Pose Coupling with Eye Movement Tracking

Ana C. Andrés del Valle & Jean-Luc Dugelay

February 2002

RR-01-058
Research Report

Table of Contents

Introduction

We consider face analysis from a video sequence as a function of the general pose of the face on the sequence, the illumination conditions under which the video is recorded and the face expression movements. Pixel changes over the analyzed image may come from any evolution at the lighting, head pose or head expression. Previous works present how we study the pose of the person in front of the camera regardless of the lighting conditions [SV99]. Head pose motion involves 'rigid' movements where the whole head can be taken as a complete unit. Expression movements involved 'elastic' model behavior in particular zones of the face, features, like eyes, mouth and eyebrows. These features are difficult to analyze as a unit and generally expression motion is studied feature by feature.

Most feature expression analysis algorithms are well defined only for a front or near-to-front position. This front position ensures the maximum area of information about the eye area with minimum distortions. In general, this area of information is enough as long as the head pose does not rotate more than 10 degrees on the Y-, Z- axis and 5 on the X-axis, partially losing precision. For instance, the algorithm is independent of translations (t_X, t_Y, t_Z) while the head remains in the camera vision field. The areas occupied by the feature and its appearance changes with the head pose.

Our eye-state algorithm [AVD01], based on the search of the minimum energy of the eye on a studied Region of Interest (ROI), proves to be very sensitive on how this region is defined. The space sampling of the ROI, which will provide us the determination of the possible state of the eye (UP-DOWN, LEFT-CENTER-RIGHT) goes along with the perfect definition of the analysis area. When dealing with the complete freedom of pose movement applied to the eyes, this first approach seems 'a priori' not strong enough. By keeping the same basis, control of illumination information, search for the minimum energy of the area and sampling of the ROI, we can reformulate the algorithm to fit it in a 3D space. Using the information provided by the pose prediction, we can foresee the evolution of the area and the feature over the video frame. Then, we extract the visual information from the useful ROI. Finally, we transform the visual information into 3D information by utilizing the knowledge of the transformation that the head pose implies on the eye area.

1. How the complete algorithm works

1. In an initialization step there has to be a very good determination of the ROI, since the measures for the state sampling are based on it. The chosen points that determined the ROI will need to be also known in the 3D synthesis. (NOTE: This is the first step where the clone is synthesized, with no expression, just after having searched the face on the video image)
2. From the general lighting study we deduce the need for some light preprocessing. For instance, simulation of eye shadows due to a very bright frontal lighting.
3. We only need one information band for the algorithm; with no other information that would be the Intensity component from the HSI. A preliminary test recognizing the color of the eyes of the user could help to take one of the band from RGB. If complete skin information were wanted (in case we search to add a backup in the form of skin tracking) then the R band would be preferred. In any case, this is only an implementation choice and has not shown dramatic changes in terms of the final results.
4. The area, which is tracked, is always a rectangle whose only requirement is to be as tangent to the eye as possible and to contain the eye: larger and proportional to the eye dimensions, and adaptive along the time depending on the pose.
5. We work under the assumption that movement between consecutive frames is smooth.
6. We assume that both eyes have the same behavior. We define the eye states based on the possible coherent combinations. Misleading results are treated deducing the states from precedent results. [AVD01]
7. To adapt the parameters of the analysis algorithm areas in the ROI, we will use the parameters of the general pose being tracked ($t_X, t_Y, t_Z, \alpha, \beta, \gamma$). We will deduce the situation of the minimal energy point in the well known sampled ROI, in the position of maximal information ($\alpha = \beta = \gamma = 0$, and $t_X = t_Y = t_Z = 0$). Defining the ROI in 3D we should avoid the projection dependence and the influence of the angles.

Figure 1 shows the general scheme that the algorithm follows.

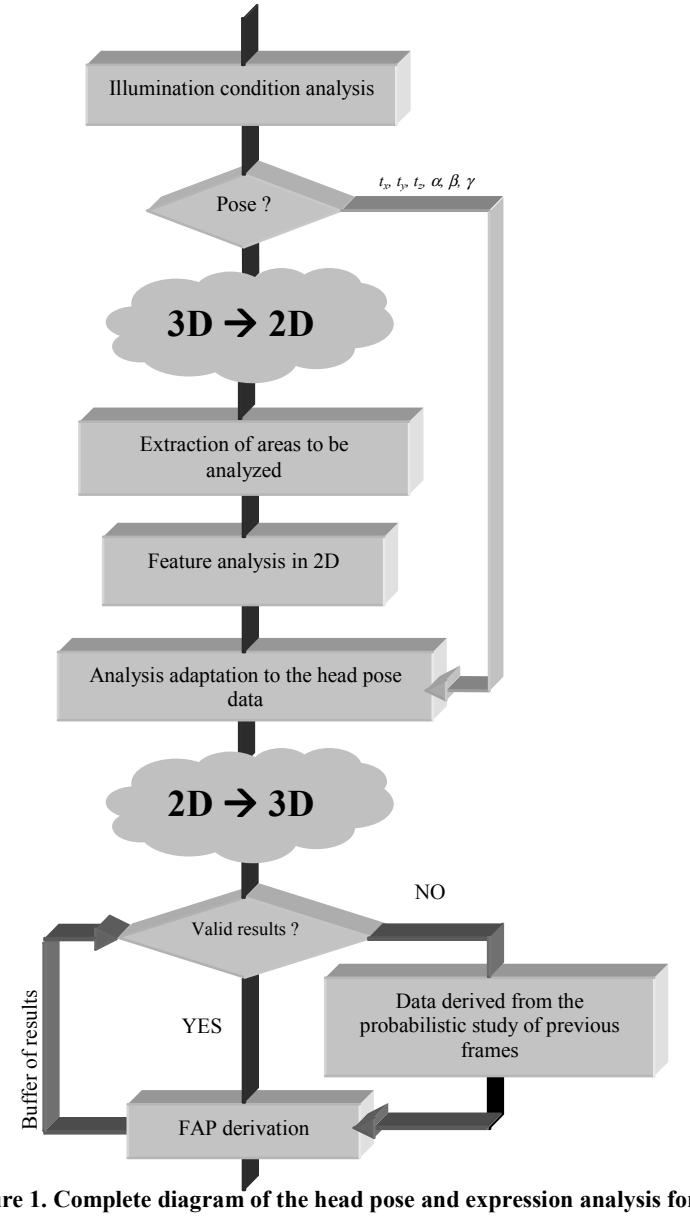


Figure 1. Complete diagram of the head pose and expression analysis for our teleconference system

2. ROI definition and model of the eye for the gaze analysis

Definition of the Region of Interest

For our algorithm the control of the area we are analyzing over the video frame is very important. The definition of a well-established Region Of Interest (ROI) has two major purposes. On the one hand, we want to extract the maximum amount of information of the eye feature from the video frame by optimizing the area to analyze (so computation can be minimized). On the other hand, we want to have the possibility of foreseeing the relevance of the information we would take from the feature before we even start analyzing it. The ROI should be adapted as much as possible to the eye; we just analyze the information we are interested in, with no interference. The projection of the eye of the user on the screen could be so small that its analysis would not give meaningful information. Thanks to the pose parameter (t_X, t_Y, t_Z, α, β, γ) prediction given by the Kalman filter, we can decide whether it is useful to analyze this feature on the video plane or not. We can also provide feedback to the user warning him about the upcoming impossibility of tracking its expressions in such conditions.

