Forschungsberichte des
Instituts für Phonetik und
Sprachliche Kommunikation
der Universität München
(FIPKM) 37 (2001) 155-178

Visual Recognition of Continuous Hand Postures

Claudia Nölker and Helge Ritter

Neuroinformatics department
Faculty of Technology
Bielefeld University, Germany
{claudia,helge}@techfak.uni-bielefeld.de

ABSTRACT

This paper[1] describes GREFIT (**G**esture **RE**cognition based on **FI**nger **T**ips), a neural network-based system which recognizes continuous hand postures from grey level video images. Our approach yields a full identification of all finger joint angles (making, however, some assumptions about joint couplings to simplify computations). This allows a full reconstruction of the 3D hand shape, using an articulated hand model with 16 segments and 20 joint angles. GREFIT uses a two-stage approach to solve this task.

In the first stage, a hierarchical system of artificial neural networks combined with a-priori knowledge locates the 2D-positions of the fingertips in the image.

In the second stage, the 2D-position information is transformed by an artificial neural net into an estimate of the 3D-configuration of an articulated hand model, which is also used for visualization. This model is designed according to the dimensions and movement possibilities of a natural human hand.

The virtual hand imitates the user's hand to a remarkable accuracy and can follow postures from grey scale images at a frame rate of 10 Hz.

1 INTRODUCTION

Human-computer-interaction (HCI) today is still mainly based on means of communication which are not natural for us. Although keyboard and mice as input devices have grown to be familiar,

[1]This paper is an abridged version of [1].

they inherently limit the speed and naturalness with which we can interact with the computer. But rather than continuing to demand that humans acquire machine-like skills, it is much better to enable machines to acquire human-like skills, such as recognizing speech and gesture. The use of hand gestures is a common means of non-verbal interaction among people and can, therefore, also provide an attractive alternative to cumbersome interface devices.

To exploit the use of gestures in HCI, it is required that configurations of the hand and/or arm become measurable by the machine. First attempts to solve this problem resulted in glove-based devices [2]. Whereas these permit a fast and reliable measurement of the joint angles, they are not quite accepted by the user due to the awkwardness of wearing them and being connected to the machine by cables.

In contrast, visual interpretation of hand gestures is more convenient for the user and can thus help in achieving the ease and naturalness desired for HCI [3]. The first approaches using video-based recognition of hand gestures simplified the computational problem by attaching colored markers on selected body landmarks. The next step was a classification of the hand shape into one of several pre-defined posture classes. This could be done on non-adorned hands but limited the type of applications severely. For future applications, it will become more and more necessary to achieve the exact parameters of the hand in order to evaluate the hand posture and with it the intention of the user. This is also evident with the development of new technologies, like MPEG for the transmission of multi-media data.

The GREFIT system introduced in this paper is a video-based non-contact interface which computes the angular values of the human hand needed for the description of the BAPs (body animation parameters) of the MPEG-4 version 2 standard. This *continuous parameterization* of a hand posture instead of the common classification opens up new application areas, where gestures are used to indicate extents of quantity, as required e.g. for object manipulation (sculpting, smoothing, joining, rotating), tele-control of a robot, or navigation.

This paper is structured as follows: First, we describe the selected approach of using the fingertips as a natural representation of the hand posture. A short survey of related work is presented in Section 3, before we describe the setup of the system for locating the fingertips in the following section. After the preprocessing (Section 5), the GREFIT system performs a global, then a local processing of the hand images which is described in Sections 6 and 7. An integration of knowledge and context information (Section 8) improves the results of locating the fingertips in the image, before the reconstruction of the hand posture is performed. We conclude with the accuracy of the GREFIT system in Section 10 and a discussion in Section 11.

2 Approach and System Architecture

We aim at a visual system for the recognition and reconstruction of a large, continuous subset of all hand postures instead of performing the usual classification into a number of discrete, fixed postures. Therefore, we need an expressive, continuous-valued representation of the hand shape for the further processing with a hierarchical approach.

2.1 Fingertips as Natural Determinants of Hand Posture

How can we describe the posture of the hand in an image by parameters? In this paper, we suggest to first localize the positions of characteristic landmarks in the image, and then to reconstruct the appropriate hand posture on the basis of these parameters.

The fingertips are a well-known set of characteristic landmarks for the human hand. This is motivated by results from Poizner et al. [4], who performed experiments with point light displays attached to different positions on arm and hand, finding that solely the movements of the fingertips are necessary for the recognition of American Sign Language (ASL). Also, research using an eye-tracker to evaluate the fixation points of subjects recognizing hand postures showed that the majority of the fixation points are near the fingertips [5]. Thus, fingertips work as natural determinants of hand posture.

This explains why fingertips are frequently used features in hand and gesture recognition. The task of locating the fingertips can be simplified by attaching specially colored markers on the hand of the user and by using color histograms for processing the images (e.g. [6]). Other approaches to find the fingertips without visual aids include different kinds of fingertip-templates (e.g. cylindrical models [7] or images of a prototype [8]). These methods all assume that the contrast of the image region in the area of the fingertip is high, i.e. that the fingers of the hand are (nearly) stretched – but this is a severe limitation to the huge number of possible hand postures.

Contrary to that, the GREFIT system is able to locate the fingertips even in areas with low contrast, i.e. when the fingertips are in front of the palm. No constraints of the possible hand postures are necessary and grey level video images are sufficient in this setup.

A big disadvantage of the usage of fingertips as features is their susceptibility to occlusions. This problem can be tackled by either using multiple cameras with different viewing directions, by tracking, or by integrating context information, e.g. the locations of the other fingertips.

2.2 Hierarchical System Architecture

We use a hierarchical approach consisting of several layers. The first layer performs a coarse processing of the entire pixel image to find an initial estimate of the location of each fingertip. Each higher layer then refines the position estimates of the previous (lower) layer by restricting processing to smaller subregions centered at the position estimates of the previous layer. Also, these subregions are processed at a higher spatial resolution (coarse-to-fine strategy). In this way, interesting areas of the image gradually get more *focal attention*. This is also a principle that can be observed in the human visual system, where the fovea is the area with the highest resolution of the image. Altogether, concentrating on the relevant data and neglecting the irrelevant parts results in a more robust recognition, and due to the reduced amount of data, the processing on the lower level can be done much faster.

In the GREFIT system, the position estimates are computed with the aid of neural networks of the LLM-type (see below)see Fig. 1. We additionally use a-priori knowledge to locate the fingertip of each finger.

2.3 System overview

After grabbing and preprocessing the image, we apply the global processing algorithm described below and then the local processing algorithm for *each* fingertip separately. This yields 5 fingertip positions in the image. For a more precise location, information about the shape and the context are included. As the next step, the obtained feature vector consisting of the 5 fingertip positions is transformed into the finger joint angles of an articulated hand model. A visualization of the results with this model completes the system. These steps are schematically depicted in Fig. 2.

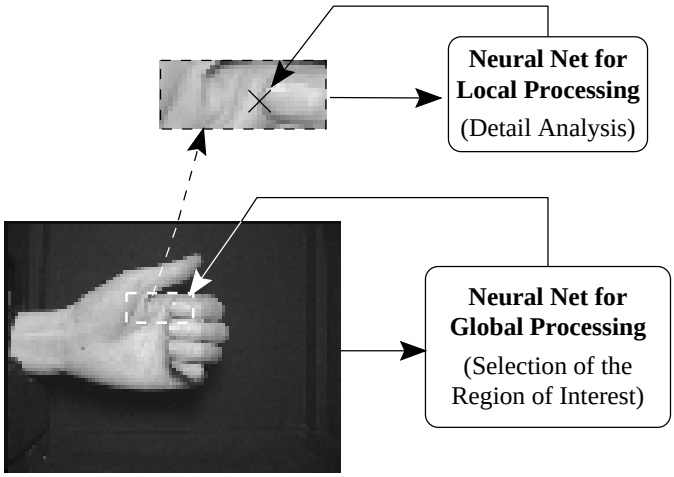

Figure 1: Hierarchical approach for the example of the fore fingertip.

