Visual Interpretation of Gaze Pointing Actions for Interaction with 2D/3D Graphic Environments

Carlo Colombo, *Member, IEEE*
Dipartimento di Elettronica per l'Automazione
Università di Brescia
Via Branze 38, I-25123 Brescia, ITALY

Alberto Del Bimbo, *Member, IEEE*
Dipartimento di Sistemi e Informatica
Università di Firenze
Via Santa Marta 3, I-50139 Firenze, ITALY

Abstract

Computer vision is emerging as a powerful technology for non intrusive human-machine interfacing. In this paper, we present a methodology which uses computer vision for interacting with 2D and 3D graphic interfaces through gaze pointing actions. Once intercepted, tracked and interpreted using novel algorithms, user shifts of visual interest as related to head or eye pupil displacements are *remapped* onto the graphic environment according to a simple metaphor providing interaction with a natural semantics. Two different *visual pointers* are introduced, which allow to use the pupil as a 2D mouse and the head as a 3D mouse (6 DOF): such non conventional pointers allow to perform navigation and selection actions in the interaction environment in a decoupled fashion. The approach, presented and evaluated here in a non immersive virtual reality context provided with hypertextual access, can be easily adapted to support different scenarios and applications.

Index Terms: Human-Computer Interaction, Vision Based Human-Machine Interfaces, Computer Vision, Visual Tracking and Remapping

1 Introduction

In the last two decades, human-computer interaction modalities have continually evolved [1], [2], [3]. Interfaces have evolved accordingly to display on the same screen multiple operating contexts using the windowing paradigm and hypertext/hypermedia facilities [4], [5]. Textual languages for keyboard-driven interaction have been gradually complemented by 2D and 2D$\frac{1}{2}$ iconic dialogue modalities based on mouse pointers; in the same way, a strong trend nowadays is to develop systems based on natural interaction modalities inspired from human behavior in the real world [6], [7]. Such advanced environments improve both the activity and satisfaction of individual users and computer supported cooperative work [8], [9].

Apart from some obvious implementation and design differences, virtual reality [10], [11], [12], augmented reality [13] and smart room [14] environments share the same principle of providing users with a more natural dialogue with (and through) the computer than in the past.

Several tools and devices have been developed, such as head-mounted displays, datagloves, touch screen systems, body markers etc. in order to support increasingly sophisticated interfaces, and handle many different interaction modalities and user needs. Yet, much work has to be done with such devices, which are often very accurate but also quite expensive, intrusive and not comfortable to use. For instance, most of the gaze driven interaction devices, which sense information related to the user's direction of gaze, are head mounted eyetrackers with some embedded infrared systems, provided with a transmitter and a receiver, which evaluates the current pupil position [15]; similar principles are exploited in the "what you look at is what you get" (WYLAIWYG) strategy to infer where the user is actually looking in the screen using infrared light equipment [16], [17], [18]; other gaze based systems, used for both experiments in psychophysics and medical diagnosis of eye pathologies, use either special light-emitting contact lenses or electrodes placed around the eye, which measure the contraction of the ocular muscles [19].

Recently, the growth of computing power, the decrease of hardware costs and the development of specific algorithms and techniques have encouraged the use of computer vision as a non intrusive technology for advanced human-machine interaction [20], [21]. Experiments have been carried out using computer vision for localizing and tracking user body parts such as head [22], [23], arms [24], hands [25], [26] and facial parameters

such as lips and eyes [27], [28], [29], [30]. Some attempts have also been made to assign a semantics to the spatial configurations and temporal evolution of the tracked parameters, with the purpose of developing interaction-oriented gesture interpretation tasks such as lip-reading [31], recognition of facial expressions [32], [33] and hand gestures [34], [35] from the machine visual analysis of human behavior.

In this paper, we present a computer vision based human-machine interaction methodology to communicate with 2D and 3D graphic environments through pupil shifts and head motion. Interaction via head and eyes can be effective for both disabled users affected by severe limb motor pathologies and general users requiring a non intrusive modality of interaction with the computer. The approach is presented in a non immersive virtual reality context provided with hypertextual access: interaction takes place separately at the bidimensional and tridimensional levels, thus allowing to support in a decoupled fashion both the 2D window based hypertextual interface and the 3D realistic graphic environment. Both head and eyes are continuously tracked in 2D, and 3D information related to head displacement and pointed screen location is then inferred using original computer vision algorithms. Once intercepted, the user's shift of interest is *remapped* onto the environment of interaction through computer graphics, according to a natural interface semantics based on the introduction of *visual pointers* mimicking the basic functionalities of dragging and clicking with conventional pointers. Two distinct visual pointers — for interaction in 2D and 3D respectively — are introduced, which can be used to perform pointing and navigation (drag) as well as selection (click) actions. The proposed framework, which allows to regard computer vision as an interaction methodology in the general context of interface design, can be easily adapted to other tasks and scenarios, so as to support interaction in multimedia, videoconferencing, telepresence, usability monitoring, and augmented reality environments.

The paper is organized as follows. In the next Section we highlight the geometric and semantic aspects of our interaction approach, and state its basic principles and constraints on operation. The visual techniques devised to estimate, track and interpret user's shifts of interest are discussed in Sect. 3. Sect. 4 describes the interaction with our augmented virtual environment, and discusses system performance and its implications in interface design. Finally, Sect. 5 is devoted to draw the conclusions and sketch directions of future work.

2 Natural Vision Based Interaction

Our operating context is composed by a camera and an on-screen multi-window environment featuring a 3D realistic graphic scene and a 2D interface using a camera placed in front of the user. The user's head and eye movements are tracked through computer vision, and encoded into a set of visual parameters reflecting the geometry of user action. Visual parameters are mapped onto commands for 2D or 3D display, and rendered via graphical synthesis according to the semantics of the interface.

In the following they will be briefly discussed geometrical issues concerning user action, and a metaphor used in the human-computer dialogue.

2.1 Geometric Issues

In our vision based interaction approach, the displacements of both the head and the eyeball subsystems are tracked and processed simultaneously: the resulting interaction device is a combination of a 3D and a 2D pointer, respectively. As the two subsystems are kinematically independent, the associated pointers can be made independent one from the other by decoupling the measurement of head and eyeball displacements through computer vision. Such a context can be described by six degrees of freedom (DOF) for the position and orientation of the head in space and two DOF for the orientation of the visual axis in space measured in some fixed reference frame. (Concerning the eyeball DOF, we will actually refer in the rest of the paper to the intersection of the visual axis, which gives the direction of gaze, with the screen plane, which provides the location currently observed.)

The basic elements of the interaction — camera (image plane), screen and user's face, i.e. the visible portion of the head, save the nose, which provides sufficient information to grab head movements — can be modeled as planar surfaces in space. While the screen and camera planes are intrinsically flat surfaces, the face is assumed approximately flat thanks to its relative distance from the other elements of the interaction. For the same reason, we model also the visible portions of the user's eyeballs, actually pseudo-spheres, as having a planar layout, and regard *any gaze shift related to an eyeball rotation as a translation of the pupil in the face plane.* Let A, B and Γ denote respectively the screen, user and camera planes, and fix the local frames α, β and γ as in Fig. 1.

A first geometric aspect of the interaction concerns the head's relative geometry

(position and orientation) w.r.t. a given reference frame ρ. This is encoded in the linear transformation between the coordinate representations of a generic point in space $\boldsymbol{p}$ in the β-frame (${}^{\beta}\boldsymbol{p}$) and in the ρ-frame (${}^{\rho}\boldsymbol{p}$). Using homogeneous coordinates for $\boldsymbol{p}$, such a transformation can be compactly described by a 4×4 matrix ${}^{\rho}_{\beta}\mathrm{T}$ s.t. $[{}^{\rho}\boldsymbol{p}^{\mathrm{T}}\ 1]^{\mathrm{T}} = {}^{\rho}_{\beta}\mathrm{T}\ [{}^{\beta}\boldsymbol{p}^{\mathrm{T}}\ 1]^{\mathrm{T}}$. Let us regard the camera as an element of the interaction: as changes of frame are composed linearly, the $\beta \mapsto \rho$ transformation can be reconstructed from the composition of the $\beta \mapsto \gamma$ and $\gamma \mapsto \rho$ transformations:

$$ {}^{\rho}_{\beta}\mathrm{T} = {}^{\rho}_{\gamma}\mathrm{T}\ {}^{\gamma}_{\beta}\mathrm{T} \quad . \tag{1} $$

The $\gamma \mapsto \rho$ transformation is time-independent, at least as long as the camera is fixed, while the $\beta \mapsto \gamma$ transformation depends on the current position of the user's head.

A second aspect of the interaction involves user's eyeball movements and screen inspection. As the result of an eyeball shift, a specific location of the screen is pointed, and its visible content is projected through the pupil onto the highest-resolution area of the retina, the fovea [36]. Thus, geometrically speaking, the pupil position $\boldsymbol{b}$ for eye $E \subset B$ (the right one in Fig. 1), can be regarded at each instant as the perspective image of the observed screen point $\boldsymbol{a}$, i.e. $\boldsymbol{b} = {}^{E}_{A}f(\boldsymbol{a})$, with ${}^{E}_{A}f : A \to E$ a nonlinear projection map depending, among other parameters, on the entries of ${}^{\beta}_{\alpha}\mathrm{T}$. Thus from a pure geometric viewpoint, *the face plane can be regarded as the image plane of a camera with only one photo-sensitive element, which is allocated to the pixel corresponding to the current pupil position* $\boldsymbol{b}$. It is worth noting that the relative distance between the user and the camera is such that, in our context, conditions are met to approximate the perspective camera with an affine camera model [37]. This allows us to say that a linear backprojection map ${}^{E}_{A}h^{-1} = {}^{A}_{E}h : B \to A$ (with ${}^{E}_{A}h \approx {}^{E}_{A}f$) exists which brings pupil positions to screen locations. The same reasoning is applied to define the camera-to-eye backprojection ${}^{E}_{\Gamma}h : \Gamma \to E$, and write the map ${}^{A}_{\Gamma}g$ between camera-projected pupil positions $\boldsymbol{c}$ and screen locations $\boldsymbol{a}$ as a composition of linear maps:

$$ {}^{A}_{\Gamma}g = {}^{A}_{E}h \circ {}^{E}_{\Gamma}h : \Gamma \to A \quad . \tag{2} $$

Such a map is a linear approximation to a perspectivity (the "picture of a picture" [37]). The camera-to-screen backprojection is time-dependent, since it is the product of two maps which involve the user frame β.

The transformations of eqs. (1) and (2) embed the information used by our 3D and 2D pointers, respectively. In Subsect. 3.2 we show how to compute and update such information based on image data.

