Template-Based Hand Pose Recognition Using Multiple Cues

Björn Stenger

Toshiba Corporate R&D Center,
1, Komukai-Toshiba-cho, Saiwai-ku, Kawasaki 212-8582, Japan
bjorn@cantab.net

Abstract. This paper presents a practical method for hypothesizing hand locations and subsequently recognizing a discrete number of poses in image sequences. In a typical setting the user is gesturing in front of a single camera and interactively performing gesture input with one hand. The approach is to identify likely hand locations in the image based on discriminative features of colour and motion. A set of exemplar templates is stored in memory and a nearest neighbour classifier is then used for hypothesis verification and pose estimation. The performance of the method is demonstrated on a number of example sequences, including recognition of static hand gestures and a *navigation by pointing* application.

1 Introduction

Detecting and tracking hands is a problem in computer vision, which has been receiving significant attention. The applications in the domain of human computer interaction (HCI) are appealing: Systems have been designed for sign language recognition [1, 2, 3], navigation and control by hand motion [4, 5, 6, 7], and capturing detailed finger articulation [8, 9, 10, 11, 12]. For a given recognition task, it is important to look at the assumptions made in each system in order to determine whether or not the method will work. Two common assumptions are that foreground segmentation is reliable and that object segmentation (the hand from the rest of the body) is feasible. Relaxing these assumptions makes the problem significantly harder, particularly when using a single CCD camera.

In this paper we consider the problem of hand tracking in an HCI application. The system first hypothesizes a number of hand locations in the image, yielding estimates of location x, y and scale s of the hand. This information can then be used in subsequent steps. The image subwindow is normalized and classified as background or one of several pre-defined hand poses. The suggested method requires little computational power thus allowing for interactive gesture recognition. The following section briefly reviews some recent work in which the problem of hand tracking and pose recognition has been addressed.

1.1 Related Work on Hand Tracking and Pose Recognition

A vision-based drawing system, working in real-time, was presented by MacCormick and Isard [6]. The 2D shape of a pointing hand parallel to the image plane was tracked using a top-down camera. The system used a particle filter together with colour segmentation and background subtraction. Recognition systems that work in more general settings typically discriminate between few poses only. For example, Triesch and von der Malsburg [7] used Gabor-jets as features in a classification task to discriminate between ten hand poses in visually complex scenes. For each hand pose an elastic graph was built, in which each node contained a Gabor-jet. These graphs were then matched to each frame independently in a coarse-to-fine fashion. Bretzner *et al.* [4] took a scale-space approach to locate a hand parallel to the image plane and to distinguish between five different poses. Wu and Huang [13] recognized a number of discrete poses using a learning approach, where a labelled data set is combined with a large unlabeled data set. Kölsch and Turk [5] combined tracking with detection in a real-time interface for a wearable camera system: The global hand position was estimated by tracking a large number of skin-coloured corner features. The detection system was able to detect a particular pose at a fixed orientation using cascaded classifiers. Such classifiers were also used in the sign language recognition system presented by Kadir *et al.* [1]. However, the background in the sequences shown was relatively simple and the emphasis was placed on reliable high-level gesture recognition. Finally, a number of methods have been suggested for the task of 3D hand pose estimation from a single view: A 3D geometric model is used to generate a large number of 2D appearances, which are stored in a database [8, 10, 11]. This makes the collection of a large training data set unnecessary. In many applications, however, it is not necessary to capture the pose at each frame, but only to detect certain types of poses.