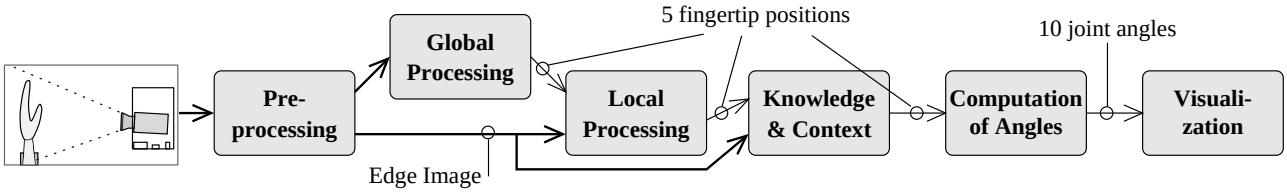

Figure 2: Overview of the GREFIT *system. Signal paths transmitting images are denoted with a thick line, data paths with a thin line.*

3 Related Work

Previous work on reconstructing hand posture from characteristic landmarks on the hand includes [9, 10, 11, 12]. In [9], a continuous parameterization of hand posture is obtained by providing a small set of basis postures from which a larger subset of actually occurring configurations can be linearly combined. A neural network is employed to identify the posture of a simulated robot hand from its image.

The hand model of DigitEyes [10] consists of 16 cylinders forming the fingers and the palm arranged as a kinematic tree, as familiar from robotics. The kinematics and geometry of the target hand must be known in advance. DigitEyes uses a set of 27 degrees of freedom (DOF). Starting with the hand in a known initial configuration, a state estimation algorithm first predicts feature positions for the next frame. Afterwards the difference between measured and predicted states is minimized. A high image sampling rate is essential for this tracking algorithm.

Lee and Kunii [11] present a hand model which is based on natural movement constraints of a human hand. With the help of color-coded gloves and stereo vision, seven "characteristic points" in 3D (all five fingertips, the wrist, and an extra point on the palm) are extracted. An iterative algorithm using external "image forces" then fits their model to the hand image. Due to the time-consuming solving of the inverse kinematics of the fingers, the algorithm is rather slow.

Millar and Crawford [12] employ a geometric model of the human hand which fulfills both the physical form and the kinematic constraints of the fingers. An analysis algorithm fits this model to six 3-dimensional marker positions at the fingertips and the center of the wrist. Since the fitting of the model is done directly, the computation is fast. The algorithm requires an initial standardization of the hand data into a predefined position and orientation. It generates a skeletal model of the hand using points and lines as its output.

Heap and Hogg [13] built a hand model with 498 vertices from a set of training examples from 3D MRI data. Afterwards, their systems constructs a Point Distribution Model and performs a PCA to find the modes of variation. The model is based on internal physical forces (to hold the model in shape) as well as external forces (to fit the model to the image data). The system can track an unmarked hand moving with 6 degrees of freedom in real time using a single video camera.

Shimada et al. [14, 15] use a 3D hand model with 23 degrees of freedom. Their system can be divided into two phases for matching of the hand posture: First, possible candidate configurations using the model are generated, second, the degree of matching with the silhouette is evaluated and extended Kalman filters are used for fitting the model to the image.

Ouhaddi and Horain [16] use an articulated 3D model of the hand that integrates bio-mechanical and anthropometric constraints and conforms to the MPEG-4 specifications, but includes only 12 degrees of freedom. The system minimizes the difference between features of the projected hand model and those extracted from the images by minimizing the non-overlapping surface in silhouette images and contour distances with classical optimization methods. The tracking system requires the hand to be in a fixed starting position and does not work in real-time.

4 Setup of the System

The setup of GREFIT is a "handbox" (see Fig. 3) containing a camera and a lamp for illumination of the scene. The box has the approximate size $30 \times 27 \times 50$ cm. The user inserts his hand through an

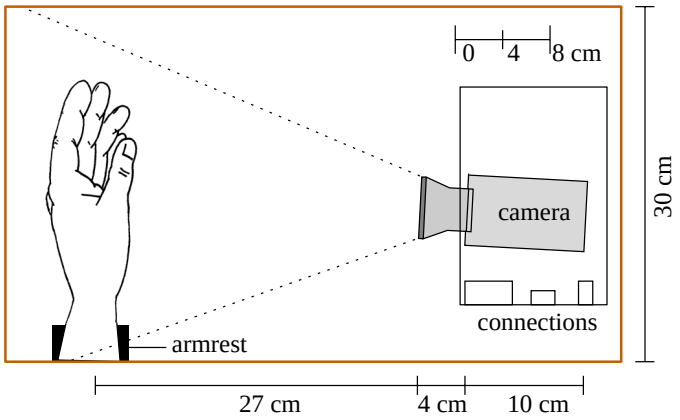

Figure 3: Outline of the handbox.

opening at the side into the handbox, an armrest prevents weariness. This setup guarantees a constant distance between the hand and the cameras. As the fingertip detection algorithm of GREFIT depends on the visibility of all 5 fingertips, the palm of the hand is required to face the camera, because this orientation reduces the possibility of occlusions.

The employed cameras of type Sony EVI-371DG are CCD cameras equipped with auto-focus lens and a controllable zoom function. A built-in shutter control normalizes the lighting. Additionally, a wide conversion lens is attached to each camera to shorten the distance between hand and cameras.

An ordinary energy-saving lamp (not shown in Fig. 3), which does not get hot when used, is placed above the cameras to illuminate the scene.

The inside of the box is painted black, facilitating segmentation sufficiently to allow the use of grey-scale images together with a simple threshold based segmentation algorithm.

The images are grabbed using a Matrox Meteor framegrabber card, the processing is done by an off-the-shelf computer (PentiumPro processor with 200 MHz) running Linux.

5 Preprocessing

The input to the system is a stream of monocular grey-scale images with a resolution of 192×144 pixels.

5.1 Image Segmentation

The background is blackened by performing a threshold operation on the image (object-background separation). This threshold is also used to obtain the silhouette image shown in Fig. 4.

Different skin colors did not raise a problem, since the automatic camera iris is able to compensate for differences in brightness. For the same reason, additional light sources (e.g. room illumination when the lid did not cover the box) do not disturb the detection process.

Figure 4: Preprocessing steps from left to right: Original image, object-background-separation, hand-arm-separation and centroid of the silhouette pixels, resulting final image.

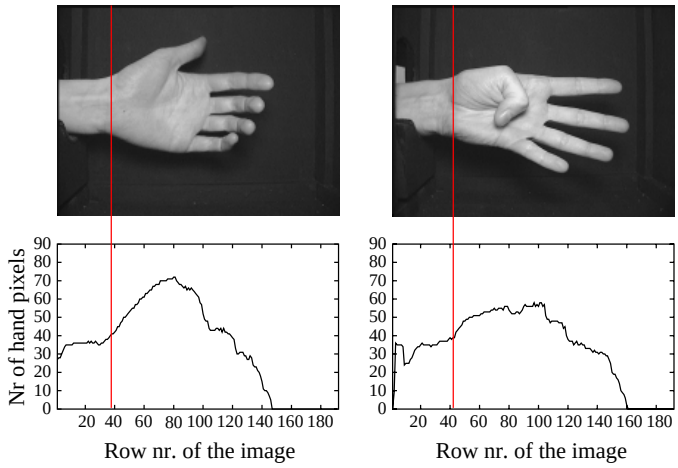

Figure 5: Hand-Arm-Separation. The images in the lower line depict the number of hand pixels per image row. The position of the wrist is assumed to be at that location where the variation of the number of object pixels per row exceeds a threshold.

5.2 Hand-Arm-Separation and Centroid-Computation

A masking of the arm is needed to meet the requirements for the subsequent centering of the hand. The masking method is pixel-based: We count the number of object pixels of each vertical image scan line. When the variation of the values exceeds a threshold, we assume that the wrist has been detected, because the hand is wider than the arm, see Fig. 5 for details.