We obtain the maximum information regarding the eye state in a frontal position (neutral position of the clone). Based on the appearance of the eye in such a position (See Fig. 4), we approximate the ROI for the eye feature with an ellipsis of axis **A** and **B** (Fig. 3). The surface of this ellipse is contained within a plane. When the head rotates or translates the ROI may show different perspectives of the eye (See Fig. 2 for example). The best way to control how the projection will affect the area of analysis is to define the ROI and its sampling for the eye states in 3D, in our case, the 3D information needed can be taken from the neutral position of the clone.

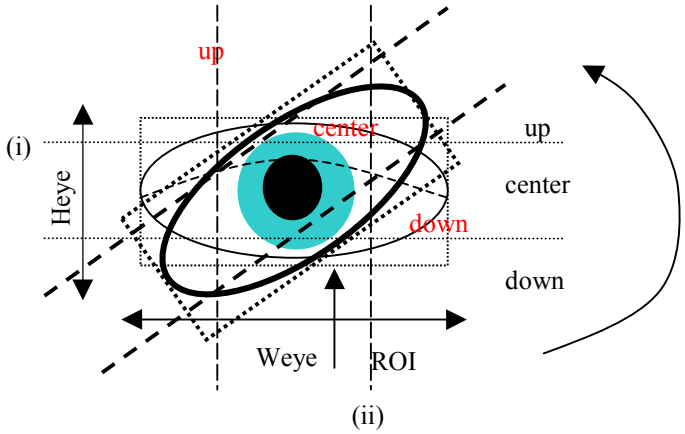


Figure 2. Rotation around the Z-axis (γ)

Eye model to recover feature information from the frame

By definition, the ROI on one video frame is a plane surface (in general, it is rectangular) to be analyzed. Our algorithm extracts the minimum energy point on this ROI. This information does not correspond to the minimum energy of a planar object but of the eye, whose 3D surface can be better approximated by a

sphere (See Figure 4 for the anatomy of the eye and Figure 3 for its equivalent three-dimensional approximation). The information to build the spherical surface can be taken from the clone of the individual.

Once the ROI is well defined and its sampled areas too, this information has to be directly related to a proper position on the spherical surface. From the video frame analysis we will get the projected position of the minimum point of energy on the ROI that belongs to our studied sphere (which will have overcome the pose transformation). To be able to relate this information with the proper state, we have to compare this datum with the defined sampling areas in 3D. At this point, either we obtain the projection of the sampling lines on the studied sphere, by transforming them from the neutral position (applying the current pose parameters) to their expected projected location; or we find the 3D position of the minimum of energy, by inverting the pose and the projection transformations, thus obtaining the position of that point in its expected neutral position.

The simplest option is to only perform one transformation and recover the 3D location of the minimum point of energy under the constraint that it belongs to the known sphere (See Section 4 for details).

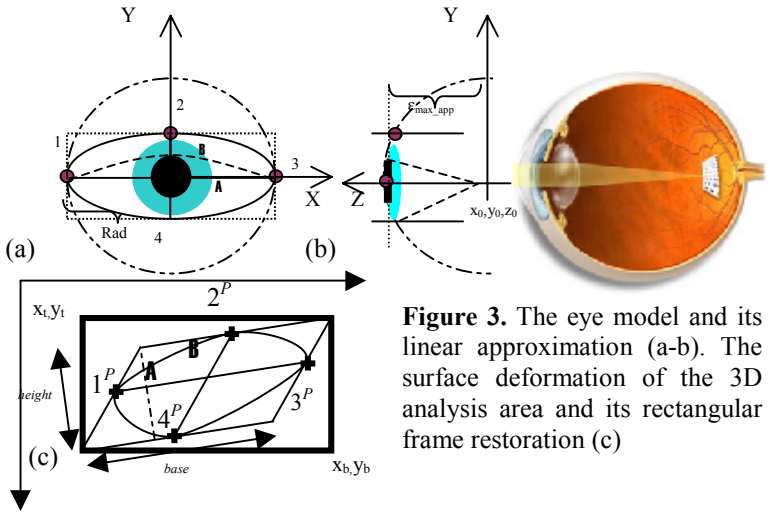


Figure 3. The eye model and its linear approximation (a-b). The surface deformation of the 3D analysis area and its rectangular frame restoration (c)

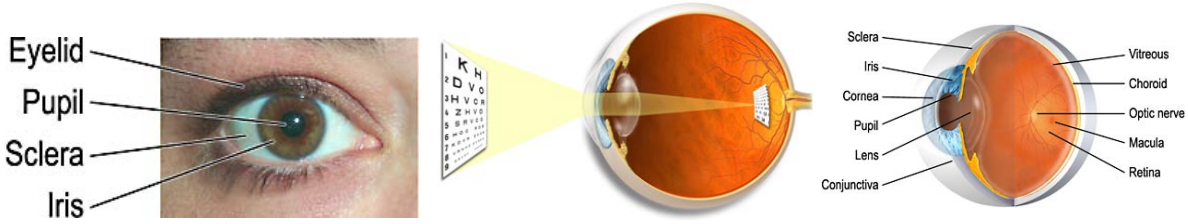


Figure 4. Anatomy of the eye.

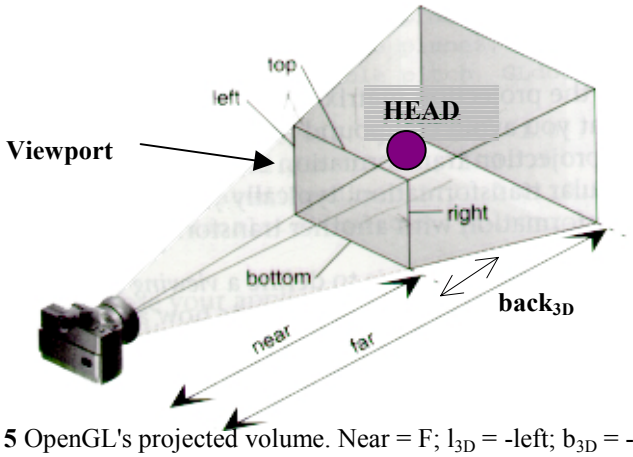


Figure 5 OpenGL's projected volume. Near = F; l_{3D} = -left; b_{3D} = -bottom

How transformations are performed

The general pose over the head is expressed in terms of three translations and three rotations. This pose information is used by the Kalman filter and by the movement synthesizer. The coordinate system, the transformation and projection used by both, filter and synthesizer, must be the same; otherwise, the synthesized projected image of the head will not match the video image the camera is acquiring.

We must define which transformations we are going to apply and their order, we also must decide the center of these transformations, which in our system is the mass center of the head. Rotating is not a commutative operation, therefore order is very important and it will have to be maintained by the synthesizer that performs the operation (OpenGL for us). From the previous work developed by Stéphane Valente [SV99] the transformation were concatenated in the following way:

$$X_{tranformed} = G \,.\, X_{neutral} \quad \text{where } G = T_{(tX,tY,tZ)} \,.\, R_{\alpha,x} \,.\, R_{\beta,y} \,.\, R_{\gamma,z} \;;$$

$$X_{projected} = P_F \,.\, T_{(0,\,0,\,F)} \,.\, X_{transformed} \qquad (1)$$

$$\begin{bmatrix} F & 0 & 0 & 0 \\ 0 & F & 0 & 0 \\ 0 & 0 & -1 & -F \\ 0 & 0 & -1 & F \end{bmatrix}$$

Where

$$T_{(tX,tY,tZ)} = \begin{bmatrix} 1 & 0 & 0 & t_X \\ 0 & 1 & 0 & t_Y \\ 0 & 0 & 1 & t_Z \\ 0 & 0 & 0 & 1 \end{bmatrix}; R_{\alpha,x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_\alpha & -s_\alpha & 0 \\ 0 & s_\alpha & c_\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \; R_{\beta,y} = \begin{bmatrix} c_\beta & 0 & s_\beta & 0 \\ 0 & 1 & 0 & 0 \\ -s_\beta & 0 & c_\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } R_{\gamma,z} = \begin{bmatrix} c_\gamma & -s_\gamma & 0 & 0 \\ s_\gamma & c_\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

For notation purposes, c_φ stands for $\cos(\varphi)$, s_φ stands for $\sin(\varphi)$ and t_φ stands for $\tan(\varphi)$, ^{P} for projected coordinates and $_n$ for neutral 3D-coordinates. Capital letters represent matrices and vectors, and lower case letters coordinates and vector components. F stands for the focal distance value of the projection system.

