ability to position the left and right cameras such that the object being viewed is always at the center of the left and right images. Active vision systems can track moving objects like a person's face in 3-D space in real-time by changing its configuration giving the user the freedom to move without restriction. Here we investigate how we position the stereo cameras such that both are directed to the same object as the gaze point. In order to simplify the configuration and computation, we assume that the left and right camera rotation angles are equal and opposite in direction. The kinematic model computation described is inspired by the work of Murray, et. al. [11].

Figure 4: A user waving his hand to gain the attention of the active vision system (upper left inset shows images from the cameras)

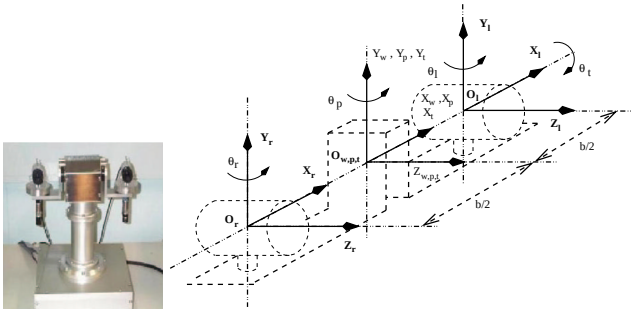

Figure 5: Active vision and its kinematic model

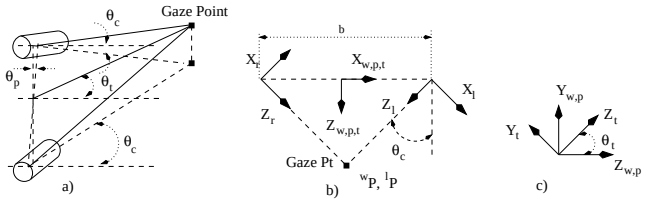

Figure 6: a) Stereo cameras directed at a gaze point, b) Coordinate systems assignment for a) when $\theta_p = \theta_t = 0$, and c) Relationship between pan and tilt coordinate systems when $\theta_t \neq 0$.

Figure 5 is an illustration of our active vision system and its equivalent kinematic model with all the coordinate systems attached. All joints are rotational and has one degree of freedom each. Here, we define the separation between left and right camera rotation axes as the baseline b.

Most computations are made with respect to the rigid world coordinate system, O_w. In most cases we are interested in the kinematics where the principal axes always intersect in a certain gaze point as shown in Figures 6 a and b. For symmetry, we let $\theta_l = -\theta_r$ where $\theta_r > 0$. To simplify our notation, we designate $\theta_c = \theta_r$ as the camera rotation angle. The forward kinematics for the general configuration shown in Figure 6a can be computed if we first assign intermediate transformation matrices that link coordinate systems from world to one of the two cameras:

$${}^w_pT = \begin{bmatrix} R_Y(\theta_p) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}, \quad {}^p_tT = \begin{bmatrix} R_X(\theta_t) & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}, \text{ and}$$

$${}^t_lT = \begin{bmatrix} R_Y(-\theta_c) & \begin{matrix} \frac{b}{2} \\ 0 \\ 0 \end{matrix} \\ \mathbf{0}^T & 1 \end{bmatrix} \tag{1}$$

where $\mathbf{0}^T = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$. We are interested in finding an expression for a general point ${}^wP = \begin{bmatrix} x_w & y_w & z_w \end{bmatrix}^T$ in terms of θ_p, θ_t, and θ_c which are measurable from the motor encoder readings. From Figure 6b, ${}^wP$ is ${}^lP = \begin{bmatrix} 0 & 0 & \frac{b}{2\sin\theta_c} \end{bmatrix}^T$. Note that this relation holds for any configuration where $\theta_c > 0$. Therefore:

$$\begin{bmatrix} {}^wP \\ 1 \end{bmatrix} = {}^w_p T\, {}^p_t T\, {}^t_l T \begin{bmatrix} {}^lP \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{b\sin\theta_p\cos\theta_t}{2\tan\theta_c} \\ \frac{-b\sin\theta_t}{2\tan\theta_c} \\ \frac{b\cos\theta_p\cos\theta_t}{2\tan\theta_c} \\ 1 \end{bmatrix}. \tag{2}$$

Equation 2 is the Forward Kinematics of the active vision mechanism with symmetrical camera rotation angles.

From Equation 2, we can derive the Inverse Kinematics giving the necessary joint angles to position the gaze point in a certain known world Cartesian coordinates:

$$\begin{bmatrix} \theta_p \\ \theta_t \\ \theta_c \end{bmatrix} = \begin{bmatrix} \arctan\left(\frac{x}{z}\right) \\ -\arctan\left(\frac{y\cos\theta_p}{z}\right) \\ \arctan\left(\frac{b\cos\theta_p\cos\theta_t}{2z}\right) \end{bmatrix} \tag{3}$$

where we drop subscript w for convenience.