2.2 Interaction Semantics: Drag and Click Metaphor

The dialogue semantics currently implemented in our visual interface is based on a *drag and click metaphor*, which lets the user interact with the environment with his head and eyeballs to perform navigational, pointing and selection actions as with conventional pointing devices. The idea behind the use of this metaphor is to achieve a natural dialogue with the machine, through an interpreter which reduces to a minimum the user's capability of producing discrete events for interaction, while exploiting at best the rich interaction possibilities at the continuous-geometric level provided by the 3D and 2D visual pointers. Explicitly, head and eyeball displacements w.r.t. a fixed reference are interpreted as pointer drags, or *navigational actions*. Pointers can trigger a *selection action* (click) as they persist in the neighborhood of a geometric configuration for a convenient time interval. Since the production of discrete events is reduced here to the on-off mechanism of time thresholding for selection, *pointers have basically the same expressivity of a 1-button mouse*.

Tab. 1 describes operation in a 2D/3D interaction environment, showing the meaning assigned to drags and clicks with the 2D and the 3D visual pointers, respectively. The head can be used to navigate or to displace objects (drag) and to select, or "freeze," a 3D scene of interest in it (click). Eyeball drags, which take into account a displacement within the scene, can be used e.g. for on-screen drawing or exploration, and eyeball clicks to select relevant 2D information when the scene has been selected.

3 Implementation

Fig. 2 offers a block-description of interface operation; the function of each block is illustrated later in this Section. Notice that sensing the user is the result of a sequential process of extracting three-dimensional information from raw bidimensional image data; this information is then processed further to produce, according to the implemented interaction semantics and state of the environment, spatio-temporal events which are

used for interface control. A typical event includes geometric information about where in the environment a change will take place, and the semantic information of whether a drag or a click operation will be carried out. The semantic process of translating user action into changes of the environment is referred here to as *remapping*.

3.1 Projection and User Modeling

3.1.1 Affine Projection Model

The relative geometry between the user and camera frames has six DOF which, as stated in Subsect. 2.1, are embedded in the matrix ${}^{\gamma}_{\beta}\mathrm{T}$ describing the change of coordinate representation for the generic point $\boldsymbol{p}$. It holds

$$\begin{bmatrix} {}^{\gamma}\boldsymbol{p} \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{\gamma}_{\beta}\mathrm{R} & {}^{\gamma}_{\beta}\boldsymbol{t} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{\beta}\boldsymbol{p} \\ 1 \end{bmatrix} \quad , \tag{3}$$

where ${}^{\gamma}_{\beta}\boldsymbol{t}$ is a translation 3-vector, and

$${}^{\gamma}_{\beta}\mathrm{R} = \begin{bmatrix} \cos\tau & -\sin\tau & 0 \\ \sin\tau & \cos\tau & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\sigma & 0 & \sin\sigma \\ 0 & -1 & 0 \\ -\sin\sigma & 0 & \cos\sigma \end{bmatrix} \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

is a 3×3 rotation matrix, completely defined by the three angles τ (tilt), σ (slant) and ϕ (orientation) shown in Fig. 3. Specifically, the slant $\sigma \in [0, \pi/2]$ is the angle between the face and the image planes; this vanishes identically if the two planes are parallel. The tilt angle $\tau \in [0, 2\pi]$ gives instead the image direction of maximum depth decrease. Thus, the face plane is expressed in camera coordinates as

$${}^{\gamma}Z({}^{\gamma}X, {}^{\gamma}Y) = c \, - \, \tan\sigma \, [\cos\tau \quad \sin\tau \,]^{\mathrm{T}} \cdot [\, {}^{\gamma}X \quad {}^{\gamma}Y \,]^{\mathrm{T}} \quad , \quad c \in \mathbb{R} \quad . \tag{5}$$

Perspective projection ${}^{\Gamma}_{E}f$ of an eye point ${}^{\gamma}\boldsymbol{p} = [\, {}^{\gamma}X \quad {}^{\gamma}Y \quad {}^{\gamma}Z \,]^{\mathrm{T}}$ onto the camera plane point $\boldsymbol{x} = [\, x \quad y \,]^{\mathrm{T}}$ is given by

$$\boldsymbol{x} = \frac{\lambda}{{}^{\gamma}Z} \begin{bmatrix} {}^{\gamma}X \\ {}^{\gamma}Y \end{bmatrix} \quad , \tag{6}$$

where λ denotes the focal length of the camera. The affine approximation ${}^{\Gamma}_{E}h$ of camera projection is obtained by recalling that, thanks to face planarity, ${}^{\beta}Z$ is equal to a constant μ, and by observing from Fig. 1 that the depth of the points of a single eye is virtually constant, and equal to the eye centroid depth ${}^{\gamma}\boldsymbol{p}_{\mathrm{E}} = {}^{\gamma}_{\beta}\boldsymbol{t} + {}^{\gamma}_{\beta}\mathrm{R}\,[0 \quad 0 \quad \mu\,]^{\mathrm{T}}$.

Using such constraints into eqs. (3) through (6), we obtain

$$\boldsymbol{x} = \mathrm{H} \begin{bmatrix} {}^{\gamma}X \\ {}^{\gamma}Y \end{bmatrix} + \boldsymbol{h} \quad , \tag{7}$$

where

$$\begin{cases} \mathrm{H} = \kappa_{\mathrm{E}} \begin{bmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{bmatrix} \begin{bmatrix} \cos\sigma & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \\ \\ \boldsymbol{h} = \kappa_{\mathrm{E}} \begin{bmatrix} {}^{\gamma}X_{\mathrm{E}} \\ {}^{\gamma}Y_{\mathrm{E}} \end{bmatrix} \quad , \quad \kappa_{\mathrm{E}} = \lambda / {}^{\gamma}Z_{\mathrm{E}} \quad . \end{cases} \tag{8}$$

Matrix H is the composition of two planar rotations by τ and ϕ, of an anisotropic scaling by $\cos\sigma$ along a specific image direction, and of an isotropic scaling of the imaged pattern by a factor κ_{E}. Notice that linearizing perspective produces a pose ambiguity, i.e. there are two distinct object poses sharing the same visual appearance. In our case, an ambiguity exists whenever $\sigma \neq 0$, as the angle triplets (τ, σ, ϕ) and $(\tau + \pi, \sigma, \phi + \pi)$ yield exactly the same H. The ambiguity problem must be solved in order to produce a correct pose estimation from an affine projection map.

As with eye-to-camera projection $(\mathrm{H}, \boldsymbol{h})$, another affine map by $(\mathrm{K}, \boldsymbol{k})$ can be associated to screen-to-eye projection. This allows us to define the camera-to-screen affine map $(\mathrm{G}, \boldsymbol{g})$. By expressing image and screen points $\boldsymbol{c}$ and $\boldsymbol{a}$ as 2-vectors using their native coordinate frames γ and α respectively, we have:

$${}^{\alpha}\boldsymbol{a} = \mathrm{G}\, {}^{\gamma}\boldsymbol{c} + \boldsymbol{g} \quad , \tag{9}$$

where $\mathrm{G} = [\mathrm{HK}]^{-1}$ and $\boldsymbol{g} = -\mathrm{G}\,\boldsymbol{h} - \mathrm{K}^{-1}\,\boldsymbol{k}$. The entries of this map depend, among other parameters, on eye focal length μ: they have to be estimated so as to infer the screen positions pointed by the user.

3.1.2 Trackable User Features

Head displacements and pupil positions can be estimated at one time by tracking the visual appearance of the user's eyes. Indeed, the *external contour* of the eye, which is fixed to the head (such a rigidity constraint assumes that any change of the visual appearance of the eye's contour is related to a head displacement. Theoretically speaking, even a nonrigid change of facial expression may influence the external eye's appearance at a short time scale. As our system operates at longer time scales, only geometric-driven

changes are significant in practice. The same applies for eye blinks) can be related to head displacements, while the internal *iris* contour, which is concentric with the pupil, moves according to user pointing actions (eyeball shifts). By analyzing separately the pupil pointing and head displacement characteristics of user's action allows us to adopt a two-step eye tracking approach:

1. *Track the external eye contour.* This determines a search bound for the pupil's position in the image.

2. *Track the pupil inside the search bound.* This is motivated by the physiological constraint that the pupil must always lie inside the external eye contour.

Image measurements for head tracking and pupil detection are obtained by coupling an elastic template with raw image data. At startup, the template is automatically initialized in the image, in order to match the location and shape of the user's eyes. For each of the two eyes, a *reference template* is thus produced, and at run-time a visual tracker is started, which keeps itself locked onto the current external eyes' visual appearance. Besides providing at any time the bound for pupil search, the current tracker's shape and location provide the time-varying information needed to measure and update user action.

3.1.3 Reference

Let us assume that the reference frame ρ is the β-frame at initialization time ($t = 0$), i.e. $\rho = \beta(0)$. Assume also that $\sigma(0) = \tau(0) = \phi(0) = 0$, i.e. the reference and camera planes are parallel, and have mutually parallel X-axes. Then, by eq. (8), the reference template models a frontoparallel view of the eyes, translated by $\boldsymbol{h}(0)$, and undergoing scaling and specular reflection w.r.t. the face plane's content by

$$\mathrm{H}(0) = \kappa_{\mathrm{E}}(0) \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad . \tag{10}$$

It is not difficult to show that there exists and affine relationship between a generic eye view $\boldsymbol{x}(t)$ and the reference eye view $\boldsymbol{x}(0)$:

$$\boldsymbol{x}(t) = \boldsymbol{x}_{\mathrm{E}}(t) + \mathrm{L}(t)\,[\boldsymbol{x}(0) - \boldsymbol{x}_{\mathrm{E}}(0)] \quad , \tag{11}$$

where

$$\mathrm{L}(t) = \mathrm{H}(t)\,\mathrm{H}^{-1}(0) \quad , \tag{12}$$

and where $\boldsymbol{x}_{\mathrm{E}}(t) = \boldsymbol{h}(t)$ is the projection of the ocular center ${}^{\gamma}\boldsymbol{p}_{\mathrm{E}}(t)$, which equals the centroid of the eye projection thanks to the linearity of the mapping.

Fixing $\beta_o = \beta(0)$ as the reference frame allows us also to describe the relative position between the current frame $\beta = \beta(t)$ and the reference as

$$ {}^{\beta_o}_{\beta}\mathrm{T} = \begin{bmatrix} & {}^{\beta_o}_{\beta}\mathrm{R} & & {}^{\beta_o}_{\beta}\boldsymbol{t} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad , \tag{13} $$

where

$$ \begin{cases} {}^{\beta_o}_{\beta}\mathrm{R} = {}^{\beta_o}_{\gamma}\mathrm{R}\,{}^{\gamma}_{\beta}\mathrm{R} \\ {}^{\beta_o}_{\beta}\boldsymbol{t} = {}^{\gamma}_{\beta_o}\mathrm{R}\left[{}^{\gamma}_{\beta}\boldsymbol{t} - {}^{\gamma}_{\beta_o}\boldsymbol{t}\right] \end{cases} \tag{14} $$

account respectively for the *relative orientation* and the *relative translation* being, from eq. (4):

$$ {}^{\beta_o}_{\gamma}\mathrm{R} = {}^{\gamma}_{\beta_o}\mathrm{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \quad . \tag{15} $$

3.2 Visual Measurements

3.2.1 Tracking the User in 2D

The reference template for each eye is split in two sub-templates, which capture the characteristics of the user's external eye and iris, respectively (Fig. 4). The external eye sub-template, made of two semi-ellipses sharing their major axis, depends on six parameters (e_1 through e_6 in the figure), namely the common major axis (e_1), the two minor axes (e_2, e_3), the ocular center's image coordinates (e_4, e_5), and the common orientation (e_6). A circle, parameterized through the coordinates of its center (c_1, c_2) and its radius (c_3), is used for the iris sub-template. Notice that part of the iris is usually occluded by the eyelids, a fact that must be explicitly taken into account in the eye tracking strategy.

