Boston University Computer Science Tech. Report No. 2000-22,Dec. 2000 (revised Apr. 2001).
To Appear in Proc. **IEEE International Conf. on Computer Vision (ICCV).** Canada. Jul. 2001.

3D Hand Pose Reconstruction Using Specialized Mappings

Rómer Rosales, Vassilis Athitsos, Leonid Sigal, and Stan Sclaroff*
Boston University, Computer Science Department
111 Cummington St., Boston, MA 02215
email:{rrosales,athitsos,lsigal,sclaroff}@bu.edu

Abstract

A system for recovering 3D hand pose from monocular color sequences is proposed. The system employs a non-linear supervised learning framework, the specialized mappings architecture (SMA), to map image features to likely 3D hand poses. The SMA's fundamental components are a set of specialized forward mapping functions, and a single feedback matching function. The forward functions are estimated directly from training data, which in our case are examples of hand joint configurations and their corresponding visual features. The joint angle data in the training set is obtained via a CyberGlove, a glove with 22 sensors that monitor the angular motions of the palm and fingers. In training, the visual features are generated using a computer graphics module that renders the hand from arbitrary viewpoints given the 22 joint angles. The viewpoint is encoded by two real values, therefore 24 real values represent a hand pose. We test our system both on synthetic sequences and on sequences taken with a color camera. The system automatically detects and tracks both hands of the user, calculates the appropriate features, and estimates the 3D hand joint angles and viewpoint from those features. Results are encouraging given the complexity of the task.

1 Introduction

The estimation of hand pose from visual cues is a key problem in the development of intuitive, non-intrusive human-computer interfaces. The shape and motion of the hand during a gesture can be used to recognize the gesture and classify it as a member of a predefined class. The importance of hand pose estimation is evident in other areas as well; *e.g.,*video coding, video indexing/retrieval, sign language understanding, computer-aided motion analysis for ergonomics, etc.

In this paper, we address the problem of recovering 3D hand pose from a monocular color sequence. Our solution to this problem makes use of concepts from stochastic visual segmentation, computer graphics, and non-linear supervised learning. Our contribution is an automatic system that tracks the hand and estimates its 3D configuration on every frame, that does not impose any restrictions on the hand shape, does not require manual initialization, and can easily recover from estimation errors.

2 Related Work

Several existing systems include automated hand detection and tracking. Such systems typically make restrictive assumptions on the domain: only hands move, the hands are the fastest moving objects in the scene [17, 38, 40, 3, 21, 23], hands are skin colored, or they are the only skin-colored objects in the scene [21, 34]. Often the background is assumed to be static, and known [21, 8]. Some systems use such assumptions to obtain several possible regions where the hands are, and use matching with appearance-based models to choose among those regions [38, 9]. Stochastic tools, such as Kalman filtering [34, 38, 40], can be used to predict the hand position in a future frame. Overall, hand detection and tracking algorithms tend to perform well in restricted environments, where assumptions about the number, location, appearance and motion of hands are valid, and the background is known. Reliable performance in more general domains, is still beyond the current state of the art.

Previous systems' representation of hand pose varies widely. For certain applications, hand trajectories can be sufficient for gesture classification [3, 23]. However, in some domains, knowledge of more detailed hand configuration must be used to disambiguate between different gestures; *e.g.,*in signed languages. Pose can be estimated in 2D or 3D. Most 2D-based approaches try to match the image of the hand with view-based models corresponding to a limited number of predefined hand poses [9, 4, 38, 11, 17, 35, 34]. In [20] the condensation algorithm is used to track the index and thumb of a hand. Such methods are valid in restricted domains, in which users are observed from a known viewpoint, performing a limited variety of motions.