The main purpose of the inverse kinematics equation is in gaze following. Given a 3-D gaze line expressed in the world coordinate system, we can search for the object by moving the gaze point along the line using Equation 3. The situation is illustrated in Figure 3, where the gaze point follows a straight line from gaze point A to C.

2.2 Description of the 3-D Gaze Line

Before we can utilize the inverse kinematics in searching for the object, we must have a formal representation of the 3-D gaze line. Like the gaze point, the gaze line must also be represented in terms of world coordinate system. Given ${}^wH = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}^T$, the head origin with respect to world coordinate system, and ${}^wG + {}^wH = \begin{bmatrix} x_r & y_r & z_r \end{bmatrix}^T$, the gaze vector described with respect to world coordinate system translated at the head origin, we can define the symmetric equations describing the gaze line:

$$\frac{x_1 - x_0}{x_r - x_0} = \frac{y_1 - y_0}{y_r - y_0} = \frac{z_1 - z_0}{z_r - z_0} \tag{4}$$

where we define ${}^wH$ as the origin , ${}^wG + {}^wH$ as the reference point and $\begin{bmatrix} x_1 & y_1 & z_1 \end{bmatrix}^T$ as any point on the line. If we wished to move a distance, $d > 0$, away from the origin, the new gaze point in terms of world coordinates would be at:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} x_0 \pm \sqrt{x_k} \\ y_0 \pm \left(\frac{y_r - y_0}{x_r - x_0}\right)\sqrt{x_k} \\ z_0 \pm \left(\frac{z_r - z_0}{x_r - x_0}\right)\sqrt{x_k} \end{bmatrix} \tag{5}$$

where $x_k = \frac{d^2}{1 + \left(\frac{y_r - y_0}{x_r - x_0}\right)^2 + \left(\frac{z_r - z_0}{x_r - x_0}\right)^2}$. The choice of sign should ensure that the point moves toward the direction of ${}^wG + {}^wH$.

When the 3-D gaze line is traced, the gaze point is first moved a certain distance (~15cm which is known as gaze point A in Figure 3) from ${}^wH$ before ZDF is applied. This ensures that the face is not mistaken as the object being searched for. Furthermore, human vision systems do not normally look at objects closer than this distance. ZDF then searches for the object until the end of the gaze line segment is reached (~65cm from ${}^wH$ which is known as Point C in Figure 3). Since we are using inverse kinematics to traverse the gaze line, we chose a small increment $d = 7.5mm$ to follow the gaze line in order to generate a smooth motion and to avoid missing the object.

2.3 Modeling Steady Gaze

Before we can initiate object search along the gaze line, we must first know when a user's gaze has stabilized. A steady gaze is assumed whenever the running standard deviation of gaze yaw and pitch go below a certain threshold. In equation form:

$$SG = \begin{cases} 1 & \sigma_{yaw} \leq \sigma_{y0}\, and\, \sigma_{pitch} \leq \sigma_{p0} \\ 0 & otherwise \end{cases} \tag{6}$$

Both σ_{yaw} and σ_{pitch} are computed from the gaze direction generated from $t - t_0$ to t with a normalized gaze confidence value $p_g(t)$ (i.e. $\sum_{t-t_0}^{t} p_g(t) = 1$). In our current setup, steady gaze (SG) becomes true whenever the standard deviations of gaze yaw and pitch fall below $\sigma_{y0} = \sigma_{p0} = 5^\circ$ for the past $t_0 = 3secs$. It can be seen that SG becomes true quickly at higher thresholds. However, the accuracy in determining the true gaze line suffers as the threshold increases. σ_{yaw} and σ_{pitch} are illustrated in Figure 3.

2.4 Zero-Disparity Filter

Once a steady gaze is detected, the object can now be searched by looking along the gaze line. One technique that determines

whether an object is present at the gaze point is by using a zero-disparity filter (ZDF) as proposed in [3]. It is known that objects at or near gaze point appear with zero disparity between the left and right fovea since they are of the same depth. Objects that exhibit zero disparity lying on the horopter can be easily eliminated if we apply a higher weight on disparity measurements at the fovea. But instead of finding the minimum disparity of vertical edges between left and right images, we maximized the normalized cross-correlation between the right and left fovea. The region of fovea is defined as a circle with a certain radius r and with origin at the image center. In human vision systems, fovea is the region in the retina containing cones where vision has the highest resolution and where the object with focus of attention is viewed. We define the object 3-D position as the gaze point where:

$$C = \frac{\left(\mathbf{I}_l - \bar{\mathbf{I}}_l\right) \cdot \left(\mathbf{I}_r - \bar{\mathbf{I}}_r\right)}{\left\|\mathbf{I}_l - \bar{\mathbf{I}}_l\right\| \left\|\mathbf{I}_r - \bar{\mathbf{I}}_r\right\|} \tag{7}$$

is at maximum. $\mathbf{I}_l$ and $\mathbf{I}_r$ represent the left and right fovea's regions respectively. In our current setting, the radius of the fovea is 100 pixels for 320-pixel × 240-pixel gray scale images.