As a last step, we compute the centroid of the pixels of the separated hand. This position will become the center for the following computation of the feature vector. In this way, we can compensate for translations. Figure 4 illustrates these preprocessing steps.

5.3 Calibration to Hand Size

Whereas normalization of illumination, background and distance between hand and camera is guaranteed by the setup, the hands of the users are varying. Although the anatomical system of the human hand is fixed, hand shapes, structure and surfaces show considerable individual differences. Therefore, a normalization is an important step to achieve good performance. Since an exact adaptation to the specific hand of the user would be presumptuous, we refrain from this and perform only a simple normalization according to the size of the hand by adapting the zoom value of the camera. We use the total number of object pixels in the silhouette image as the criterion. Therefore, we ask the user to stretch out his/her fingers during the short calibration process. The zoom value of the camera is adjusted until the percentage of hand pixels in the image is approximately 25 percent.

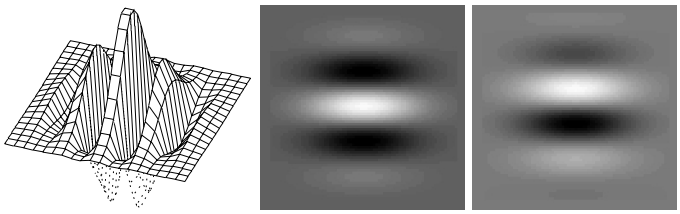

Figure 6: Left: Typical shape of a Gabor-function. Right: Filter kernels of a Cosine- and Sine-Gabor filter.

5.4 Edge Detection

As a further preprocessing step, we perform edge detection by applying a Laplace filter to the image. The edge image is used for the local processing in Section 7. The edge image is also used in order to significantly reduce the number of possible fingertip positions through the integration of a-priori knowledge, since we require the fingertip position to coincide with an edge pixel (see Section 8).

We applied an orientation-invariant Laplacian operator on the image data, and thresholded the results to obtain binary edge maps. The thresholds are chosen so that the edge detection yields good results especially in areas with low contrast, e.g. if the fingertips are in front of the palm.

6 Global Processing

In order to get a first estimate of the fingertip positions, we compute a lower-dimensional representation of the image by a set of Gabor filters, and apply an artificial neural network to the resulting feature vector.

6.1 Gabor-Feature-Vector

The input vector of 6912 pixel intensities of an image (subsampled at half resolution of 96×72 pixels) is much too high-dimensional for a direct use as a feature vector. To get a low-dimensional representation of the image, we apply Gabor filters at several positions in the image. 2D-Gabor filters have been found by Jones and Palmer [17] to resemble the response profile of simple cells in the visual cortex of cats. The 2D-Gabor filter kernels have the shape of a Gaussian modulated by a sinusoidal plane wave:

$$g(x,y) = \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) f\left(u_0 x + v_0 y\right) \tag{1}$$

with $f(x) = \sin(x)$ or $f(x) = \cos(x)$, where σ is the width of the Gaussian and u_0, v_0 are parameters to specify the frequency and orientation of the filter. Figure 6 depicts the typical shape of a Gabor function and the filter kernels of a Cosine- and a Sine-Gabor filter. Due to possible ambiguities in the filter response, Gabor filters are usually applied as a filter pair, for instance Cosine and Sine filter.

In order to compute the feature-vector, a cross-correlation of the filter with the image is performed. High resemblance of the picture with the filter kernel yields high values in the filter response. As Gabor filters are sensitive to directions, a number of filters with different orientations are combined ("Gaborjet").

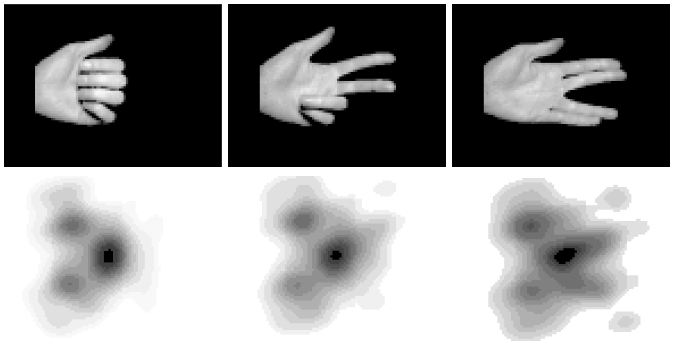

Figure 7: Visualization of the application of the Gaborjets to the preprocessed images. The intensities of the pixel values picture a product of the 35-dimensional Gaborjet vector with the filters for each component of the vector. The darker the pixels, the higher is the conformity with the filters at that pixel position.

At five points (arranged like the "Five" on a dice) in the image[2], we applied a Gaborjet consisting of Cosine- and Sine-Gabor filters of 3 different orientations each ($0°$, $60°$, $120°$) plus an additional isotropic Gaussian. This results in a 35-dimensional feature vector. The frequencies as well as the width σ of the Gaussian were adjusted to the size of the image, so that the filters cover the area of the hand. In Fig. 7, we take the 35 Gaborjet coefficients, multiply each coefficient by its corresponding filter and sum all the weighted filters to produce the images which are shown.

6.2 LLM-Net

For our approach we make use of an LLM (Local-Linear-Mapping)-network (for further explanation see [18, 23]), which is related to self-organizing maps [19], to generate a smooth mapping from the input variables (in our application a feature vector $\in \mathbb{R}^{35}$) onto the output variables (position $\in \mathbb{R}^2$ in the image).

An LLM-net has a fixed number n of units (or nodes), each of which consists of a reference vector $\vec{w}_i^{\text{in}} \in \mathbb{R}^L$ in the input space and an associated reference vector $\vec{w}_i^{\text{out}} \in \mathbb{R}^M$ in the output space ($i = 1, \ldots, n$) together with a matrix A_i defining a local linear mapping from $\mathbb{R}^L$ to $\mathbb{R}^M$.

The task of the matrices A_i is to approximate the mapping linearly in the vicinity of the prototype vectors $\vec{w}_i^{\text{in}}$ (cf. Eq. (3) below).

[2]The central point is placed on the centroid of the object pixels (see Fig. 4 and Section 5).

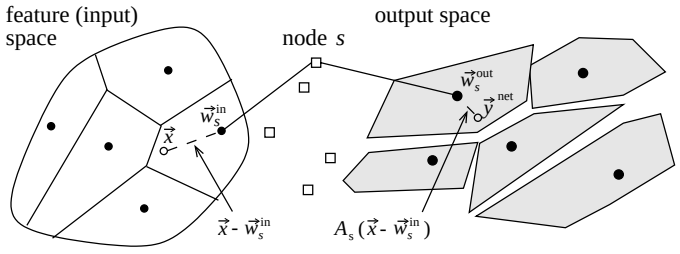

Figure 8: Local linear mapping network.

During training, the net gets new training examples each of which consists of a feature vector $\vec{x} \in \mathbb{R}^L$ and the associated output vector $\vec{y} \in \mathbb{R}^M$. By comparison of $\vec{x}$ and the reference vectors of the input space, the reference vector $\vec{w}_s^{\text{in}}$ with the smallest Euclidean distance d_s

$$d_s = \min_i d_i, \quad \text{where} \quad d_i = ||\vec{x} - \vec{w}_i^{\text{in}}|| \tag{2}$$

is determined.

