Social Environment: Towards an Intuitive User Interface

Michael Hoch and Georg Trogemann

Academy of Media Arts
Peter-Welter-Platz 2, 50676 Cologne, Germany,
e-mail: {micha,georg}@khm.de

Abstract

In this paper we describe a scenario for a so called "Social Environment" which allows for a more intuitive man-machine communication by using the entire space in front of a large back-projection screen. Computer vision, recognizing movements, gestures and posture, and speech recognition are used, placing the user in a non-intrusive environment. The interaction is related to the location and movement of the user within this space and is described in two exemplary applications. For a pointing gesture we will describe a simple algorithm that is based on the intersection of planes. The interface has a special focus on the needs of people in creative fields such as visual arts or film.

1 Introduction

Creative people are often unfamiliar with computers and hesitate before getting involved with mouse and keyboard, because of the symbolic and alphanumeric abstraction that is necessary. In this paper we describe the first step towards a new user interface titled "Social Environment" that tries to close this gap by being non-intrusive and intuitive to learn. Our interface is based on the fact that people already have experience in relating locations to entities [15], as used in games like multi-user dungeons that are often based on the metaphor of rooms to facilitate navigation. A second example are 3D-file browsers, that use 3D visualization for data retrieval. These approaches facilitate exploring and manipulating data, but they do not facilitate the use of the computer itself.

Storing information at locations in the mind has been a technique called "Ars Memoriae" which was used by orators in antiquity to memorize long speeches [15]. The orator would create icons of the subjects to be memorized and place them at chosen locations in an imaginary architecture. The icons are used as links to the information and as the orator (in his mind) walks to the memorized locations he is able to retrieve the icons and with them the information. Our scenario uses this metaphor by allowing the user to put icons at chosen locations in real space and to retrieve these icons and the linked information at a later time. The computer will help to retrieve the icons by displaying them on the screen and will offer the linked information when requested. Our system uses the real space in front of a back-projection screen. Computer vision is used for sensing user posture and simple gestures like pointing, together with speech recognition to trigger commands. For speech recognition we use a wireless head-set microphone and PC-based, continuous, speaker independent speech recognition software that allows the recognizable commands to be defined in a "Backus Naur" type of grammar [12].

In this paper we will first present related work. Next, we describe the scenario of our system and present two applications as examples. Thereafter, the gesture recognition process will be outlined, and

Proceedings „Herbsttagung '96 - 3D Bildanalyse und -synthese", Universität Erlangen-Nürnberg, 18-19. Nov. 1996

we will describe the algorithm for the pointing gesture in more detail.

2 Related Work

Using body movements to control game-style applications is described in [1]. We extend this approach to applications in the creative field, such as an interactive film planning system [7], [9] in which the real space is used as a location where data is placed and retrieved, and not only as a relative reference system to navigate in a virtual space. An "intelligent room" is described in [14] that supports the user's daily activities in an office or meeting environment. In our approach, space and gestures are not only used to trigger commands, but real space becomes part of the computer's data space and the body movements in this space become part of the interface. Using two-dimensional graphic space for data-management is described in [2], using real space as a location to situate windows of a standard window system is described in [6]. We extend these approaches by freeing the user from tracking devices and allowing her to use the entire room to store any kind of information.

Unlike other gesture recognition systems, we do not seek to recognize complex gestures which, although they often allow great functionality, they also have a large learning curve and are sometimes awkward to use, for example the different hand gestures in [3] and [13]. Our commands are as simple as pointing gestures.

3 Scenario

In the environment presented here, the user controls an application by means of moving and navigating in real space in front of a wall-size back-projection screen. The movements are utilized for navigating and manipulating data, i.e. moving data to places in real space, or moving objects like a virtual camera to a specific position. Commands are triggered by using pointing gestures and speech. The focus lies on the location of the user and the overall movement of hands and body. These influence the way commands are either interpreted or decrease the functionality of commands, e.g. blending out the finer control commands if the user seems to move in a nervous manner.
The locations the user chooses can be next to objects or areas in the room that are easy to remember, like a table or a window, as well as locations like the sides or the center of the room. Height can also be used to organize information, e.g. putting the less important entities on the floor and the most important at eye-level. Furthermore, by relating data to locations in real space the system is open for self organizing strategies as described in [8] where, for example, a self organizing sound data base could be explored in real space. Here, data entities form groups of similar quality and other entities, as well as groups, move around in space to find appropriate groups to join.