We now have an active gaze tracking system that can track a person's head, generate gaze and head measurements and search for the object a user is looking at when steady gaze is initiated. But before presenting the results, we will discuss another important skill. It is the ability of the system to bootstrap itself by setting its focus of attention on things that are moving and with skin color. This skill is also important in making sure that the face is always tracked in times when the gaze tracking system fails to generate head measurements.

3. GAINING ATTENTION

A system that can respond to basic human behaviors such as motion creates an impression that it is highly interactive and easy to use. Here, we describe how the active vision system is able to detect motion in its surrounding and track the target in real-time time. Since the system should only be interested in motion coming from a person, we use skin color to filter out activity from other moving objects. The effects on motion estimate due to camera ego-motion is also taken into consideration. The measurement of motion and skin color is done at the pixel level. For each pixel that we consider, we assign: 1) a vector of motion flow estimate and its corresponding confidence value and 2) a skin color probability. A modified condensation or particle filter fuses these two visual cues together to locate the object 2-D centroid. A method of determining the probability that the object is a person is also formulated. To track the object in real-time, its 3-D position is estimated from the 2-D position of the object from each image. To generate a smooth output, we use a Kalman Filter. Each axis controller then uses the 3-D position to generate a command output.

3.1 Estimating Motion Using Optic Flow

We used the algorithm described in [12] to measure the flow vector mean, $\mu_{\mathbf{v}}$, and covariance, $\mathbf{\Lambda}_{\mathbf{v}}$. It is basically a weighted version of the gradient-based method by Lucas and Kanade [8] with a Bayesian probabilistic model used to include uncertainty in the computation [15]. In order to generate a scalar confidence value from the 2×2 $\mathbf{\Lambda}_{\mathbf{v}}$, we use the minimum Eigenvalue, λ_{min}, of ${\mathbf{\Lambda}_{\mathbf{v}}}^{-1}$. This is equivalent to decoupling the flow components. λ_{min} is then mapped into the probability range $\begin{bmatrix} 0 & 1 \end{bmatrix}$ by a constant factor to generate a normalized confidence value, p_m. The final output is a measure of the optical flow, $\mu_{\mathbf{v}} = \mathbf{v} = \begin{bmatrix} v_x & v_y \end{bmatrix}^T$ of the pixel (x,y) at the center of the image patch being considered, and its corresponding probability value, p_m. In our experiment, each 320 × 240 image is divided into 80 ×60 image patches to compute the optical flow in real-time.

3.1.1 Compensating for Camera Ego-Motion

Optical flow computation is much simpler if a static stereo camera configuration is used. In our case, when the active vision starts tracking an object, the motion of the camera itself (called ego-motion) induces optical flow making the background scene indistinguishable from an independent moving object. To determine the object flow vector, we estimate the theoretical background scene motion field and subtract it from the measured flow vector [10]. The object flow vector is therefore: $\mathbf{v}_o = \mathbf{v} - k_v \mathbf{v}_s$, where $\mathbf{v}_s = \mathbf{v}_s^T + \mathbf{v}_s^\omega$ is the background scene motion field and k_v is a constant to account for computational time delays. If $\|\mathbf{v}_o\| < v_{min}$, the motion probability is set to 0 eliminating the background scene flow components. However, computing the translational component $\mathbf{v}_s^T$ is difficult since the depth of the scene is unknown. Fortunately, this can be neglected if the depth of the scene is larger than the stereo camera baseline as shown in [12]. The problem then becomes straightforward since the rotational component, $\mathbf{v}_s^\omega$, is only dependent on the camera focal length determined from calibration and on camera rotational speed. The camera rotational speed can be computed by propagating link velocities from the pan to the left camera coordinate system:

$$\begin{bmatrix} \omega_{lx} \\ \omega_{ly} \\ \omega_{lz} \end{bmatrix} = \begin{bmatrix} \cos\theta_l \dot{\theta}_t + \sin\theta_l \sin\theta_t \dot{\theta}_p \\ \cos\theta_t \dot{\theta}_p + \dot{\theta}_l \\ \sin\theta_l \dot{\theta}_t - \cos\theta_l \sin\theta_t \dot{\theta}_p \end{bmatrix} \tag{8}$$

. By replacing θ_l with θ_r, the right camera rotation vector can be computed as well.

3.2 Measure of Similarity with Skin Color

In our earlier paper [1], we described a method to measure the probability of a pixel with a certain RGB components is a skin pixel. The method as proposed by Cai and Goshtasby [2] computes the chrominance of every candidate pixel in a uniform color space (CIE Lab). To calculate the probability of a pixel being a skin color, it is compared to a previously prepared chrominance chart of skin images from different persons. The comparison generates a skin probability estimate of the pixel, p_s. In our experiment, we use the center pixel of every image patch in the motion flow computation described previously to generate a dense skin probability measure.