Template initialization. At system startup, a raw estimate of left and right eye location and shape in the image is derived, and their reference templates are accordingly initialized.

To speed up processing, the two image regions containing the eyes are first roughly located by means of *edge dominance maps*, i.e. maps which take into account the dominance of brightness gradient $\nabla E(x, y)$ in a given direction [27]. To accomplish this task,

two maps are successively computed at low image resolution. First, an horizontal map $M_h(x, y)$ defined as

$$\begin{cases} M_h(x,y) = 0 & \text{if} \quad |\partial E(x,y)/\partial y| \leq |\partial E(x,y)/\partial x| \\ M_h(x,y) = |\partial E(x,y)/\partial y| & \text{otherwise} \end{cases} \tag{16}$$

is computed, whose histogram in the horizontal direction, $\sum_x M_h(x, y)$, allows us to compute an approximate location of the stripe including the eyes. Then, the desired eye regions are obtained similarly by means of a vertical histogram of the same map, limited to the previously extracted stripe.

Once the regions including the eyes are found, the templates are adjusted against image data by an energy-minimization criterion (see e.g. [29]). A quadratic energy term, function of the $9 = 6 + 3$ eye template parameters, is minimized, so that the template is relaxed inside the eye regions by gradient descent. To the energy term contribute both peaks and valleys of image brightness (modeling respectively the sclera and the iris), and brightness discontinuities. After relaxation, each obtained reference template is used to initialize the run-time tracker, a lightweight process which allows a fast eye tracking behavior.

Tracking the external eye. The tracker is composed of a discrete set of points, initialized at startup by uniformly sampling the reference template's external eye ellipses. The initialized tracker is stored as a *reference tracker* for later use. After the reference tracker is obtained, external eye tracking and iris search can start.

At run-time, the tracker's parameters are refined and updated by means of a simple tracker-to-image fitting approach, based on least squares and the extraction of brightness edge points (Fig. 5, *right*). Thanks to the affine projection model, the tracker can be made quite robust by allowing it to deform only in an affine fashion.

Explicitly, let us regard the tracking process as equivalent to estimating a *dynamic visual state* composed, at a generic time $t \geq 0$, by the tracker points $\{\boldsymbol{x}_i(t)\}$, $i = 1 \ldots n$ and their image velocities. Let the tracker's centroid be $\boldsymbol{x}_{\mathrm{E}}(t)$ and the reference tracker be $\{\boldsymbol{x}_i(0)\}$ with centroid $\boldsymbol{x}_{\mathrm{E}}(0)$ and zero initial velocity. Then external eye tracking proceeds as follows.

1. *State prediction.* A new (time $t + 1$) visual state is predicted based on the current state and a constant velocity model.

2. *Image search.* In a neighborhood — whose size depends on the local estimate uncertainty — of each predicted tracker's point, a local search of edge points (brightness gradient maxima) takes place at each of the n tracker's points and along normal directions to the tracker itself. The set $\{\widetilde{\boldsymbol{x}}_i(t+1)\}$ of edge points (measurement set) is computed by means of a 5-point, recursive coarse-to-fine algorithm based on finite differences [38]. Search intervals along each direction are adaptively adjusted based on the previous tracking results.

3. *Least squares fit.* The LS approximation of the new template centroid $\boldsymbol{x}_{\mathrm{E}}(t+1)$ is simply the centroid (average point) of the measurement set: $\widetilde{\boldsymbol{x}}_{\mathrm{E}}(t+1) = \frac{1}{n}\sum_i \widetilde{\boldsymbol{x}}_i(t+1)$. The 2×2 matrix $\mathrm{L}(t+1)$ of eq. (11) is also estimated via LS as the best approximation $\widetilde{\mathrm{L}}(t+1)$ of the affine transformation about the origin between the measurement set and the reference template. This is done by minimizing the quadratic cost

$$\mathcal{E} = \sum_i \left\| [\widetilde{\boldsymbol{x}}_i(t+1) - \widetilde{\boldsymbol{x}}_{\mathrm{E}}(t+1)] - \widetilde{\mathrm{L}}(t+1)\,[\boldsymbol{x}_i(0) - \boldsymbol{x}_{\mathrm{E}}(0)] \right\|^2 \qquad (17)$$

via the solution of the linear homogeneous system obtained by partial differentiation of $\mathcal{E}$ w.r.t. the unknown four parameters (the entries of $\widetilde{\mathrm{L}}$).

4. *Filtering.* The six parameters of the affine image transformation $[\widetilde{\mathrm{L}}(t+1), \widetilde{\boldsymbol{x}}_{\mathrm{E}}(t+1)]$ are smoothed using a mobile mean filter. For the generic parameter p, the filtered value is

$$\widehat{p}\,(t+1) = w_p\, p(t+1) + (1 - w_p)\,\widehat{p}\,(t) \quad . \qquad (18)$$

To achieve a better control of the tracking process, a different filter gain $w_p \in [0, 1]$ is assigned to each parameter.

5. *Affine state projection.* Finally, once the affine transformation $[\widehat{L}(t+1), \widehat{\boldsymbol{x}}_{\mathrm{E}}(t+1)]$ is obtained, the new tracker is computed as

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_{\mathrm{E}}(t+1) + \widehat{\mathrm{L}}(t+1)\,[\boldsymbol{x}_i(0) - \boldsymbol{x}_{\mathrm{E}}(0)] \quad , \quad i = 1 \ldots n \quad , \qquad (19)$$

with $\boldsymbol{x}_{\mathrm{E}}(t+1) = \widehat{\boldsymbol{x}}_{\mathrm{E}}(t+1)$. The new tracker point velocities can now be estimated from the LS comparison between the new (time $t+1$) and old (time t) tracker instances. The affine projection ensures that at each time t the tracker is an affine-transformed instance of the reference tracker obtained at $t = 0$. Besides,

notice that this tracking approach is independent of the specific shape of the tracker being used, so that different shapes can be tracked by simply changing the reference template — say, using parabolic arcs instead of elliptic arcs.

Locating the pupil. Once the new external eye tracker instance is found, the current iris shape and position are searched according to a technique akin to the one used before. At run-time the iris tracker, initialized at startup using the reference template's iris, can undergo any affine departure from the circular shape, and attain an elliptic shape. A strongly elliptic iris is obtained for large head slant angles. Since the shape of the iris is known in advance, for iris tracking the steps of tracker fitting and affine projection are simultaneous, and carried out by explicitly using the ellipse's analytic expression into the objective function to be minimized. The new raw ellipse parameters are extracted by fitting the predicted points $\widetilde{\boldsymbol{y}} = [\,\widetilde{x} \quad \widetilde{y}\,]^{\mathrm{T}}$ with the generic conic via LS, and then imposing the conic to be an ellipse. That is, once the generic five conic parameters γ_l which minimize the squared error

$$\mathcal{F} = \sum_i \left(\widetilde{x}_i^2 + \gamma_1\,\widetilde{x}_i\widetilde{y}_i + \gamma_2\,\widetilde{y}_i^2 + \gamma_3\,\widetilde{x}_i + \gamma_4\,\widetilde{y}_i + \gamma_5\right)^2 \tag{20}$$

are found, they can be expressed in terms of ellipse center, major and minor axes, and orientation as shown in [39]. After mobile mean filtering, the current center of the iris is taken as the new location of the pupil, and used to update the 2D pointer.

During the prediction step, care should be taken to avoid erroneous results due to possible partial iris occlusions. To this end, the edge points belonging to the external eye tracker are automatically excluded from LS fitting with the iris tracker; in this way, an external contour point is not taken into account as a predicted iris point. To avoid occlusions, we use edge search lines which are radial paths connecting external eye tracker's points with the tracker's centroid: the search proceeds inwards and, as shown in Fig. 5 (*left*), only negative gradient points are taken into account.

3.2.2 3D User Parameter Estimation

In order to remap user movements into 2D and 3D pointer commands for the system, the 2D information extracted as expounded in the previous Paragraph is used to derive 3D estimates related both to head movements and eye pointing actions.

3D pointer. As reported by eq. (14), obtaining the head displacements relative to the reference frame involves the estimation of the three head orientation angles σ, τ, ϕ which enter into the definition of ${}^{\gamma}_{\beta}\mathrm{R}$ of eq. (4), and of the relative translation 3-vector ${}^{\beta o}_{\beta}\boldsymbol{t}$. These quantities can be estimated from a comparison between the current (time t) and reference (time 0) trackers associated to the left and right eyes. We express this comparison in the spirit of eq. (11) through the transformations $(\mathrm{L}_{\mathrm{L}}, \boldsymbol{x}_{\mathrm{L}})$ and $(\mathrm{L}_{\mathrm{R}}, \boldsymbol{x}_{\mathrm{R}})$.

An estimate of relative translation and orientation angles can be obtained, in principle, from a single affine transformation $(\mathrm{L}_{\mathrm{E}}, \boldsymbol{x}_{\mathrm{E}})$, based on $\boldsymbol{x}_{\mathrm{E}}(t)$, $\boldsymbol{x}_{\mathrm{E}}(0)$, and the manipulation of $\mathrm{L}_{\mathrm{E}}(t)$ [40], [41]. This methods requires a disambiguation strategy for the τ and ϕ angles, and proves to be not robust enough for this operational context. We prefer therefore a more direct approach, which exploits the *piecewise affine* camera model of Subsect. 2.1 — according to which the projection is affine for the two eyes individually, but not when they are considered together — to obtain a robust and unambiguous estimate of the head pose angles using the current and reference trackers of both eyes. Such an approach is used also to evaluate relative translation, and is based on computing relative quantities related to the change in shape and position of the two trackers. The quantities of interest are the ocular centers $\boldsymbol{x}_{\mathrm{L}}$ and $\boldsymbol{x}_{\mathrm{R}}$, the *depth ratio* $\eta = {}^{\gamma}Z_{\mathrm{R}}/{}^{\gamma}Z_{\mathrm{L}}$ and the *weighted interocular difference* $\delta\boldsymbol{x} = \boldsymbol{x}_{\mathrm{L}} - \eta\,\boldsymbol{x}_{\mathrm{R}}$. We see from eqs. (8) and (12) that η can be estimated as

$$\eta = \sqrt{\frac{\mathrm{det}\mathrm{L}_{\mathrm{L}}}{\mathrm{det}\mathrm{L}_{\mathrm{R}}}} \quad , \tag{21}$$

where the determinant is a measure related to area changes of the imaged pattern. Furthermore, from the face planarity assumption of eq. (5) and the projection model of eq. (8) we easily get that the σ and τ angles are related one to the other by

$$\tan\sigma = \frac{\lambda\,(\eta - 1)}{\left[\,\cos\tau \quad \sin\tau\,\right]^{\mathrm{T}} \cdot \delta\boldsymbol{x}} \geq 0 \quad . \tag{22}$$

Fig. 6 shows the relationship between tracker's shape and head position when the head is in an arbitrary (*left*) and in the reference (*right*) position. We assume that, independently from head cyclotorsions (rotations about the normal to the face plane), *the slant angle σ coincides always with the head pan angle* (rotation about the neck axis). A consequence of equating pan and slant angles is that the ϕ angle equals 0 for $\eta \geq 1$, and π otherwise. Another consequence is that the direction of the line passing through the left and right ocular centers coincides with the direction of maximum depth variation,

i.e. the absolute value of the denominator in eq. (22) is maximum. This, and the fact that the left-hand side of eq. (22) must be positive or zero, yields the following equality which can be used to compute τ:

$$[\cos\tau \quad \sin\tau]^{\mathrm{T}} = \mathrm{sgn}(\eta - 1)\,\frac{\delta\boldsymbol{x}}{\|\delta\boldsymbol{x}\|} \quad , \tag{23}$$

where $\mathrm{sgn}(0) = 1$. Using the computed value of τ in eq. (22) provides us finally with an estimate of the pan angle σ.