One limitation in view-based methods is that pose recog-

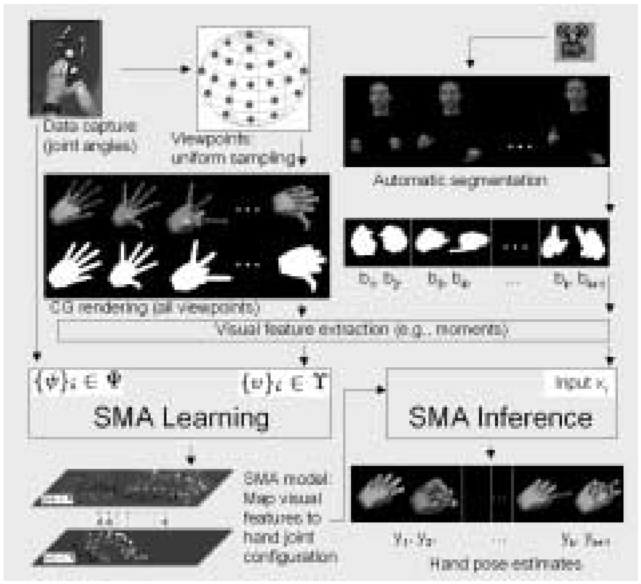


Figure 1: Hand pose estimation overview.

*This work was supported in part through Office of Naval Research Young Investigator Award N00014-96-1-0661, and National Science Foundation grants IIS-9912573 and EIA-9809340.

nition is not viewpoint invariant. Images of the same 3D hand shape from different viewpoints, or even rotated examples of the same image would be considered different poses. Some of those limits have been addressed by using multiple cameras [35], and stereo [8]; naturally, such methods will not work in monocular sequences. Our approach will avoid this limitation through the use of probabilistic modeling, Specialized Mappings (SMA), to map image features to likely 3D hand poses.

A related approach to SMA is described in [39], where a system is trained with views corresponding to many different hand orientations and viewpoints. Some training views are labeled with the 3D pose category they correspond to, but most of them are unlabeled. The categories of the unlabeled data are treated as missing values in a D-EM (Discriminant Expectation-Maximization) framework. The system can recognize 14 hand configurations, observed from a variety of viewpoints. A difference between that approach and ours is that, in their system, the configuration estimation is formulated as a classification problem, in which a finite number of classes are defined. Our SMA approach is based on regression rather than classification, allowing for theoretically continuous solutions of the estimation problem.

Sometimes, such continuous solutions are preferable to simply recognizing a limited number of classes. For example, in a virtual reality application, we may want to accurately reconstruct the hand of the user in the virtual environment and estimate the effects of that particular configuration on the environment. Even in cases where the ultimate goal is classification, accurate 3D information can improve recognition by making it robust to viewpoint variations. An important decision in estimating 3D pose is the representation and parameterization. Link-and-joint models are used by [25, 31], whereas a mesh model is used by [10]. In those three systems, the hand configuration at the beginning of a sequence must be known *a priori*. In addition, self-occlusions and fast motions make it hard to maintain accuracy while tracking. Our proposed SMA approach avoids these drawbacks.

SMA is related to machine learning models [16, 12, 6, 28] that use the principle of divide-and-conquer to reduce the complexity of the learning problem by splitting it into several simpler ones. In general these algorithms try to fit surfaces to the observed data by (1) splitting the input space into several regions, and (2) approximating simpler functions to fit the input-output relationship inside these regions. The splitting process may create a new problem: how to optimally partition the problem such that we obtain several sub-problems that can be solved using the specific solver capabilities (*i.e.*,form of mapping functions). In SMA's, we address this problem by solving for the partitions and the mappings simultaneously.

In the work of [6], *hard* splits of the data were used, *i.e.*,the parameters in one region only depend on the data falling in that region. In [16], some of the drawbacks of the hard-split approach were pointed out (*e.g.*,increase in the variance of the estimator), and an architecture that uses *soft* splits of the data, the Hierarchical Mixture of Experts, was described. In this architecture, as in [12], at each level of the tree, a gating network is used to control the influence (weight) of the expert units (mapping functions) to model the data. However, in [12] arbitrary subsets of the experts units can be chosen. Unlike these architectures, in SMA's the mapping selection is done using a feedback matching process, currently in a winner-take-all fashion, but *soft* splitting is done during training. In applications where a feedback map can be computed easily and accurately, this is an important advantage. Also, the shape of the regions that determine ownership to given specialized functions is general; therefore, we do not assume any fixed functional form or discriminant function to define these regions (gating networks).