It is understood that we define the head in its neutral position when $\alpha = \beta = \gamma = 0$ and $t_X = t_Y = t_Z = 0$. The origin of the head coordinates must be defined there where the rotations will be held (which may or may not be the real center of the model)

The matrices of transformations are set to transform the 3D or 2D coordinates expressed in their homogeneous form:

$$[x\ y\ z\ w] \rightarrow [x/w\ y/w\ z/w]\ ;\ [x\ y\ w] \rightarrow [x/w\ x/w]\ ;$$

The homogeneous coordinate, w, indicates the proportion to the view in the synthesized world. Unless specifically stated, w is 1 for the original 3D coordinates from the 3D model provided. If not $w < 1$ enlarges the given coordinates and $w > 1$ diminishes the given coordinates. OpenGL's representation on the screen (from projections) will go from [$-w$ to $+w$]. OpenGL also performs another transformation to set the image on the image plane in pixels and in its proper proportions by taking into account the Viewport (physical image plane), see Figure 5. We refer to [OGL99] for details and we develop the different projection expression with OpenGL's intrinsic parameters in section 5. The following explanation covers the general behavior of any synthesizer that follows the operation expected by the Kalman filter.

If we want to obtain the homogeneous coordinates from their 2D projections we have to invert the system:

$X_{neutral} = G^{-1} . X_{tranformed} = R_{\gamma, z}^{-1} . R_{\beta, y}^{-1} . R_{\alpha, x}^{-1} . T_{(tX, tY, tZ)}^{-1} . X_{transformed}$;

knowing that $R_{\alpha,z}^{-1} = R_{-\alpha,z}$ and $T_{(x,y,z)}^{-1} = T_{(-x, -y, -z)}^{-1}$:

$$X_{neutral} = G^{-1} . X_{tranformed} = R_{-\gamma, z} . R_{-\beta, y} . R_{-\alpha, x} . T_{(-tX, -tY, -tZ)} . X_{transformed}. \quad (2)$$

From equation 1 we can deduce the expression of $X_{transformed}$ as

$$X_{transformed} = [P_F . T_{(0, 0, F)}]^{-1} . X_{projected} \quad (3)$$

$$\begin{bmatrix} 1/F & 0 & 0 & 0 \\ 0 & 1/F & 0 & 0 \\ 0 & 0 & -1/2 & -1/2 \\ 0 & 0 & -1/2F & 1/2F \end{bmatrix}$$

So we can finally substitute in equation 2:

$$X_{neutral} = G^{-1} . X_{tranformed} = R_{-\gamma, z} . R_{-\beta, y} . R_{-\alpha, x} . T_{(-tX, -tY, -tZ)} . [P_F . T_{(0, 0, F)}]^{-1} . X_{projected}. \quad (4)$$

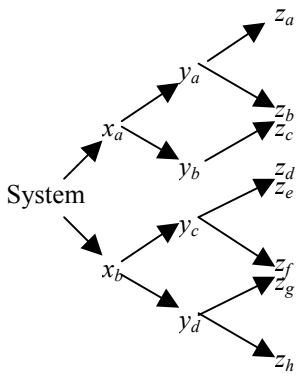
3. Obtaining the 3D coordinates of the minimum energy point of the ROI

Equation 3 shows the relationship of the 3D coordinates of the object and its projection on the screen under any pose. $X_{projected}$ and $X_{transformed}$ are the homogeneous correspondence of those coordinates. The projected information can come from a synthesized world or from real data. We cannot extract homogenous coordinate data from a video sequence. Thus, we are not able to inverse the chain of transforms to directly recover the 3D coordinates of the minimum energy point from a video frame. From the projection, we generate equivalence (1) but no depth information can be recovered. To obtain the z (depth) coordinate, we have to introduce a new constraint, which for this feature, will be the knowledge that this minimum energy point belongs to a well defined sphere $(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = Rad^2$

Solving the following system:

$$(5)\ \begin{cases} (x_n - x_0)^2 + (y_n - y_0)^2 + (z_n - z_0)^2 = Rad^2 \\ [(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)x_P - Fc_\beta c_\gamma]x_n + [(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)x_P + Fc_\beta s_\gamma]y_n + [-c_\alpha c_\beta x_P - Fs_\beta]z_n = (t_X - x_P)F + t_Z x_P \\ [(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)y_P - (s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma)F]x_n + [(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)y_P + (s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma)F]y_n + [-c_\alpha c_\beta y_P + s_\alpha c_\beta F]z_n = (t_Y - y_P)F + t_Z y_P \end{cases}$$

generates a tree of solutions that can be expressed with the following solution tree:



From all the results, only two will be feasible, the two points that represent the intersection between the projection ray from the camera to the screen with the sphere. From these possible solutions, we will only take the one whose depth coordinate during the analysis is greater (higher $z_{transformed}$ value) because it will show the one that is currently visible on the image plane. Figure 6 shows the situation of the head regarding the coordinate system utilized; when projecting the head, some vertices hide other vertices that have lower z values and lie on the same XY plane.

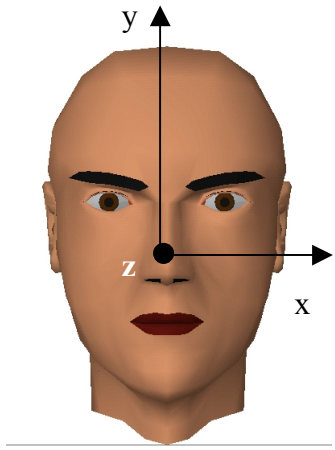


Figure 6. Coordinate system associated to the 3D head model.

CASE WHERE WE APPROXIMATE BY A PLANE

In practice, it is not convenient to solve a second-order equation. Linearizing the equation of the sphere becomes necessary. We are interested in obtaining the plane that best represents the spherical eye information in the front position. We have chosen the pupil in its neutral position (frontal) as the point around which we will develop the linear approximation. This choice implies developing the plane tangent to the sphere at the pupil point. A plane tangent to the sphere is such that its normal is the vector **R**, which is given by:

R = $(A, B, C) = (x_s - x_0, y_s - y_0, z_s - z_0)$ where x_s, y_s, z_s are the expressions of the coordinates in the sphere general equation (Equation 5). Any group of parallel planes can be described by this normal as $A'x + B'y + C'z + D' = 0$. From this family we take to those tangent to the sphere, applying $A'=A$, $B'=B$, $C'=C$, and

frontal, that is, belonging to the plane XY = 0 (z = M). To choose M we develop the sphere equation with z=M which generates a family of circles:

$F(x,y) = (x-x_0)^2+(y-y_0)^2=Rad^2-(M-z_0)^2$

from this family we are interested in those whose radius is equal to 0 because they define the point of the plane that are tangent to the sphere.

$F(x,y) = 0 \rightarrow M = z_0 + Rad$ or, $M = z_0 - Rad$.

