HEAD ORIENTATION AND GAZE DETECTION FROM A SINGLE IMAGE

Jeremy Yirmeyahu Kaminski
Computer Science Department, Holon Academic Institute of Technology, Holon, Israel
Email: kaminsj@hait.ac.il

Adi Shavit and Dotan Knaan
Gentech Corporation, Israel Labs, Jerusalem, Israel
Email: adish.dolan@gentech.co.il

Mina Teicher
Bar-Ilan University, Ramat-Gan, Israel
Email: teicher@macs.biu.ac.il

Keywords: Head Orientation, Gaze, Resultant

Abstract: Head orientation is an important part of many advanced human-machine interaction systems. We present a single image based head pose computation algorithm. It is deduced from anthropometric data. This approach allows us to use a single camera and requires no cooperation from the user. Using a single image avoids the complexities associated with a multi-camera system. Evaluation tests show that our approach is accurate, fast and can be used in a variety of contexts. Application to gaze detection, with a working system, is also demonstrated.

1 Introduction

Numerous systems need to compute head motion and orientation. Instances are driver attention monitoring and human-computer interface for multimedia or medical purposes.

Therefore, for a large range of applications, the need for a simple and efficient algorithm for computing head orientation is crucial. Systems that must recover at some stage some information on the 3D position of the user’s head, run by definition in a complex environment, because of the variety of human faces and behaviors. That is the reason why, such a system will gain in stability and usability if the underlying algorithm uses a simple setting. In that context, it is worth noting that our algorithm is based on a single camera and does not require from the user any calibration.

Moreover, we shall present how our algorithm for head orientation can be used for gaze detection too. We describe an overall system, which performs gaze detection from a single image. Our system turned out to be accurate and fast, which makes it convenient for a large variety of applications.

1.1 Comparison With Other Approaches

Before introducing the core of our method, we present a short comparison with previous works. Since the problem of head position and gaze detection has received a huge amount of attention in the past decade, we do not pretend to establish a comprehensive review of previous work. However we consider hereafter what seems to be the most relevant references to show the novelty of our approach.

The large number of proposed algorithms for gaze detection proves that no solution is completely satisfying. In (Glenstrup and Engell-Nielsen, 1995), one can find a good survey of several gaze detection techniques. In (Ji and Yang, 2002; A. Perez, 2003), one can find a stereo system for gaze and face pose computation, which is particularly suitable for monitoring driver vigilance. Both systems are based on two cameras, one being a narrow field camera (which provides a high resolution image of the eyes by tracking a small area) and the second being a large field camera (which tracks the whole face). Besides the computationally complex difficulties arising from multiple cameras and controlling these pan-tilt cameras, the system hardware is quite costly. In (T. Ohno and Yoshikawa, 2002), a monocular system is presented, which uses a personal calibration process for each user and does not allow large head motions. Limiting the head motion is typical for systems that utilize only a single camera. (T. Ohno and Yoshikawa, 2002) uses a (motorized) auto-focus lens to estimate the distance of the face from the camera. In (J.G. Wang and Venkateswarku, 2003), the eye gaze is computed by using the fact that the iris contour, while being a cir-
acle in 3D is perspective an ellipse in the image. The
drawback in this approach is that a high resolution im-
age of the iris area is necessary. This severely limits
the possible motions of the user, unless an additional
wide-angle camera is used.

In this paper we introduce a new approach with
several advantages. The system is monocular, hence
the difficulties associated with multiple cameras are
avoided. The camera parameters are maintained con-
tant in time. The system requires no personal cali-
bration and the head is allowed to move freely. This is
achieved by using a model of the face, deduced from
anthropometric features. This kind of method has al-
ready received some attention in past (T. Horprasert
and Davis, 1997; Gee and Cipolla, 1994a; Gee and
Cipolla, 1994b). However, our approach is simpler,
requires less points to be tracked and is eventually
more robust and practical.

In (Gee and Cipolla, 1994a; Gee and Cipolla,
1994b), the head orientation is estimated under the
assumption of the weak perspective image model. This
algorithm works using four points: the mouth cor-
ners and the external corners of the eyes. Once those
points are precisely detected, the head orientation
is computed by using the ratio of the lengths \( L_e \) and
\( L_f \), where \( L_e \) is the distance between the external
eyes corners and \( L_f \) between the mouth and the eyes. In
(T. Horprasert and Davis, 1997), a five points algo-

