Hand-Gesture Recognition as a 3-D Input Technique

Ulrich Bröckl-Fox

Universität Karlsruhe
Institut für Betriebs- und Dialogsysteme
Lehrstuhl Prof. Dr. A. Schmitt
Postfach 6980; 76128 Karlsruhe, Germany
EMAIL: ub@ira.uka.de

Abstract. 3-D input techniques based on vision systems are presented: human extremities (hands and head) are observed by a video camera. Certain 2-D parameters of the depicted extremities are used as input parameters for 3-D metaphors. The 2-D parameters needed and the real-time, contour-based computation of these on standard hardware is presented. Among these parameters, stable hand-shape classification is the key to simple and robust 3-D metaphors. A filter algorithm to separate hand and forearm is described. Different mappings of these parameters into 3-D movements and their metaphorical meanings behind these mappings are shown.
To evaluate these approaches a network-distributed *Virtual Squash* game was implemented, and the effectiveness of the video-based metaphors was compared directly to that of the space-ball. The results reveal that the precision of translations is slightly worse, the number of hits and the velocity of the Virtual squash-racket is slightly better, but the precision of rotations is supreme for video-based hand-gesture input.

1 Introduction

When designing a widget set for interactive user interfaces manipulating 3-D data one has to incorporate easy to grasp metaphors enabling the user to manipulate 3-D objects in a simple way. For 3-D metaphors the following problem arises: a standard workstation neither has a 3-D input medium that can be mapped directly into the 3-D object space nor does it have a 3-D output device. Therefore a special-purpose 3-D input device should be incorporated as an additional input medium into the widget set. A large variety of these devices exists on the market, among these the data-glove rules the virtual-reality market and the space-ball the desktop market. The latter was shown in [10] to be nearly as efficient as the data-glove, both are stated to be up to two times more effective than mouse-driven metaphors such as those described in [7].
On the other hand, vision systems are becoming more and more a standard equipment for current workstations used mainly for tele-conferencing. Towards the data glove using these systems for 3-D interaction offers the advantage of keeping both hands free for typing, mouse selections etc.. According to [24] only 14% of desktop applications currently developed use 3-D graphics. Of these nearly none is pure 3-D. At least some choices, menu selections or options have to be entered by mouse or keyboard. Most applications constantly swap between 3-D object space and the 2-D space of the graphical

user interface. For such applications a data-glove is more likely to be a handicap than an advantage. Towards the space-ball device vision offers at the first sight the advantage of being available without extra cost. As it will be shown by direct comparison in the *Evaluation* section, the video-based 3-D interaction technique is even more efficient than the space-ball device.

The idea to use video input as an input device for human computer-interaction is not new: [18] made the proposal to extend his camera-based system, used to record finger and hand movements in 2-D, into a 3-D system using multiple cameras. The paper presents a technique that implements this idea, but using several cameras is avoided in order to keep costs and computational efforts low.

There exist several other video-based approaches for 3-D interaction: [2] constructed an optical tracker with a scalable work area whereas [21, 22] proposes monocular approaches for desktop applications. [27] presents a hand-gesture taxonomy, the image-flow-based system is still under development. [1] describe their system to control the view on 3-D geometries by observation of the user's head. The same group developed a hand-gesture recognition system [9], introduced the usage of extended Kalman filters for structure and motion estimation of video-based systems in [11] and currently works on full-body interaction as described in [20]. [28] try to detect 27 degrees of freedom using a video-based approach, they obtained 3 degrees of freedom. Prior work in this field of the team in which the author works is summarized in [6].

The new contributions of this paper can be summarized by the following hypotheses that will be underpinned in the rest of the paper:

1. There is almost no need to provide six degrees of freedom at once with an input device: users, especially when untrained, tend to divide a manipulation task requiring both translation and rotation into several single-operation phases.

2. In order to switch between these phases a stable hand-shape recognition system is needed. Only then a change in hand-shape can be used as an event that triggers switching to a different operation.

3. Provided that we are able to implement such a system, it is easy to find three 2-D parameters in the depicted hand that can be used as input parameters for a 3-D metaphor. All these 2-D parameters can be won by integration, not by differentiation operators on the image, which makes the system less sensitive to noise.

4. In applying these simple 3-D metaphors for three extremities (hands and head) instead of one (the hand or head only) in parallel, a simultaneous but intuitive specification of six degrees of freedom is possible again: one hand translates the tool, the other rotates the object while the head's movements manipulate the view on the object and the tool. Through motion parallax the user gets — even if no special 3-D output device such as a shutter stereoscopic is available — a good estimation of the tool's and object's 3-D relation. For using hands in parallel similar findings are reported for the usage of two data-gloves in [3].

2 2-D Parameters

This section gives an overview of the few 2-D parameters needed for a video-based 3-D interaction system, and their efficient calculation.

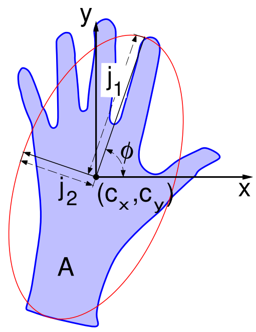

Fig. 1. Needed parameters of the hand.

Contours are computed from the given grey-scale images by a compass-mask algorithm. The integer-valued contours are transformed into real-valued ones by moving the contour points along the grey-scale gradient that is calculated by Sobel operators. The description of these steps can be found in [25]. Finally the real-valued contours are transformed into cubic B-spline curves which acts as a low-pass filter [13]. From this point on we only work with real-valued contours or cubic B-spline curves.
From the cubic B-spline curves the moments $m_{i,j}$ up to the second order are calculated. From these the area $A = m_{0,0}$, the length of the contour l, the center of gravity $c = (c_x, c_y) = (m_{1,0}, m_{0,1})/m_{0,0}$ and the main moments of inertia j_1, j_2 of the hand[1] are computed as described in [17].
The hand's 2-D orientation ϕ is won as a byproduct of the hand-shape classification described below.