So far with the orientation estimation. Concerning the translations we notice that, since $\|{}^{\gamma}\boldsymbol{p}_{\mathrm{R}}\| \gg \mu$, we can neglect the unknown value μ and estimate ${}^{\beta_o}_{\beta(t)}\boldsymbol{t}$ simply as ${}^{\gamma}\boldsymbol{p}_{\mathrm{R}}(t) - {}^{\gamma}\boldsymbol{p}_{\mathrm{R}}(0)$. From eqs. (5), (8) and (23) we have:

$$\kappa_{\mathrm{R}}(0) \cdot {}^{\beta_o}_{\beta(t)}\boldsymbol{t} = {}^{\gamma}_{\beta_o}\mathrm{R} \left\{ \zeta_{\mathrm{R}}(t) \begin{bmatrix} \boldsymbol{x}_{\mathrm{R}}(t) \\ \lambda \end{bmatrix} - \begin{bmatrix} \boldsymbol{x}_{\mathrm{R}}(0) \\ \lambda \end{bmatrix} \right\} \quad , \tag{24}$$

where the relative depth for the right eye $\zeta_{\mathrm{R}}(t) = {}^{\gamma}Z_{\mathrm{R}}(t)/{}^{\gamma}Z_{\mathrm{R}}(0)$ is computed using the weighted interocular differences, the depth ratio and the slant:

$$\zeta_{\mathrm{R}}(t) = \frac{\|\delta\boldsymbol{x}(0)\|}{\|\delta\boldsymbol{x}(t)\|}\,\eta(t)\,\cos\sigma(t) \quad . \tag{25}$$

(In the case of frontoparallel interaction is $\eta = 1$, relative depth is the same for the two eyes, and equals the *interocular distance ratio* $\|\boldsymbol{x}_{\mathrm{L}}(0) - \boldsymbol{x}_{\mathrm{R}}(0)\|/\|\boldsymbol{x}_{\mathrm{L}}(t) - \boldsymbol{x}_{\mathrm{R}}(t)\|$.) The translation parameters can be recovered using eq. (24) up to the unknown scale factor $\kappa_{\mathrm{R}}(0)$. This indetermination — which has no practical drawbacks in our operating context, as it will be explained in Par. 3.3.1 — originates from modeling the eyes not from how they look like in the face plane but from their image appearance.

2D pointer. In Subsect. 2.1 we have pointed out that, assumed a linear camera and eye projection model, an affine transformation relates imaged pupil locations and observed screen points. The transformation, referred to as $(\mathrm{G}, \boldsymbol{g})$ in Subsect. 3.1, involves time-dependent quantities, some of which are unknown or difficult to know, such as the distance from eye to screen and eye focal length. Nevertheless, such a camera-to-screen mapping can be estimated at startup via a two-phase *plane-to-plane calibration* procedure:

1. *Observation.* A set $\{{}^{\gamma}\boldsymbol{c}_i\}$ of $m \geq 3$ image observations of pupil positions is collected and recorded, obtained by tracking the pupil while letting the user execute a

"scanpath," i.e. a sequence of gaze fixations, on a set of given screen locations $\{{}^{\alpha}\boldsymbol{a}_i\}$. To reduce errors during this phase, each new observation is obtained as an average of subsequent image measurements.

2. *Estimation.* The affine model is then estimated using data and observations, as the LS solution of an overdetermined system obtained from eq. (9). LS computations can be carried out in two steps, by first computing data and observation centroids ${}^{\alpha}\boldsymbol{a}_{\mathrm{B}}$ and ${}^{\gamma}\boldsymbol{c}_{\mathrm{B}}$ and estimating the matrix G as with affine tracking from $({}^{\alpha}\boldsymbol{a} - {}^{\alpha}\boldsymbol{a}_{\mathrm{B}}) = \mathrm{G}\,({}^{\gamma}\boldsymbol{c} - {}^{\gamma}\boldsymbol{c}_{\mathrm{B}})$, and then by obtaining the translation with $\boldsymbol{g} = {}^{\alpha}\boldsymbol{a}_{\mathrm{B}} - \mathrm{G}\,{}^{\gamma}\boldsymbol{c}_{\mathrm{B}}$.

During calibration, head motions are not allowed, in principle, as they determine a calibration error which depends on the extent of user displacements. Head motions during calibration can be compensated by normalizing the image plane observations with respect to a reference. This is obtained by replacing pupil observations $\{{}^{\gamma}\boldsymbol{c}_i\}$ with a new observation set $\{{}^{\gamma}\boldsymbol{c}_i^c\}$ s.t.

$$ {}^{\gamma}\boldsymbol{c}_i^c = {}^{\gamma}\boldsymbol{c}_i - [\boldsymbol{x}_{\mathrm{E}}(t_i) - \boldsymbol{x}_{\mathrm{E}}(t_0)] \tag{26} $$

thus taking into account, as compensation term, the amount of head translation in the image as the difference between the ocular centroids at the current (t_i) and initial (t_0) observation times.

Fig. 7 shows the results of a calibration session using $m = 25$ (a higher number of calibration points proves to yield a better accuracy, but also to get the user bored of performing calibration). The screen points to be fixated by the user during calibration are arranged in the 5×5 grid shown in Fig. 7 (*top left*). This is also the theoretical grid, since if calibration was perfect, the grid should be exactly reconstructed using image observations and the computed map $(\mathrm{G}, \boldsymbol{g})$. Figs. 7 (*top right, bottom left*) show respectively the reconstructed grid without and with compensation. It is evident from the figures that head motion compensation — the correction terms are shown in the needle diagram of Fig. 7 (*bottom right*) — has a strongly beneficial effect on calibration performance: indeed, a non perfect compensation of head displacements is the major source of error in the use of the 2D pointer. Another source of calibration errors is related to inaccuracies in the phase of pupil localization: in this respect, zooming with the camera on the user's eye region so as to track better both user eyes and pupils improves system performance.

3.3 3D Parameter Interpretation and Graphical Remapping

3.3.1 3D Pointer

To remap user head displacements into our virtual environment, we introduce the concept of *virtual camera*, representing the imaginary device used to obtain the 2D on-screen view of the virtual scene from a given point of the 3D graphic environment [42]. Specifically, any 3D user head motion is replicated in the environment, as if the virtual camera was moving in his place. This implies that translation and orientation displacements relative to the reference position be remapped as virtual camera displacements w.r.t. its own reference viewpoint and attitude: the remapping is one-one for orientation angles, and proportional for translations, with a constant of proportionality controlling interface sensitivity to the amount of user translation:

$$ {}^{\chi}\boldsymbol{p} = {}^{\beta_o}_{\beta}\mathrm{R}^{\mathrm{T}} \left[{}^{\chi_o}\boldsymbol{p} - \mathrm{diag}(k_x, k_y, k_z) \, {}^{\beta_o}_{\beta}\boldsymbol{t} \right] \quad , \tag{27} $$

where $\mathrm{diag}(k_x, k_y, k_z)$ is the translation scaling diagonal matrix. In eq. (27) we assume that the 3D virtual scene is defined in terms of χ_o coordinates, and that the virtual camera frame at time t is $\chi = \chi(t)$, with $\chi(0) = \chi_o$. Using this method, as the user approaches the screen without rotating the head (translation in the positive ${}^{\beta_o}Z$ direction), the objects in the observed scene come closer, with an amount of scale change depending directly on the remapping constant k_z.

Fig. 8 shows the remapping results — obtained with all filter gains set to 0.5, thus assigning the same confidence to the previous estimate and to the prediction — for two different interaction sessions of about 40 s each (time is expressed in number of frames). In the first session (Fig. 8, *top*), the user first approaches the screen while translating leftwards, and then moves away from the screen while translating the head downwards and rightwards. As the remapping constants are all unitary, translations are expressed in pixels. Fig. 8 (*bottom*) illustrates system behavior in the presence of head rotations. In a first phase, the user rotates his head leftwards ($\tau = \phi = 180$ deg) until reaching a pan angle σ of about 45 deg; then he gradually returns to the reference position ($\sigma = \phi = \tau = 0$, frontoparallel pose). He performs then a head pan rightwards which, as before, is followed by going back to the reference. In the last phase, composite cyclotorsion and pan movements are executed.

Proportional viewpoint control (in the present implementation the remapping con-

stants are set manually in the system; adaptive user interfaces can set them automatically after a training session) is basically a *qualitative* way of performing remapping which, besides allowing the unknown scale factor of eq. (24) to be absorbed into the remapping constant, has the advantage of providing the user with a natural feedback as the result of his action. Non proportional viewpoint control or the presence of large 3D parameter estimation errors would produce an unexpected interface behavior, with the result of confusing the user. This, in turn, would produce a synchronization delay between user actions and interface answers. With our remapping strategy instead, especially after some training, the user gradually forgets about the semantics, experiments direct interaction, and is mentally projected into the environment. This is a psychological phenomenon which is well known to trained car drivers, who are seldom conscious of using steering wheel, gear, and brakes.

3.3.2 2D Pointer

Remapping the 2D pointer has mainly a *quantitative* nature. The key idea for such a pointer is indeed to be able to know where the user is looking in the screen, and to reproduce, using a *vision based WYLAIWYG paradigm*, the functionality of a 2D mouse. This involves deeply having a metric knowledge of interaction and such knowledge, embedded into the calibration map $(\mathrm{G}, \boldsymbol{g})$, needs to be updated at run-time by head motion compensation based on current visual data and camera parameters, in a way akin to that used in Par. 3.2.2 to compensate during calibration [43].

Results for a 2D remapping session are provided in Tab. 2. In this session, the user is asked to fixate successively the points of a 3×3 on-screen grid, while the corresponding computed pupil positions in the image are remapped onto the screen using the calibration map and eq. (9). The table reports the average and maximum mismatch between the set of calibration points ("ground truth") and the remapped points both for the two coordinate directions and in magnitude. Magnitude errors provide us with a resolution limit for our 2D pointer, which must be taken into account in the design of the graphic interface.