rithm is proposed to recover the 3D orientation of
the head, under full perspective projection. The in-
ternal and external eyes corners provide four points,
while the fifth point is the bottom of the nose. The
first four points approximately lie on a line. There-
fore the authors use the cross-ratio of these points as
an algebraic constraint on the 3D orientation of the
head. It is worth noting that the cross-ratio is known
to be very sensitive to noise. Consider for exam-
ple, four points \( A, B, C, D \) lying on the x-axis which
x-coordinates are respectively \( 5, 10, 15, 20 \). These
points can be typically the eye corners. Then the
cross-ratio \( [A, B, C, D] = (5 - 15)/(5 - 20) \times (10 -
20)/(10 - 15) = 10/3 = 3.333 \). Now if \( A \) is de-
tected at \( 4 \) and \( B \) at \( 11 \), then the cross-ratio becomes
\( [A, B, C, D] = (4 - 15)/(5 - 20) \times (11 - 20)/(11 -
15) = 99/64 \approx 1.54 \). This simple computation
shows that using the cross-ratio as a constraint on the
3D structure, requires detection with a precision gen-

erally beyond the capability of a vision system.

In contrast, our approach is based on three points
only and works with a full perspective model. The
three points are the eye centers and the middle point
between the nostrils. Using these three points, we can
calculate several algebraic constraints on the 3D head
orientation, based on an anthropomorphic model of the
human face. These constraints are explicitly formu-
lated in section 2.

Once the head orientation is recovered, further
computations are possible. In this paper, we show an
application to gaze detection. This approach, of a me-
chanically simple, automatic and non-intrusive sys-
tem, allows eye-gazing to be used in a variety of ap-
plications where eye-gaze detection was not an option
before. For example, such a system may be installed
in mass produced cars. With the growing concern of
accidents, customers and regulators are demanding
safer cars. Active sensors that may prevent acci-
dents are actively perused. A non-intrusive, cheaply
produced, one-size-fits-all eye-gazing system could
monitor driver vigilance at all times. Drowsiness and
inattention can immediately generate alarms. In con-
junction with other active sensors, such as radar, ob-
stacle detection, etc. the driver may be warned of an
unnoticed hazard outside the car.

Psychophysical and psychological tests and experi-
ments with uncooperative subjects such as children
and/or primates, may also benefit from such a static
(no moving parts) system, which allows the subject
to focus solely on the task at hand while remaining
oblivious to the eye-gaze system.

In conjunction with additional higher-level sys-
tems, a covert eye-gazing system may be useful in se-
curity applications. For example, monitoring the eye-
gaze of ATM clients. In automated airport checkin
counters, such a system may alert of suspiciously be-
having individuals.

The paper is organized as follows. In section 2,
we present the core of the paper, the face model that
we use and how this model leads to the compu-
tation of the Euclidean face 3D orientation and position. We
present simulations, that show the results are robust to
time. The system requires no personal cali-
bation and the head is allowed to move freely. This is
achieved by using a model of the face, deduced from
anthropometric features. This kind of method has al-
ready received some attention in past (T. Horprasert
and Davis, 1997; Gee and Cipolla, 1994), we assume the following model of a generic
human face. Let \( A \) and \( B \) be the centers of the eyes,
and let \( C \) be the middle point between the nostrils.
Then we assume the following model:

\[
d(A, C) = d(B, C) \quad (1)
\]
\[
d(A, B) = r d(A, C) \quad (2)
\]
\[
d(A, B) = 6.5 cm \quad (3)
\]

where \( r = 1.0833 \). The two first equations allow
computing the orientation of the face, while the third
equation is necessary for computing the distance between the camera and the face. The face model is illustrated in figure 1. The data gathered in (Farkas, 1994) shows that our model is widely valid over the human population. Of course variations exist, but the simulations presented in section 2.5 show that our algorithm is quite robust over the whole spectrum of human faces.

2.2 3D Face Orientation

Let \( M \) be the camera matrix. All the computations are done in the coordinate system of the camera. Therefore the camera matrix has the following expression:

\[
M = K[I:0],
\]

where \( K \) is the matrix of internal parameters (Hartley and Zisserman, 2000; Faugeras and Luong, 2001).