2 Subwindow Localization

This section proposes an efficient method for hypothesizing a number of locations and scale (x, y, s) of hands based on colour and motion features with no prior knowledge. This is done by searching for maxima in scale-space of the different feature likelihoods. The principle is similar to multi-scale blob detectors that look for scale-space extrema, e.g. using a Laplacian or difference of Gaussian

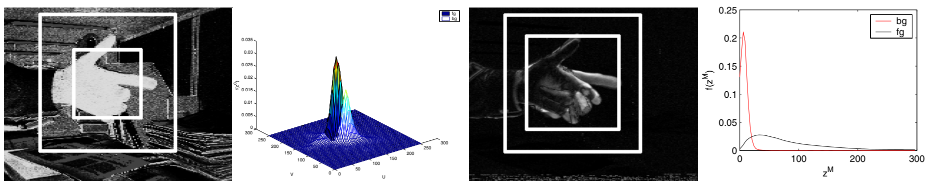

Fig. 1. Discriminative features for localization. (left) Colour and **(right)** motion features are used to hypothesize hand regions. The feature distributions of foreground and background regions are used to compute likelihoods for subregions in an image, which can be done efficiently using cumulative likelihood maps.

Table 1. Run-time comparison. This table shows the run-times to generate location hypotheses by finding maxima in scale space of a likelihood map, **(top)** using DoG filters by computing a Gaussian pyramid, and **(bottom)** using box filters on the original image (measured on a 3 GHz Pentium IV). The resulting top hypotheses were the same for both methods on the test sequences. The overhead for computing the Gaussian pyramid is high, whereas the cost for the box filter grows linearly with the number of scales. In the recognition experiments 10 scales are used.

Number of scales	3	6	9	12	15
DoG filter [14]	80 ms	99 ms	106 ms	108 ms	111 ms
Box filter	12 ms	24 ms	36 ms	48 ms	61 ms

filter [4, 14]. However, instead of working on an image pyramid we use an efficient box filter. At each image location square subwindows of different sizes are placed, and each subwindow is divided into centre foreground and surrounding background region [15], see figure 1. The relative size of the regions is fixed and is 1/8 for the case of colour features and 1/4 for the case of motion features (the inside region is larger since observable motion is often around the object silhouette). The likelihood for each subwindow in eqn (1) can be computed efficiently using integral images of the likelihood maps [15, 16]. Table 1 shows a run-time comparison of generating hypotheses by computing scale-space extrema using box filters and difference of Gaussian filters [14].

A likelihood model is defined, which explains the observation in each subwindow based on a foreground and a background feature distribution. If a hand is located within the centre region of such a subwindow, surrounded by background, the correct feature distribution will be chosen for most pixels, leading to a high subwindow likelihood. The state variable $\mathbf{x}$ is given by the parameters that define the location and scale of a chosen subwindow: $\mathbf{x} = [x, y, s]$. The observation vector $\mathbf{z}$ is described in terms of image features. At each time instant, a foreground feature distribution p^{fg} and a background distribution p^{bg} are given. Let $\mathbf{z}(\mathbf{u})$ be the observation at image location $\mathbf{u} \in \mathbb{R}^2$, and $fg(\mathbf{x})$ and $bg(\mathbf{x})$ the foreground and background image regions, respectively, given by the subwindow with parameters $\mathbf{x}$. We write the log-likelihood for a subwindow as

$$\log p(\mathbf{z}|\mathbf{x}) = \sum_{\mathbf{u} \in fg(\mathbf{x})} \log p^{fg}(\mathbf{z}(\mathbf{u})|\mathbf{x}) + \sum_{\mathbf{u} \in bg(\mathbf{x})} \log p^{bg}(\mathbf{z}(\mathbf{u})|\mathbf{x}). \tag{1}$$

The features are colour features $\mathbf{z}^C$ and motion features $\mathbf{z}^M$, described in detail in the following sections. Colour and motion features often complement each other in practice. For example, when a hand moves in front of a static skin coloured background, such as the face, motion features can still be discriminative. On the other hand, if the scene contains moving background objects, the colour features are more discriminative. Thus we treat the the features independently at this stage and for both colour and motion features separately sort the subwindows according to their likelihood values $\log p(\mathbf{z}|\mathbf{x})$ normalized by the subwindow size. Local maxima are extracted, while applying non-maximal suppression in order