In the winner-take-all variant of the net, merely the best-match unit s determines the answer $\vec{y}^{\text{net}}$ in the output space. Smoother results can be achieved if the output of the network is a weighted superposition of the outputs of the individual nets. In both cases, the difference $\vec{x} - \vec{w}_s^{\text{in}}$ between the input value and the reference vector in the input space is converted into a linear correction term (see Fig. 8). Therefore, the answer of the net is

$$\vec{y}^{\text{net}} = \vec{w}_s^{\text{out}} + A_s(\vec{x} - \vec{w}_s^{\text{in}}). \tag{3}$$

At the end of each training step an adaptation of the reference vectors $\vec{w}_s^{\text{in}}$, $\vec{w}_s^{\text{out}}$ and the matrix A_s is performed according to the equations

$$\begin{aligned} \Delta\vec{w}_s^{\text{in}} &= \varepsilon_1(\vec{x} - \vec{w}_s^{\text{in}}), \\ \Delta\vec{w}_s^{\text{out}} &= \varepsilon_2(\vec{y} - \vec{y}^{\text{net}}) + A_s\Delta\vec{w}_s^{\text{in}}, \qquad (4) \\ \Delta A_s &= \varepsilon_3(d_s^2)^{-1}(\vec{y} - \vec{y}^{\text{net}})(\vec{x} - \vec{w}_s^{\text{in}})^T, \qquad (5) \end{aligned}$$

where the learning step sizes $\varepsilon_1, \varepsilon_2, \varepsilon_3$ decrease slowly. Due to this adaptation step, the reference vectors accumulate in areas with many training examples. This partitions the input space into a Voronoi tessellation.

6.3 Training and Results

For each finger, separate networks were trained on the same feature vectors. The input space is 35-dimensional (Gabor-feature vector), the output space is 2-dimensional and represents the (x,y)-position in the image. The LLM-networks have $n = 6$ nodes, the learning step sizes $\varepsilon_1, \varepsilon_2, \varepsilon_3$ were chosen to be 0.8 at the beginning and gradually decreased to 0.0001. The training examples were presented in random order for 5 epochs.

A data set of 1000 example images consisting of 10 sequences with manually marked fingertip positions was prepared for training and testing. These images were divided up into 10 different partitionings consisting of 900 images for training and 100 images for testing. We then trained 10 neural networks for each training/test-set and computed the average from these data as the overall result.

All errors – distances between the correct position and the detected positions – are measured in pixels (192×144 image size). Figure 9 presents a box plot of the results of the training and the test runs. The average error is denoted with a bar inside each box, the variance is indicated by the dashed lines. The size of the box denotes the 95%-limit which means that the error of 95% of all examples is below this value. The maximum error is shown as a whisker in the plot. The training results are displayed with white boxes, the test results with grey boxes.

The results in Fig. 9 show that the maximum error in the test is always smaller than the same value in the training, the other values are quite similar. This signifies that the networks did not just specialize on the training images, but achieved a good generalization. Only slight differences occurred within the 10 different training/test-sets.

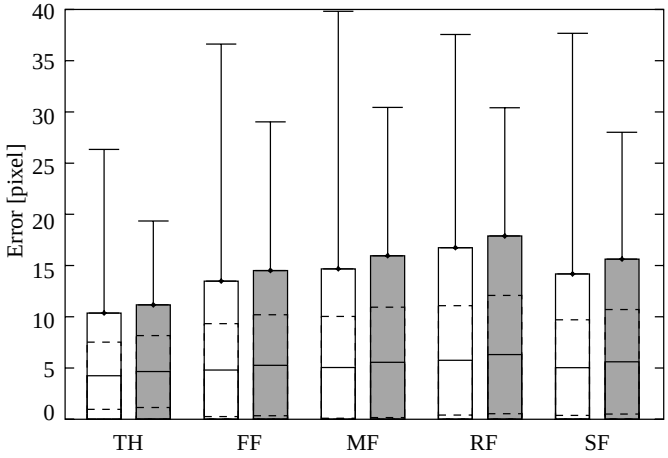

Figure 9: Results of training (white boxes) and test (grey boxes) of the LLM-nets with 6 nodes and a 35-dimensional Gabor feature-vector.

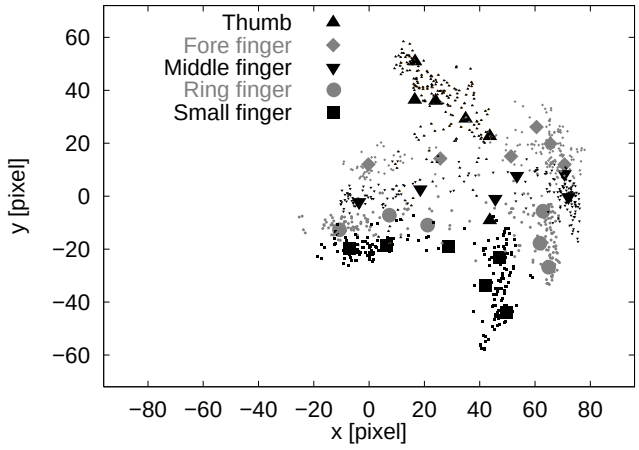

Figure 10: Movement areas of the fingertips (relative to the centroid) in a typical sequence of 100 images (tiny symbols), and positions of the 6 reference vectors of the LLM-networks in the output-space for the thumb and fingers.

The networks for the thumb achieve the best results, whereas the positions of the middle finger and ring finger seem to be the most difficult to learn. A possible reason may be that a movement of the thumb produces more pronounced changes in the feature vector than a movement of the fingers: Since we computed orientation-dependent Gabor-filter features, the usually parallel movements of the fingers have less effect in the feature vector than the predominantly perpendicular movements of the thumb in the image.

Figure 10 displays the positions of the fingertips in a typical sequence of 100 images relative to the image centroids. It shows that, after training, the reference vectors in the output space are arranged so that they cover the movement area of the fingertips quite well.

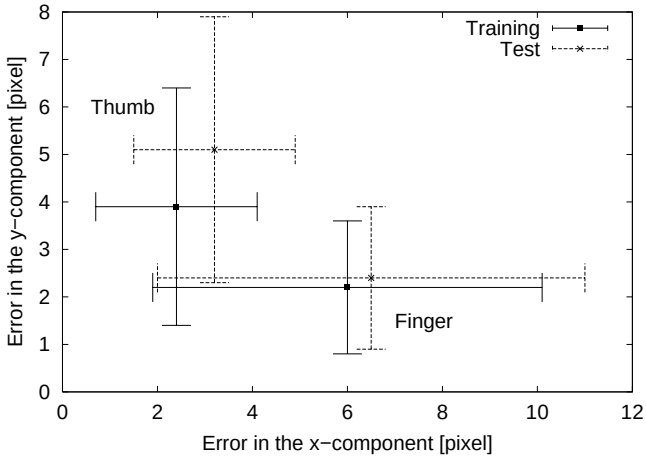

Figure 11: Error and standard deviation of the error of the global processing by the LLM-networks for the thumb and the other fingers in training and test.

7 Local Processing

The fingertip position estimates from the preceding, global processing stage are now used to define smaller regions of interest (ROI's) for the subsequent local processing stage. Each ROI is centered at one of the estimated fingertip positions and is a sample with the original, full pixel resolution. Again, we first compute a feature vector from which then a second ANN (one for each ROI) computes a correction to the initial position estimate.

7.1 Size of Region-Of-Interest

An optimally chosen size of each ROI is crucial for the further processing, since it determines the search-area and should cover the uncertainty range of the estimates of the first (global) processing stage. Therefore, we chose the sizes so that they are in line with the error and its standard deviation of the results of the global processing. As can be seen in Fig. 11, error and variance for the finger position estimates[3] in the x-direction are approximately three times the size as in the y-direction, therefore we selected ROI's of 36×12 pixel size.

For the thumb, the y-coordinate contains the largest error and variance, therefore, the thumb ROI size was set to be 16×24 pixels.

7.2 Feature-Vector

Similarly to the global processing step, we again apply Gaborjets to the edge image of each ROI in order to compute a feature vector. The Gaborjets were arranged on the sub-images so that they fit to the respective size and shape. Again, we use Gaborjets consisting of 7 filters at 5 positions to obtain a 35-dimensional feature vector.

[3]No significant difference occurred within the results for each finger, so the results were summarized.