With respect to work on learning based approaches for estimating articulated body pose, Point Distribution Models have been applied to recovering upper-body pose from silhouettes or skin-colored blobs [1, 24]. In [13], a Gaussian probability model for short human motion sequences was built. However, this method assumes that 2D tracking of joints in the image is given. In [2], the manifold of human body configurations was modeled via a hidden Markov model and learned via entropy minimization. In [33] dynamic programming is used to calculate the best global labeling of the joint probability density function of the position and velocity of body features; it was also assumed that it is possible to track these features for pairs of frames. These last three approaches model the dynamics of motion, a problem that in general requires much more training data to build a reasonable approximation to the underlying probability distribution.

3 Overview

An overview of our approach can be seen in Fig. 1. First is first trained, given a number J of example hand joint configurations are acquired using a CyberGlove (at approx. 15 Hz). The CyberGlove measures 22 angular DOF of the hand. Computer graphics software can be used to render a shaded view of any hand configuration captured by the CyberGlove. Using this computer graphics rendering function, we can generate a uniform sampling (with size S) on the whole view sphere, and render views (images) of every hand configuration from all sampled viewpoints. We can then use image processing to extract visual feature υ_i vector from each of the images generated; in our case we extract moment based-features, but other features are possible [13]. This process yields a set $\{\psi_i\} = \Psi$, where ψ_i is each of the hand joint configurations from each viewpoint [1], and $\{\upsilon_i\} = \Upsilon$, where υ_i is a vector of visual features corresponding to each ψ_i.

These sets Ψ and Υ constitute samples from the input-output relationship that we will attempt to learn using our architecture. Given a new image of a hand, we will compute its visual feature vector $\mathbf{x}$. We then compute the mapping from $\mathbf{x}$ to the most likely 24 DOF hand configuration. Note that this mapping is highly ambiguous. In fact the relationship is many to many; therefore no single function can perform this task. Using the Specialized Mapping Architecture (SMA), we split (partition) this mapping into many mappings. Each of these hopefully simpler problems is then solved using a different specialized function. The SMA learning scheme solves for partitions and mappings simultaneously.

The SMA tries to learn a multiple mapping so that, when performing inference, given a vector of visual features $\mathbf{x}$, an output in the output space of hand configurations can be

[1]This vector is then composed of 22 internal pose parameters plus two global orientation parameters.

provided. On the right column of Fig. 1, a diagram of the inference process is shown. First video input is obtained, and using a segmentation module, regions with high likelihood of being skin colored are found. From these regions we extract visual features (*e.g.,*moments). Then the given vector of visual features $\mathbf{x}$ is presented to SMA, which generates several output estimates, one of which is chosen using a defined cost function. Most of the details, including the processes of learning and inference by SMA are presented in the following sections.

Our approach can easily integrate different choices of features. Furthermore, the same approach can be used to estimate the pose of articulated objects other than hands.

4 Hand Shape Representation

The hand model that we use is implemented in the VirtualHand programming library [36]. The parameters of the model are 22 joint angles. For the index, middle, ring and pinky finger, there is an angle for each of the distal, proximal and metacarpophalangeal joints. For the thumb, there is an inner joint angle, an outer joint angle and two angles for the trapeziometacarpal joint. There are also abduction angles between the following pairs of successive fingers: index/middle, middle/ring and ring/pinky. Finally, there is an angle for the palm arch, an angle measuring wrist flexion and an angle measuring wrist bending towards the pinky finger.

The VirtualHand library provides tools that can render an artificial hand from an arbitrary viewpoint, given values for the 22 angles. Fig. 3 shows examples of hand renderings. Using a CyberGlove (manufactured by VirtualTechnologies) we collected about 2,400 examples of hand poses (parameterized as vectors containing the 22 angles). We rendered each pose from 86 different viewpoints. Those viewpoints formed an approximately uniformly distributed set on the surface of a sphere centered at the hand. The synthetic images obtained this way were used for training and testing as described in the experimental results. The VirtualHand library was also used to reconstruct the estimated 3D hand shape for testing data, based on the output of our system.