Let \((a, b, c)\) be the projection of \((A, B, C)\) onto the image. In the equations below, the image points \(a, b, c\) are given by their projective coordinates in the image plane, while the 3D points \(A, B, C\) are given by their Euclidean coordinates in \(\mathbb{R}^3\). Given these notations, the projection equations are:

\[
a \sim KA
\]

\[
b \sim KB
\]

\[
c \sim KC
\]

where \(\sim\) means equality up to a scale factor. Therefore the 3D points are given by the following expressions:

\[
A = \alpha K^{-1}a
\]

\[
B = \beta K^{-1}b
\]

\[
C = \gamma K^{-1}c
\]

where \(\alpha, \beta, \gamma\) are unknown scale factor. These could also be deduced by considering the points at infinity of the optical rays generated by the image points \(a, b, c\) and the camera center. These points at infinity are simply given in projective coordinates by: \([K^{-1}a, 0]^t, [K^{-1}b, 0]^t, [K^{-1}c, 0]^t\). Then the points \(A, B, C\) are given in projective coordinates by \([\alpha K^{-1}a, 1]^t, [\beta K^{-1}b, 1]^t, [\gamma K^{-1}c, 1]^t\). These expressions naturally yield the equations (7), (8), (9) giving the Euclidean coordinates of the points.

Plugging these expressions of \(A, B, C\) into the two first equations of the model (1) and (2), leads to two homogeneous quadratic equations in \(\alpha, \beta, \gamma\):

\[
f(\alpha, \beta, \gamma) = 0
\]

\[
g(\alpha, \beta, \gamma) = 0
\]

Thus finding the points \(A, B, C\) is now reduced in finding the intersection of two conics in the projective plane. Moreover since no solution is on the line defined by \(\gamma = 0\) (since the nose of the user is not located at the camera center!), one can reduce the computation of the affine piece defined by \(\gamma = 1\). Hence we shall now focus our attention on the following system:

\[
f(\alpha, \beta, 1) = 0
\]

\[
g(\alpha, \beta, 1) = 0
\]

This system defines the intersection of two conics in the affine plane. The following subsection is devoted to the computation of the solutions of this system.

2.3 Computing the Intersection of Conics in the Affine Plane

For sake of completeness, we shall recall shortly one way of computing the solutions of the system above. For more details, see (Sturmfels, 2002). Consider first two polynomials \(f, g \in \mathbb{C}[x]\). The resultant gives a way to know if the two polynomials have a common root. Write the polynomials as follows:

\[
\begin{align*}
 f &= a_nx^n + \ldots + a_1x + a_0 \\
 g &= b_px^p + \ldots + b_1x + b_0
\end{align*}
\]

The resultant of \(f\) and \(g\) is a polynomial \(r\), which is a combination of monomials in \(\{a_i\}_{i=1,...,n}\) and \(\{b_j\}_{j=1,...,p}\) with coefficients in \(\mathbb{Z}\), that is \(r \in \mathbb{Z}[a_i, b_j]\). The resultant \(r\) vanishes if and only if either \(a_n\) or \(b_p\) is zero or the polynomials have a common root in \(\mathbb{C}\). The resultant can be computed as the determinant of a polynomial matrix. There exist several matrices whose determinant is equal to the resultant. The best known and simplest matrix is the so-called Sylvester matrix, defined as follows:

\[
S(f, g) =
\begin{bmatrix}
  a_n & 0 & 0 & \ldots & 0 & b_p & 0 & 0 & \ldots & 0 \\
  a_{n-1} & a_n & 0 & \ldots & 0 & b_{p-1} & b_p & 0 & \ldots & 0 \\
  \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots
\end{bmatrix}
\]

Therefore, we have:

\[
r(x) = \det(Syl(f, g)).
\]
In addition to this expression which gives a practical way to compute the resultant, there exists another formula of theoretical interest:

$$r(x) = a_n b_p \Pi_{x,a}(x_i^n - x_i^p),$$

where $x_i^a$ are the roots of $f$ and $x_i^p$ are those of $g$. It can be shown that the resultant is a polynomial of degree $np$.

An important point is that the resultant is also defined and has the same properties if the coefficients of the polynomials are not only numbers but also polynomials in another variable. Hence, consider now that $f, g \in \mathbb{C}[x, y]$ and write:

$$\begin{align*}
  f &= a_0(x)y^n + \ldots + a_1(x)y + a_0(x) \\
  g &= b_0(x)y^n + \ldots + b_1(x)y + b_0(x)
\end{align*}$$

The question is now the following: given a value $x_0$ of $x$, do the two polynomials $f(x_0, y)$ and $g(x_0, y)$ have a common root? The answer to this question is based on the computation of the resultant of $f$ and $g$ with respect to $y$ (i.e., using the presentation given by (14)). This is a univariate polynomial in $x$, denoted by $r(x) = \text{res}(f, g, y)$.