The types of interaction which take place in our scenario are: placing and retrieving data, switching between different sets of entities, and storing entities at fixed positions in the surroundings of the user:

- To place an entity at a certain location the user selects it on the screen by issuing a speech command and/or a pointing gesture. Next, he moves to a location in the room and puts the selected entity at that location triggered by another speech command.
- For retrieving entities the user moves to the location were he remembers the entity to be located. The computer will show the current collection on the screen, from which an entity can be selected. Different 2D or 3D views of the collections can be used to match the user's preferences.
- Switching between different sets of entities is possible by using speech commands, which enables the user to use the room for different sets of data, i.e. the same locations can be used for storing and retrieving a new set of information [15].

- Entities can also be stored at a fixed position around the user, as the "surround-fixed-windows" described in [6]. These entities are fixed to a position relative to the user's position and are accessable even when the user moves around. This mode can be used to select a group of entities, such as hyperlinks or camera control functions, and freely move around with them until the group is released again.

4 Applications

We created two exemplary applications to demonstrate the idea of our environment:

Managing and maintaining information entities

This example application demonstrates the use of space for manipulating and retrieving data. The user can place information entities, i.e. hyperlinks, data, or sound files, at any location in the space in front of the projection, and retrieve the information by simply moving to that location. The current collection is shown on the screen when the user moves within the perimeter of the entities in real space. Figure 1 shows a typical screen of our demo system where the current collection is a set of netscape hyperlinks that are superimposed over a lab scene.

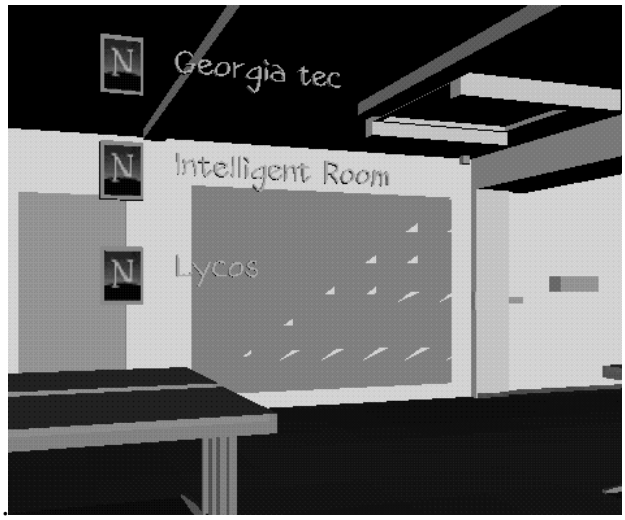

Figure 1: Hyperlinks superimposed on lab scene

Links or data files are activated by pointing to them and/or issuing a speech command. The idea is that the user will memorize the location where data about a certain topic have been placed more easily than browsing a hierarchical file system.

Film Planning System

In film production storyboards are normally used to plan a film. Storyboards are drawn sketches of the key scenes of a sequence. These sketches show camera perspective and give a good description of the scene setting, but they lack the final impression of the image, and movements cannot be shown directly. In our application, the system is used for directing a computer generated scene for film planning purposes [7], [9]. The example scene shows a typical set of a living room. Objects in the scene can be moved around by simply pointing to them and issuing a speech command. The user can modify camera position and point of view by changing his position in real space, and define camera movements that can be recorded and played back by moving in front of the scene. The entire space in front of the display is used to direct the scene. This is a familiar working environment for film people as opposed to the desktop environment of traditional PC. Instead of a storyboard, the result is a sample sequence of the film. The system creates an interactive environment where image analysis is used to control an image synthesis process, i.e. creating a computer generated film. For a more detailed description of the concepts see [7] and [9].

5 Gesture Recognition

For gesture recognition we currently use a single and a stereo camera system connected to a Silicon Graphics workstation. The single camera system is used to determine the location of the user in 3D space and to account for body movements. The stereo camera system is used for a pointing gesture, which will be described in more detail.

Single camera system

A single camera view is used to determine the position and posture of the user. First, the user is ex-

tracted from the background using a constant background image. A blob analysis procedure detects the biggest connected region in the image. Thereafter, head and hands are located using curvature cues on the boundary curve of the silhouette. Simple heuristics about head and hand position, similar to those described in [4] and [11], are used to guide the search process and fill in missing data when parts cannot be extracted reliable. We assume that the user is most often facing the camera and therefore expect to find the left hand on the left side of the silhouette. The search regions for hands are located at the left and right edge of the bounding box and have a 10 percent width of the entire contour (Figure 2). If no curvature cues are detected in this region we assume that the hand is at a rest position, as shown on the right side of Figure 2, i.e. at 45% of the height of the contour. If the hand is found, we continuously update the new position as long as the displacement does not exceed 30% of the bounding box width. The same scheme is used for the head, with the search area depicted in Figure 2. For the feet we use a different bounding box that covers the lower 34% of the contour. The search regions are chosen similarly to the hand regions. The rest position is located on the lower edge of the bounding box.