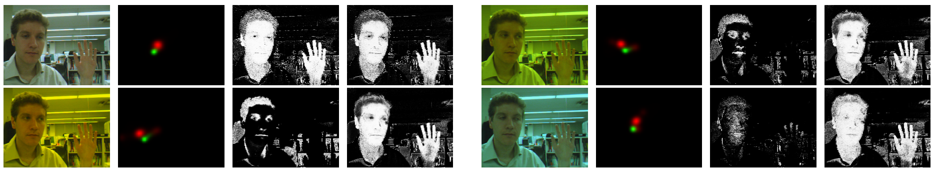

Fig. 2. Colour adaptation based on face detection. Each of the four examples shows the input frame from a sequence with varying colour balance, the smoothed skin colour (green) and background distributions (red) in UV-colour space, probabilities for a static colour model model in normalized UV-space, probabilities for the adaptive model based on face detection.

to obtain a better spatial distribution of the extracted regions. We take the k local maxima of each likelihood map as hypothetical object regions ($k = 3$ in the experiments).

2.1 Colour Model

Skin colour is a useful cue for hand tracking, however one major practical issue is to maintain a robust model under varying conditions: changing illumination, different skin tone of different people and varying background scenes. A popular approach is to work in an intensity-normalized colour space. However, this alone is often not sufficient and a method that is able to initialize and adapt the colour model is required [17, 18, 5, 15]. In this paper the skin colour model is obtained from a frontal face detector, which is run at every kth frame ($k = 30$) and which does not rely on colour information [19]. The assumption is that the skin colour of face and hands have similar distributions, allowing the system to adapt to a specific user as well as the current lighting conditions, even in extreme cases (see figure 2). Colour is represented using a 2D histogram of size 64^2 for vectors $\mathbf{z}^C = [U, V]$, containing the chromatic components of the YUV representation. When a face is detected, the colour values within an ellipse centered and scaled according to a detected location give the distribution $p(\mathbf{z}^C|x^C = \text{skin})$, estimated by histogramming and smoothing the distribution. The background distribution is estimated as a mixture of a uniform distribution and a colour distribution obtained from image regions adjacent to (left, right, and above) the detected face locations.

2.2 Motion Model

Motion is another valuable cue that has been extensively used in HCI applications [18] as it is robust under a wide range of conditions. The motion feature $\mathbf{z}^M$ is computed from the difference image as the L^1-norm of the pixel-wise RGB difference vectors at times t and $t-1$:

$$\mathbf{z}^M = |R^t - R^{t-1}| + |G^t - G^{t-1}| + |B^t - B^{t-1}| \ . \quad (2)$$

We differentiate between static background and moving objects, not between different foreground objects at this point. The distributions $p(\mathbf{z}^M|x^M = \text{motion})$

and $p(\mathbf{z}^M|x^M = \text{static})$ depend on a number of factors, such as the background scene, the illumination and the motion velocity. We obtain estimates from example sequences of difference images: the distribution for static background is computed from sequences with no moving objects and represents accounts for noise and some lighting variation. The distribution for moving objects is estimated from sequences of a moving hand. The distributions (see figure 1) were found to be reasonably stable across sequences with different backgrounds.

3 Hand Pose Estimation

Given a hypothesized image subregion, the aim is to determine whether the region contains a hand and if so, which pose it is in. The regions are normalized to a size of 40 × 40 pixels and are classified using nearest neighbour classification. We propose two distance measures based on oriented intensity edges and skin colour likelihood, respectively. The next section introduces a distance measure, which is based on oriented edges and avoids the thresholding step of a binary edge detector.