The resultant can be used in many contexts. For our purpose, we will use it to compute the intersection points of two planar algebraic curves. Consider the curve $C_1$ (respectively $C_2$) defined as the set of points $(x, y)$ which are roots of $f(x, y)$ (respectively $g(x, y)$). We want to compute the intersection of $C_1$ and $C_2$. Algebraically, this is equivalent to computing the common roots of $f$ and $g$. Therefore, we use the following procedure:

- Compute the resultant $r(x) = \text{res}(f, g, y) \in \mathbb{C}[x]$.
- Find the roots of $r(x)$: $x_1, \ldots, x_t$
- For each $i = 1, \ldots, t$, compute the common roots of $f(x_i, y)$ and $g(x_i, y)$ in $\mathbb{C}[y]$: $y_{i1}, \ldots, y_{ik_i}$.
- The intersection of $C_1$ and $C_2$ is therefore:

$$\{(x_1, y_{i1}), \ldots, (x_1, y_{1k_1}), \ldots, (x_t, y_{1k_1}), \ldots, (x_t, y_{tk_t})\}.$$

In our context, the resultant $r$ is polynomial of degree 4 and so $t \leq 4$ and $k_i \leq 2$. To complete the picture, we just need to mention an efficient and reliable way to compute the roots of a univariate polynomial. The algorithm that we will describe is very efficient and robust for low degree polynomials. Given a univariate polynomial $p(x) = a_n x^n + \ldots + a_1 x + a_0$, one can follow the forming matrix, called the companion matrix of $p$:

$$C(p) = \begin{bmatrix}
  0 & 1 & 0 & \ldots & 0 \\
  0 & 0 & 1 & \ldots & 0 \\
  \vdots & \vdots & \ddots & \vdots & \vdots \\
  -a_0/a_n & -a_1/a_n & -a_2/a_n & \ldots & -a_{n-1}/a_n
\end{bmatrix}$$

A short computation shows that the characteristic polynomial of $C(p)$ is equal to $-\frac{1}{a_n}p$. Thus the roots of $p$ are exactly the eigenvalues of $C(p)$. This provides one practical way to compute the roots of a univariate polynomial.

### 2.4 3D Face Orientation

Therefore, we solve the system $S$ defined by equations (12) and (13) using the approach presented above. By Bezout’s theorem (or simply by looking at the degree of the resultant), we know that there are at most 4 complex solutions to this system. Experiments show that a system generated by the image of a human face has only two real roots. The ambiguity between these two roots is easily handled, since one solution leads to non realistic eye-to-eye distance. Let $(\alpha_0, \beta_0)$ be the right solution. Then the points $A_0, B_0$ and $C_0$ are known up to a unique scale factor. We shall denote $A_0, B_0$ and $C_0$ the points obtained by the solution $(\alpha_0, \beta_0)$. Thus we have the following expression:

$$\begin{align*}
  A_0 &= \alpha_0 K^{-1} a \\
  B_0 &= \beta_0 K^{-1} b \\
  C_0 &= K^{-1} c
\end{align*}$$

Thus we have the following relations too: $A = \gamma A_0, B = \gamma B_0$ and $C = \gamma C_0$.

The computation of $\gamma$ is done using the third model equation (3). Once the face points are computed, we can compute the distance between the user’s face and the camera and so the 3D orientation of the face. Indeed the normal to the plane defined by $A, B$ and $C$ is given by:

$$\vec{N} = \overrightarrow{AB} \land \overrightarrow{AC},$$

where $\land$ is the cross product.

### 2.5 Robustness to Errors in Model and Detection

In order to estimate the sensitivity of this algorithm to errors in model and in detection, we performed several simulations. As we shall detail in subsection 3.1, we use a rather high resolution camera. Therefore in the simulation, we start from the following setting:

- The focal length $f = 4000$ in pixels (which is very close to the value of the actual camera used in the system),
- The principal point is at the image center,
- The distance between the camera and the face is 60cm (which is the actual setting of the system).

The simulations are done according to the following protocol. An artificial face, defined by three points in space, say $A, B$ and $C$, is projected onto a known
camera. Given a parameter $p$, we perform a perturbation of $p$ by a white Gaussian noise of standard deviation $\sigma$. For each value of $\sigma$, we perform 100 random perturbations. For each value of $p$, obtained by this process, we compute the error in the 3D reconstruction as the mean of the square errors.

The first simulation (see figure 2) shows that the system is very robust to errors in the estimation of the focal length, since for a noise with standard deviation of 100 (in pixels), the reconstruction error is $1.2cm$, less than 1% of the distance between the camera and the user.