3.3.3 Visual Pointer Timings

An issue of relevant practical importance is the time responsiveness of the interface; this is directly related to the time threshold used to measure persistence and assign the proper semantics to user action. Choosing the right threshold value is of key importance to have a good balance between speed of operation and naturality of interaction. If the threshold value is too low, then every action is interpreted as a click command, an undesirable situation usually referred to as *Midas touch* [18]. A good dwell time for the 2D pointer, which takes into account the high mobility of the pupil, is 1 s, which on the one hand guarantees a fast response, and on the other limits the occurrence of false alarms. Since head motions are slower, a time threshold about two times longer (2 s) can be used for the 3D pointer.

A second timing mechanism is implemented in the interface, which avoids the occurrence of "interference" between the pointers. Consider for example the case of a 3D click, which occurs when the head persists in a fixed position for two seconds or more. If, in the same time period, the pupil is still, then a 2D click event is generated, which is most probably involuntary. To solve the interference problem, the 2D and 3D pointers are decoupled, by letting the pupil be actually remapped only if the head is in a suitable neighborhood of its reference position — i.e. if the current tracker's parameters differ not too significantly from those of the reference tracker.

4 Interacting with the Interface

The principles and methods exposed so far have been used as a framework for the design and implementation of an interaction system with a 2D/3D graphic environment, visualizing a museum containing a digital collection of canvases by famous 20th century artists such as Klimt, Picasso, Modigliani, etc. and complemented by a 2D on-screen hypertext providing the museum with on-line catalogue facilities. The virtual environment allows to navigate around in the museum, to inspect each single canvas and to visualize the associated hypertextual information links. Experiments have been performed with several users and different interaction conditions.

4.1 Interaction Media and Operation

The interface uses the OpenGL graphic libraries and runs on a Silicon Graphics Indy workstation (MIPS R4000/R4010 with 100 MHz clock). The vision subsystem software also runs on the Indy, and gets raw image data through a VINO frame grabber board from an inexpensive B/W camera provided with zoom facilities.

Visual pointers allow the user to interact easily with the system. The head is used to change viewpoint, and navigate from place to place in the museum. The eye pupil is used to select hot points of hypertextual information, which can be accessed by persistently fixating the canvases on the walls. The hypertext is organized in several menus, whose different levels can be opened by successive selections from a window based 2D interface.

To achieve natural interaction, the loop delay must be short enough to provide the user with feedback about his latest action. The overall interaction loop time for our system is the sum of the time spent doing visual computations and graphic environment manipulation. Initializating visual algorithms involves automatic eye extraction and template initialization and takes around 450 ms to complete. At run-time visual tracking runs at video rate (25 Hz) instead, using $n = 64$ sampling points for both external eye and iris search. Without special hardware for graphics acceleration, most of the loop time is taken by 3D graphic remapping (some hundreds ms at an intermediate picture quality).

4.2 Interaction Sessions

Examples of typical interaction sessions using the 3D and 2D pointers are illustrated in Figs. 9–10 and in Fig. 11 respectively. To emphasize the relationship between user action and graphic environment changes, a specular image of the user's face and eyes — with associated 2D trackers — is shown superimposed to the 3D graphics. The temporal order of the sequences is top to bottom and left to right.

Fig. 9 shows a zoom-in sequence. Zooming is obtained by approaching the screen with the head; this causes the painting in the middle of the wall to be displayed at full resolution. A new viewpoint can always be selected by head fixation: however, in this case the user decides to go back to the initial view by moving away from the screen (last image of the sequence). Fig. 10 illustrates the generation of a viewpoint change determined by a head rotation: a leftward head pan — similar to that of the first part

of Fig. 8 (*bottom*) — causes the graphic environment to move rightwards, and display a previously invisible museum wall. Notice the simultaneous presence of a slight leftward head translation.

Once a specific viewpoint has been selected by head rotation, the user can go back to the reference position, inspect the on-screen scene and possibly learn more about a given painting in it, or about its author, by selecting a canvas by pupil pointing (Fig. 11, *snapshots 1–4*). An important aspect of the interface concerns the screen regions which can accept eye-gaze commands, which we refer to as *active windows*. At menu level 0, as commands are issued directly by fixating a canvas, each canvas is an active window. At menu level 1 (Fig. 11, *4th snapshot*), the screen is partitioned in two regions. The left-hand region has six active windows, four of which allow to access respectively to information about author, historical period, selected and related paintings; the remaining two windows are used for paging control ("back," "next"). Selection of the desired canvases or buttons is performed according to the timings described in Par. 3.3.3. The right-hand side of the screen contains output information, and as such it can be fixed by the user without timing constraints (*passive window*). As the first menu level is accessed, the passive window is filled with the painting which was selected in the museum. Fig. 11 (*snapshots 5–8*) shows the process of getting information about the author of such painting: the "author" button is repeatedly fixated, and this causes the author information to be copied into the passive window to be read.

The spacing and size of active windows strictly depend on the magnitude of the pupil remapping error (see again Tab. 2). The average pointing error can be used to properly designing the 2D interface and size-up active windows in order to reduce the risk of false alarms during slightly incorrect pointing.

5 Conclusion

In this paper, we have presented an approach to advanced man-machine interfacing based on merging computer vision and computer graphics. Computer vision is used to develop non intrusive pointer devices based on user head and eye motions, and to couple them with a simple drag and click semantics for interaction in 2D and 3D graphic environments. The pointers capture significant parameters of user action, which are then remapped onto the environment. The main operations which the user can perform are

environment exploration and data selection. Head tracking is used to interact with a 3D graphic environment, while eye pupil tracking allows to control interaction with 2D interfaces.

A trade-off exists between the advantages offered by this approach and its relative accuracy with respect to more conventional pointers (think for example to 2D mice or 3D track-balls). However, accuracy limitations may be overcome by facing the problem of interface design as a whole, i.e. by taking into account the characteristics of the visual interaction devices directly in the specification of the interaction semantics, in order to exploit at best their possibilities.

Future developments of this work are concerned with the definition of dynamically changing interfaces, which can adapt to the current user and dialogue conditions by the active modification of interface parameters such as color or window size.

Our interaction framework can also be extended so as to deal with remote environments, and allow applications such as telepresence and teleconferencing. Indeed, while interaction with the on-screen environment always takes place locally, the displayed scene needs not to be necessarily a graphic environment, but could be e.g. the output of a remote — possibly motorized — camera. Similarly, the framework can be easily adapted to interactive 3D video scenarios.

Another direction of future research is to extend the semantic expressivity of the interface. A natural way to do that in a computer vision context can be that of the interpretation of gestures and expressions.

Acknowledgements

The authors wish to thank Mr. Silvio De Magistris for his help in the implementation of the virtual museum interface. C. Colombo wishes to thank Mrs. Lydia Spiganti for her support.

References

[1] J.D. Foley. Interfaces for advanced computing. *Scientific American*, 1990.

[2] A. Marcus. Human communication issues in advanced UIS. *Communications of the ACM*, 36(4), 1990.

[3] Composing user interfaces. *IEEE Computer*, 22(2), 1989. (Special issue).

[4] J. Nielsen. Traditional dialogue applied to modern user interfaces. *Communications of the ACM*, 33(10), 1990.

[5] J. Nielsen. Noncommand user interfaces. *Communications of the ACM*, 36(4), 1993.

[6] S.K. Chang, editor. *Visual Programming Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1990.

[7] E. Glinert. Nontextual programming environments. In S.K. Chang, editor, *Visual Programming Systems.* Prentice-Hall, Englewood Cliffs, NJ, 1990.

[8] A. Dix, J. Finlay, G. Abowd, and R. Beale. *Human-Computer Interaction.* Prentice-Hall, Englewood Cliffs, NJ, 1993.

[9] J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, and T. Carey. *Human-Computer Interaction.* Addison-Wesley, Reading, MA, 1994.

[10] M. Krueger. *Artificial Reality II.* Addison-Wesley, Reading, MA, 1991.

[11] S.K. Helsel and J.J. Roth. *Virtual Reality: Theory, Practice, and Promise.* Meckler Publishing Ed., Westport, London, 1991.

[12] Virtual environments. *IEEE Computer*, 28(7), 1995. D. Pratt, M. Zyda and K. Keileher, editors (special issue).

[13] Computer augmented environments. *Communications of the ACM*, 36(7), 1993. P. Wellner, R. Gold and W. Mackay, editors (special issue).

[14] A.P. Pentland. Smart rooms. *Scientific American*, 274(4):54–62, 1996.

[15] R. Razdan and A. Kielar. Eye tracking for man/machine interfaces. *Sensors*, pages 39–42, 1988.

[16] T.E. Hutchinson, K. Preston White jr. and W.N. Martin, K.C. Reichert, and L.A. Frey. Human-computer interaction using eye-gaze input. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1527–1534, 1989.

[17] D. Cleveland and N. Cleveland. Eyegaze eyetracking system. In *Proc. 11th Imagina International Forum on New Images IMAGINA'92, Montecarlo, Monaco*, pages 17–23, 1992.

[18] R.J.K. Jacob. What you look at is what you get. *IEEE Computer*, 26(7):65–66, 1993.

[19] J. Gips, P. Olivieri, and J. Tecce. Direct control of the computer through electrodes placed around the eyes. In *Proc. 5th International Conference on Human-Computer Interaction, Japan, 1993*, pages 630–635, 1993.

[20] A. Azarbayejani, T. Starner, B. Horowitz, and A. Pentland. Visually controlled graphics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):602–605, 1993.

[21] R. Cipolla and N.J. Hollinghurst. Human-robot interface by pointing with uncalibrated stereo vision. *Image and Vision Computing*, 14(3):171–178, 1996.

[22] A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure, and focal length. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):562–575, 1995.

[23] A.H. Gee and R. Cipolla. Fast visual tracking by temporal consensus. *Image and Vision Computing*, 14(2):105–114, 1996.

[24] L. Goncalves, E. Di Bernardo, E. Ursella, and P. Perona. Monocular tracking of the human arm in 3D. In *Proc. 5th International Conference on Computer Vision ICCV'95, Cambridge, MA*, pages 764–770, 1995.

[25] J.M. Rehg and T. Kanade. Visual tracking of high DOF articulated structures: an application to human hand tracking. In *Proc. 3rd European Conference on Computer Vision ECCV'94, Stockholm, Sweden*, pages 35–46, 1994.

[26] J.J. Kuch and T.S. Huang. Vision based hand modeling and tracking for virtual teleconferencing and telecollaboration. In *Proc. 5th International Conference on Computer Vision ICCV'95, Cambridge, MA*, pages 666–671, 1995.

[27] R. Brunelli and T. Poggio. Face recognition: Features versus templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052, 1993.

[28] D. Terzopoulos and R. Szeliski. Tracking with Kalman snakes. In A. Blake and A. Yuille, editors, *Active Vision*, chapter 1, pages 3–20. MIT Press, 1992.

[29] A.L. Yuille, P.W. Hallinan, and D.S. Cohen. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–111, 1992.