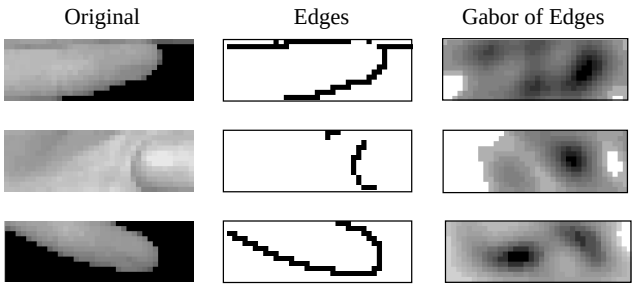

Figure 12: Examples of original images, edge images and visualization of the Gaborjet vector of the sub-images.

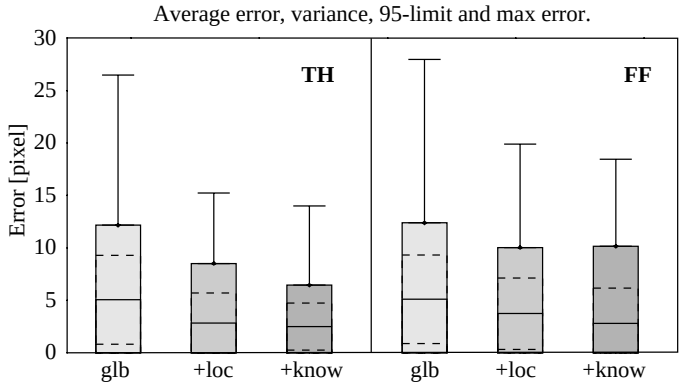

Figure 13: Results of the hierarchical approach for the thumb (TH) and the forefinger (FF). The results of the global processing (glb) are significantly improved by the application of local processing (+loc). Subsequent integration of knowledge and context (+know) improves the results further.

7.3 Training and Results

Since it is not possible to distinguish the fingers by their tips alone (their local shape and size are nearly identical), we used only two LLM-networks for the local stage: one network is trained to find the tip of the thumb, another network for the remaining fingers, which are sufficiently similar to allow the use of a single, shared network.

To this end, we generated the data for the training and test of the LLM-networks by randomly selecting subimage patches containing a fingertip, where the average distance of the fingertip position to the center of the image was set to 7 pixels $\pm$ a variance of 3 pixels. The size of the training set was 2400 subimages, the test set contained 600 sub-images.

As in the global processing, we partitioned the subimages into 10 different training/test-sets. For each set 10 neural networks with 3 nodes were trained by presenting training examples in random order for 5 epochs. The reported data are average values for all sets and network results. The display in Fig. 13 is analogous as in Fig. 9.

The improvements (see Fig. 13) achieved by the local processing for a number of example images are illustrated in Fig. 14.

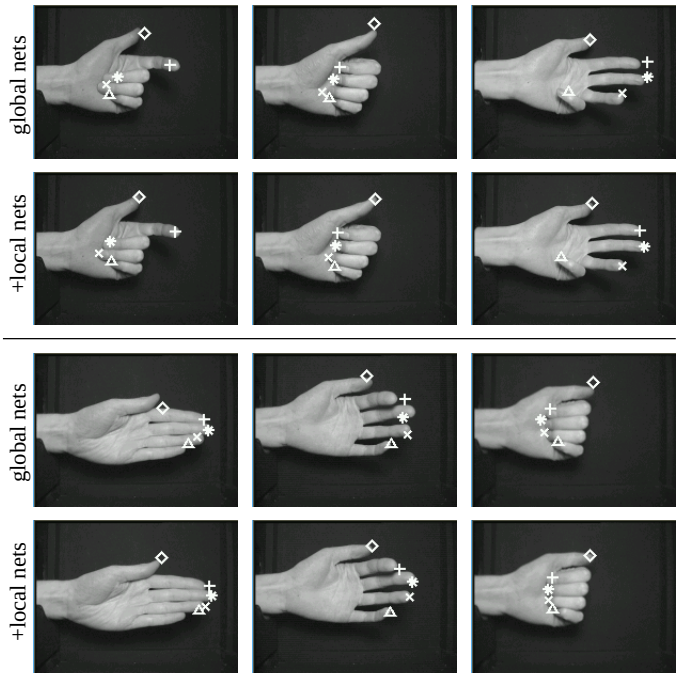

Figure 14: Results of the global networks with 6 nodes in the upper rows and the subsequent results when adding the local networks (lower rows).

8 Integration of Knowledge and Context

In this section, we discuss improvements in the GREFIT system by making use of additional information sources, namely a-priori knowledge about the shape of the hand and context information about the special setup. As this is also a local processing step, we are working on sub-images which are centered on the detected fingertip position.

8.1 Prior Knowledge

No individual technique of computer vision is universal. Instead, only the combination of several – typically complementary – information channels leads to reliable and robust systems [20]. In the present case, we can improve performance by combining complementary results of the processing of the sub-image into a "confidence image" representing our confidence for the presence of a fingertip at each of the pixels of the image.

Three channels of information are integrated in the GREFIT system: The results of the global+local computation is represented as a Gaussian activity profile with maximum value (i.e. the highest confidence) at the center of the sub-image and decreasing values for the more distant pixels, see Fig. 15, second row, for the Gaussian combined with the edge image. Second, since the fingertips are on the borders of the fingers, we restricted the possible positions to the edge pixels. To this end, we used the edge images computed in Section 5, which limited the number of possible points severely. Third, a simple matching with a "fingertip filter" of the size 5×5 square pixels for the shape of the fingertips evaluated the conformity of the edges with a typical "fingertip edge", i.e. a short vertical line with a rounding at one or both sides (see Fig. 15, second and third row).

All three information channels are normalized to yield images with confidence values in $[0, 1]$. These

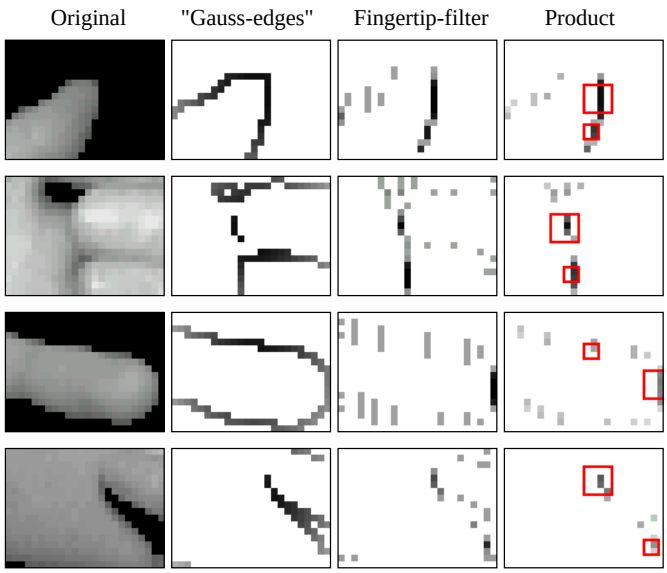

Figure 15: Combination of different information channels (from left to right): Original image, Gaussian activity profile combined with the edge image, fingertip-filter and the product of these confidence images. The pixel with the highest confidence for a fingertip is indicated by a big square, a possible alternate position with a small square.

solitary confidence images are pixel-wise multiplied to get the overall result. The two highest values in the product image with a minimum separation of 4 pixels are then used as the most probable candidate and an alternative position. Figure 15 visualizes the original sub-image and its confidence images with the most reliable fingertip locations indicated by squares.

8.2 Context

The detection of features can be further improved, if constraints about the scene or the object(s) can be integrated in the process of analysis. This additional knowledge can for instance help with the interpretation of the scene by providing background information about typical shapes, patterns of movement, etc.

In the GREFIT system, an implicit restriction to the area is already achieved by the trained global neural networks, each of which has learned the typical area of movement for its finger, see Fig. 10. As additional context information, we impose the constraint that two fingertips can not be at the same position. To this end, we replace the best-guess candidates with alternative positions whose confidence values are high whenever the chosen candidates of adjacent fingers would be less than 4 pixels apart.