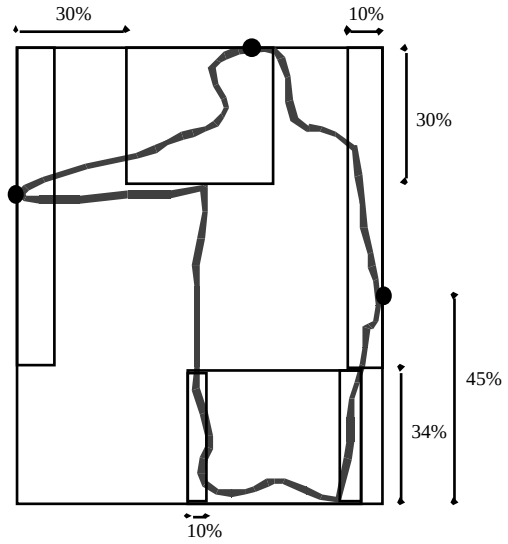

Figure 2: Silhouette of the user shown with search regions and detected hands and head.

If we cannot find a position in the current frame with the above rules, we use the ratio of the position relative to the bounding box found in the last frame. This ratio applied to the current bounding box gives the new position used. If we cannot find a new position for more than five frames, we assume that we lost the hand and interpolate the current position to the rest position, so that the hand smoothly moves back to it's rest position. The size of the search regions and the other heuristics were determined using experimentation.

An approximation of the 3D position of the user is determined by using the position of the feet in the image. Furthermore, the velocities of body and hand movements are determined to account for interactions where the amount of movement is important. Finally, a Kalman filter is used to smooth the data. The Kalman filter, together with the heuristics, result in a stable system.

Stereo camera system

The stereo camera system is used for a pointing operation which determines the position on the screen pointed at by the user. Currently, we use a colored stick that is segmented in the stereo pair by using color cues (Figure 3). We are working on a version that is not dependent on a stick, but rather extracts the user's forearm automatically, as well as integrating the functionality of the single camera system.

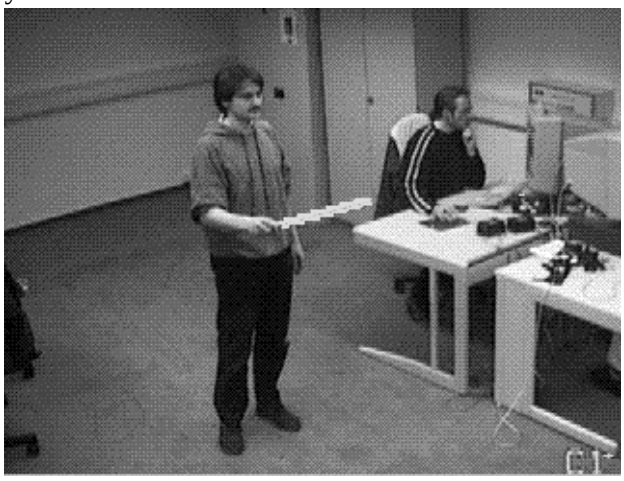

Figure 3: Left camera image with segmented colored stick.

The algorithm runs in real time, i.e. a screen cursor follows the change of the user's pointing direction without noticeable delay. The general idea of the 3-D pointing system is to approximate the position of the user's forearm by a three-dimensional line and then calculate the intersection of that line with the screen plane. The first processing step is to extract the forearm of the user from the camera image and to approximate it by a straight line. Standard image processing algorithms can be applied for the straight line approximation. The result of this step is an analytical description of a line in camera coordinates. But a line seen by one camera does not define a unique three-dimensional line. The 2-D line just imposes constraints on the 3-D line's position in space, i.e. the line must lie in the plane E defined by the 2-D line and the optical center of the camera. When two cameras are available, the system has, in general, a unique solution. If more than 2 cameras are used, the system is over-constrained and the least-squares solution for the intersection of the planes should be applied [5].