3.3 Particle Filtering with Motion and Skin Color

Once a measure of motion probability, p_m, and skin probability, p_s, for every pixel under consideration is available, it is now possible to generate a hypotheses to track a moving skin colored object. The first method we tried uses simple voting and biggest region segmentation to determine the object to track. Since the skin color detection algorithm is highly sensitive to camera color settings, the centroid of the biggest skin region is unstable. Furthermore, if a person is far from the camera, the size of the biggest region with both motion and skin color becomes comparable to the one generated by noise. The voting method does not provide an adequate measure of probability that the object it is tracking is a person. Although the second method is more complex and takes more CPU resources, it does not suffer from the limitations of the voting method. The second method uses the Particle Filter as proposed by Isaard and Blake [6] to fuse motion and skin color cues together.

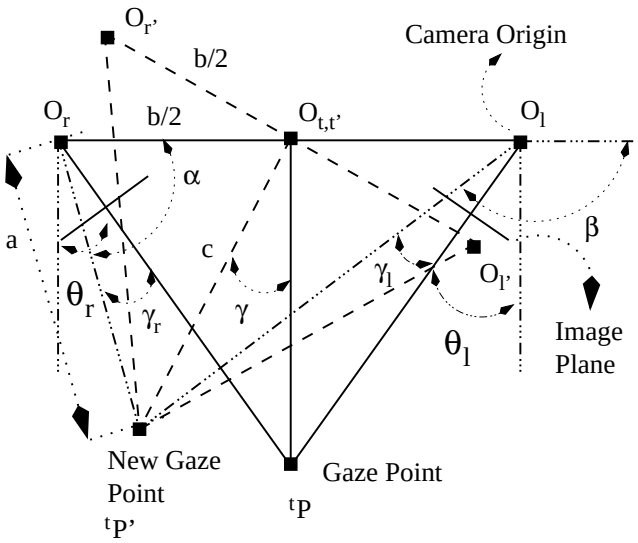

Figure 7: Old and new gaze point positions

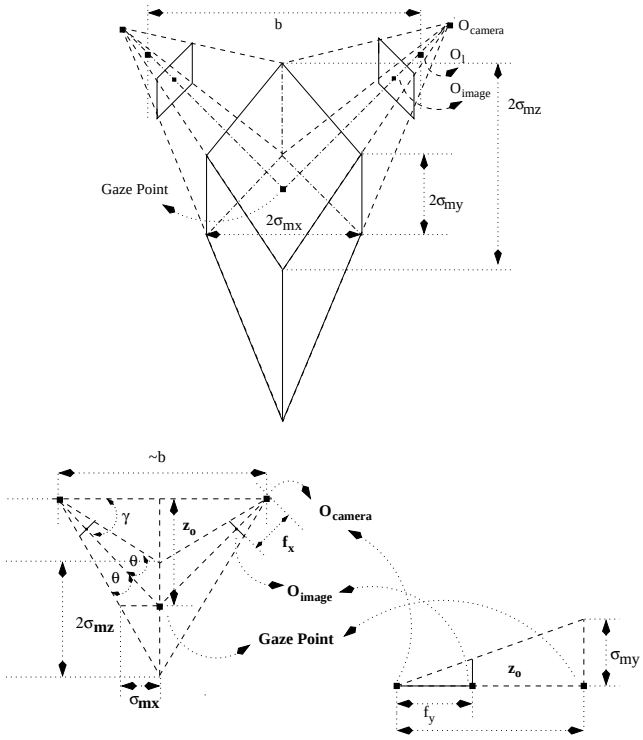

Figure 8: ***Top:*** **Estimate of the stereo camera 3-D field of view as shown by the intersection of the left and right camera field of views.** ***Bottom:*** **Field of view on 2-D. The dimensions of field of view are used in Kalman Filtering.**

We consider N particles to form the initial sample set. In the prediction stage, we model our system as having a constant velocity on each cycle. We assume that the measurement is corrupted by a certain uniform zero-mean noise. The x component of the state of the nth particle can be written as:

$$\mathbf{x}_t^{(n)} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t-1} \\ v_{t-1} \end{bmatrix} + \eta_x(t) \tag{9}$$

where x_{t-1} is the x pixel coordinate from the selection stage, v_{t-1} is the x-component of the velocity of the object being tracked obtained from the optical flow computation and η_x is the uniform zero-mean noise w pixels wide. The same model can be easily obtained for the y component of the state. Similar to [7], we resample only 90% of the particles to avoid the scenario where particles are trapped in a surrounding fixture with color similar to skin. Furthermore, the new weight of the pixel with coordinate (x, y) obtained from the prediction stage is determined by:

$$\pi_t^{(n)} = (p_m(1-\alpha_m)+\alpha_m)(p_s(1-\alpha_s)+\alpha_s) \tag{10}$$

where $0 < \alpha_m, \alpha_s < 1$ to prevent a pixel with zero motion or skin probability from zeroing out the weight. In our experiment, we used $\alpha_m = \alpha_s = 0.1$ for $N = 500$ particles. As mentioned earlier, we can generate a measure of confidence value to tell whether the object being tracked is from a user or not. This is equal to $p_o = \frac{\sum_{n=1}^{N} \pi_t^{(n)}}{N}$. In our experiment, a $p_o > 0.1$ indicates that a person is being tracked. The object centroid is computed as the expectation of the state after normalizing all weights such that $\sum_{n=1}^{N} \pi_t^{(n)} = 1$:

$$\mathbf{p}_{oc} = \begin{bmatrix} x_{oc} \\ y_{oc} \end{bmatrix} = \begin{bmatrix} \sum_{n=1}^{N} x_t^{(n)} \pi_t^{(n)} \\ \sum_{n=1}^{N} y_t^{(n)} \pi_t^{(n)} \end{bmatrix} \tag{11}$$

3.4 Robust 3-D Tracking

The preceding section describes finding the 2-D coordinates of the user in each image. To estimate the position in 3-D, we utilize the camera intrinsic parameters and stereo camera configuration. In our earlier paper [1], we reported a control algorithm to track a skin colored object in real-time where camera rotation angles are not necessarily equal. As mentioned earlier, we now restrict our camera rotation angles to be symmetrical.

From this point, we assume all computations are done with respect to O_t where the object 3-D position can be easily visualized from its top view ($x-z$ plane). Figure 7 is the case when the object moved to a new point located on the right half on both images. The other three cases are not shown here but they easily follow. Since the depth of the object being tracked is much larger than the camera origin displacement from the rotation axis, we can assume that the camera is rotating in its origin. From Figure 7 and our earlier paper [1], the object 3-D position can be estimated as:

$${}^tP' = \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} = \begin{bmatrix} -c\sin\gamma \\ z_o \tan\theta \\ c\cos\gamma \end{bmatrix} \tag{12}$$

where $\gamma = \arccos\left(\frac{a\sin\alpha}{c}\right)$, $c = \sqrt{a^2 + \frac{b^2}{4} - ab\cos\alpha}$, $a = b\frac{\sin(\pi-\beta)}{\sin(\beta-\alpha)}$, $\alpha = \frac{\pi}{2} - \theta_r + \gamma_r$, $\beta = \frac{\pi}{2} + \theta_l + \gamma_l$, $\gamma_r = \arctan\left(\frac{x_{ocr} - \frac{w}{2}}{f_x}\right)$, $\gamma_l = \arctan\left(\frac{x_{ocl} - \frac{w}{2}}{f_x}\right)$, θ_r and θ_l can be measured from the mo-

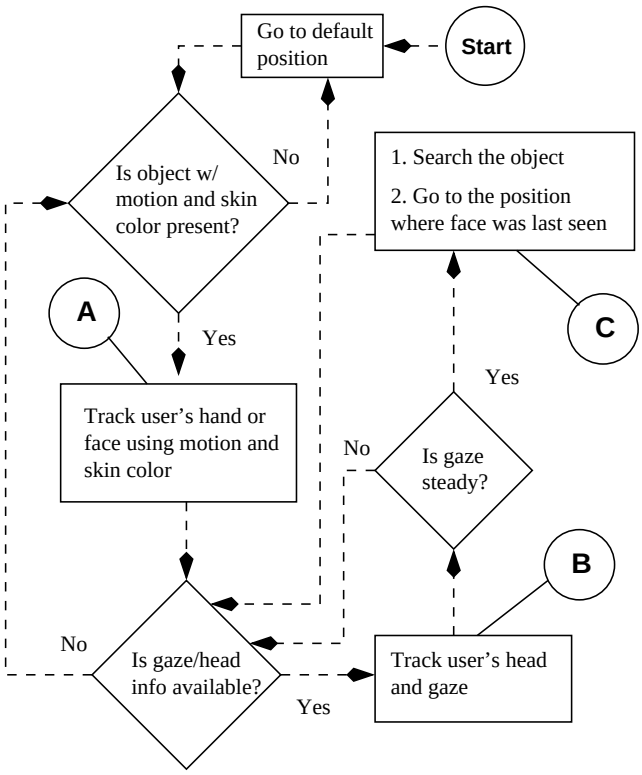

Figure 9: General behavior of the active gaze tracking system when the attention and object searching skills are integrated

tor encoder readings, $\theta = \arctan\left(\frac{y_{ocr}+y_{ocl}-h}{2f_y}\right)$,

$\mathbf{p}_{ocr} = \begin{bmatrix} x_{ocr} & y_{ocr} \end{bmatrix}^T$ is the object 2-D centroid as computed earlier on the right image ($\mathbf{p}_{ocl}$ is for left image), $w \times h$ is the image dimension and f_x (f_y) is the focal length in $x(y)$ pixel coordinate determined from camera calibration. The expression $\frac{y_{ocr}+y_{ocl}-h}{2}$ in the equation for θ ensures that two rays from the each camera center going through the object 2-D centroid on left and right images intersect in 3-D space. It is equivalent to setting the y-coordinate of left and right images 2-D centroid to their mean value.