We take the point belonging to the frontal plane:

$z = z_0 + Rad$.

We finally define the limited squared analysis area on this plane.

If we recover a point in this plane we know that the error committed by the approximation is limited by $|\varepsilon|<Rad$ and it appears when analyzing the more extreme points of the defined area. Knowing the anatomy of the eyeball, the chosen rectangular area is normally never as big as the maximum defined by the radius of the sphere. Figure 3 shows the physical interpretation of the error committed when linearizing the eyeball sphere.

Study of the solution with the general expression of a plane:

If a plane around the point of study approximates the sphere, the system is linear and simplified. This plane has the structure $A'x + B'y + C'z + D' = 0$. This structure can be seen as a homogeneous function applied to a set of points $P^T \cdot X=0$, where $P^T = [A'\ B'\ C'\ D']$ and $X = [x\ y\ z\ w]$.

With these premises, we know from (4) that this equality could be written as $P^T \cdot (G^{-1} \cdot X)=0$.

Defining $G^{-1} = \begin{bmatrix} h & i & j & k \\ l & m & n & o \\ p & q & r & s \\ t & u & v & w \end{bmatrix}$, we could write de new constraint as

$(Ah + Bl + Cp + Dt)x_n + (Ai + Bm + Cq + Du)y_n + (Aj + Bn + Cr + Dv)z_n + (Ak + Bo + Cs + Dw) = 0$ (taking w=1).

The values for G^{-1} are:

$h = c_\gamma c_\beta$ $\quad$ $l = -s_\gamma c_\beta$

$i = s_\gamma c_\alpha + c_\gamma s_\beta s_\alpha$ $\quad$ $m = c_\gamma c_\alpha - s_\gamma s_\beta s_\alpha$

$j = s_\gamma s_\alpha - c_\gamma s_\beta c_\alpha$ $\quad$ $n = c_\gamma s_\alpha + s_\gamma s_\beta c_\alpha$

$k = -c_\gamma c_\beta t_X - (s_\gamma c_\alpha + c_\gamma s_\beta s_\alpha)t_Y - (s_\gamma s_\alpha - c_\gamma s_\beta c_\alpha)t_Z$ $\quad$ $o = s_\gamma c_\beta t_X - (c_\gamma c_\alpha - s_\gamma s_\beta s_\alpha)t_Y - (c_\gamma s_\alpha + s_\gamma s_\beta c_\alpha)t_Z$

$p = s_\beta$ $\quad$ $t = 0$

$q = -c_\beta s_\alpha$ $\quad$ $u = 0$

$r = c_\beta c_\alpha$ $\quad$ $v = 0$

$s = -s_\beta t_X + c_\beta s_\alpha t_Y - c_\beta c_\alpha t_Z$ $\quad$ $w = 1$

Obtaining this way the final expression:

$(Ah + Bl + Cp)x_n + (Ai + Bm + Cq)y_n + (Aj + Bn + Cr)z_n + (Ak + Bo + Cs + D) = 0$

From the new plane constraints and equivalence (3) we could obtain the 3D position of the projected point.

As stated before, from our analysis data, it is much more simple and more efficient to directly solve for the vector $X_{neutral}$ from equation (1) combined with the constraint of the plane defined in its neutral position $A'x$

$+ B'y + C'y + D'=0$. Then, we fit $X_{neutral}$ coordinates onto the quantized areas in their neutral state. We can compactly express the complete system as follows:

$$\begin{bmatrix} A' & B' & C' \\ (c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)x_P - Fc_\beta c_\gamma & (-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)x_P + Fc_\beta s_\gamma & -c_\alpha c_\beta x_P - Fs_\beta \\ (c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)y_P - (s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma)F & (-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)y_P + (s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma)F & -c_\alpha c_\beta y_P + s_\alpha c_\beta F \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = \begin{bmatrix} -D' \\ (t_X - x_P)F + t_Z x_P \\ (t_Y - y_P)F + t_Z y_P \end{bmatrix}$$

We must see if there is always a solution when the plane constraint is reinforced. Checking when the determinant of the system is zero we obtain the following non desired situations:

$$A \cdot [[(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)x_P + Fc_\beta s_\gamma][-c_\alpha c_\beta y_P + s_\alpha c_\beta F] - [-c_\alpha c_\beta x_P - Fs_\beta][(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)y_P + (s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma)F]] = 0$$

&&

$$B \cdot [[(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)x_P - Fc_\beta c_\gamma][-c_\alpha c_\beta y_P + s_\alpha c_\beta F] - [-c_\alpha c_\beta x_P - Fs_\beta][(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)y_P - (s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma)F]] = 0$$

&&

$$C \cdot [[(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)x_P - Fc_\beta c_\gamma][(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)y_P + (s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma)F] - [(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma)x_P + Fc_\beta s_\gamma][(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)y_P - (s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma)F]] = 0.$$

4. Equivalence with OpenGL's projection system

The inverse of the OpenGL projection (our practical scenario) would give the following matrices:

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = \begin{bmatrix} a_4 \\ b_4 \end{bmatrix}$$

where

$$a_1 = x_p.l_{3D}(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) + Fc_\beta c_\gamma \qquad b_1 = y_p.b_{3D}(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) + F(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma)$$
$$a_2 = x_p.l_{3D}(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma) - Fc_\beta s_\gamma \qquad b_2 = y_p.b_{3D}(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma) + F(-s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma)$$
$$a_3 = x_p.l_{3D}(-c_\alpha c_\beta) + Fs_\beta \qquad b_3 = y_p.b_{3D}(-c_\alpha c_\beta) - Fs_\alpha c_\beta$$
$$a_4 = x_p.l_{3D}(t_Z - F - back_{3D}) - F.t_X \qquad b_4 = y_p.b_{3D}(t_Z - F - back_{3D}) - F.t_Y$$

l_{3D} and b_{3D} represent the values of the Viewport dimension and $back_{3D}$ is the extra displacement along the z-axis added to the model so it is correctly viewed, see Figure 5 for details.

If we add the plane constraint our linear system becomes:

$\begin{bmatrix} A' & B' & C' \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} = \begin{bmatrix} -D' \\ a_4 \\ b_4 \end{bmatrix}$ easily solvable when the plane is defined on XY (z = M):

$\begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix} \begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} a_4 - Ma_3 \\ b_4 - Mb_3 \end{bmatrix}$ which has always a solution if $a_1.b_1\text{-}a_2.b_2 \neq 0$

We need to avoid the obvious cases: $a_1=a_2=0$ or $b_1=b_2=0$, and then the forbidden combinations $a_1=b_1 = 0$ or $a_2=b_2=0$. Generally, evaluating the cases where $a_1=0$, $a_2=0$, $b_1=0$ and $b_2=0$ we detect the most conflictive situations.

$a_1=0 \rightarrow$
$\beta = \pi/2;\ \alpha = \pm\pi/4;\ \gamma = \pm\pi/4$
$\beta = \pi/2;\ \alpha = \pm 3\pi/4;\ \gamma = \pm 3\pi/4$
$b_1=0 \rightarrow$
$\alpha = \pm\pi/2;\ \gamma = \pm\pi/2;\ \forall\beta$
$a_2=0 \rightarrow$
$\beta = \pm\pi/2;\ \alpha = \gamma;\ \alpha = -\gamma$
$b_2=0 \rightarrow$
$t_\gamma.s_\beta = y_p.b_{3D}/F$
$\alpha = \pi/4;\ \gamma = \beta \approx 2\pi/7$

From this analysis, we conclude that the pose-feature analysis coupling is completely stable to head translations and becomes unstable for those head angles beyond $\pm\pi/4$. Nevertheless, our tracking system finds its limitation beyond these angles because the tracked data start to fall onto the same projected points.