2.1 Computation of 2-D Parameters

To determine the k-th order moments of the hand efficiently several authors proposed to calculate these from the hand's contour using triangulations [19] or run-length encodings [8, 29]. The remainder of this section presents a generalization for polynomials of the approach described in [16] that is based on Greene's integral-theorem.

Theorem 1 *Greene's integral-theorem* (proof see e.g. [15])*: If $B \in \mathbb{R}^2$ is a region and ∂B is the positively oriented boundary of B and P and Q have continuous partial derivatives on B then*

$$\int_B \left(\frac{\partial Q}{\partial x} - \frac{\partial P}{\partial y} \right) d(x,y) \;=\; \int_{\partial B} P\,dx + Q\,dy.$$

By means of this theorem we are able to compute the moments from contours that are bounded by polynomial curves:

Corollary 2 *Let be B bounded by n cubic polynomial segments*
$\gamma_1, \ldots, \gamma_n : \mathbb{R} \to \mathbb{R}^2$,

$$\gamma_i(t) \;=\; \begin{pmatrix} \gamma_{i,x}(t) \\ \gamma_{i,y}(t) \end{pmatrix},\; t \in [0,1]$$

$$\overset{(e.g.)}{=} \quad \begin{pmatrix} c_{i,x,0} + c_{i,x,1}\,t + c_{i,x,2}\,t^2 + c_{i,x,3}\,t^3 \\ c_{i,y,0} + c_{i,y,1}\,t + c_{i,y,2}\,t^2 + c_{i,y,3}\,t^3 \end{pmatrix}.$$

[1]The techniques described in this section apply to the face in the same way.

Then

$$m_{p,q}(B) = \sum_{i=1}^{n} \left(\int_0^1 \frac{\gamma_{i,x}^{p+1}(t)}{p+1} \gamma_{i,y}^{q}(t)\, \dot{\gamma}_{i,y}(t)\, dt \right).$$

Proof: Assume that $P = 0$ in theorem 1. Further let be $Q = \gamma_{i,y}^{q}(t)\, \gamma_{i,x}^{p+1}(t)/(p+1)$. Then $\partial Q/\partial x = x^p\, y^q$ what is equal to the definition of moments. If we finally set $dy = \dot{\gamma}_{i,y}(t)\, dt$ we obtain the corollary.
This corollary can be easily formulated in a computer algebra system, in the appendix the moments up to the first order are stated in table 1.

2.2 Filters

Beside the grey-scale gradient filter mentioned before and the low-pass filtering induced by the B-spline curve approximation, a third somewhat more hand-gesture related filter is needed. While the first two filters are mainly needed to reduce camera and motion noise the aim of the third filter is to avoid the effects of the forearm entering and leaving the picture. Especially when the observed camera volume is big the parameters of the hand are falsified or even dominated by the forearm's parameters. For desktop applications the following algorithm offers a simple method to eliminate the forearm segments of the hand contour.

The principle of the recursive algorithm is depicted in figure 2. In each recursion step the segments lying in a $45°$ sector left and right to the negative major inertia axis (which can be determined from moments up to the second order, see above and [6, 17]) are replaced by a helical arc. Only in this sector forearm segments can be found due to camera/desktop relation. The recursion is stopped as soon as the quotient j_1/j_2 of the major and minor moments of inertia is less than a constant $k > 1$, normally $k = 2$.

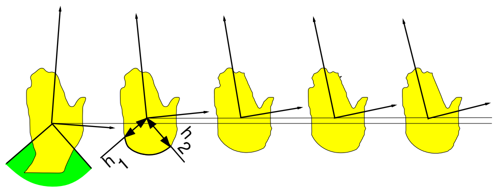

Fig. 2. Some recursions of the forearm-filter. Recursion was repeated 4 times for documentation purpose, $j_1/j_2 < 2$ became true after the first recursion.

Algorithm 3 (Replacing forearm segments of the hand contour):

Input: Contour $K \in (\mathbb{R} \times \mathbb{R})^n$, $n \in \mathbb{N}$, a periodic sequence of positive-oriented 2-D points, the center of gravity (c_x, c_y), the orientation ϕ of the negative major-inertia axis. The first point of a contour segmented s is denoted by $s.p_0$, the second by $s.p_1$.

Output: Contour K', with the forearm segments replaced by helical segments.

PROCEDURE forearm_filter(VAR K: Contour, VAR c: 2D_Point, VAR ϕ: Angle);
VAR p,c_{new}: 2D_Point; i,n: INTEGER; j_1, j_2: REAL;
BEGIN
  Determine segments s_i, s_j, intersecting with the $\phi \pm 45°$-rotated X-axis
  and the number n of segments lying in between.

```
  { Determine radii of the helix: }
  h_0 := sqrt((s_i.p_1.x - c_x)^2 + (s_i.p_1.y - c_y)^2);
  h_1 := sqrt((s_j.p_0.x - c_x)^2 + (s_i.p_0.y - c_y)^2);
  FORALL p ∈ {s_i.p_1, ..., s_j.p_0} DO BEGIN
    { Replace contour points by helical points: }
    p.x := c_x + (i/n (h_1 - h_0) + h_0) cos(φ - 45° + i/n 90°);
    p.y := c_y + (i/n (h_1 - h_0) + h_0) sin(φ - 45° + i/n 90°); i := i + 1;
  END;
  Determine_Center_of_Gravity_and_Moments_of_Inertia_and_Orientation
                                                        (K, c_new, j_1, j_2,φ);
  IF j_1/j_2 ≥ 2 THEN   forearm_filter(K, c_new, φ);
END;
```

Theorem 4 *If the width/height quotient q of the depicted arm is always $q < 20$ the algorithm stops after at most 6 recursions.*

Proof: In order to estimate the number of recursion steps we consider figure 3. Here hand and forearm are modeled by a rectangle. Without loss of generality the rectangle is parallel to the X-axis and its height K_0 is 1. This is a worst-case estimation since due to the extension of the wrist and palm the depicted hand is always 'thicker' in Y-direction compared to the forearm. Therefore the rectangle segments replaced by helical segments cause a greater quotient j_1/j_2 than the replaced forearm segments and thus a slower convergence. A further upper bound estimation is replacing the helical arc by a rectangle of the width $4\,K/5$ and height K (see [5] for details on that estimation).