9 Reconstruction of Hand Posture

In order to reconstruct the joint angles of the hand from the positions of the fingertips, we employ a geometric model of the hand. This requires us to solve the inverse kinematics. As we will show below, a very convenient way to solve this task is the use of a PSOM (Parameterized Self-Organizing Map), which is trained in the forward direction and can then be easily inverted. Finally, a plausibility

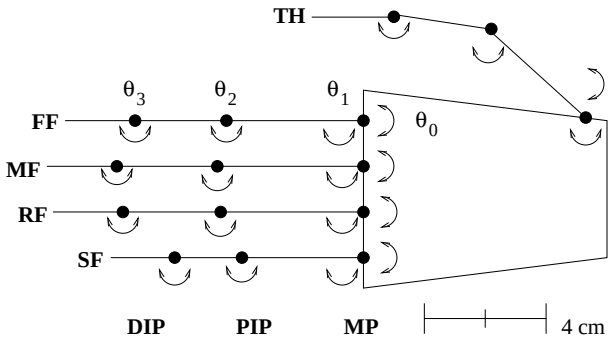

Figure 16: Structure of the hand model and naming of joints. See text for abbreviations.

test checks if some additional movement constraints of adjacent fingers are fulfilled.

9.1 Articulated Model of the Hand

For the reconstruction and visualization of the hand posture, we developed a virtual model of the hand. It represents the shape and incorporates constraints based on the natural movement of the human hand. With the hand model *any* hand posture can be imitated, which is essential for the computation of the finger angles as well as the output of GREFIT .

Our hand model is composed of 16 rigid elements, one for the palm, and three segments for each finger and the thumb. The sizes of the segments have been taken according to a medium sized human hand and are depicted in Fig. 16. In order to achieve a great variety of finger movements, we provide four degrees of freedom (DOF) for each finger and the thumb (see Fig. 16).

The fingers and the thumb have two DOF at the anchoring joint (or the metacarpophalangeal joint, MP) allowing flexion with an angle θ_1 as well as spreading movement expressed by θ_0. The proximal interphalangeal (PIP) and the distal interphalangeal (DIP) joint have one DOF each. The angles θ_2 and θ_3 represent their degree of flexion. Altogether, our hand model has 20 degrees of freedom.

Indeed, not all of these joint angles of a human hand are capable of fully independent actuation, but are correlated (see [21]). An example are the last two joints of the fingers, i.e. the PIP and DIP, which are driven by the same tendon. Approximately, their relation is

$$\theta_3 = \frac{2}{3}\theta_2 . \tag{6}$$

Another simplification is to equate the joint angles θ_2 and θ_1 that determine the flexion of the fingers:

$$\theta_2 = \theta_1 . \tag{7}$$

This constraint does not exactly correspond to the situation of the human hand, but is a good approximation.

With these constraints, a 10-dimensional state vector represents the hand posture. Using this 2-dimensional position data, we can reconstruct the 3-dimensional hand posture.

We also included the convergence of the fingers when the fist is clenched (see [21]): The axes of the fingers are not perpendicular to the finger segments, but are slightly oblique. As a consequence, the fingers are closely side-by-side at complete flexion.

Table 1: Allowable ranges of joint angles of the finger and thumb. (after [21]).

	$\theta_0^{\min}$	$\theta_0^{\max}$	$\theta_1^{\min}$	$\theta_1^{\max}$
TH	-30	20	-20	40
FF	-5	25	0	90
MF	-10	10	0	90
RF	-15	5	0	95
SF	-20	5	0	100

Even though the allowable ranges of finger joint angles vary from person to person, general ranges as in Table 1 can be specified. Here, we only consider the active movements of the joints. These are activated by the tendons and muscles of the hand, whereas the passive movement is externally forced. The zero position of the thumb is defined as the state where no thumb muscles are active. The fingers are stretched when in zero position, their axes are parallel.

9.2 PSOM-Net

We employ a PSOM (Parameterized Self-Organizing Map) [22] to compute for each finger the appropriate finger joint angles from the 2-dimensional position of the fingertip in the image. Here, the employed neural network can only be described briefly, for details see [24, 25].

The PSOM is an extension of the well-known SOM by Kohonen [26]. It uses a set of nonlinear basis manifolds to construct a mapping through a number of "topologically ordered" reference vectors. The advantage of the PSOM in comparison to the SOM is that fewer reference vectors are required for a good result. Also, learning is very fast.

The PSOM replaces the discrete lattice of a SOM by a continuous "mapping manifold" S. Therefore, a reference vector $\mathbf{w_a}$ is assigned to each grid point $\mathbf{a} \in \mathbf{A} \subset S$ (see Fig. 17). The reference vectors have to be topologically ordered, i.e. neighbors on the lattice are neighbors in the data space, too. A basis function $H(\mathbf{a}, \mathbf{s})$ is also associated with each grid point $\mathbf{a} \in \mathbf{A}$. The basis functions realize a smooth interpolation of intermediate positions between the grid points. The manifold M in the embedding space X can then be described as the image of S under $\mathbf{w}(\cdot)$ where

$$\mathbf{w}(\mathbf{s}) = \sum_{\mathbf{a} \in \mathbf{A}} H(\mathbf{a}, \mathbf{s})\, \mathbf{w_a}. \tag{8}$$

The basis functions have to meet two conditions. The first *orthonormality* condition

$$H(\mathbf{a}_i, \mathbf{a}_j) = \delta_{ij} \quad \forall \mathbf{a}_i, \mathbf{a}_j \in \mathbf{A} \tag{9}$$

ensures that the manifold M given by Eq. (8) passes through all desired support points $\mathbf{w_a}$, $\mathbf{a} \in \mathbf{A}$. Second, the sum of all contributing weights should be one:

$$\sum_{\mathbf{a} \in \mathbf{A}} H(\mathbf{a}, \mathbf{s}) = 1 \quad \forall \mathbf{s} \in \mathbf{S}. \tag{10}$$

This ensures that a translation of all reference vectors entails a corresponding translation of the PSOM manifold without changing its shape.

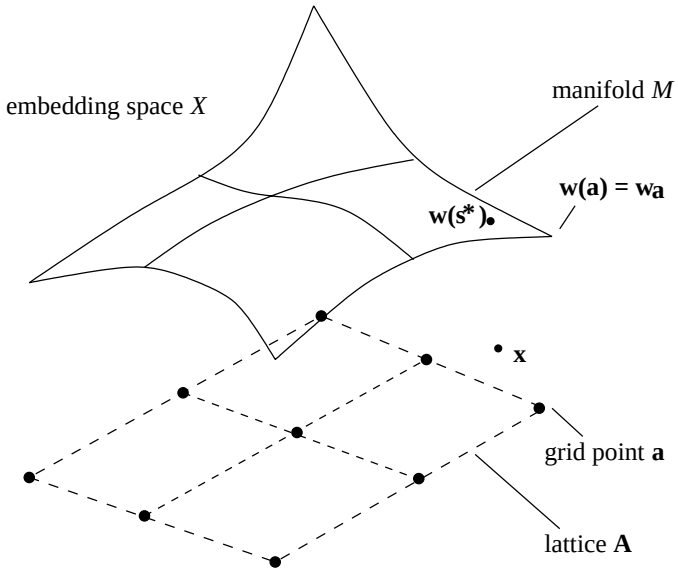

Figure 17: Illustration of a training setup for a 3×3 PSOM.

During application, an initial point $\mathbf{x}$ in the data space X is mapped onto the nearest point $\mathbf{w}(\mathbf{s}^*) \in M$ with

$$\mathbf{s}^* = \arg\min_{\mathbf{s}} \|\mathbf{x} - \mathbf{w}(\mathbf{s})\|. \tag{11}$$

This minimization is done using an iterative procedure: We start with the point $\mathbf{a}$ on the lattice $\mathbf{A}$ which has the smallest distance to $\mathbf{x}$, take the associated reference vector $\mathbf{w_a}$, and apply gradient descent along the manifold M to find the best approximation. The output $\mathbf{w}(\mathbf{s}^*)$ of the PSOM is then computed using Eq. (8).