3.1 Oriented Edge Distance

Given a hand image in a normalized window, we compute a descriptor using oriented edge energy [20]. This is based on the response to a pair of even and odd filters, f^e_θ and f^o_θ, respectively, at orientation θ:

$$u^{OE}_\theta = (f^e_\theta * I)^2 + (f^o_\theta * I)^2. \tag{3}$$

Figure 3 shows the filter responses for four example images, showing the advantage of these features in complex scenes over using binary edge maps obtained

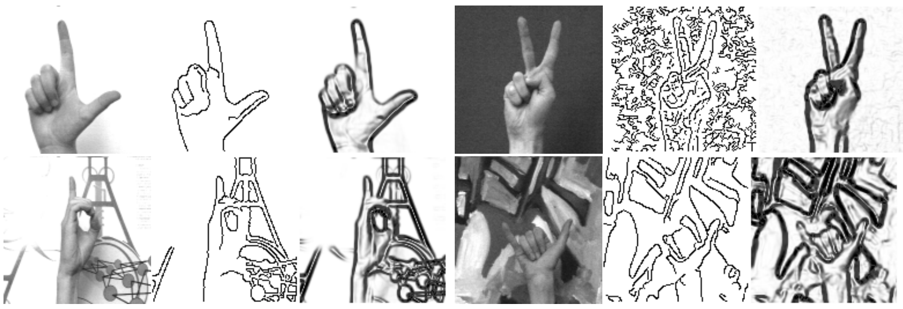

Fig. 3. Oriented edge energy vs. Canny edges. This figure shows the results of the filtering operations on example images from the database of hand images from Triesch and von der Malsburg [7]. For each image triple: **left:** input images, **middle:** Canny edges (constant parameter settings), **right:** oriented edge energy (pixel-wise maximum of responses to four filters with different orientation). The Canny detector often outputs spurious background edges or leads to premature loss of edge information.

with a standard Canny detector, which are widely used to extract geometric information [8, 11]. The value u_θ^{OE} is computed at four discrete orientations for each pixel location, $\theta = \{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\}$, yielding vectors $\{\mathbf{u}_i\}_{i=1,\ldots,4}$, each of which is then normalized and stacked into one vector $\mathbf{u}^{OE}$ [21]. The distance measure between two oriented edge maps $\mathbf{u}^{OE}$ and $\mathbf{v}^{OE}$ is defined as

$$d^{OE}(\mathbf{u}^{OE}, \mathbf{v}^{OE}) = 1 - \frac{1}{4}\sum_{i=1}^{4}\langle \mathbf{u}_i^{OE}, \mathbf{v}_i^{OE}\rangle. \tag{4}$$

The values in each feature vector $\mathbf{u}_i^{OE}$ are then multiplied with the pixel-wise skin colour probability.

3.2 Colour Based Distance

In order to compare templates using colour, the oriented filters are applied to the colour likelihood map. These vectors are also normalized and stacked into one feature vector $\mathbf{u}^C$. The distance measure is defined similar to the previous section as

$$d^{C}(\mathbf{u}^{C}, \mathbf{v}^{C}) = 1 - \frac{1}{4}\sum_{i=1}^{4}\langle \mathbf{u}_i^{C}, \mathbf{v}_i^{C}\rangle. \tag{5}$$

By using the filter responses directly in the distance function, binary thresholding of edges or skin colour at an early stage is avoided.

3.3 Local Template Registration

In order to compute a more exact distance, a continuous image alignment transformation $\mathbf{T}_\alpha$ with parameters α is computed for the best 50 matches. This is necessary to reduce jittered motion, but also to better discriminate between similar poses. A similarity transform is used for this purpose, represented by the 3×3 homogeneous matrix

$$\mathbf{T}_\alpha = \begin{bmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix}, \tag{6}$$

where $\alpha = \{s, \mathbf{R}, \mathbf{t}\}$, $s \in \mathbb{R}$ is a scale factor, $\mathbf{R} \in SO_3$ a rotation matrix and $\mathbf{t} \in \mathbb{R}^{2\times 1}$ a translation vector. Similarity transforms are chosen over affine transformations, because shapes under affine transform can look indistinguishable from the hand shape in a different view. The transformation parameters are found by searching over a grid of size 3^4 within the four dimensional parameter space.