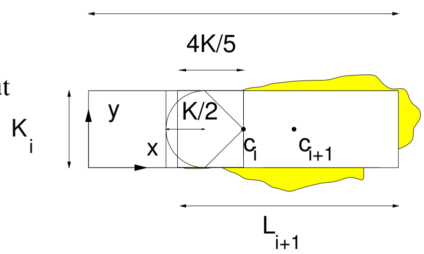

Fig. 3. Identifiers for proof of theorem 4.

Now we are able to determine the length L_i of the i-th recursion step by the following recurrence relation:

$$L_i = \begin{cases} L_0 & \text{if } i = 0, \\ L_{i-1}/2 + 4/5 & \text{if } i > 0. \end{cases} \tag{1}$$

This recurrence yields — easily to be proven by induction — the following solution:

$$L_n = \frac{8}{5} + (L_0 - \frac{8}{5})\, 2^{-n}. \tag{2}$$

Therefore the recursion stops after at most n steps, with n as follows:

$$n > \log_2 \left(\frac{5\, L_0}{2} - 4 \right). \tag{3}$$

Setting $L_0 := q = 20$ (remember $K = 1$ for all steps in the estimation) we obtain $n > 5.52$; that is, $n = 6$ is a upper bound.

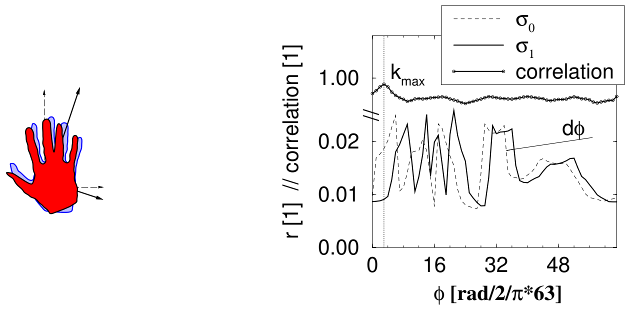

Fig. 4. Using signatures to determine 2-D orientation of the hand, and the class of the hand-shape.

2.3 Stable Classification of Hand-Shapes

In order to determine the class to which a given hand-shape belongs, the hand's contour is transformed into a discrete, one-dimensional function, a so-called *signature*, as depicted above. A simple way to obtain a signature is to transform the centralized contour into polar coordinates. Then the angular axis is split into n intervals and the length of the radius of the contour is integrated for each interval. The signatures are normalized by their autocorrelation. If the contour is rotated as in figure 4 two things can be observed for the signatures: the shape of the signature remains similar, but it is phase-shifted. The phase shift is one parameter we are searching for. It is calculated as the quadratic interpolation of the maximum k_{max} of the correlation of the two signatures. The correlation itself can be calculated efficiently applying the convolution theorem in $O(n \ \log \ n)$ steps [12].
To use the correlation of signatures as a means to classify hand-shapes, we use pre-learned signatures of sample-signs and correlate these with a given hand-shape of an unknown class. The correlation's maximum i) determines the class the hand-shape belongs to and ii) gives a measurement for the fidelity of the classification process. It can be shown, that this approach is (2-D) rotation-, translation and scaling-invariant und thus quite user-invariant. It should be noted here that the phase shift $d\phi$ i.e. the 2-D orientation calculated is much more robust towards noise and 3-D rotations of the hand than moment-based approaches. Figure 5 shows the invariance of this simple algorithm towards different users and different orientations of the hand-shape. The columns show the correlation's results for three shape samples, each row stands for a hand sign given from different users in different 2-D orientations.

2.4 Experiments with the Classificator

To evaluate this method it was used to classify dynamic hand-gestures such as hello- and bye-waving and pointing-gestures in the same arrangement as described in [9]. At a recognition speed of 14 Hz a recognition rate of 97.4 % was obtained on an *Intel 486* processor at a clock speed of 50 MHz. No false classifications occurred during the experiments. These results were obtained even though the training gestures were taken from *one* individual, whereas 19 different persons used the hand-gesture recognizer.
Even better recognition rates can be obtained if several differently defined signatures (e.g. the slope density function from [26] instead of radius versus angle) are used to

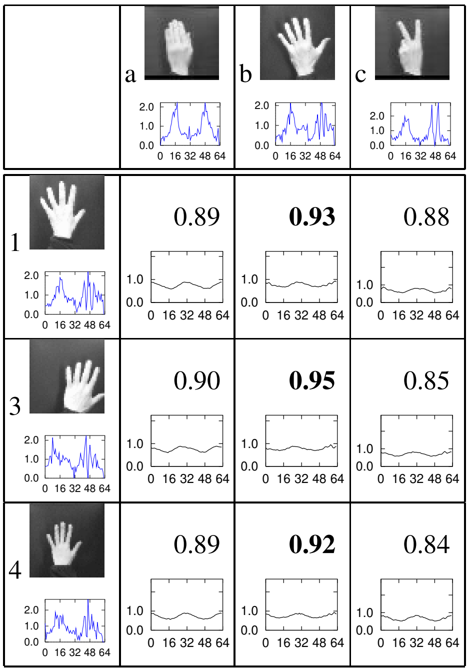

Fig. 5. Examples of hand-shape classifications.

feed a classification algorithm based on [30]. Here for eight different hand-shapes a recognition rate of 98.0 % was achieved. If we skip the user invariance, that is if we use the training-shapes individually, even 99.2 % can be achieved. Details can be found in [5].