6. Error expressions: study of the error generated by both, the Kalman prediction (t_X, t_Y, t_Z, α, β, γ) and the data (x_p, y_p) imprecision

Algorithmically, we can summarize the coupling of pose tracking with eye-state feature analysis as the achievement of the correct inverse projection-pose transformations of the lower energy point on the video eye 2D ROI (x_p, y_p) to obtain its 3D coordinates in their neutral position (x_n, y_n, z_n). These coordinates will be compared to the sampled eye states defined over the neutral 3D ROI so to derive the needed FAPs that describe the eye-state found. We express the inverse projection-pose transformation as:

$$x_n = \frac{1}{\det|A|}\cdot\left[M.B(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma) + M.Cc_\beta c_\gamma + M.F^2(s_\alpha s_\gamma - c_\alpha s_\beta c_\gamma) + Bt_x(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma) + F^2 t_x(s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma) - F^2 t_y c_\beta s_\gamma + C.H(c_\alpha c_\gamma - s_\alpha s_\beta s_\gamma) - Ct_y(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma) + B.Hc_\beta s_\gamma\right]$$

$$y_n = \frac{1}{\det|A|}\cdot\left[M.B(c_\alpha c_\gamma - s_\alpha s_\beta s_\gamma) - M.Cc_\beta c_\gamma + M.F^2(s_\alpha c_\gamma + c_\alpha s_\beta c_\gamma) + Bt_x(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) - F^2 t_x(s_\alpha s_\beta c_\gamma + c_\alpha c_\gamma) - F^2 t_y c_\beta c_\gamma - C.H(s_\alpha s_\beta c_\gamma + c_\alpha c_\gamma) + Ct_y(s_\alpha s_\gamma - c_\alpha s_\beta c_\gamma) + B.Hc_\beta c_\gamma\right]$$

$$z_n = M$$

where

$$B = y_p b_{3D} F \;;\; C = x_p l_{3D} F \;;\; H = t_z - F - back_{3D}$$

$$\det|A| = F(Fc_\alpha c_\beta + x_p l_{3D} s_\beta - y_p b_{3D} s_\alpha c_\beta)$$

These expressions have taken into account all model assumptions. Defining the eye-states over the 3D ROI ensures the correct interpretation of the eye movement. The final action can be misled by the error cumulated from the two algorithms involved and their coupling.

In this section, we estimate the influence the precision of the image analysis algorithm and the pose prediction have over the final results in terms of the error correlation and evolution found in the final neutral coordinates when errors are introduced during the image analysis or the Kalman pose parameter prediction separately.

The inaccuracy additive error (ε) on the obtained parameters ($\tilde{x}_p, \tilde{y}_p, \tilde{\alpha}, \tilde{\beta}, \tilde{\gamma}, \tilde{t}_x, \tilde{t}_y, \tilde{t}_z$) carries on in the expression of the final values. We express the proximate final values, $\tilde{x}_n$ and $\tilde{y}_n$, in terms of the theoretical right values, x_n and y_n, and the affecting multiplying errors (ε_{mult}) and additive errors (ε_{add}) coming from the analysis and the pose prediction.

ERROR EXPRESSIONS

- Error derived from data inaccuracy during image analysis:

$$\tilde{x}_p = x_p + \varepsilon_x \;\; ; \;\; \tilde{y}_p = y_p + \varepsilon_y$$

$$\tilde{x}_n = x_n . \varepsilon_{x\_mult} + \varepsilon_{x\_add}$$

where

$$\varepsilon_{x\_mult} = \frac{1}{1 + \frac{\varepsilon_x l_{3D} s_\beta - \varepsilon_y b_{3D} s_\alpha c_\beta}{x_p l_{3D} s_\beta + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

$$\varepsilon_{x\_add} = \frac{\frac{\varepsilon_y b_{3D}((M(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma) + t_x(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma) + H c_\beta s_\gamma) + \varepsilon_x l_{3D}(H(c_\alpha c_\gamma - s_\alpha s_\beta s_\gamma) - M c_\beta c_\gamma - t_y(s_\alpha c_\gamma + c_\alpha s_\beta s_\gamma))}{x_p l_{3D} s_\beta + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}{1 + \frac{\varepsilon_x l_{3D} s_\beta - \varepsilon_y b_{3D} s_\alpha c_\beta}{x_p l_{3D} s_\beta + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

$$\tilde{y}_n = y_n . \varepsilon_{y\_mult} + \varepsilon_{y\_add}$$

where

$$\varepsilon_{y\_mult} = \frac{1}{1 + \frac{\varepsilon_x l_{3D} s_\beta - \varepsilon_y b_{3D} s_\alpha c_\beta}{x_p l_{3D} s_\beta + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

$$\varepsilon_{y\_add} = \frac{\frac{\varepsilon_y b_{3D}((M(c_\alpha c_\gamma - s_\alpha s_\beta s_\gamma) + t_x(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) + H c_\beta c_\gamma) + \varepsilon_x l_{3D}(H(c_\alpha s_\gamma - s_\alpha s_\beta c_\gamma) + M c_\beta c_\gamma + t_y(s_\alpha s_\gamma - c_\alpha s_\beta c_\gamma))}{x_p l_{3D} s_\beta + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}{1 + \frac{\varepsilon_x l_{3D} s_\beta - \varepsilon_y b_{3D} s_\alpha c_\beta}{x_p l_{3D} s_\beta + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

- Error derived from imprecision in prediction from the Kalman filter:

❖ $\tilde{\alpha} = \alpha + \varepsilon_\alpha$

$$\tilde{x}_n = x_n . \varepsilon_{\alpha\_x\_mult} + \varepsilon_{\alpha\_x\_add}$$

$$\varepsilon_{\alpha\_x\_mult} = \frac{1}{1 - \frac{F s_\alpha c_\beta + y_p b_{3D} c_\alpha c_\beta}{x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\alpha} + \frac{(1 - c_{\varepsilon_\alpha})}{(c_{\varepsilon_\alpha})} \cdot \frac{x_p l_{3D} s_\beta}{x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

$$\varepsilon_{\alpha\_x\_add} = \frac{\frac{1}{F(x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)} \cdot \left[\frac{(1 - c_{\varepsilon_\alpha})}{c_{\varepsilon_\alpha}} \cdot (M.C c_\beta c_\gamma - F^2 t_y c_\beta s_\gamma + B.H c_\beta s_\gamma) + T t_{\varepsilon_\alpha} \right]}{1 - \frac{F s_\alpha c_\beta + y_p b_{3D} c_\alpha c_\beta}{x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\alpha} + \frac{(1 - c_{\varepsilon_\alpha})}{(c_{\varepsilon_\alpha})} \cdot \frac{x_p l_{3D} s_\beta}{x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

$$T = [M.B(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) + M.F^2(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma) + B.t_x(-s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma) + F^2 t_x(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma) - C.H(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma) + C.t_y(s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma)]$$

$$\tilde{y}_n = y_p . \varepsilon_{\alpha\_y\_mult} + \varepsilon_{\alpha\_y\_add}$$

$$\varepsilon_{\alpha\_y\_mult} = \frac{1}{1 - \frac{F s_\alpha c_\beta + y_p b_{3D} c_\alpha c_\beta}{x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\alpha} + \frac{(1 - c_{\varepsilon_\alpha})}{(c_{\varepsilon_\alpha})} \cdot \frac{x_p l_{3D} s_\beta}{x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

$$\varepsilon_{\alpha\_y\_add} = \frac{\frac{1}{F(x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)} \cdot \left[\frac{(1 - c_{\varepsilon_\alpha})}{c_{\varepsilon_\alpha}} \cdot (-M.Cc_\beta c_\gamma - F^2 t_y c_\beta c_\gamma + B.Hc_\beta c_\gamma) + Tt_{\varepsilon_\alpha} \right]}{1 - \frac{Fs_\alpha c_\beta + y_p b_{3D} c_\alpha c_\beta}{x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\alpha} + \frac{(1 - c_{\varepsilon_\alpha})}{(c_{\varepsilon_\alpha})} \cdot \frac{x_p l_{3D} s_\beta}{x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta}}$$

$$T = [M.B(-c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma) - M.F^2(s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma) - Bt_x(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma) - F^2 t_x(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) - C.H(c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma) + Ct_y(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma)]$$