To describe the whole process mathematically, we have to deal with several coordinate systems: a world coordinate system in which the screen is described, camera coordinate systems, and the coordinates of the 2-D image plane. An obvious choice for the world coordinate system is to assign the projection screen to one of the planes defined by the axes of this coordinate system, e.g. the $x-y$ plane. For the camera coordinate system the origin can be at the optical center. A standard set-up for a two camera system is depicted in Figure 4.

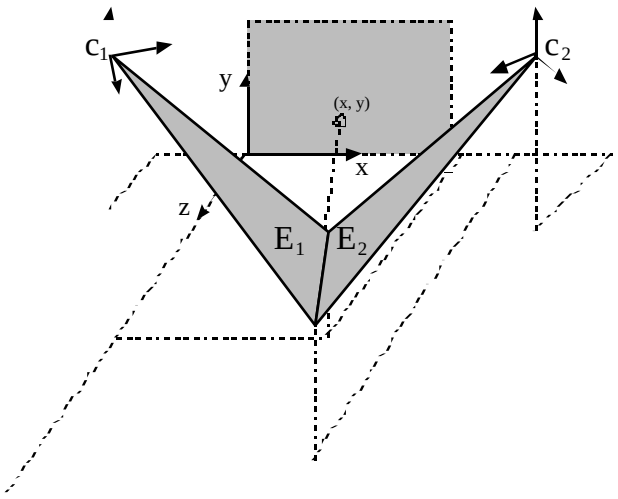

Figure 4: Schematic view of camera set-up and segmented pointing stick.

For camera calibration we use a semi-automatic procedure that first determines the angle of view by using a test pattern of known dimensions. Next, camera position and the location of an object in the scene are measured manually. Thereafter, the rotation angles in x- and y-axes are determined automatically using these values and the location of the object in the image plane.

Determining the 3-D pointing direction of a user is closely related to the image formation process in the pinhole camera model. In homogeneous coordinates, points in three-dimensional space are four-component vectors $\mathbf{X} = (hx, hy, hz, h)^{\mathrm{T}}$. The image coordinates $\mathbf{Y}$ are obtained by

$$\mathbf{Y} = \mathbf{M}\mathbf{X}.$$

The complete 4 x 4 transformation matrix, $\mathbf{M}$, from world coordinates to image coordinates, is composed of several elementary matrices

$$\mathbf{M} = \mathbf{N}\mathbf{P}\mathbf{R}_z\mathbf{R}_y\mathbf{R}_x\mathbf{T}.$$

The translation, $\mathbf{T}$, shifts the origin of the world coordinate system to the origin of the camera coordinate system. $\mathbf{R}_z$, $\mathbf{R}_y$, and $\mathbf{R}_x$ change the orientation of the coordinate system by rotations

about suitable axes. **P** performs the perspective projection, and **N** transforms (scales and translates) the two-dimensional image plane [10].

To reconstruct the 3D straight line L, the transformations of the pinhole camera model have to be inverted. L is given by the intersection of the two planes E_1 and E_2 (see Figure 4). Using vector notation for the planes, E_i, $i = 1,2$ can be described by one point $\mathbf{c}_i$ and two vectors $\mathbf{r}_i^1, \mathbf{r}_i^2$. The center of the camera coordinate system can be chosen for the point $\mathbf{c}_i$. Suitable vectors $\mathbf{r}_i^1, \mathbf{r}_i^2$, $i = 1,2$ are given by $\mathbf{c}_i$ and the endpoints of the line segments in the image plane, which can easily be determined from the image coordinates of the segmented region. The construction of planes E_i can be seen as an attempt to reverse the projection **P** (which is of course not possible but yields the above mentioned constrains for the 3-D position of the line). In the next step, we transform the camera coordinates of E_1 and E_2 to world coordinates by applying inverse transformations $\mathbf{T}^{-1}$, $\mathbf{R}_x^{-1}$, and $\mathbf{R}_y^{-1}$ (rotation about the z-axis is not necessary with our camera set-up). When using vector notation, we just need to apply the inverse translation on $\mathbf{c}_i$, and inverse rotations on $\mathbf{r}_i^1, \mathbf{r}_i^2$ to obtain the representation of E_1 and E_2 in world coordinates.