[30] X. Xie, R. Sudhakar, and H. Zhuang. Real-time eye feature tracking from a video image sequence using Kalman filter. *IEEE Transactions on Systems, Man and Cybernetics*, 25(12):1568–1577, 1995.

[31] R. Kaucic, B. Dalton, and A. Blake. Real-time lip tracking for audio-visual speech recognition applications. In *Proc. 4th European Conference on Computer Vision ECCV'96, Cambridge, England*, pages 376–387, 1996.

[32] I.A. Essa and A. Pentland. A vision system for observing and extracting facial action parameters. Technical Report 247, MIT Media Laboratory Perceptual Computing Section, June 1994.

[33] Y. Moses, D. Reynard, and A. Blake. Determining facial expressions in real time. In *Proc. 5th International Conference on Computer Vision ICCV'95, Cambridge, MA*, pages 296–301, 1995.

[34] J. Davis and M. Shah. Recognizing hand gestures. In *Proc. 3rd European Conference on Computer Vision ECCV'94, Stockholm, Sweden*, pages 331–340, 1994.

[35] T. Starner and A. Pentland. Real-time american sign language from video using hidden Markov models. In *International Symposium on Computer Vision*. IEEE Computer Society Press, 1995.

[36] R.H.S. Carpenter. *Movements of the Eyes*. Pion, London, England, 1977.

[37] J.L. Mundy and A. Zisserman. Projective geometry for machine vision. In J.L. Mundy and A. Zisserman, editors, *Geometric Invariance in Computer Vision*. MIT Press, 1992.

[38] R. Curwen and A. Blake. Dynamic contours: Real-time active splines. In A. Blake and A. Yuille, editors, *Active Vision*, chapter 3, pages 39–57. MIT Press, 1992.

[39] D. Ballard and C.M. Brown. *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

[40] R. Horaud, S. Christy, F. Dornaika, and B. Lamiroy. Object pose: Links between paraperspective and perspective. In *Proc. 5th IEEE International Conference on Computer Vision ICCV'95, Cambridge, MA*, pages 426–433, 1995.

[41] C. Colombo and J.L. Crowley. Uncalibrated visual tasks via linear interaction. In *Proc. 4th European Conference on Computer Vision ECCV'96, Cambridge, England*, pages 583–592, 1996.

[42] J. Foley and A. van Dam. *Fundamentals of Interactive Computer Graphics.* Addison-Wesley, Reading, MA, 1982.

[43] C. Colombo and A. Del Bimbo. Interacting through eyes. *Robotics and Autonomous Systems*, 1997. (To appear).

Figures

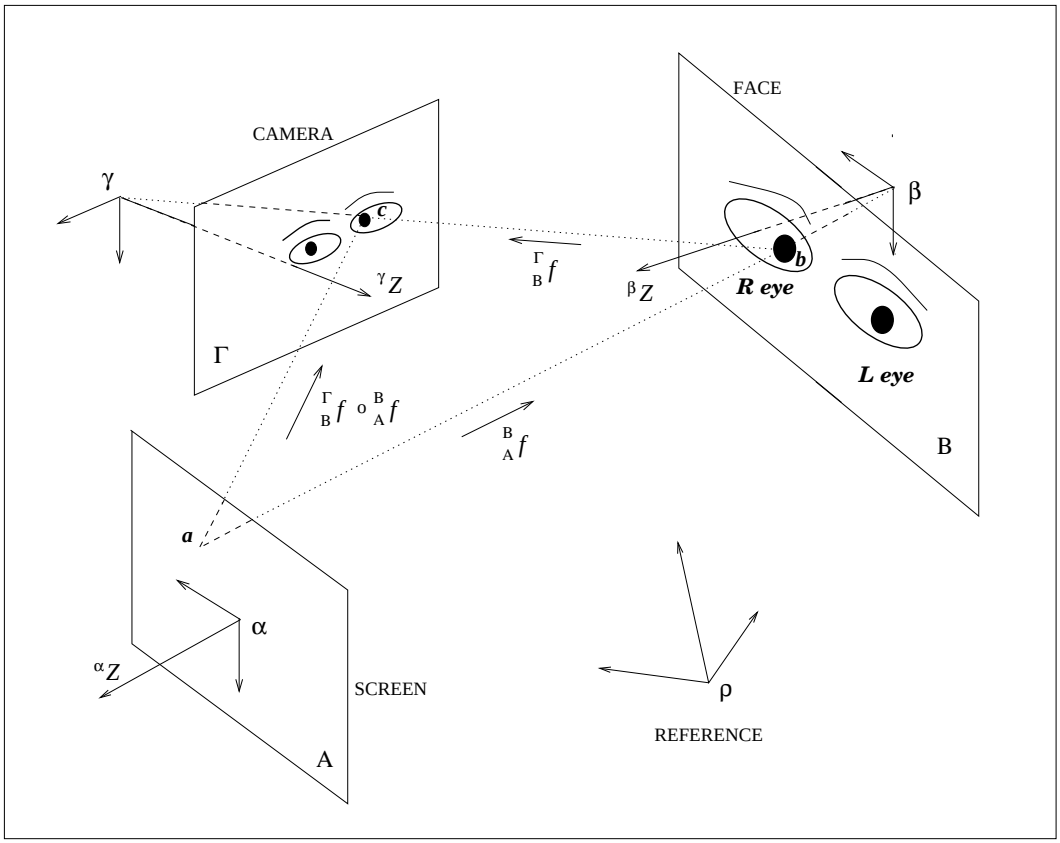

Figure 1: Visual interaction: geometric aspects.

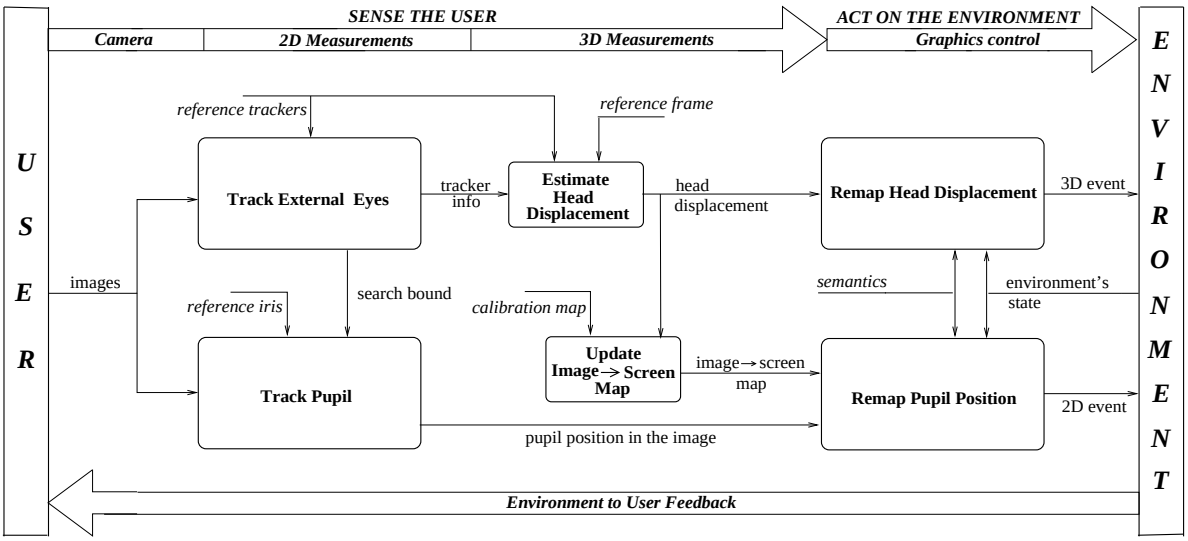

Figure 2: Block-description of our interface (see text). Non bold italics font refers to information available at startup, non bold roman to time-varying information.

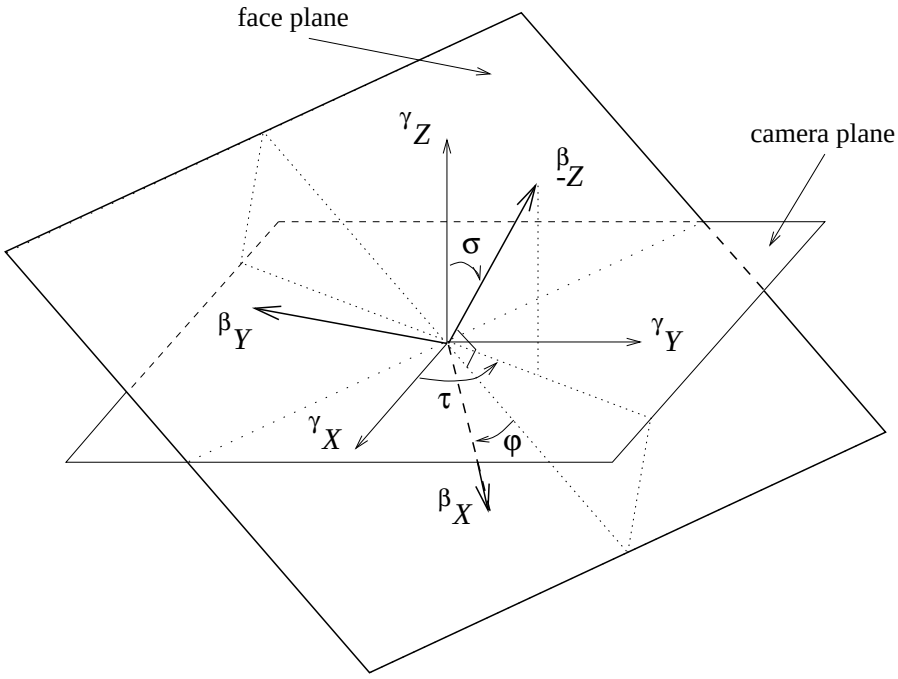

Figure 3: Definition of head pose parameters.

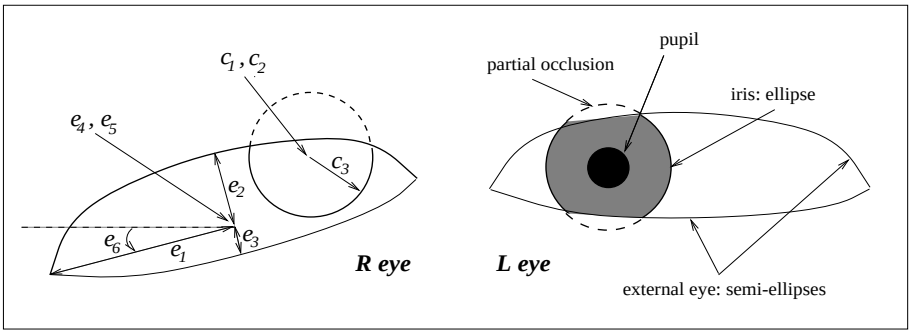

Figure 4: Reference template for the left and right eyes.