❖ $\widetilde{\beta} = \beta + \varepsilon_\beta$

$$\widetilde{x}_n = x_n . \varepsilon_{\beta\_x\_mult} + \varepsilon_{\beta\_x\_add}$$

$$\varepsilon_{\beta\_x\_mult} = \frac{1}{1 - \frac{x_p l_{3D} c_\beta - Fc_\alpha s_\beta + y_p b_{3D} s_\alpha s_\beta}{x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\beta}}$$

$$\varepsilon_{\beta\_x\_add} = \frac{\frac{1}{F(x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)} \cdot \left[\frac{(1 - c_{\varepsilon_\beta})}{c_{\varepsilon_\beta}} \cdot (M.Bc_\alpha s_\gamma + M.F^2 s_\alpha s_\gamma + (C.H - F^2 t_x) c_\alpha c_\gamma + (Bt_x - Ct_y) s_\alpha c_\gamma) + Tt_{\varepsilon_\beta} \right]}{1 - \frac{x_p l_{3D} c_\beta - Fc_\alpha s_\beta + y_p b_{3D} s_\alpha s_\beta}{x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\beta}}$$

$$T = M.B.s_\alpha c_\beta c_\gamma - M.Cs_\beta c_\gamma - M.F^2 c_\alpha c_\beta c_\gamma + (F^2 t_x - C.H) s_\alpha c_\beta s_\gamma + F^2 t_y s_\beta s_\gamma + (Bt_x - Ct_y) c_\alpha c_\beta s_\gamma - B.Hs_\beta s_\gamma$$

$$\widetilde{y}_n = y_n . \varepsilon_{\beta\_y\_mult} + \varepsilon_{\beta\_y\_add}$$

$$\varepsilon_{\beta\_y\_mult} = \frac{1}{1 - \frac{x_p l_{3D} c_\beta - Fc_\alpha s_\beta + y_p b_{3D} s_\alpha s_\beta}{x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\beta}}$$

$$\varepsilon_{\beta\_y\_add} = \frac{\frac{1}{F(x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)} \cdot \left[\frac{(1 - c_{\varepsilon_\beta})}{c_{\varepsilon_\beta}} \cdot (M.Bc_\alpha c_\gamma + M.F^2 s_\alpha c_\gamma + (C.H - F^2 t_x) c_\alpha s_\gamma - (Bt_x - Ct_y) s_\alpha c_\gamma) + Tt_{\varepsilon_\beta} \right]}{1 - \frac{x_p l_{3D} c_\beta - Fc_\alpha s_\beta + y_p b_{3D} s_\alpha s_\beta}{x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta} \cdot t_{\varepsilon_\beta}}$$

$$T = -M.Bs_\alpha c_\beta s_\gamma + M.Cs_\beta c_\gamma + M.F^2 c_\alpha c_\beta s_\gamma - (F^2 t_x + C.H) s_\alpha c_\beta c_\gamma + F^2 t_y s_\beta c_\gamma + (Bt_x - Ct_y) c_\alpha c_\beta c_\gamma - B.Hs_\beta c_\gamma$$

❖ $\widetilde{\gamma} = \gamma + \varepsilon_\gamma$

$$\widetilde{x}_n = x_n . \varepsilon_{\gamma\_x\_mult} + \varepsilon_{\gamma\_x\_add}$$

$$\varepsilon_{\gamma\_x\_mult} = c_{\varepsilon\gamma}$$

$$\varepsilon_{\gamma\_x\_add} = \frac{s_{\varepsilon_\gamma}}{F(x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)} \cdot [M.B(c_\alpha c_\gamma - s_\alpha s_\beta s_\gamma) - M.Cc_\beta s_\gamma + M.F^2(s_\alpha c_\gamma + c_\alpha s_\beta s_\gamma) + Bt_x(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) + F^2 t_x(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma) - F^2 t_y . c_\beta c_\gamma - C.H(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma) - C.t_y(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma) + B.Hc_\beta c_\gamma$$

$$\widetilde{y}_n = y_n . \varepsilon_{\gamma\_y\_mult} + \varepsilon_{\gamma\_y\_add}$$

$$\varepsilon_{\gamma\_y\_mult} = c_{\varepsilon\gamma}$$

$$\varepsilon_{\gamma\_y\_add} = \frac{s_{\varepsilon_\gamma}}{F(x_p l_{3D} + Fc_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)} \cdot [-M.B(c_\alpha s_\gamma + s_\alpha s_\beta c_\gamma) + M.Cc_\beta s_\gamma - M.F^2(s_\alpha s_\gamma - c_\alpha s_\beta c_\gamma) - Bt_x(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma) + F^2 t_x(s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma) + F^2 t_y . c_\beta s_\gamma + C.H(s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma) + C.t_y(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma) - B.Hc_\beta s_\gamma$$

❖ $\widetilde{t}_x = t_x + \varepsilon_{t_x}$

$$\tilde{x}_n = x_n + \varepsilon_{tx\_x\_add}$$

$$\varepsilon_{tx\_x\_add} = \varepsilon_{t_x} \cdot \frac{F^2(s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma) + B(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma)}{F(x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)}$$

$$\tilde{y}_n = y_n + \varepsilon_{tx\_y\_add}$$

$$\varepsilon_{tx\_y\_add} = \varepsilon_{t_x} \cdot \frac{-F^2(s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma) + B(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)}{F(x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)}$$

❖ $\tilde{t}_y = t_y + \varepsilon_{t_x}$

$$\tilde{x}_n = x_n + \varepsilon_{ty\_x\_add}$$

$$\varepsilon_{ty\_x\_add} = -\varepsilon_{t_y} \cdot \frac{F^2 c_\beta s_\gamma + C(c_\alpha s_\beta s_\gamma + s_\alpha c_\gamma)}{F(x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)}$$

$$\tilde{y}_n = y_n + \varepsilon_{ty\_y\_add}$$

$$\varepsilon_{ty\_y\_add} = -\varepsilon_{t_y} \cdot \frac{-F^2 . c_\beta c_\gamma - C(c_\alpha s_\beta c_\gamma - s_\alpha s_\gamma)}{F(x_p l_{3D} + F . c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)}$$

❖ $\tilde{t}_z = t_z + \varepsilon_{t_z}$

$$\tilde{x}_n = x_n + \varepsilon_{t_z\_x\_add}$$

$$\varepsilon_{t_z\_x\_add} = -\varepsilon_{t_z} \cdot \frac{B c_\beta c_\gamma - C(s_\alpha s_\beta s_\gamma - c_\alpha c_\gamma)}{F(x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)}$$

$$\tilde{y}_n = y_n + \varepsilon_{t_z\_y\_add}$$

$$\varepsilon_{t_z\_y\_add} = -\varepsilon_{t_z} \cdot \frac{B c_\beta c_\gamma - C(s_\alpha s_\beta c_\gamma + c_\alpha s_\gamma)}{F(x_p l_{3D} + F c_\alpha c_\beta - y_p b_{3D} s_\alpha c_\beta)}$$

REMARKS

Error expressions depend on the specific projection system parameters (F, l_{3D}, b_{3D}, etc.), the pose that the head model shows (α, β, γ, t_x, t_y, t_z) and the value of the projected coordinates obtained from the analysis (x_p, y_p).