3 3-D Metaphors

To ease the description for the following metaphors, only three different hand-shapes, the ones depicted in figure 5, are used. Nonetheless in our 3-D widget-set (cf. [4]) we use more hand-shape-classes as shown in figure 6. The possibility to steer the dialog flow by different hand-shapes is not the *only* means to trigger events, the traditional WIMP[2] interaction paradigm is left to the user. By doing so, we have a simple means of introducing and depicting the possibilities the hand gesture recognizer offers to the user. Possible modes in 3-D interaction are *"Camera in hand"*, *"Scene in hand"* and *"Fly through scene"* with the same functionality as the metaphors described in [31]. The *"Nodo"* operation has the same meaning as lifting the mouse in 2-D graphical user interfaces, i.e., it enables the user to change the hand's position without performing any graphical actions and thus "walk" long distances through the 3-D scene.

[2] WIMP = Windows, Icons, Menus, and Pointers

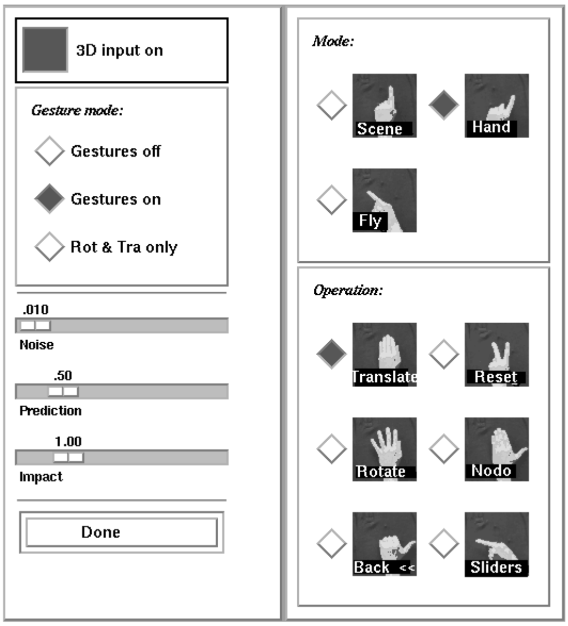

Fig. 6. Using several hand-shapes to steer a 3-D widget.

3.1 The Trinocular Metaphor

Fig. 7. The mirror-box.

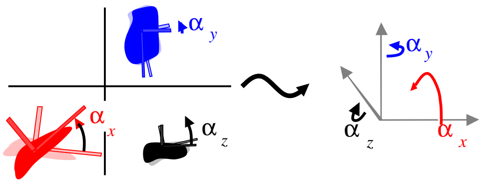

Fig. 8. Mapping 2-D orientations of the three 2-D views onto 3-D Euler angles.

One of our first approaches was quite simple: we took three orthogonal views of the user's hand and mapped the 2-D centers of gravity calculated by first order moments onto the 3-D center of gravity of the hand. To obtain three 2-D views the user interacts in 3-D space by moving his hand into the *"mirror-box"* (figure 7). Two mirrors are arranged in an approximate angle of 45 degrees to the camera plane. Therefore the three views of the scene intersect nearly rectangularly.
The hand's 3-D orientation can be determined by directly mapping the three 2-D orientations of the three views onto 3-D Euler angles (figure 8). Details can be found again in [6].

3.2 The Screw Metaphor

Fig. 9. Screw metaphor.

In order to avoid the necessity to put a rather large mirror box onto the desktop monocular metaphors were investigated more closely. As it will be shown in the evaluation section these monocular metaphors were also found to be more efficient. Monocular metaphors generally steer the operation to perform by a dedicated hand symbol as in the examples on figure 9: the straddled hand is used to rotate things; the closed hand translates them. The screw metaphor got its name from the concept for translations (right side of figure 9): the user translates in x- and z-direction by moving his hand into the appropriate direction. For y-translations one thinks of turning a screw: turning right means increasing y-coordinates, turning left decreasing y-coordinates. The rotational case, shown on the left of the figure, is a straight forward adaptation of the rolling-ball metaphor [14] for three-dimensional input devices:
The 2-D x- and y-translation of the center of gravity of the hand (c_x, c_y) steer the rotational angles α_y and α_x. The hand's 2-D orientation ϕ is mapped directly onto the rotation α_z about the z-axis. The metaphorical concepts, especially for translations, require some training time ($\approx$ 1 min) before they can be used efficiently. But once learned, the metaphor has a decisive advantage, especially for desktop applications: since the hand rests for all manipulations in the x-y-plane it is quite unfatiguing.

3.3 The Thumb Metaphor

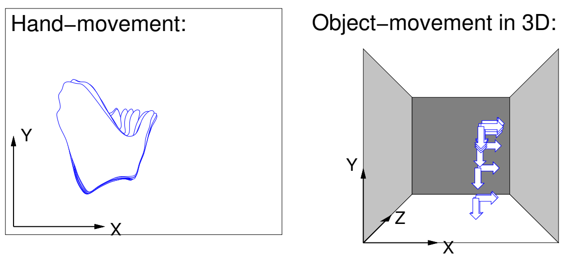

Fig. 10. Thumb Metaphor.

The metaphor is equal to the screw metaphor in its rotational part. For translations the y-translation is specified by the abduction angle of the thumb. This angle can again be calculated by means of signature-correlation. Translations are then characterized by two sample hand-shapes: one for the totally closed hand (σ_1), and another one for the thumb abducted (σ_2). Let denote $\kappa_{\sigma_i,\sigma_k}$ the maximum correlation of two given hand-shapes. Then we are able to calculate the abduction angle (and likewise any other hand-shape alteration, in particular finger movement) for a given signature σ_k by the following definition:

Definition 1 *Two-class specific signature distance:*

$$\tau(\sigma_k) := \left| \frac{\kappa_{(1),k} - \kappa_{(2),k}}{2 - 2\,\kappa_{(1),(2)}} + \frac{1}{2} \right| .$$

As in the screw metaphor the hand rests in the x-y-plane for all manipulations. Furthermore the user has no problems with unvoluntary changes of the 2D-orientation of his hand induced by x- and z-translations. Figure 11 shows an example plot of a two-class specific signature distance as defined in definition 1.