To generate a smooth output for velocity and position control, ${}^tP'$ is passed through a Kalman Filter. We treat ${}^tP'$ as having three independent components moving at a constant velocity. The general form of the dynamic system for the x component using the notation in [4] is:

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ v_t \end{bmatrix} \sim N\left(\begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t-1} \\ v_{t-1} \end{bmatrix}, \quad \mathbf{Q} \right) \tag{13}$$

where the process covariance $\mathbf{Q} = \sigma_v \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 \end{bmatrix}$ as shown in [12]. The measurement model for the x component is given by:

$$\mathbf{m}_t \sim N\left(\begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x}_t, \quad \sigma_{mx}^2 \right) \tag{14}$$

where the measurement standard deviation is half of the approximate width of the field of view as shown in Figure 8. Therefore, $\sigma_{mx} = \frac{wz_o}{2f_x}$.

Figure 10: ***Top:*** **Active vision gaining attention and tracking the face.** ***Bottom:*** **Particle filters as shown in the inset (circle = face centroid)**

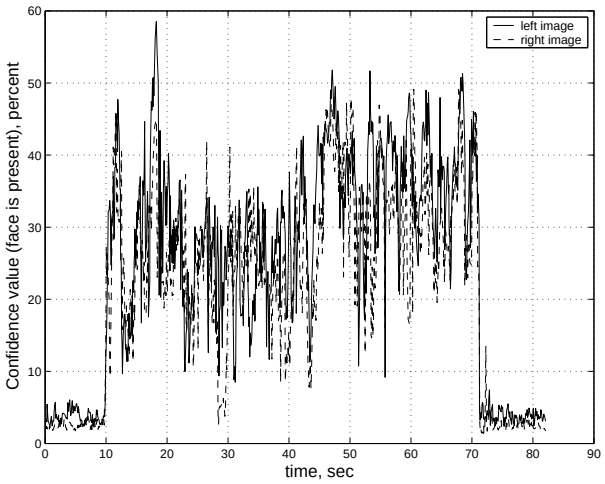

Figure 11: Confidence value (face is present) vs time

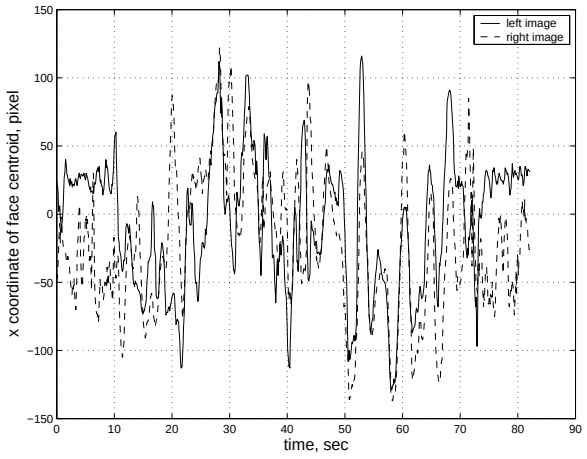

Figure 12: Position controller error in x pixel coordinate vs time

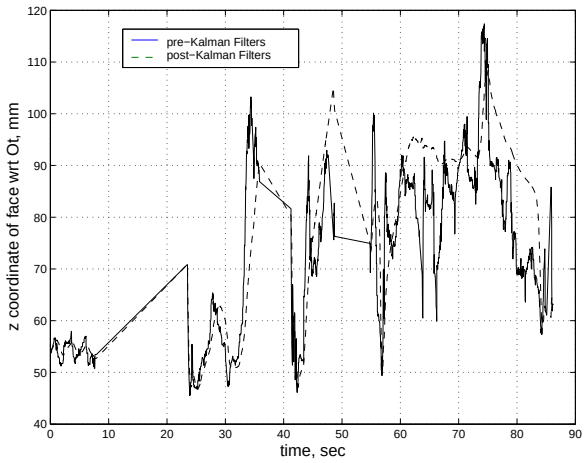

Figure 13: Depth of the person being tracked before and after Kalman Filtering vs time

The y and z components follow similarly and are given the same process covariance. From Figure 8, the measurement standard deviations are given by: $\sigma_{my} = \frac{hz_o}{2f_y}$ and

$\sigma_{mz} = \frac{b}{4}\left(\tan\left(\gamma+\theta\right) - \tan\left(\gamma-\theta\right)\right)$ where $\theta = \arctan\left(\frac{w}{2f_x}\right)$ and $\gamma = \arctan\left(\frac{2z_o}{b}\right)$.