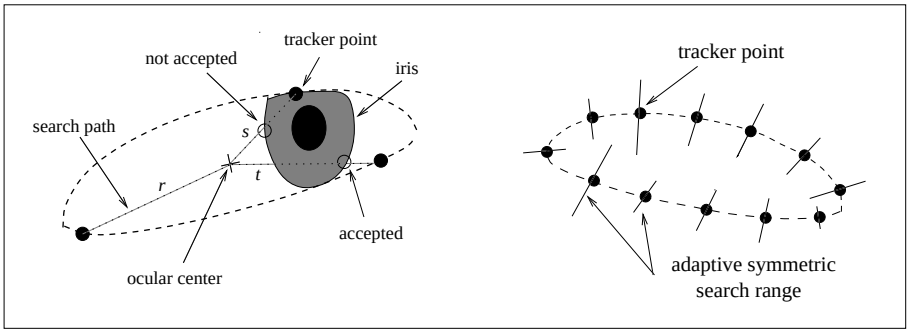

Figure 5: Affine-deformable eye tracker and search of iris candidate points.

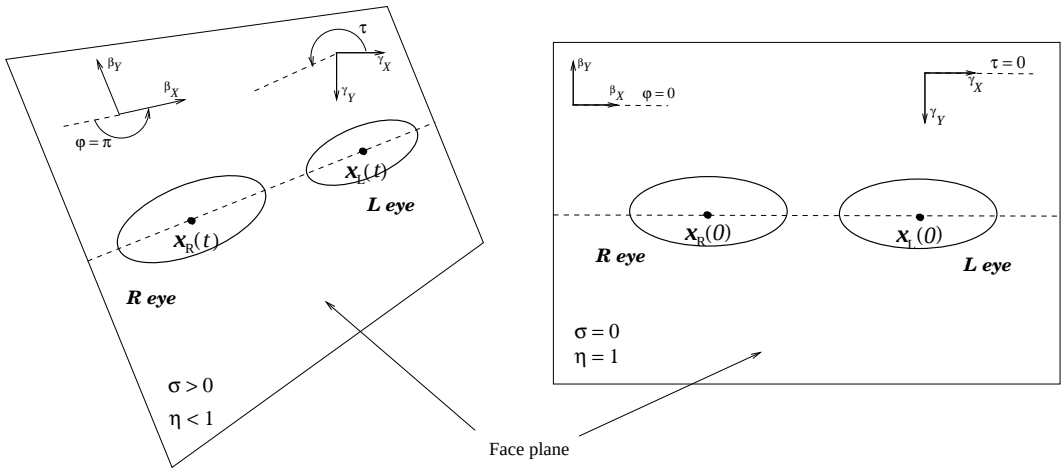

Figure 6: The 2D parameters used for 3D head displacement estimation.

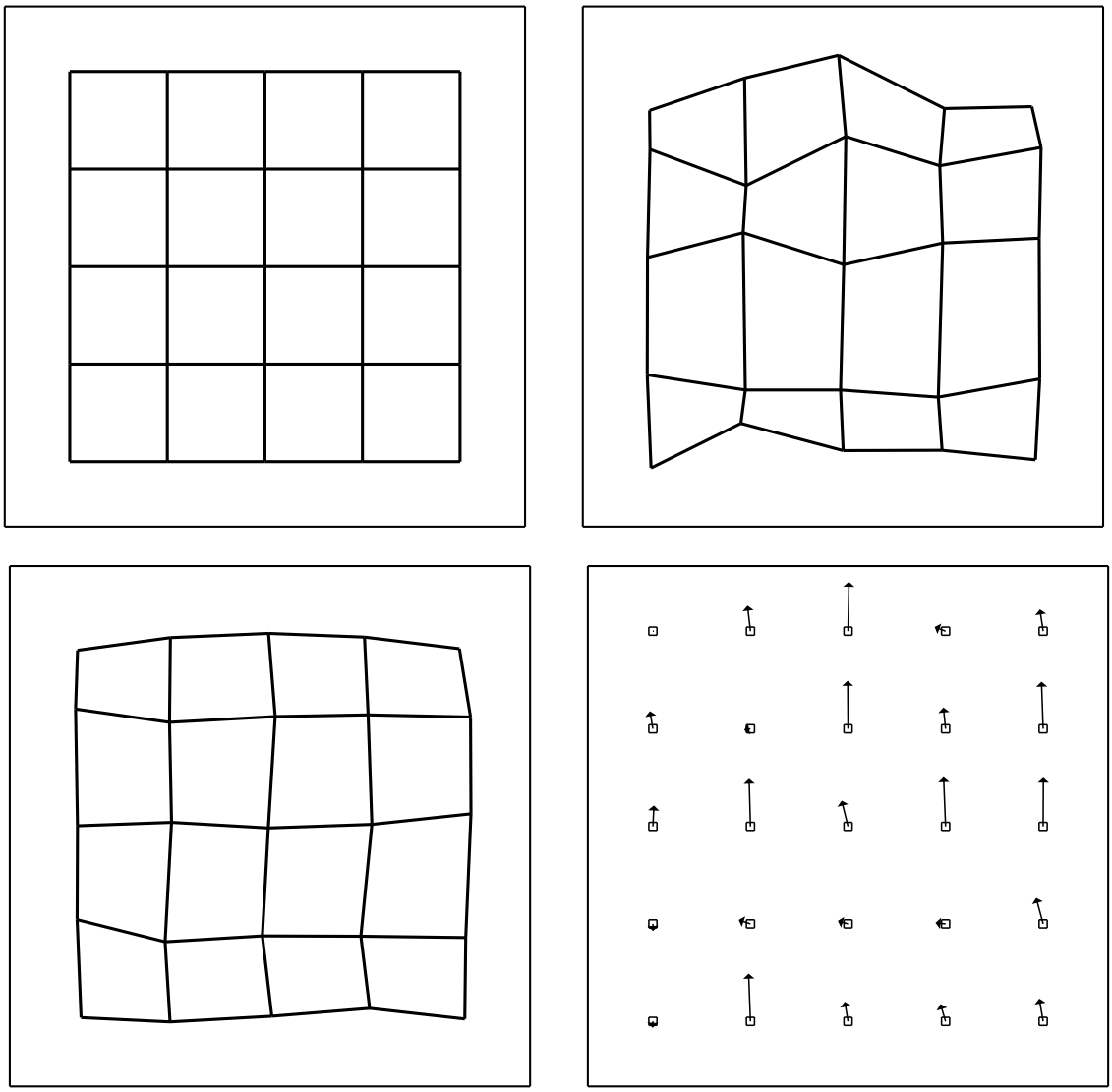

Figure 7: Calibration results. *Top left*: theoretical calibration grid; *top right*: uncompensated grid; *bottom left*: compensated grid; *bottom right*: compensation vectors.

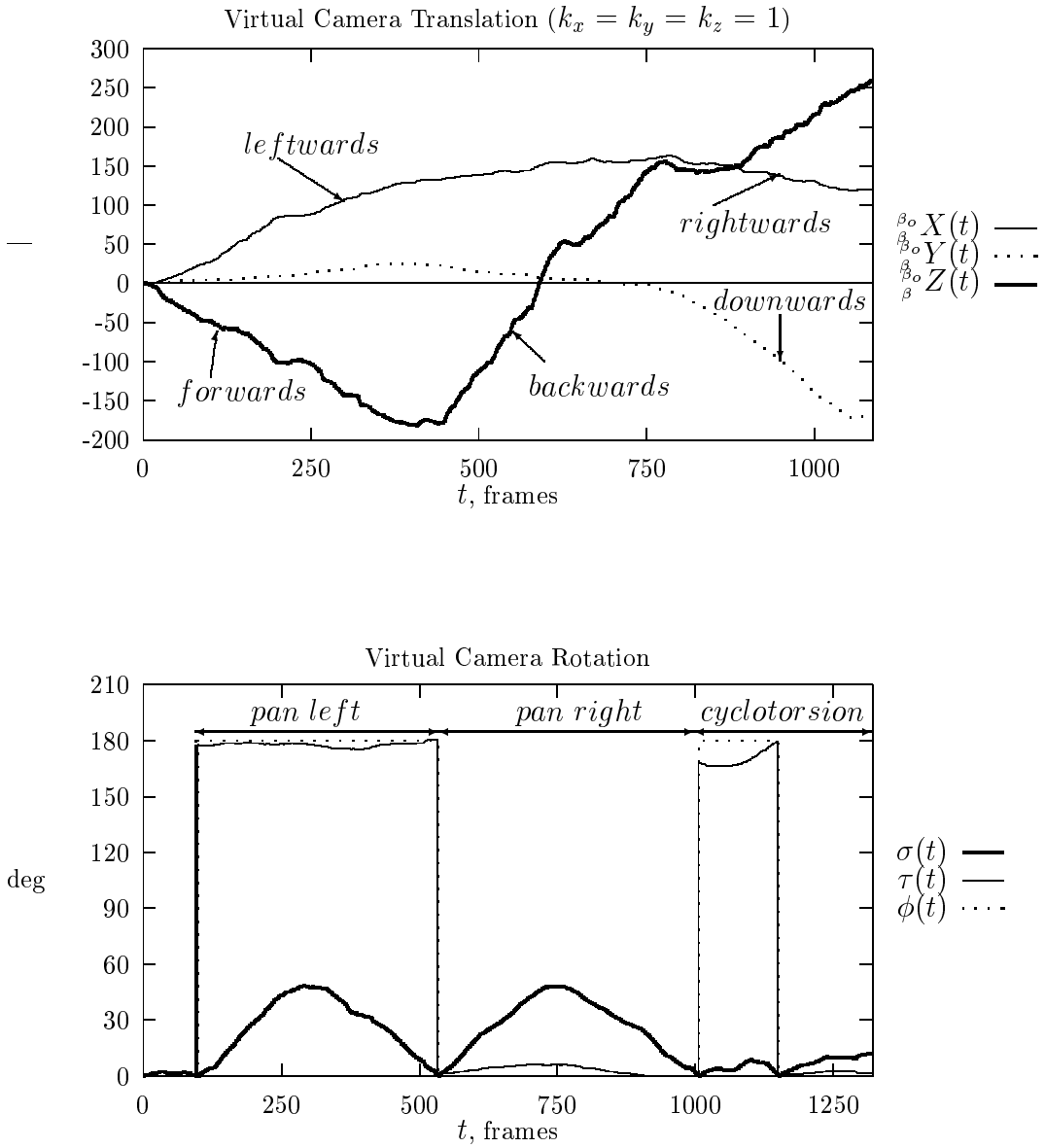

Figure 8: Remapping relative translation (*top*) and relative orientation (*bottom*).