The next two simulation aim at measuring the influence of errors in model. First, the assumed eye-to-eye distance is corrupted by a Gaussian white noise (figure 3). The mean value is $6.5cm$ as mentioned in section 2. For a standard deviation of 0.5, which represents an extreme anomaly with respect to the standard human morphology, the reconstruction error is about $3.3cm$, less than 2% of the distance between the camera and the user. The influence of the human ratio $r$, as defined in equation (2), is also tested by adding a Gaussian white noise, centered at the "universal" value 1.0833 (figure 4). For a standard deviation 0.15, which also represents a very strong anomaly, the reconstruction is $1.75cm$, just over 1% of the distance between the camera and the user.

After measuring the influence of errors in camera calibration and model, the next step is to evaluate the sensitivity to input data perturbation. The image points are corrupted by a Gaussian white noise (figure 5). For a noise of 10 pixels, which is a large error in detection, the reconstruction error is less $2cm$, about 1.15% of the distance between the camera and the user.

The accuracy of the system is mainly due to the fact that the focal length is high ($f = 4000$ in pixels). Indeed when computing the optical rays generated by the image points, as in equations (7,8,9), we use the
inverse of $K$, which is roughly equivalent to multiplying the image points coordinates by $1/f$. Hence the larger $f$ is, the less impact a detection error has on the computation.

3 Application: Gaze Detection System

In this section, we show how the ideas presented above can be used to build a gaze detection system, that does not require any user calibration or interaction.

3.1 System Architecture

The main practical goal of this work was to create a non-intrusive gaze detection system, that would require no user cooperation while keeping the system complexity low. We use a high-resolution 15 fps, 1392x1040 video camera with a 25mm fixed-focus lens. The CCD pixel is a square of length equal to 6.45 microns. Thus the focal length is 3875, which is the same order of magnitude as the value used in the simulation. This setup allows both a wide field of view, for a broad range of head positions, and high resolution images of the eyes. Since we can estimate the 3D head position from a single image, we can use a fixed focus lens instead of a motorized auto-focus lens. This makes the camera calibration simpler and the calibration of the internal parameters is done only once. The system uses an IR LED at a known position to illuminate the user’s face.

3.2 System Overview

The general flow of the system is depicted in Figure 6. For every new frame, the glints, that is the reflections of the LED light from the eye corneas as seen by the camera, are detected and their corresponding pupils are found. The search area for the nose is then defined, and the nose bottom is found. Given the two glints and nose position, we can reconstruct the complete Euclidean 3D face position and orientation relative to the camera, using the geometric algorithm presented in section 2. This reconstruction gives us the exact 3D position of the glints and pupils. Then, for each eye, the 3D cornea center is computed using the knowledge of LED position, as shown in figure 8. This model is similar to the eye model used in (T. Ohno and Yoshikawa, 2002). The following sub-sections 3.3 and 3.4 will describe these stages in more detail.
3.3.2 Nose Detection

The detection of the nose-bottom is done by searching for dark-bright-dark patterns in the area just below the eyes. Indeed, the nostrils appear as dark blobs in the image thanks to the relative position of the camera and the face as shown in figure 7. The size and orientation of this search area is determined by the distance and orientation of the chosen glint-pair. Once dark-bright-dark patterns are found, we use connected component blob analysis on this region to identify only those dark blobs that obey certain size, shape, distance and relative angle constraints that yield plausible nostrils. The nose bottom is selected as the point just between the two nostrils.

3.4 Gaze Detection

Given the glints and the bottom point of the nose, one can apply the geometric algorithm presented in section 2 to compute the 3D face orientation. As seen in subsection 2.5, even if the glints are not exactly located in the center of the eye, the system returns an accurate answer. Then for each eye, the cornea center is computed using the knowledge of LED position, as shown in figure 8.

The gaze line is defined as being the line joining the cornea center and the pupil center in 3D.

The pupil center is first detected in the image and computed in 3D as follows. The distance between the pupil center and the cornea center is a known measure of the human anatomy. It is equal to 0.45 cm. Consider then a sphere $S$ centered at the cornea center, with radius equal to 0.45 cm. The pupil center lies on the optical ray generated by its projection onto the image and the camera center. This ray intersects the sphere $S$ in two points. The closest of these points to the camera is the pupil center.

5 Discussion

We proposed an automatic, non-intrusive eye-gaze system. It uses an anthropomorphic model of the human face to calculate the face distance, orientation and gaze angle, without requiring any user-specific calibration. This generality, as seen in subsection 2.5, does not introduce large errors into the gaze direction computation.

While the benefits of a calibration-free system al-
Figure 10: The detected triangle, eyes’ centers and the nose bottom, together with the gaze line.

low for a broad range of previously impossible applications, the system design allows for easy plugging of user-specific calibration data, which will increase the accuracy even more.

REFERENCES


Figure 11: The detected triangle, eyes’ centers and the nose bottom, together with the gaze line.