In the final step, we switch to coordinate notation of E_i, $i = 1,2$, $E_i : A_i x + B_i y + C_i z + D_i = 0$. To obtain the point (x, y) were the line L intersects with the screen we just solve for $E_1 = E_2$, using the knowledge that the z-value of L is equal to zero. For the two camera models we get $\mathbf{X} = \mathbf{K}^{-1}\mathbf{D}$, which in detail is:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} A_1 & B_1 \\ A_2 & B_2 \end{pmatrix}^{-1} \begin{pmatrix} -D_1 \\ -D_2 \end{pmatrix}.$$

If more than 2 cameras are used, the system is overconstrained and the least-squares solution $\mathbf{X} = (\mathbf{K}^T\mathbf{K})^{-1}\mathbf{K}^T\mathbf{D}$ should be used. This method provides successive (x, y) values in real-time and Kalman filtering is applied to smooth the data. Tests have shown that the visual feedback that is given by the screen cursor is far more important than accuracy, similar to hand-eye feedback with mouse and mouse pointer. By modifying the last step, the pointing operation can also be used for pointing at any object in the scene if the 3D coordinates of the objects are known.

6 Conclusion

We presented a first step towards a "Social Environment" that incorporates the space in front of a back-projection screen in the user-interface. Computer vision is used to recognize the movements and gestures of the user to achieve a non-intrusive interface. Simple gestures and speech recognition are used to trigger commands. The focus of the interaction lies on the user's location and movements in real space, leading to an interface that is intuitive and easy to learn. The system is designed for the special needs of people in the creative arts and forms an intuitive user interface in this context. For future work we plan to implement more applications that create a familiar working environment for visual artists and film makers.

References

[1] Ali Azarbayejani, Christopher Wren and Alex Pentland, "Real-Time 3-D Tracking of the Human Body", Proceedings of IMAGE'COM 96, Bordeaux, France, May 1996.

[2] Richard A. Bolt, "The Human Interface", Lifetime Learning Publications, Belmont, California, 1984.

[3] Ulrich Bröckl-Fox, "Real-Time 3D Interaction with up to 16 Degrees of Freedom from Monocular Video Image Flows", International Workshop on Automatic Face- and Gesture-Recognition, June 26-28, Zürich, Switzerland, pp. 172-178, 1995.

[4] Trevor Darrell et. al., “A Novel Environment for Situated Vision and Behavior“, Proc. of CVPR-94 Workshop for Visual Behaviors, pp. 68-72, Seattle, Washington, June 1994.

[5] Olivier Faugeras, Three-Dimensional Computer Vision: A Geometric Viewpoint, The MIT Press, Cambridge, 1993.

[6] Steven Feiner et. al., “Windows on the World: 2D Windows for 3D Augmented Reality“, Proc. UIST ‘93: ACM Symp. On User Interface Software and Technology, Atlanta GA, November 3-5, pp. 145-155, 1993.

[7] Georg Fleischmann, Michael Hoch and Detlev Schwabe, "FilmPlan: ein interaktives Filmplanungssystem", Lab 1, Magazin der Kunsthochschule für Medien Köln, pp. 34-39, 1994.

[8] Georg Fleischmann, Michael Hoch, Detlev Schwabe et. al., “smdk: An Interactive Self-Organizing Sound Environment”, AAAI-94 Workshop, Artificial Intelligence, Artificial Life, & Entertainment, Working Notes, Seattle, Washington, 1994.

[9] Georg Fleischmann and Detlev Schwabe, "Interaktive Simulationsumgebung für die Filmplanung", Tagungsband des 40. internationalen wissenschaftlichen Kolloquiums der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität Ilmenau, 1995.

[10] Bernd Jähne, “Digital Image Processing: Concepts, Algorithms, and Scientific Applications“, Springer-Verlag, Berlin 1995.

[11] Michael P. Johnson, Pattie Maes, and Trevor Darrel, “Evolving Visual Routines“, Artificial Life, 1(4), 1994.

[12] Lernout & Hauspie Speech Products, “User’s Reference and Programming Guide“, Sint-Krispijnstraat 7, 8900 Ieper, Belgium, V3.010, July 1996.

[13] Christoph Maggioni, “GestureComputer - New Ways of Operating a Computer”, International Workshop on Automatic Face- and Gesture-Recognition, June 26-28, Zürich, Switzerland, pp 166-171, 1995.

[14] Mark C. Torrance, “Advances in Human-Computer Interaction: The Intelligent Room“, Working Notes of CHI ‘95 Research Symposium, May 6-7, Denver, Colorado, 1995.

[15] Frances A. Yates, “The Art of Memory”, Routledge & Kegan Paul/PLC, London, 1966.