The PSOM can be applied to map an input vector $\mathbf{x} \in X$ onto a map coordinate $\mathbf{s}$. Beside this mapping task, it is also able to perform an associative completion of fragmentary input. We took advantage of this ability and applied the PSOM to the inverse kinematics task.

9.3 Inverse Kinematics with the PSOM

One PSOM for each finger is trained by providing the training data in topological order. The training data are generated by varying the joint angles θ_0 and θ_1 (here, the lattice $\mathbf{A}$ is 2-dimensional) in their ranges of movement (see Table 1) and computing the resulting fingertip positions by forward kinematics with the hand model. This produces 4-dimensional reference vectors of joint angles and the respective fingertip positions (here, $X = \mathbb{R}^4$), which can be split into a 2-dimensional subset of input variables (the 2D-positions) and two remaining output variables (the joint angles)

$$(\theta_0, \theta_1, x, y).$$

After training, the PSOM realizes the mapping from the 2-dimensional position of a fingertip in the image onto the joint angles of the respective finger

$$(*, *, x, y) \rightarrow (\theta_0, \theta_1, x, y).$$

With this learned mapping, the PSOM completes the partially-known vector to a completely-known vector.

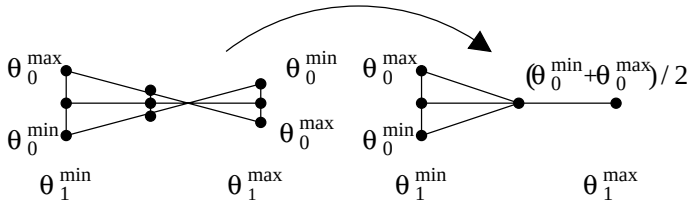

Figure 18: Structure of the PSOM where θ_0 denotes a spreading of the hand, and θ_1 identifies variation in flexion for the example of a 3×3 grid. Left: Ambiguity in the joint angles at the intersection point. Right: Improved structure.

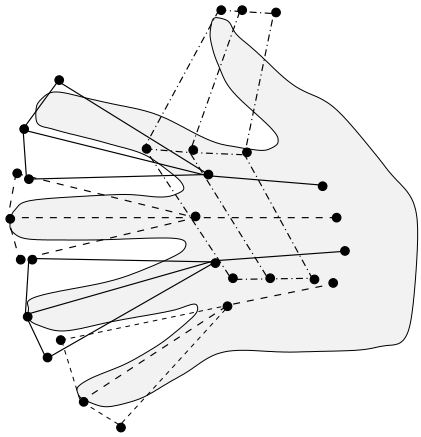

Figure 19: Movement areas of the fingers and the thumb.

The topological order of the training data is ensured by varying the joint angles on a grid structure, e.g. $(\theta_0^{\min}, \theta_1^{\min})$, $(\theta_0^{\min}, \frac{1}{2}(\theta_1^{\min} + \theta_1^{\max}))$, $(\theta_0^{\min}, \theta_1^{\max})$, $\ldots$ $(\theta_0^{\max}, \theta_1^{\max})$. Depicting the resulting fingertip positions yielded the shape of an "horizontal egg-timer" (see Fig. 18, left) with an intersection in the middle. This particular 2D-position can be achieved with many different joint angle combinations and represents a singular point of the kinematic transformation. Due to this ambiguity in the joint angles, the transformation between the 2D-position and the angles can not be solved by the PSOM at this singularity.

So far, our modeling of the ranges of motion was based on the assumption that the angles θ_0 and θ_1 of the fingers are *not* correlated. However, a closer look indicated that the abduction and adduction angle θ_0 decreases as the flexion angle θ_1 increases. In other words, the freedom of motion in θ_0 can only be fully exploited when the fingers are roughly stretched.

Since the grid structure had to be preserved for the application of the PSOM, we merged those grid points with large values in the flexion angle. The range of movement of the fingers is now limited to a "broom"-shaped area (see Fig. 18, right). As this changed structure has no singularity, the PSOMs perform well. The internal grid structure of the PSOM was maintained, but we provide fewer different reference vectors during training than points on the grid exist.

The thumb is the exception to this method. Since no singularities occur for its range of movement, the training of the PSOM is based upon the original grid. The resulting ranges of motion for all 4 fingers and the thumb are illustrated in Fig. 19 for a grid of size 3×3.

In addition, we disabled the ability of the PSOM to extrapolate, i.e. to infer values outside of the movement area. This ensures that the fingers do not take on "unhealthy" values for the joint angles.

This constraint equips the PSOM with implicit model information; anatomically injurious postures cannot occur.

9.4 Application

We tested the PSOMs with 100 randomly chosen test examples in the movement area of the respective finger. For evaluation, the results of the PSOMs were again transformed into the 2D-positions and directly compared with the input values. A comparison of PSOMs with different dimensions of the axes showed that a 4×4 grid yields the best results. Independent of the finger (and thumb), the average error is approximately 0.14 cm and the maximum error is always below 0.4 cm. (The length of the hand is 19 cm, the width of a finger 1.5 cm.)

9.5 Restrictions of movements of adjacent fingers

Whereas the qualitative facts are known, quantitative data for the couplings between angular values of adjacent fingers is rare. However, it is exactly these couplings which determine whether the resulting hand posture looks natural or not. In the GREFIT system, a plausibility test for the compliance of the posture is done on the basis of a number of inequalities proposed by Lee and Kunii [11]. With the help of the static joint angle limits in Table 1, they describe the dynamic joint angle limits for the flexion of the MP joint depending on the values of adjacent fingers. For example, the angle $\theta_{\text{MF,1}}$ of the MP joint of the middle finger can only take on values which obey the rules

$$\theta_{\text{MF,1}} \geq \max(\theta_{\text{FF,1}} - 25,\, \theta_{\text{RF,1}} - 45,\, \theta^{\text{min}}_{\text{MF,1}}) \quad \text{(MFmax)}$$
$$\theta_{\text{MF,1}} \leq \min(\theta_{\text{FF,1}} + 54,\, \theta_{\text{RF,1}} + 20,\, \theta^{\text{max}}_{\text{MF,1}}) \quad \text{(MFmin)}$$

which depend on the angular values of fore finger and ring finger.

Due to their structure, non-fulfilled inequalities always occur at least as a pair (e.g. (FFmax) and (MFmin)). If exactly two inequalities are not fulfilled, this can be fixed by sole changing of the joint angle value of either fore or small finger. Otherwise, at least one of the joint angles of the middle or ring finger has to be adjusted. In rare cases, where 4 inequalities are not fulfilled, a second checking-pass is necessary.

We included this plausibility test as the last step before displaying the hand posture with the articulated model.

10 Results

Here, we illustrate the performance of the GREFIT system. Figure 20 (upper rows) depicts a number of different hand postures offered to the system. The lower rows shows the corresponding 3D-reconstructions using the articulated hand model described schematically in Fig. 16. While Fig. 20 can only illustrate the identification of a discrete set of hand postures, we would like to stress that the system indeed can process a continuous sequence of hand postures of which the shown samples are only “snapshots”. See also website [27] for a short movie and an example application.