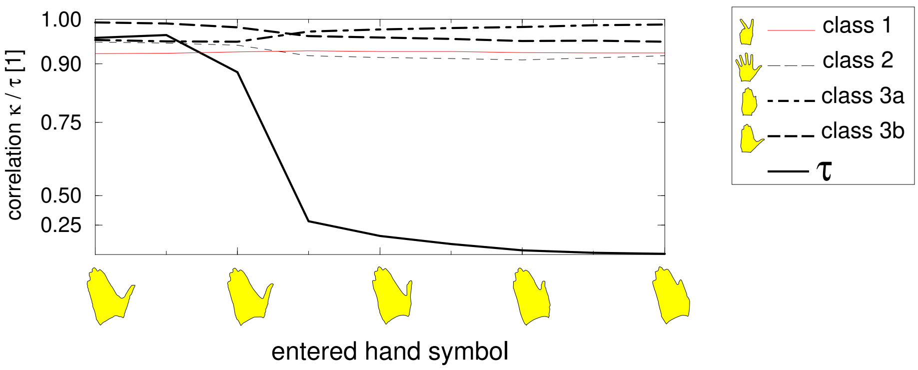

Fig. 11. Example of a two-class specific signature distance.

3.4 The Gear Metaphor

The gear metaphor is more intuitive: translations are a 1:1 metaphor, that means as the hand translates, the displayed geometries are translated. The changes in (x, y, z) coordinates are obtained from the changes of the center of gravity c_x, c_y and the area A of the depicted hand. To calculate this, a camera-calibration algorithm is needed, see [6] for details on this algorithm. The rotational part of the metaphor gave the metaphor its name: it is a 1:4 metaphor, i.e. turning the hand about an angle of $45°$ implies a rotation of the displayed geometries about $4 \times 45° = 180°$. The reason for doing so is the fact that there is no means of determining the 3-D orientation from a 2-D image unambiguously [23]. But if we restrict the allowed rotations to an interval of $[-45°, 45°]$ we can make the following observation (see figure 12): as the

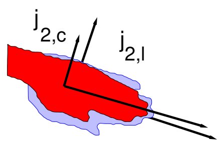

Fig. 12. Changing moments of inertia under rotation.

hand is rotated about its major inertia axis the last ($j_{2,l}$) and current ($j_{2,c}$) inertia moments change with the rotation angle approximately linear to the sine of the angle. The same applies for rotations about the second inertia axis and all combinations of these rotations. One problem remains: we do not know the sign of the rotation's sine. To avoid this problem we put the camera at an Euler angle of $(-45°, -45°, 0)$ and restrict the allowed rotations to $[-45°, 45°]$.
Now we know that all rotations within that interval have a bijective effect on the inertia moments. Setting *a* to the sine of the rotation about the major axis of inertia, *b* likewise for the minor axis of inertia and *f* to the unknown depth of the hand along the focal axis of the camera, we obtain the following nonlinear system of equations:

$$\left\{ s_c = \frac{a\ b\ s_l}{f}, j_{1,c} = \frac{a\ j_{1,l}}{f}, j_{2,c} = \frac{b\ j_{2,l}}{f} \right\}, \tag{4}$$

where s_c and s_l stand for the current and last length of the hand contour. The solution of this system is:

$$\left\{ b = \frac{s_c\ j_{1,l}}{j_{1,c}\ s_l}, f = \frac{s_c\ j_{2,l}\ \ j_{1,l}}{j_{1,c}\ \ j_{2,c}\ \ s_l}, a = \frac{s_c\ j_{2,l}}{j_{2,c}\ \ s_l} \right\} \tag{5}$$

Thus we are able to compute the angles of rotation about the axis of inertia from the moments of inertia and the contour length. Together with the 2-D hand orientation ϕ and the rotational part of the camera-calibration matrix we can transform these parameters into Euler angles in world coordinates.

4 Carving Metaphor

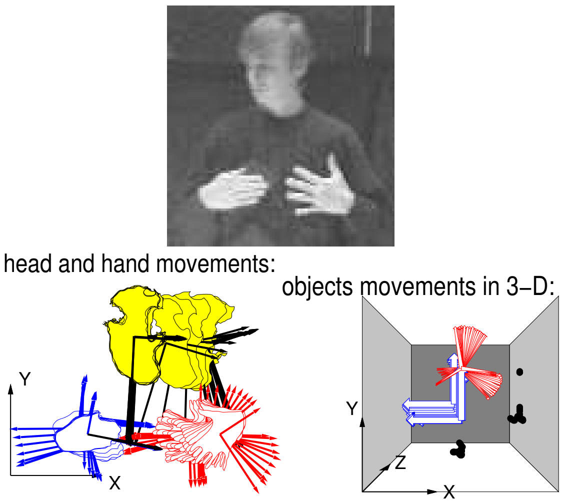

Fig. 13. An example of the carving metaphor.

The metaphors above can be used in parallel for several extremities. Figure 13 shows a composition of these metaphors: the left and right hand operate the screw metaphor, the head only operates in the rotational part of the screw metaphor, since it is not possible with the approaches stated above to do a face-gesture classification. Nonetheless the change of area A of the head's contour can be mapped onto a scaling factor of the displayed geometries. Thus the total number of degrees of freedom is $4 + 2 \times 6 = 16$. In the middle of the figure, the extracted contour curves and the flow of these is depicted. The right part of the figure shows the effects of the left hand on the manipulated object (represented by the dark 3-D axes) and the effects of the right hand onto the tool (represented by the light 3-D axes). The black dots on the floor and side-wall represent the translation of these objects. The effects of the head-movements are not superimposed for illustrating purposes.