3.4 Template Likelihoods

Based on the distances d^{OE} and d^C, a likelihood function is defined for a hand being in a particular pose based on matching to a set of exemplar templates $\{\mathbf{y}_j\}_{j=1,\ldots,N_j}$. The exemplars are then clustered such that each of the original examples is within a certain distance of at least one of the cluster centres

$\{\hat{\mathbf{y}}_k\}_{k=1,\ldots,N_k}$ for both features, edges as well as colour. The feature likelihood of a hand pose is estimated as [22]

$$p(\mathbf{z}|\alpha, k) = \frac{1}{2\pi\sigma^2} \exp -d(\mathbf{z}, \mathbf{T}_\alpha \hat{\mathbf{y}}_k)/2\sigma^2 \tag{7}$$

where $\mathbf{z}$ is the current image observation and $\mathbf{T}_\alpha \mathbf{y}_k$ is the exemplar template $\hat{\mathbf{y}_k}$, transformed by $\mathbf{T}_\alpha$. This term is computed for both oriented edge and colour features, where the parameters are estimated off-line using a set of 500 registered hand images containing variation in illumination and skin coloured background. A pose is detected when the distance value for either oriented edges or colour is below a threshold value, chosen in the experiments as 2σ.

4 Experimental Results

This section shows experimental results the localization task and on two different applications using pose estimation. The first application is a recognition task of ten different hand poses. The second task is a *navigation by pointing* application, where the user can indicate a direction by pointing at the camera. A system overview is given in figure 4. Both experiments are carried out using 320×240 images from a single camera directed at the user in an office environment. The skin colour model is initialized using frontal face detection (not shown in the figures). The likelihoods for subwindows is computed at 10 scales.

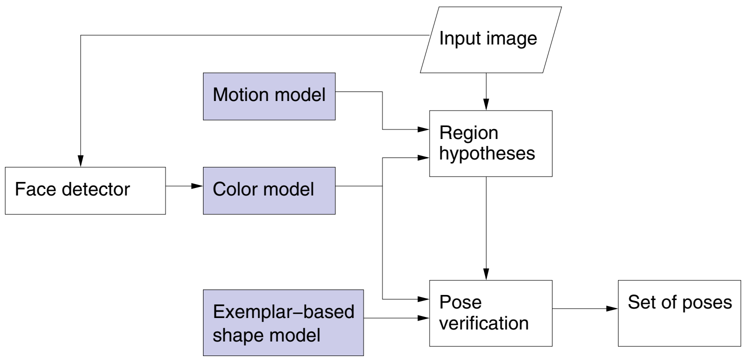

Fig. 4. System overview. In each frame motion and colour features are used to hypothesize image subregions, which are subsequently verified using an exemplar based shape model that includes oriented edge and colour features.

4.1 Pose Recognition

The top of figure 5 shows the ten different poses to be recognized. They are the same poses that have been used in [7]. First, a number of templates are obtained from a sequence taken in front of neutral background. For the recognition stage

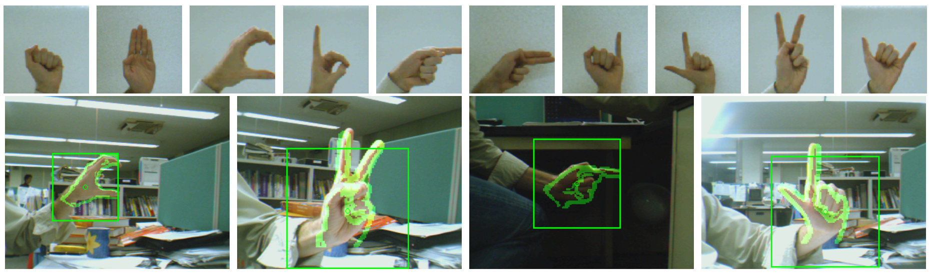

Fig. 5. Hand posture recognition. top: the ten hand postures to be recognized, recorded against a neutral background, **below:** example frames from the sequence with recognition results superimposed. In this sequence the camera is moved for a number of times and there is large variation in lighting conditions.