The output of the Kalman Filter is a smoothed measure of the object 3-D location, ${}^tP'_k = \left[\begin{array}{ccc} x_{ok} & y_{ok} & z_{ok} \end{array}\right]^T$. The command output of the pan axis velocity controller is directly proportional to $\arctan\left(\frac{x_{ok}}{z_{ok}}\right)$ ($\arctan\left(\frac{y_{ok}}{z_{ok}}\right)$ for tilt axis). The rotation angles are now determined by $\theta_c = \arctan\left(\frac{b}{2z_{ok}}\right)$ and directly used as the position command output.

Results from the experiments performed to validate the performance of the new interactive skills are discussed next.

4. RESULTS AND DISCUSSION

The summary of the activity of the active vision system is shown Figure 9. It generally describes how the two skills come together to

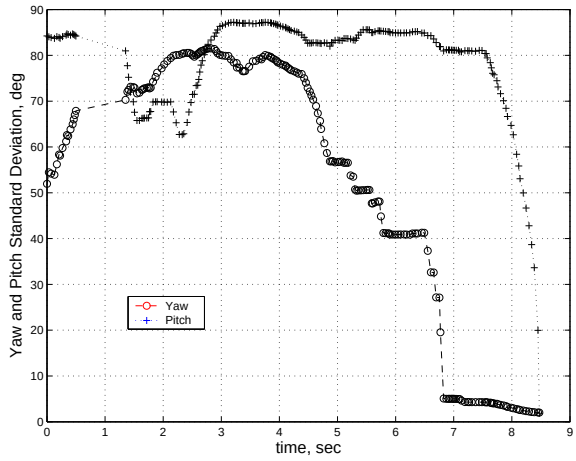

Figure 14: Gaze and yaw pitch standard deviations vs time

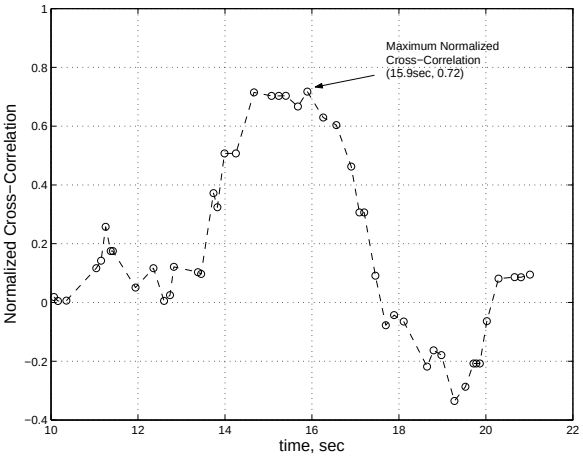

Figure 15: Normalized cross-correlation along 3-D gaze line vs time

bootstrap the active vision and search the object a user is looking at. The activity starts with the active vision in the default position. Once an object with both motion and skin color is present, tracking starts (Process A). Eye gaze, head rotation and head translation are calculated whenever possible. If the gaze and head information is available, it supersedes the motion and skin color information and used instead to track the user more accurately (Process B). Once a steady gaze is detected, the object with focus of attention is searched. After showing the user the object the system has found, the gaze point is transferred back to the position where the face was last seen (Process C). One of the videos of the experiments done is submitted with this paper. All active gaze tracking videos can be found in: http://www.syseng.anu.edu.au/rsl/rsl_demos.html . The details of the results are discussed next.

4.1 Robust 3-D Tracking

The first experiment verifies our robust tracking algorithm (Process A in Figure 9). Figure 10 shows snapshots from the time the active vision is: 1) idle at the default position (0 sec), 2) gaining focus of attention (15 secs), and 3) tracking (>22 secs). Figure 11