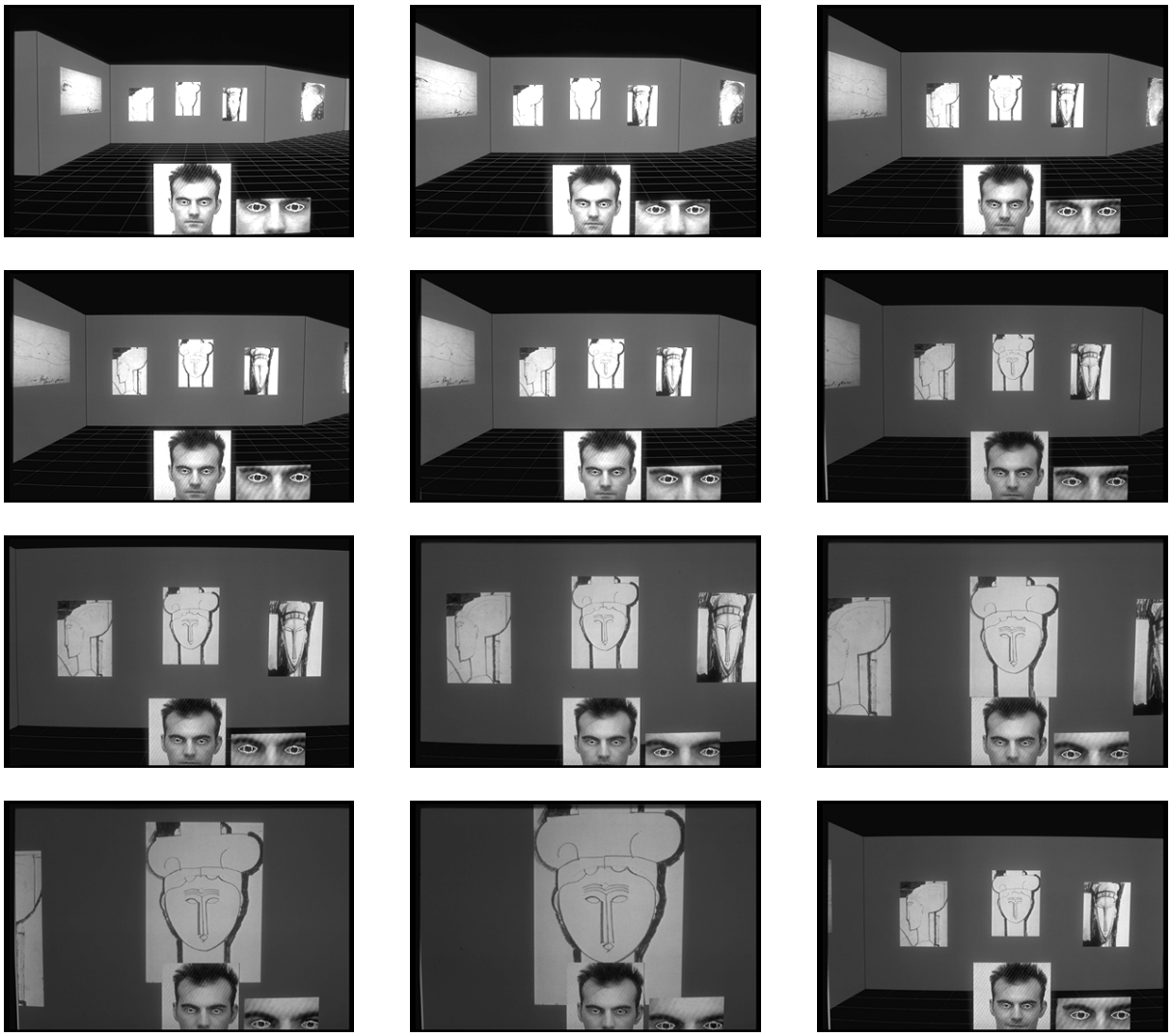

Figure 9: 3D pointer: zoom sequence.

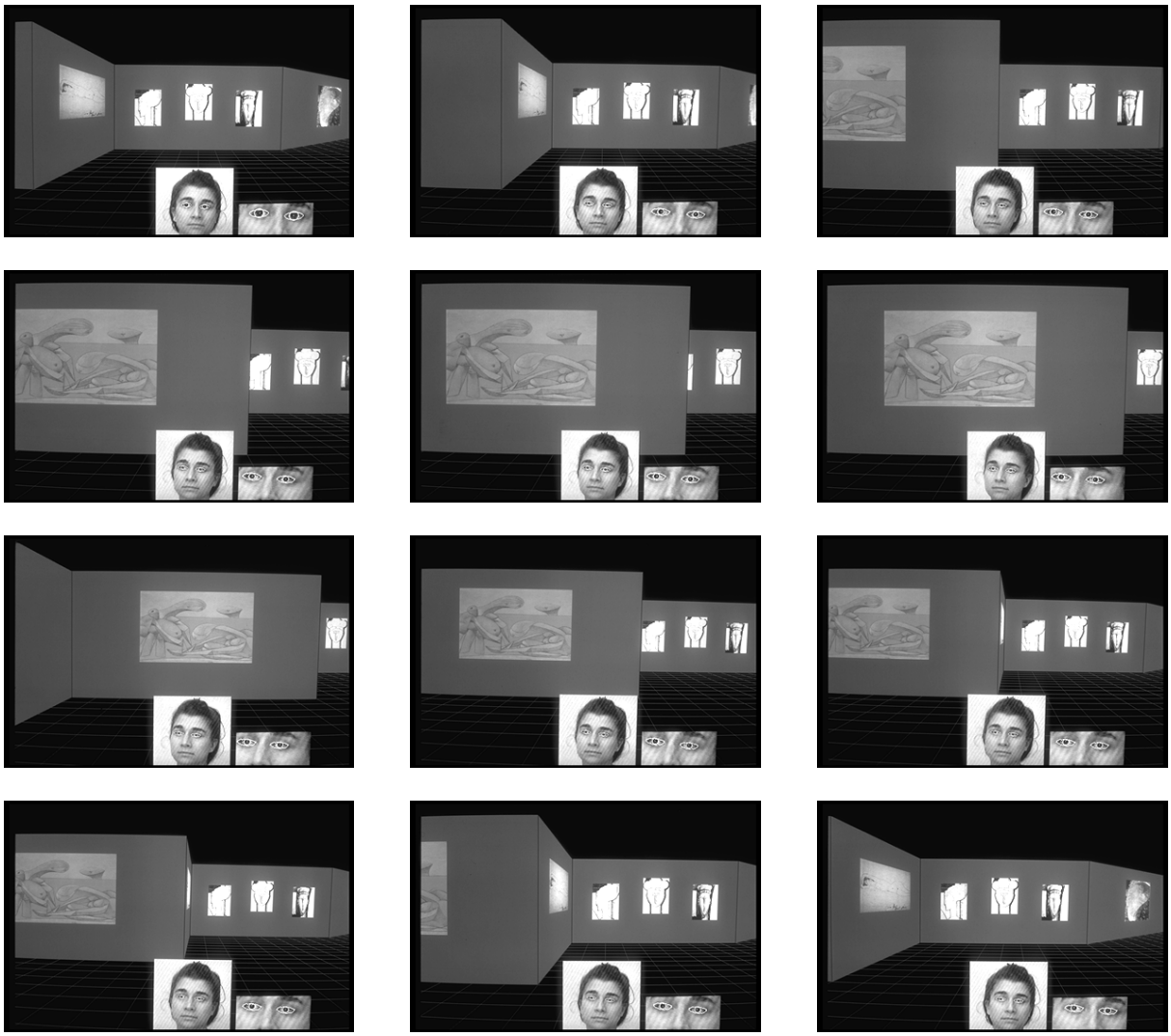

Figure 10: 3D pointer: pan sequence.

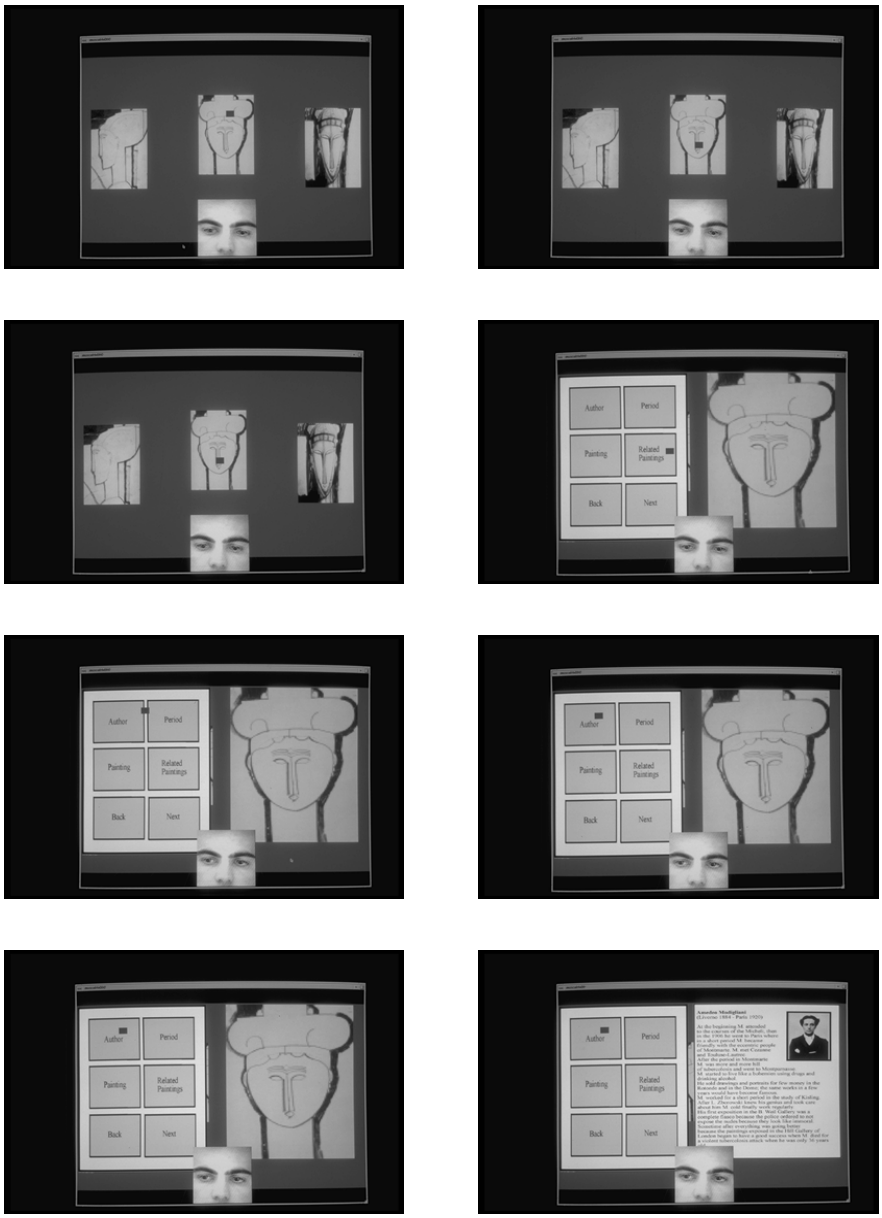

Figure 11: Selection with the 2D pointer. *Snapshots 1–4*: The 1st menu level is accessed by steadily pointing the eye to a specific painting. *Snapshots 5–8*: access to the second menu level.

Tables

Visual Pointer	*Drag*	*Click*
Eyeball	2D point change	2D point selection
Head	3D scene change	3D scene selection

Table 1: Operation with the Drag and Click metaphor.

Remapping Accuracy	*Error on* x	*Error on* y	*Uncertainty radius*
Average (mm)	6.23	10.69	12.38
Maximum (mm)	17.51	19.93	26.53

Table 2: 2D pointer remapping accuracy.

List of Figures

List of Tables