Table 2. Recognition results for (left) static gesture recognition. This table shows the correctly classified poses on three image sequences (3000 frames each). Each column shows the results of different combinations of localization features (colour/motion) and template features (colour/edges). Different rows show the classification result if the same pose has been classified consistently over a certain number of consecutive frames. **(right) Results for the navigation application.** This table shows the correctly classified poses on four image sequences (3000 frames each) taken of different users.

Consec	col win	mot win	col win	mot win
frames	col tmpl	col tmpl	edge tmpl	edge tmpl
1	0.85	0.33	0.67	0.29
2	0.91	0.50	0.78	0.42
3	0.94	0.63	0.84	0.51
4	0.95	0.73	0.87	0.59
5	0.96	0.79	0.88	0.66

Consec	col win	mot win	col win	mot win
frames	col tmpl	col tmpl	edge tmpl	edge tmpl
1	0.73	0.61	0.57	0.35
2	0.81	0.65	0.69	0.42
3	0.84	0.73	0.74	0.49
4	0.86	0.79	0.80	0.53
5	0.88	0.83	0.84	0.54

40 templates for each pose are used, encoding variation in pose. To increase tolerance to small rotations, for each template a further four templates are generated by rotation around the image centre (-10 to 10 degrees). Three sequences of 3000 frames each were recorded in which a user performed all ten gestures five times. The bottom row of figure 5 shows example frames, demonstrating the variation in scale and lighting conditions. Projected onto each frame is the pose estimate, represented by the closest exemplar template. Table 2 (left) shows the average classification rates in the image sequences according to individual features. Temporal continuity is added by requiring detection of the same pose over a number of consecutive frames. Most errors occur due to confusion of two similar poses (pointing with one or two fingers, respectively). Another error source are highlights on the hand, where pixels have low skin colour probability. Using 1000 templates, the system currently runs at 6 fps on a 3 GHz Pentium IV.

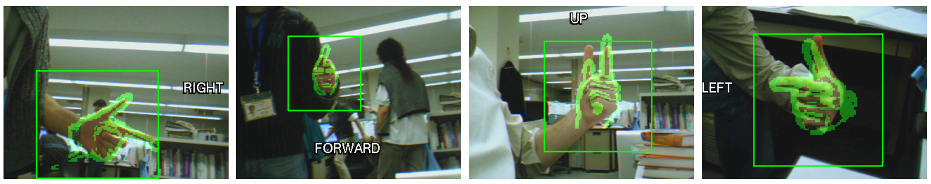

Fig. 6. Navigation by pointing at the camera. This figure shows frames from a test sequence. The overlays represent the estimated pose. Each template is labeled as one of the classes *forward, left, right, up, down, stop.* The examples are from four sequences where three different users perform the gestures.

4.2 Navigation by Pointing

The second application we consider is a navigation by pointing application. In this experiment 300 templates are used for each pose, covering the range of motion of a hand pointing toward the camera. No rotated templates are generated in this case as the orientation is critical for determining the pointing direction. The templates are manually labeled as one of six classes, *forward, left, right, upward, downward, stop.* Four sequences of 3000 frames each were recorded from three different users. Example frames are shown in figure 6. The pose estimate is superimposed, showing that the poses are indeed recognized fairly accurately. Table 2 (right) shows the results on the example sequences. The difficult case here is when the hand is pointing upward, which looks similar to when it points forward with the thumb extended. The system currently runs at 5 fps.

5 Summary and Conclusions

This paper presents a solution to hand pose recognition in cluttered scenes. Initial location hypotheses are obtained using colour and motion, which are verified using normalized template matching with a pre-selected number of templates. The method has been applied to a recognition task with ten poses as well as a navigation by pointing application. A number of techniques have been combined in this system: The colour model is initialized and updated by a frontal face detector. Hand locations and scale are hypothesized efficiently using cumulative likelihood maps, and the hand pose is estimated by normalized template matching. The system lifts several restrictions which are often present in real-time systems: In contrast to other methods, the system in this paper uses neither background subtraction [6] nor does it rely on binary colour segmentation [17, 2, 3] or gesturing at a fixed distance to the camera [23]. Finally, the method is efficient enough to detect the hand in each frame independently. Future extensions of the system include a tracking element that helps to improve the efficiency and accuracy of the pose estimation, as well as an increase in the number of poses.