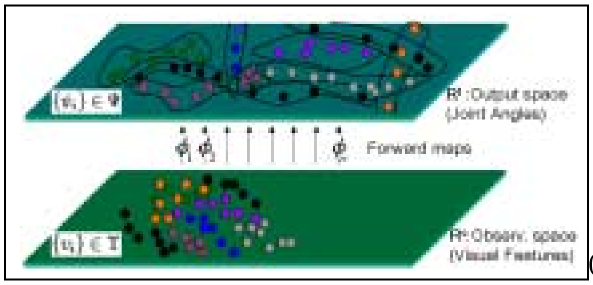
5 Learning Algorithm

The estimation paradigm used in this work consist of mapping the observed low-level visual features to hand joint configurations. The underlying approach for finding this mapping is based on the Specialized Mappings Architecture (SMA), a non-linear supervised learning architecture.

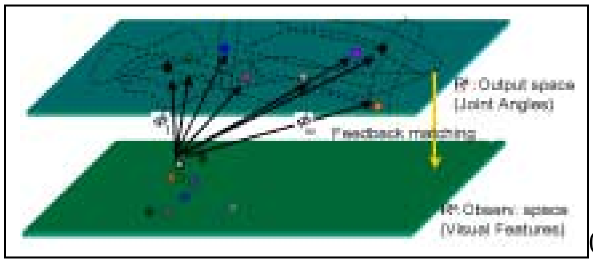
Given an input and output space $\Re^c$ and $\Re^t$ respectively, SMA's consist of several specialized *forward* mapping functions $\phi_k : \Re^c \rightarrow \Re^t$ and a *feedback matching* function $\zeta : \Re^t \rightarrow \Re^c$, which in this case is known (visual features can be obtained given the joint configurations by using computer graphics based rendering).

In order to estimate these mappings, we use a supervised learning approach with training data $\mathcal{Z} = \{\mathbf{z}_i\}_{i=1..n}$, with $\mathbf{z}_i = (\upsilon_i, \psi_i)$ an input-output pair (visual features and hand joint angles respectively).

Our architecture generates a series of m functions ϕ_k in which each of these functions is specialized to map certain inputs (their specialized domain) better than others. The specialized domain can be for example a region of the input space. However, this specialized domain of ϕ_k can be more general than just a connected region in the input space. We propose to determine these specialized domains and functions simultaneously.



(a)



(b)

Figure 2: SMA diagram illustrating (a) an estimated SMA model with m specialized functions mapping subsets of the training data (each subset is drawn with a different color) and (b) the inference process in which a given observation is mapped by all the specialized functions, and then a feedback matching step is performed to choose the best of the m estimates.

Fig. 2(a) illustrates the basic idea of this model. We use different colors (gray-levels) to represent the domain of each specialized function. At initialization random colors are assigned to each point, the goal is to find an optimal mapping and partition that is efficient in reducing some error function. Once the model has been learned our mapping may look like Fig. 2(a), in which each function ϕ_i is in charge of mapping certain inputs only.

5.1 Probabilistic Model

Let the training sets of output-input observations be $\mathbf{\Psi} = \{\psi_1, \psi_2, ..., \psi_n\}$, and $\mathbf{\Upsilon} = \{\upsilon_1, \upsilon_2, ..., \upsilon_n\}$ respectively. We will use $\mathbf{z}_i = (\psi_i, \upsilon_i)$ to define a given output-input training pair, $\mathcal{Z} = \{\mathbf{z}_1 ... \mathbf{z}_n\}$ represents our observed training set.

Define the unobserved random variables y_i with $i = \{1..n\}$ and $\mathbf{y} = (y_1, y_2, ..., y_n)$. In our model the variables y_i have domain the discrete set $\mathcal{C} = \{1..m\}$ of labels for the specialized functions, and can be thought as the function number used to map data point i, therefore m is the number of specialized functions in the model.

Define the model parameters $\theta = (\theta_1, \theta_2, ...\theta_m, \lambda)$, where θ_i represents the parameters of the mapping function i. The vector $\lambda = (\lambda_1, \lambda_2, ..., \lambda_m)$, where λ_k represent $P(y_i = k|\theta)$.