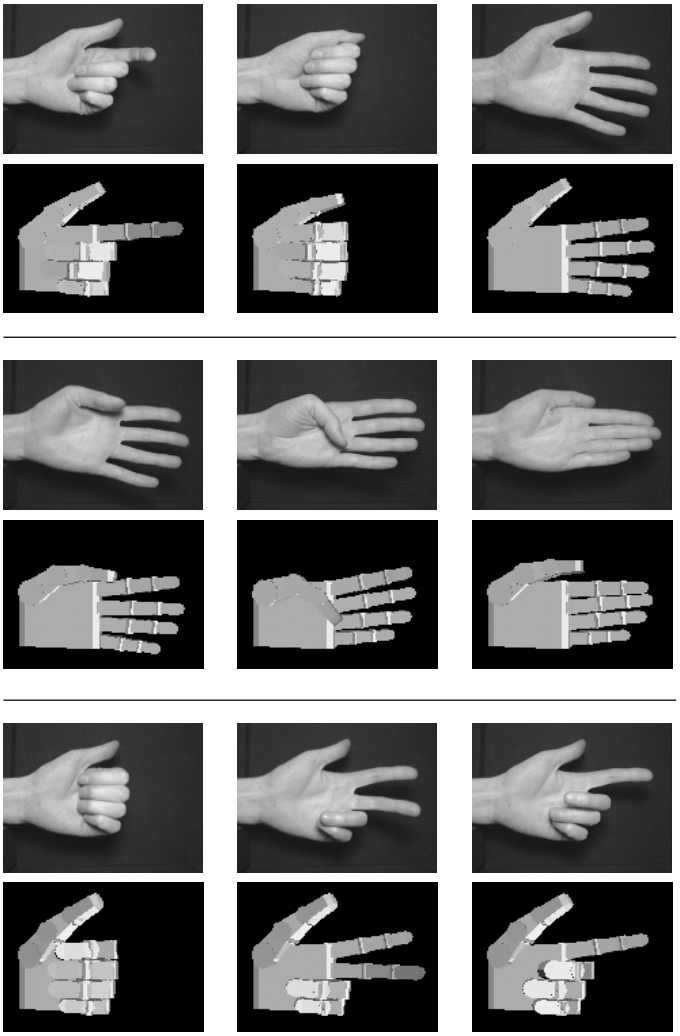

Figure 20: Visual control of the hand model using natural hand postures.

11 Discussion

As Fig. 20 shows, the original hand posture and the reconstruction match very closely. However, a comparison can only be done visually. This is standard practice. An exact, quantitative analysis would require a measurement with a DataGlove in order to compare the achieved values of the joint angles. But the external finger sensors will also disturb the visual recognition significantly and falsify the results. Therefore, such an evaluation is not feasible.

The GREFIT system is able to reliably reconstruct the hand posture at a frame rate of 10 Hertz. Currently, each image is processed separately. A further improvement could be achieved by including information about the preceding images in the sequence.

The perception of motion is often the first processing step in the visual system. Consequently, spatio-temporal variation in an image sequence is frequently used in computer vision, since it can reduce problems from temporary occlusions and restricts the search-area in the image, which results in less processing time. However, this demands the processing of image sequences with a frame-rate that is suitable for the moving object.

The movement of the fingers is potentially very fast. Therefore, the images have to be grabbed with a high frame rate in order to include sequence information. Recent hardware developments open up this possibility at reasonable cost.

The current vision-based interface provides one main building block towards making manual operations understandable for machines. Additional important steps will be the recognition of grasping postures and the analysis of bimanual hand-object configurations. Achieving this by purely visual means constitutes a major challenge for neural network based adaptive vision systems and is the objective of future work.

References

[1] C. Nölker and H. Ritter, "Visual Recognition of Continuous Hand Postures," to appear in *IEEE Transaction on Neural Networks, Special Issue Multimedia Processing*, July 2002.

[2] D. Sturman and D. Zeltzer, "A survey of glove-based input," *IEEE Computer Graphics and Application*, vol. 14, no. 1, pp. 30–39, 1994.

[3] V.I. Pavlović, R. Sharma, and T.S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 677–695, 1997, http://www.ifp.uiuc.edu/∼vladimir/research.html.

[4] H. Poizner, U. Bellugi, and V. Lutes-Driscoll, "Perception of American Sign Language in dynamic point-light displays," *Journal of Experimental Psychology: Human Performance and Perception*, vol. 7, no. 2, pp. 432–440, 1981.

[5] M.J. Mataric and M. Pomplun, "Fixation behavior in observation and imitation of human movement," *Cognitive Brain Research*, vol. 7, pp. 191–202, 1998.

[6] Y. Iwai, K. Watanabe, Y. Yagi, and M. Yachida, "Gesture recognition using colored gloves," in *International Conference on Pattern Recognition 96*, 1996, pp. A 662–666.

[7] J.M. Rehg and T. Kanade, "Model-based tracking of self-accluding articulated objects," in *Proc. Int. Conference on Computer Vision*, jun 1995, pp. 612–617.

[8] Y. Sato, Y. Kobayashi, and H. Koike, "Fast tracking of hands and fingertips in infrared images for augmented desk interface," in *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition (FG 2000)*, 2000.

[9] A. Meyering and H. Ritter, "Learning 3D-shape perception with Local Linear Maps," *Proc. Int. Joint Conf. on Neural Networks*, vol. 4, pp. 432–436, 1992.

[10] J.M. Rehg and T. Kanade, "Visual tracking of high DOF articulated structures: an application to human hand tracking," in *Computer Vision – ECCV'94*, J.-O. Eklundh, Ed., Berlin Heidelberg, 1994, pp. 35–46, Springer Verlag, Lecture Notes in Computer Science 801.

[11] J. Lee and T.L. Kunii, "Model-based analysis of hand posture," *IEEE Computer Graphics and Applications*, vol. 15, no. 5, pp. 77–86, 1995.

[12] R.J. Millar and G.F. Crawford, "A mathematical model for hand-shape analysis," in *Progress in Gestural Interaction - Proceedings of Gesture Workshop'96*, P.A. Harling and A.D.N. Edwards, Eds. 1996, pp. 235–245, Springer.

[13] A.J. Heap and D.C. Hogg, "3D deformable hand models," in *Proc. Gesture Workshop*, Mar 1996.

[14] N. Shimada and Y. Shirai, "3-D hand pose estimation and shape model refinement from a monocular image sequence," in *Proc. of VSMM'96 in GIFU*, 1996, pp. 423–428.

[15] N. Shimada, Y. Shirai, Y. Kuno, and I. Miura, "Hand gesture estimation and model refinement using monocular camera – ambiguity limitation by inequality constraints," in *Proc. of the 3rd Int. Conf. on Automatic Face and Gesture Recognition*, 1998, pp. 268–273.

[16] H. Ouhaddi and P. Horain, "3D hand gesture tracking by model registration," in *Proc. Int. Workshop on Synthetic-Natural Hybrid Coding and Three Dimensional Imaging*, Sep 1999, pp. 74–78, http://www.sim.int-evry.fr/∼hand/.

[17] J. P. Jones and L. A. Palmer, "An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex," *J. Neurophysiology*, vol. 58, no. 6, pp. 1233–1258, Dec 1987.

[18] H. Ritter, "Learning with the self-organizing map," in *Artificial Neural Networks*, T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas, Eds., pp. 379–384. Elsevier Science Publishers B. V., 1991.

[19] T. Kohonen, *Self-organizing maps*, Springer-Verlag, Heidelberg, 1997.

[20] J. Martin and J.L. Crowley, "An appearance-based approach to gesture-recognition," in *9th Int. Conf. on Image Analysis and Processing*, A.D. Bimbo, Ed., Florence, Italy, September 1997, Springer-Verlag, Lecture Notes in Computer Science, Nr 1311.

[21] I.A. Kapandji, *The physiology of the joints: Upper limbs*, Churchill Livingstone, 1982.

[22] H. Ritter, "Parametrized self-organizing maps," *Artificial Neural Networks*, vol. 3, pp. 568–577, 1993.

[23] H. Ritter and T. Martinetz and K. Schulten, "Neural Computation and Self-Organizing Maps", Addison Wesley, 1992.

[24] J. Walter, *Rapid learning in Robotics*, Cuvillier Verlag Göttingen, 1996.

[25] J. Walter and H. Ritter, "Rapid learning with parametrized self-organizing maps," *Neurocomputing*, vol. 12, pp. 131–153, 1996.

[26] T. Kohonen, "The self-organizing map," in *Proc. IEEE 78*, 1990, pp. 1464–1480.

[27] C. Nölker, GREFIT web-site, http://www.TechFak.Uni-Bielefeld.de/∼claudia/vishand.html.