Tables 1, 2, 3, 4, 5, 6, 7 and 8 illustrate the error evolution when the analyzed head is almost its neutral position ($\alpha = \beta = \gamma = t_x = t_y = t_z \approx 0$), assuming only one source of error at a time. Additive and multiplying error behavior evolves with the changes in the additive error of the analyzed results. Tables 1 and 2 show the error evolution in terms of percentage of error committed when analyzing the projected data (x_p, y_p); Tables 6, 7 and 8 the percentage of error from the Kalman filter translation parameter prediction; and Tables 3, 4 and 5 show the error in radian units due to the Kalman prediction inaccuracy over the pose angles. $f(\ )$, $f'(\ )$, $g(\ )$ and $g'(\ )$ are different functions that depend on the specific projection system parameters. $h(\ )$, $h'(\ )$, $h''(\ )$, $h'''(\ )$, $k(\ )$, $k'(\ )$, $k''(\ )$, $k'''(\ )$, $m(\ )$, $m'(\ )$, $n(\ )$, $n'(\ )$, $q(\ )$, $q'(\ )$, $p(\ )$ and $p'(\ )$ are functions that depend on the system parameters and the analyzed values x_p and y_p.

Bearing in mind that these tables represent the results over a specific situation, we can still study the influence trends over the error behavior and which parameters seem to be more critical for a correct coupling result. Exact error values depend on the instant system characteristics but all specific functions have the same order of magnitude ($O(10^0)$) and it is independent of the system and the obtained results.

In the evolution study, we see that the instant pose can greatly influence the error behavior. For instance, the projected data multiplying error is minimal when the head is in its neutral position (Tables 1 and 2). In all cases and for small inaccuracy errors, the overall behavior shows a clean linear evolution. Pose parameters related to the x and y-axis (α, β, t_x and t_y) have similar comportment. Rotations around the x and the y axis have stronger action over the error results than the rotation related to the z-axis. In fixed pose conditions γ rotation, t_x and t_y do not change the image appearance of the studied ROI and therefore errors due to inaccuracy in their prediction have less impact on the neutral coordinates.

As a final conclusion, we deduce that the Kalman filter error influence is larger than the accuracy of the analyzed data. First, it involves the greatest amount of parameters; second, it is the basis for correct ROI framing that helps to obtain the optimal minimal energy point search. Analyzing the expressions, we realize that when coupling pose prediction with feature expression analysis, the inaccuracy of pose predicting becomes the major source of error. Unlike the error introduced when analyzing the video image, the prediction error could hardly be minimized by any image analysis technique and therefore cannot be easily controlled. This error analysis shows how critical the accuracy of the prediction is for our method. Figure 7[1] depicts several graphs were the Kalman predicted filter results are plotted against the real pose data. Movement must be smooth in order to obtain the best Kalman filter action and minimize the filter artifacts. Otherwise, they would introduce too many errors and therefore they would mislead the feature expression analysis results.

[1] Extracted from [SV99]

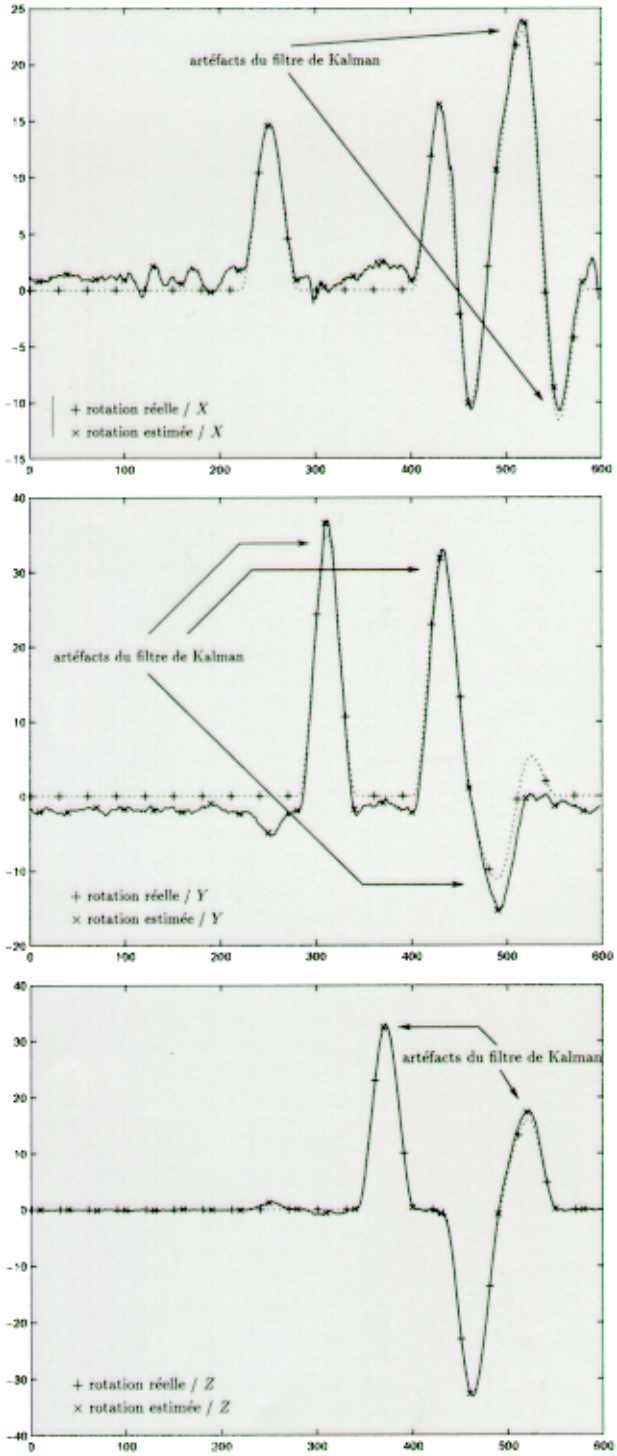


Figure 7. Comparing real and estimated pose parameters from the Kalman filter.

Table 1. Evolution of the error due to the inaccuracy on the x component of projected data analysis.