5 Evaluation

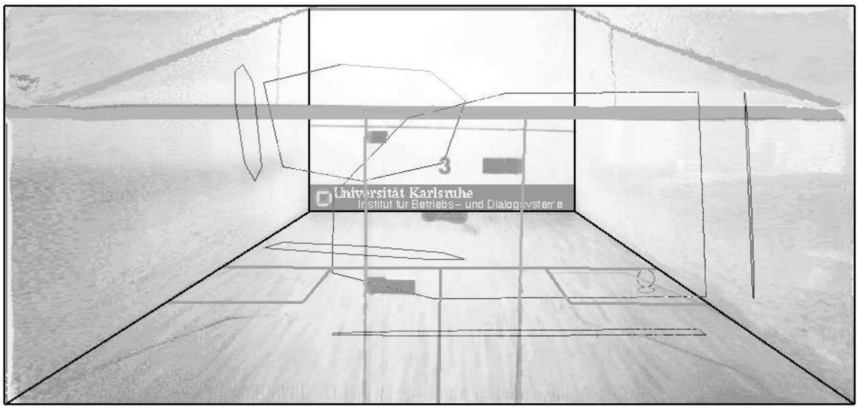

Fig. 14. Virtual squash-court.

To evaluate the video-based 3-D input method it was compared directly to the space-ball, *the* alternative device for desktop 3-D interaction. Therefore a network-distributed *Virtual Squash* game was implemented. The graphics output was quite simple: the squash-rackets were modeled by polygons and superimposed onto a digitized picture of a real squash-court. In order to enhance the 3-D impression shadows of the rackets were plotted on the walls of the squash-court. To get many contacts of the symbolic racket and target the size of the rackets was chosen big. This avoids rare and randomly distributed evaluation data.

5.1 Results

The following figure summarizes the results of the squash experiment. The results were obtained from 12 individuals during a 15-minute game composed of a training phase and three phases where the speed and the translation's and rotation's precision were used to calculate the final-score for each player. The figure shows the results obtained for each phase, respectively. The abbreviations are: s-b = space-ball, s-m = screw metaphor, g-

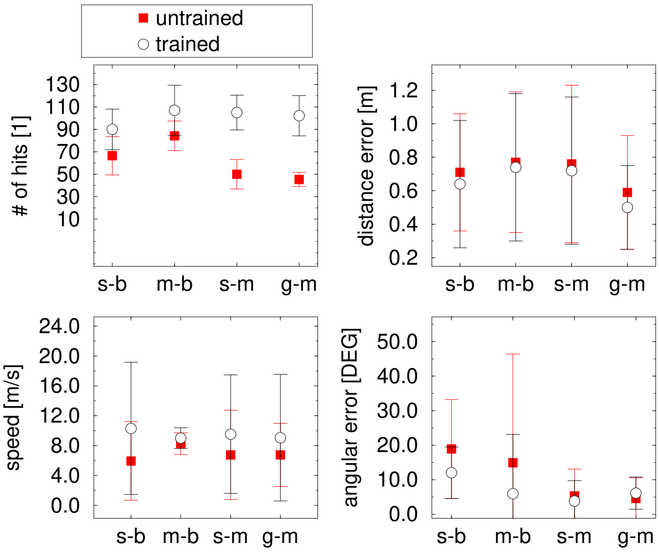

Fig. 15. Results.

m = gear metaphor and m-b stands for the mirror-box. One interesting result was found with this device: it was possible either to use translation and rotation isolated or in combination (i.e. 6 DOF), again triggered by different hand-shapes. The decision which variant should be taken was left to the user. Averaged for all users and operations in 3-D only in 20.3 % of all cases the 6 DOF variant was taken, all other operations were split into rotations (34.5 %) and translations(45.2 %). One explanation are the average Euler angles that were induced during translations by the rotational joints of the human arm and hand: for each translation an average of $(3.44, 3.83, 5.76)$ degrees of rotation was measured, for unique rotational tasks we found an average of $(4.51, 5.02, 7.56)$ degrees. Thus nonvoluntary rotations during translations make up more than the half of those for voluntary rotations. Both facts together are a strong hint for the hypothesis 1 in the introduction. Concerning the claim of real-time capability, here only one number is presented to give an estimate of the computational resources needed for the carving metaphor: on an *Intel 486* processor at a clock speed of 50 MHz we obtained a frame rate of 10.6 Hz.

6 Conclusion

Taking all results into consideration it reveals that the precision of translations is slightly worse (factor 0.90), the number of hits (factor 1.19) and the velocity of the Virtual squash-racket (factor 1.25) is slightly better, and the precision of rotations (factor 2.00) is supreme for video-based hand-gesture 3-D input. Thus the results show that video-based 3-D input techniques — even if we use metaphors as simple as the screw metaphor — are competitive to the space-ball. Especially for defining rotations, one even can claim superiority of these techniques. The thumb metaphor seems to be the ideal candidate for the 3-D input device extension to our 3-D widget system, since it is easy to implement, robust towards varying lighting situations (if we use a mouse-pad-sized black underground where the hand rests), and finally, it is unfatiguing. For 3-D interaction off the desktop (virtual reality applications e.g.) the carving metaphor is a promising approach. As a future development a color separation as proposed in [21] should be used here to extract the contours instead of the simple threshold method used above.

Appendix

Table 1. First order moments for cubic polynomial splines.