References

1. Kadir, T., Bowden, R., Ong, E.J., Zisserman, A.: Minimal training, large lexicon, unconstrained sign language recognition. In: Proc. BMVC. (2004)
2. Lockton, R., Fitzgibbon, A.W.: Real-time gesture recognition using deterministic boosting. In: Proc. BMVC. Volume II. (2002) 817–826
3. Tomasi, C., Petrov, S., Sastry, A.K.: 3D tracking = classification + interpolation. In: Proc. 9th ICCV. Volume II. (2003) 1441–1448
4. Bretzner, L., Laptev, I., Lindeberg, T.: Hand gesture recognition using multi-scale colour features, hierarchical models and particle filtering. In: Proc. Face and Gesture. (2002) 423–428
5. Kölsch, M., Turk, M.: Fast 2D hand tracking with flocks of features and multi-cue integration. In: Workshop on Real-Time Vision for HCI. (2004)
6. MacCormick, J., Isard, M.: Partitioned sampling, articulated objects, and interface-quality hand tracking. In: Proc. 6th ECCV. Volume 2. (2000) 3–19
7. Triesch, J., von der Malsburg, C.: Classification of hand postures against complex backgrounds using elastic graph matching. IVC **20** (2002) 937–943
8. Athitsos, V., Sclaroff, S.: Estimating 3D hand pose from a cluttered image. In: Proc. CVPR. Volume II. (2003) 432–439
9. Rehg, J.M., Kanade, T.: Model-based tracking of self-occluding articulated objects. In: Proc. 5th ICCV. (1995) 612–617
10. Shimada, N., Kimura, K., Shirai, Y.: Real-time 3-D hand posture estimation based on 2-D appearance retrieval using monocular camera. In: Proc. Int. WS. RATFG-RTS. (2001) 23–30
11. Stenger, B., Thayananthan, A., Torr, P.H.S., Cipolla, R.: Filtering using a tree-based estimator. In: Proc. 9th ICCV. Volume II. (2003) 1063–1070
12. Wu, Y., Lin, J.Y., Huang, T.S.: Capturing natural hand articulation. In: Proc. 8th ICCV. Volume II. (2001) 426–432
13. Wu, Y., Huang, T.S.: View-independent recognition of hand postures. In: Proc. CVPR. Volume II. (2000) 88–94
14. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
15. Micilotta, A., Bowden, R.: View-based location and tracking of body parts for visual interaction. In: Proc. BMVC. (2004) 849–858
16. Movellan, J.R., Hershey, J., Susskind, J.: Real-time video tracking using convolution HMMs. In: Workshop on Generative Models for Vision. (2004)
17. Argyros, A.A., Lourakis, M.I.A.: Real-time tracking of multiple skin-colored objects with a possibly moving camera. In: ECCV. (2004) 368–379
18. Crowley, J.L., Berard, F.: Multi-modal tracking of faces for video communications. In: Proc. CVPR. (1997) 640–645
19. Viola, P., Jones, M.J.: Robust real-time face detection. IJCV **57** (2004) 137–154
20. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. PAMI **26** (2004) 530–549
21. Everingham, M.R., Zisserman, A.: Automated person identification in video. In: 3rd Int. Conf. on Image and Video Retrieval. (2004) 289–298
22. Toyama, K., Blake, A.: Probabilistic tracking with exemplars in a metric space. IJCV **48** (2002) 9–19
23. Von Hardenberg, C., Bérard, F.: Bare-hand human-computer interaction. In: Proc. ACM Workshop on Perceptive User Interfaces. (2001)