Error % x_p	error % x_{mult}	error % x_{add}	error % y_{mult}	error % y_{add}
10	0	10.f(system)	0	10.f'(system)
5	0	5.f(system)	0	5.f'(system)
1	0	f(system)	0	f'(system)
0.5	0	0.5.f(system)	0	0.53.f'(system)
0.1	0	0.1.f(system)	0	0.1.f'(system)
0.05	0	0.05.f(system)	0	0.05.f'(system)
0.01	0	0.01.f(system)	0	0.01.f'(system)

Table 2. Evolution of the error due to the inaccuracy on the y component of projected data analysis.

error % y_p	error % x_{mult}	error % x_{add}	error % y_{mult}	error % y_{add}
10	0	10.g(system)	0	10.g'(system)
5	0	5.g(system)	0	5.g'(system)
1	0	g(system)	0	g'(system)
0.5	0	0.5.g(system)	0	0.5.g'(system)
0.1	0	0.1.g(system)	0	0.1.g'(system)
0.05	0	0.05.g(system)	0	0.05.g'(system)
0.01	0	0.01.g(system)	0	0.01.g'(system)

Table 3. Evolution of the error due to the inaccuracy on the α pose parameter prediction.

error α rad	error x_{mult}	error x_{add}	error y_{mult}	error y_{add}
1	1/(1-h(sys,coor). 0.0175)	(0.000152.h'(sys,coor)+0.0175.h''(sys,coor))/ (1-h(sys,coor). 0.0175)	1/(1-h(sys,coor). 0.0175)	(0.000152.h'''(sys,coor)+0.0175.h''(sys,coor))/ (1-h(sys,coor).0.0175)
0.5	1/(1-h(sys,coor).0.00873)	(0.0000381.h'(sys,coor)+0.00873.h''(sys,coor))/ (1-h(sys,coor).0.00873)	1/(1-h(sys,coor).0.00873)	(0.0000381.h'''(sys,coor)+0.00873.h''(sys,coor))/ (1-h(sys,coor).0.00873)
0.1	1/(1-h(sys,coor).0.00175)	(0.00000152.h'(sys,coor)+0.00175.h''(sys,coor))/ (1-h(sys,coor).0.00175)	1/(1-h(sys,coor).0.00175)	(0.00000152.h'''(sys,coor)+0.00175.h''(sys,coor))/ (1-h(sys,coor).0.00175)
0.05	1/(1-h(sys,coor)*0.000873)	(0.000000381.h'(sys,coor)+0.000873.h''(sys,coor))/ (1-h(sys,coor).0.000873)	1/(1-h(sys,coor).0.000873)	(0.000000381.h'''(sys,coor)+0.000873.h''(sys,coor))/ (1-h(sys,coor),0.000873)
0.01	1/(1-h(sys,coor)*0.000175)	(0.0000000152.h'(sys,coor)+0.000175.h''(sys,coor))/ (1-h(sys,coor).0.000175)	1/(1-h(sys,coor).0.000175)	(0.0000000152.h'''(sys,coor)+0.000175.h''(sys,coor))/ (1-h(sys,coor).0.000175)

Table 4. Evolution of the error due to the inaccuracy on the β pose parameter prediction.

error β rad	error x_{mult}	error x_{add}	error y_{mult}	error y_{add}
1	1/(1-k(sys,coor).0.0175)	(0.000152.k''(sys,coor)+0.0175.k'(sys,coor))/ (1-k(sys,coor).0.0175)	1/(1-k(sys,coor).0.0175)	(0.000152.k'''(sys,coor)+0.0175.k''(sys,coor))/ (1-k(sys,coor).0.0175)
0.5	1/(1-k(sys,coor).0.00873)	(0.0000381.k'(sys,coor)+0.00873.k''(sys,coor))/ (1-k(sys,coor).0.00873)	1/(1-k(sys,coor).0.00873)	(0.0000381.k'''(sys,coor)+0.00873.k''(sys,coor))/ (1-k(sys,coor).0.00873)
0.1	1/(1-k(sys,coor).0.00175)	(0.00000152.k'(sys,coor)+0.00175.k''(sys,coor))/ (1-k(sys,coor).0.00175)	1/(1-k(sys,coor).0.00175)	(0.00000152.k'''(sys,coor)+0.00175.k''(sys,coor))/ (1-k(sys,coor).0.00175)
0.05	1/(1-k(sys,coor).0.000873)	(0.000000381.k'(sys,coor)+0.000873.k''(sys,coor))/ (1-k(sys,coor).0.000873)	1/(1-k(sys,coor).0.000873)	(0.000000381.k'''(sys,coor)+0.000873.k''(sys,coor))/ (1-k(sys,coor).0.000873)
0.01	1/(1-k(sys,coor).0.000175)	(0.0000000152.k'(sys,coor)+0.000175.k''(sys,coor))/ (1-k(sys,coor).0.000175)	1/(1-k(sys,coor).0.000175)	(0.0000000152.k'''(sys,coor)+0.000175.k''(sys,coor))/ (1-k(sys,coor).0.000175)

Table 5. Evolution of the error due to the inaccuracy on the γ pose parameter prediction.

error γ rad	error x_{mult}	error x_{add}	error y_{mult}	error y_{add}
1	0.99984	0.0175.m(sys,coor)	0.0175	0.99984.m'(sys,coor)
0.5	0.999961	0.00873.m(sys,coor)	0.00873	0.999961.m'(sys,coor)
0.1	0.9999984	0.00175.m(sys,coor)	0.00175	0.9999984.m'(sys,coor)
0.05	0.99999961	0.000873.m(sys,coor)	0.000873	0.99999961.m'(sys,coor)
0.01	0.999999984	0.000175.m(sys,coor)	0.000175	0.999999984.m'(sys,coor)

Table 6. Evolution of the error due to the inaccuracy on the t_x pose parameter prediction.

error % t_x	error % x_{mult}	error % x_{add}	error % y_{mult}	error % y_{add}
10	none	10.*n*(system,coor)	none	10.*n*'(system,coor)
5	none	5.*n*(system,coor)	none	5.*n*'(system,coor)
1	none	*n*(system,coor)	none	*n*'(system,coor)
0.5	none	0.5.*n*(system,coor)	none	0.5.*n*'(system,coor)
0.1	none	0.1.*n*(system,coor)	none	0.1.*n*'(system,coor)
0.05	none	0.05.*n*(system,coor)	none	0.05.*n*'(system,coor)
0.01	none	0.01.*n*(system,coor)	none	0.01.*n*'(system,coor)

Table 7. Evolution of the error due to the inaccuracy on the t_y pose parameter prediction.

error % t_y	error % x_{mult}	error % x_{add}	error % y_{mult}	error % y_{add}
10	none	10.*p*(system,coor)	none	10.*p*'(system,coor)
5	none	5.*p*(system,coor)	none	5.*p*'(system,coor)
1	none	*p*(system,coor)	none	*p*'(system,coor)
0.5	none	0.5.*p*(system,coor)	none	0.5.*p*'(system,coor)
0.1	none	0.1.*p*(system,coor)	none	0.1.*p*'(system,coor)
0.05	none	0.05.*p*(system,coor)	none	0.05.*p*'(system,coor)
0.01	none	0.01.*p*(system,coor)	none	0.01.*p*'(system,coor)

Table 8. Evolution of the error due to the inaccuracy on the t_z pose parameter prediction.

error % *tz*	error % x_{mult}	error % x_{add}	error % y_{mult}	error % y_{add}
10	none	10.*q*(system,coor)	none	10.*q*'(system,coor)
5	none	5.*q*(system,coor)	none	5.*q*'(system,coor)
1	none	*q*(system,coor)	none	*q*'(system,coor)
0.5	none	0.5.*q*(system,coor)	none	0.5.*q*'(system,coor)
0.1	none	0.1.*q*(system,coor)	none	0.1.*q*'(system,coor)
0.05	none	0.05.*q*(system,coor)	none	0.05.*q*'(system,coor)
0.01	none	0.01.*q*(system,coor)	none	0.01.*q*'(system,coor)

References

[SV99] S. Valente, "Analyse, Synthèse et Animation de Clones dans un Contexte de Téléreunion Virtuelle," *Ph.D. Thesis* - Institut Eurecom - EPFL, November 1999.

[OGL99] M. Woo et al., "OpenGL Programming guide," Third Edition. *Addison-Wesley*. 1999.

[AVD01] A. C. Andrés del Valle, and J.-L. Dugelay, "Eye State Tracking for Face Cloning," *ICIP 2001*, Thessaloniki - Greece, 7-10 October 2001.