Parametric equation, $t \in [0, 1]$: $\gamma(t) = \begin{pmatrix} c_{x,0} + c_{x,1}t + c_{x,2}t^2 + c_{x,3}t^3 \\ c_{y,0} + c_{y,1}t + c_{y,2}t^2 + c_{y,3}t^3 \end{pmatrix}$	
$m_{0,0}$	$\frac{c_{x,2}c_{y,1}}{3} + \frac{c_{x,3}c_{y,1}}{4} + \frac{3\,c_{x,1}c_{y,3}}{4} + c_{x,0}c_{y,2} + c_{x,0}c_{y,3} + \frac{2\,c_{x,1}c_{y,2}}{3} + c_{x,0}c_{y,1} + \frac{c_{x,2}c_{y,2}}{2} + \frac{3\,c_{x,2}c_{y,3}}{5} + \frac{2\,c_{x,3}c_{y,2}}{5} + \frac{c_{x,1}c_{y,1}}{2} + \frac{c_{x,3}c_{y,3}}{2}$
$m_{0,1}$	$\frac{3\,c_{x,2}c_{y,3}^2}{8} + \frac{3\,c_{x,1}c_{y,3}^2}{7} + \frac{2\,c_{x,3}c_{y,2}^2}{7} + \frac{c_{x,0}c_{y,3}^2}{2} + \frac{c_{x,2}c_{y,2}^2}{3} + \frac{2\,c_{x,1}c_{y,2}^2}{5} + \frac{c_{x,3}c_{y,1}^2}{5} + \frac{c_{x,0}c_{y,2}^2}{2} + \frac{c_{x,2}c_{y,1}^2}{4} + \frac{c_{x,1}c_{y,1}^2}{3} + \frac{c_{x,0}c_{y,1}^2}{2} + \frac{c_{x,3}c_{y,3}^2}{3} + c_{x,0}c_{y,0}c_{y,1} + c_{x,0}c_{y,0}c_{y,2} + c_{x,0}c_{y,0}c_{y,3} + \frac{4\,c_{x,3}c_{y,3}c_{y,1}}{7} + \frac{5\,c_{x,3}c_{y,3}c_{y,2}}{8} + \frac{5\,c_{y,3}c_{x,2}c_{y,2}}{7} + \frac{5\,c_{y,3}c_{x,1}c_{y,2}}{6} + \frac{2\,c_{y,3}c_{x,2}c_{y,1}}{3} + \frac{c_{y,3}c_{x,3}c_{y,0}}{2} + \frac{c_{y,2}c_{x,3}c_{y,1}}{2} + c_{y,3}c_{x,0}c_{y,2} + \frac{4\,c_{y,3}c_{x,1}c_{y,1}}{5} + \frac{3\,c_{y,3}c_{x,2}c_{y,0}}{5} + \frac{3\,c_{y,2}c_{x,2}c_{y,1}}{5} + \frac{2\,c_{y,2}c_{x,3}c_{y,0}}{5} + c_{y,3}c_{x,0}c_{y,1} + \frac{3\,c_{y,3}c_{x,1}c_{y,0}}{4} + \frac{3\,c_{y,2}c_{x,1}c_{y,1}}{4} + \frac{c_{y,2}c_{x,2}c_{y,0}}{2} + \frac{c_{y,1}c_{x,3}c_{y,0}}{4} + c_{y,2}c_{x,0}c_{y,1} + \frac{2\,c_{y,2}c_{x,1}c_{y,0}}{3} + \frac{c_{y,1}c_{x,2}c_{y,0}}{3} + \frac{c_{y,1}c_{x,1}c_{y,0}}{2}$
$m_{1,0}$	$\frac{3\,c_{x,2}c_{x,3}c_{y,3}}{8} + \frac{2\,c_{x,2}c_{x,3}c_{y,2}}{7} + \frac{c_{y,1}c_{x,1}^2}{6} + \frac{c_{y,1}c_{x,0}c_{x,2}}{3} + \frac{c_{y,1}c_{x,2}^2}{10} + \frac{c_{y,2}c_{x,1}^2}{4} + \frac{2\,c_{y,2}c_{x,0}c_{x,3}}{5} + \frac{c_{y,2}c_{x,2}^2}{6} + \frac{3\,c_{y,3}c_{x,1}^2}{10} + \frac{3\,c_{y,3}c_{x,2}^2}{14} + \frac{3\,c_{x,0}c_{x,1}c_{y,3}}{4} + \frac{c_{y,3}c_{x,1}c_{x,2}}{2} + \frac{c_{y,3}c_{x,0}c_{x,3}}{2} + \frac{3\,c_{y,3}c_{x,1}c_{x,3}}{7} + \frac{c_{y,2}c_{x,1}c_{x,3}}{3} + \frac{c_{y,1}c_{x,1}c_{x,3}}{5} + \frac{2\,c_{y,2}c_{x,1}c_{x,2}}{5} + \frac{2\,c_{x,0}c_{x,1}c_{y,2}}{3} + \frac{3\,c_{y,3}c_{x,0}c_{x,2}}{5} + \frac{c_{y,1}c_{x,1}c_{x,2}}{4} + \frac{c_{y,1}c_{x,0}c_{x,3}}{4} + \frac{c_{y,2}c_{x,0}c_{x,2}}{2} + \frac{c_{x,0}c_{x,1}c_{y,1}}{2} + \frac{c_{x,2}c_{x,3}c_{y,1}}{6} + \frac{c_{x,0}^2c_{y,3}}{2} + \frac{c_{x,0}^2c_{y,2}}{2} + \frac{c_{x,0}^2c_{y,1}}{2} + \frac{c_{x,3}^2c_{y,2}}{8} + \frac{c_{x,3}^2c_{y,3}}{6} + \frac{c_{x,3}^2c_{y,1}}{14}$

References

[1] A. Azarbayejani, T. Starner, B. Horowitz, A. Pentland. Visually Controlled Graph-

ics. *Transactions on Pattern Analysis and Machine Intelligence*, 15(6):602–605, June 1993.