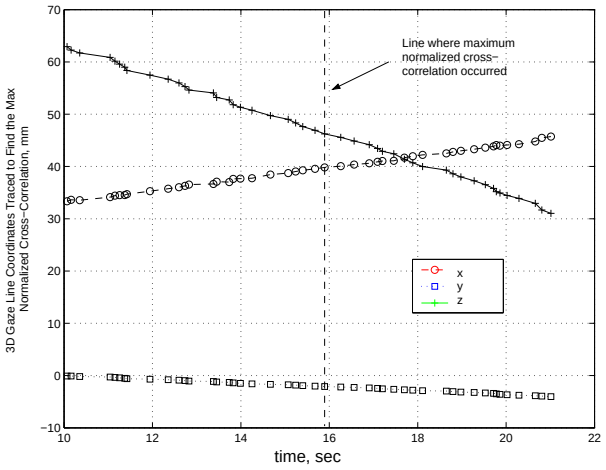

Figure 16: 3-D position of the gaze point as the 3-D gaze line is traced vs time

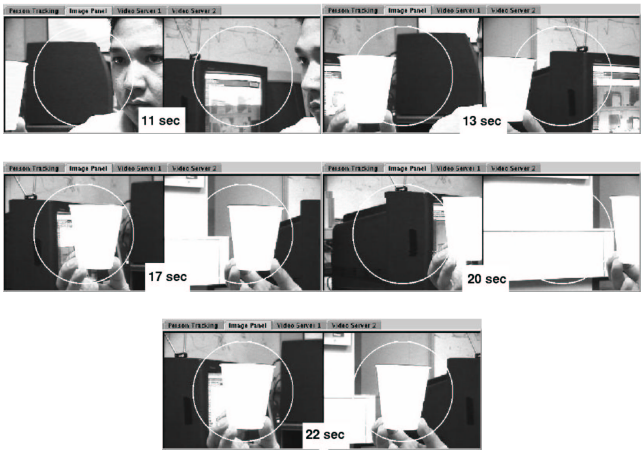

Figure 17: ZDF images along the gaze line (circle = fovea)

shows the confidence value to indicate when a person is present or not. For time less than 10 secs, the active vision is not tracking since the probability that a person is present is below a threshold. The time between 10 and 71 secs is when the active vision starts gaining attention and then continuously tracking the user's face. After 71 secs, the person disappeared and the active vision returns to its default position. The effectiveness in zeroing out positional error in the horizontal direction is shown in Figure 12. During tracking, the mean error of the face centroid is -15.48 pixels for the left image (22.48 for the right image) while the standard deviation is 52.39 pixels (56.18 for the right image) in the x coordinate. We believe that the mean and standard deviation will improve by tuning our controllers. The smoothing effect of Kalman Filter on depth measurement is shown in Figure 13. Experiment shows that if Kalman Filter is not applied, tracking is not possible since all axis controllers become unstable due to oscillations in the 3-D position estimate.

4.2 Pointing to an Object Using 3-D Gaze

In this experiment, a user is sitting in front of the active vision system while looking steadily on an object (a white cup) he is holding as shown in Figure 2 (Processes B and C in Figure 9). As shown in Figure 14, time less than 8.5 secs was spent tracking the user's head and gaze while waiting for the standard deviations of gaze yaw and pitch to go below their threshold (5°) to initiate steady gaze. Time between 8.5 to 10 secs was spent transferring the gaze point from the user's face to Point A in Figure 3 (~15cm along the gaze line). The search for the object started at 10 secs and lasted until 21 secs as shown in Figure 15 where the maximum cross-correlation equal to 0.72 (minimum disparity) is found at 15.9 secs. This gaze point of minimum disparity is recorded as shown in Figure 16. Some camera images along the gaze line are shown in Figure 17. The scene with minimum disparity is shown at 22 secs after the search is completed. In our future experiment, we intend to segment this object at the scene of minimum disparity to validate the accuracy of the search. After finding the object, the attention is again transferred to the user.

5. CONCLUSION

We demonstrated a unique experiment where a user can direct the active vision to search for the object with focus of attention. It is a step forward in making human-machine interaction more similar to human-human interaction. We envision to perform an experiment where a robot can be directed to pick up objects using the user's gaze.

6. REFERENCES

[1] Atienza, R. and Zelinsky, A., Active Gaze Tracking for Human-Robot Interaction, ICMI 2002, PA USA, 2002.

[2] Cai, J. and Goshtasby, A., Detecting human faces in color images, Image and Vision Computing. 18: 63-75, 1999.

[3] Coombs, D. and Brown, C., Real-time binocular smooth pursuit. IJCV, 11(2):147-164, 1993.

[4] Forsyth, D. and Ponce, J., Computer vision a modern approach, Prentice Hall, NJ, 2003.

[5] Fletcher, L., et. al., Driver support systems for smart cars, IEEE Symposium on Intelligent Vehicles, Italy, 2003.

[6] Isard, M. and Blake, A., Condensation-condensational density propagation for visual tracking, IJCV, 29(1): 5-28 , 1998.

[7] Loy, G., et. al., An adaptive fusion architecture for target tracking, Proc. Face and Gesture Recognition 2002.

[8] Lucas, B. and Kanade, T., An iterative image registration technique with an application to stereo vision. In Proc. DARPA Image Understanding Workshop, 1981.

[9] Maglio, P., et. al., Gaze and speech in attentive user interfaces, ICMI 2000.

[10] Murray, D., et. al., Driving saccade to pursuit using image motion, IJCV, 16(3): 205-228, 1995.

[11] Murray, D., et. al., Design of stereo heads, in Active Vision eds. Blake, A. and Yuille, A., 1992.

[12] Rougeaux, S., Real-time active vision for versatile interaction, Ph.D. Dissertation, Univ. d'Evry and Electrotechnical Lab, 1999.

[13] Scassellati, B., Investigating models of social development using a humanoid robot, in Biorobotics eds. Webb, B. and Consi, T., MIT Press, 2000.

[14] Seeing Machines Inc., www.seeingmachines.com.

[15] Simoncelli, E., et. al., Probability distributions of optical flow, Proc. CPVR, 1991.