[2] R. Azuma, M. Ward, R. Bennett, S. Gottschalk, H. Fuchs. A Demonstrated Optical Tracker With Scalable Work Area for Head-Mounted Display Systems. In *Proceedings of the 1992 Symposium on Interactive 3D Graphics*, S. 43–51, 1992.

[3] R. Bolt, E. Herranz. Two-Handed Gesture in Multi-Modal Natural Dialog. In *Symposium on User Interface Software and Technology*, S. 7–14. ACM, November 1992.

[4] U. Bröckl, A. J. Klingert, A. Schmitt. Towards Standardized User Interfaces for Three-Dimensional Interaction: the Development of 3D Widgets Based on PEX and Motif. In *Proceedings of the Fourth Annual Conference of the European X User Group (EXUG)*, 1992.

[5] U. Bröckl-Fox. *Untersuchung neuer, gestenbasierter Methoden für die 3D-Interaktion*. Verlag Shaker, Aachen, FRG, 1995.

[6] U. Bröckl-Fox, L. Kettner, A. Klingert, L. Kobbelt. *Artificial Life and Virtual Reality*, Kapitel XII, Using Three Dimensional Hand-Gesture Recognition as a New 3D Input Technique, S. 173–187. Wiley and Sons, 1994.

[7] Michael Chen, S. Joy Mountford, Abigail Sellen. A Study in Interactive 3-D Rotation Using 2-D Control Devices. *Computer Graphics*, 22(4):121–129, August 1988.

[8] M. Dai, P. Baylou, M. Najim. An Efficient Algorithm for Computation of Shape Moments from Run-Length Codes or Chain Codes. *Pattern Recognition*, 25(10):1119–1128, 1992.

[9] T. Darrell, A. Pentland. Space-Time Gestures. In *IEEE Conference Computer Vision and Pattern Recognition*, S. 294–299, New York City, June 15–17 1993.

[10] Wolfgang Felger. How Interactive Visualization Can Benefit from Multidimensional Input Devices. In *Proceedings of the SPIE - The International Society for Optical Engineering*, Band 1668, Visual Data Interpretation, S. 15–24, 1992.

[11] Martin Friedmann, Thad Starner, Alex Pentland. Device Synchronisation Using an Optimal Linear Filter. In *Proceedings of the 1992 Symposium on Interactive 3D Graphics, Special Issue of Computer Graphics, Band 26*, S. 57–62, 1992.

[12] R. C. Gonzalez, P. Wintz. *Digital Image Processing*. Addison-Wesley, Reading, MA, 2 edition, 1987.

[13] A. Goshtasby, F. Cheng, B. A. Barsky. B-Spline Curves and Surfaces Viewed as Digital Filters. *Computer Vision, Graphics, and Image Processing*, 52:264–275, 1990.

[14] Andrew J. Hanson. The Rolling Ball. In David Kirk (Hrsg.), *Graphics Gems III*, S. 51–60. Academic Press, 1992.

[15] H. Heuser. *Lehrbuch der Analysis, Teil 2*. B.G. Teubner Stuttgart, 1983.

[16] X. Y. Jiang, H. Bunke. Ein konturbasierter Ansatz zur Berechnung von Momenten. In *Informatik Fachberichte 290*, S. 143–150. Bernd Radig, October 1991.

[17] B. Klaus, P. Horn. *Robot Vision*. The MIT Press, McGraw-Hill Book Company, 1986.

[18] M. W. Krueger. *Artificial Reality*. Addison-Wesley, 1981.

[19] Jia-Guu Leu. Computing A Shape's Moments from Its Boundary. *Pattern Recognition*, 24(10):949–957, 1991.

[20] P. Maes, T. Darrell, B. Blumberg, S. Pentland. Interacting with Animated Autonomous Agents. In *Announced paper for SIGGRAPH 94, Orlando, FL*, 1994.

[21] C. Maggioni. A Novel Gestural Input Device for Virtual Reality. In *Proceedings of IEEE 1993 Virtual Reality Annual International Symposium, VRAIS*, S. 118–124, 1993.

[22] C. Maggioni. Non Immersive Control of Virtual Environments. In *Virtual Reality, Anwendungen & Trends, IPA/IAO-Forum*, S. 129–142, Stuttgart–Vaihingen, 9–10 Februar 1994. Springer Verlag.

[23] J. S. C. Yuan. Mukundan. Estimation of Quaternion Parameters from Two Dimensional Image Moments. *Graphical Models and Image Processing*, 54:345–350, 1992.

[24] B. A. Myers, M. Rosson. *Survey on User Interface Programming*. Number (#77629) in RC 17624. IBM Research Division, January 1992.

[25] M. Nadler. A Note on Coefficients of Compass Mask Coefficients. *Computer Vision, Graphic, and Image Processing*, 51:96–101, 1990.

[26] P. Nahim. The Theory of Measurement of a Silhouette Description for Image Processing and Recognition. *Pattern Recognition*, 6(2):85–95, 1974.

[27] F. Quek. Toward a Vision-Based Hand Gesture Interface. In *Conference on Virtual Reality Software and Technology*, Singapore, August 23-26 1994.

[28] J. Rehg, T. Kanade. DigitEyes: Vision-Based Human Hand Tracking. *File: CMU-CS-93-220.ps.Z ftp:reports.adm.cs.cmu.edu:/usr0/anon/1993*, S. 1–17, 1993.

[29] I. Rothe, K. Voss. *Orientierungsbestimmung von Objekten durch Momentinvarianten*, S. 42–49. Informatik Aktuell. Springer Verlag: S. Fuchs, R. Hoffmann, 1992.

[30] D. Rubine. Specifying Gestures by Example. *Computer Graphics*, 25(4):329–337, 1991.

[31] C. Ware, S. Osborne. Exploration and Virtual Camera Control in Virtual Three Dimensional Environments. In *ACM Transactions On Computer Graphics*, S. 175–183, Oct. 1990.