Using Bayes' rule and assuming independence among observations, we have the joint probability of the observed and hidden variables conditioned on our model parameters:

$$P(\mathcal{Z}, \mathbf{y}|\theta) = P(\mathcal{Z}|\mathbf{y}, \theta)P(\mathbf{y}|\theta) = \prod_i P(\mathbf{z}_i|y_i, \theta)P(y_i|\theta) \tag{1}$$

5.2 SMA Parameter Estimation and the EM Algorithm

The optimization problem defined by Eq. 1 is computationally very expensive. Here, the probabilistic parameter estimation problem is approached under the Expectation Maximization (EM) algorithm framework [5]. We use the notation followed by [22].

Note that Eq. 1 makes reference to a still undefined distribution $P(\mathbf{z}_i|y_i,\theta)$. Several options had been proposed [27]. Here we will use a Gaussian distribution with mean defined by the error incurred in using the possibly non-linear function ϕ_y as a mapping function, and a variance Σ_y:

$$P(\mathbf{z}|y,\theta) = \mathcal{N}(\psi; \phi_y(\upsilon,\theta), \Sigma_y) \tag{2}$$

Using this distribution, the E-step consists of finding $\tilde{P}(\mathbf{y}) = P(\mathbf{y}|\mathcal{Z},\theta)$. In our case, this factorizes as:

$$\begin{aligned} \tilde{P}(\mathbf{y}) &= \prod_i \frac{\lambda_{y_i} P(\mathbf{z}_i|y_i,\theta)}{\sum_{k\in\mathcal{C}} \lambda_k P(\mathbf{z}_i|y_i=k,\theta)} && (3) \\ &= \prod_i \frac{\lambda_{y_i} e^{(\psi_i - \phi_{y_i}(\upsilon_i,\theta_{y_i}))^\top \Sigma^{-1} (\psi_i - \phi_{y_i}(\upsilon_i,\theta_{y_i}))}}{\sum_{k\in\mathcal{C}} \lambda_k e^{(\psi_i - \phi_k(\upsilon_i,\theta_k))^\top \Sigma^{-1} (\psi_i - \phi_k(\upsilon_i,\theta_k))}} && (4) \end{aligned}$$

The M-step consists of finding $\theta^{(t)} = \arg\max_\theta E_{\tilde{P}^{(t)}}[\log P(\mathbf{y},\mathcal{Z}|\theta)]$. Using our model, it can be shown that:

$$\theta^{(t)} = \arg\max_\theta \sum_i \sum_{\mathbf{y}_i\in\mathcal{C}} \tilde{P}^{(t)}(y_i)[\log P(\mathbf{z}_i|y_i,\theta) + \log P(y_i|\theta)]. \tag{5}$$

This gives the following update rules for λ_k and Σ_k (where Lagrange multipliers were used to incorporate the constraint $\sum_k \lambda_k = 1$).

$$\begin{aligned} \lambda_k &= \frac{1}{n}\sum_i P(y_i = k|\mathbf{z}_i,\theta) && (6) \\ \Sigma_k &= \frac{\sum_i \tilde{P}^{(t)}(y_i=k)(\psi_i - \phi_k(\upsilon_i,\theta_k))(\psi_i - \phi_k(\upsilon_i,\theta_k))^\top}{\sum_i \tilde{P}^{(t)}(y_i=k)} && (7) \end{aligned}$$

The update for θ_k depends on the form of ϕ_k. Here we have chosen a non-linear function of the form:

$$\{\phi_k(\mathbf{x},\theta_k)\}_q = g_2(\sum_{j=0}^{l_2} w_{qj}^{(2)} g_1(\sum_{i=0}^{l_1} w_{ji}^{(1)} x_i)), \tag{8}$$

where x_i is the $i-th$ component of the visual feature vector, $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ are weights and biases (part of θ_k), g_1 and g_2 are a sigmoidal and linear function respectively, l_1 and l_2 are the number of nodes in each layer, and q is just the dimension index of the output vector. This is a 1-hidden layer feed-forward network.

Unfortunately, using this function (as it would be by using most non-linear functions) forces us to use iterative optimization for the M-step.

5.3 Stochastic Learning

The update equations described above are useful to find a local minimum given the initial values of the parameters. In order to improve this process, and avoid some of the local minima that inevitably arise, we use an annealing schedule on the $\tilde{P_i}^{(t)}$ probabilities during the M-step. In this way, we redefine:

$$\tilde{P_i}^{(t)}(y_i = j) = \frac{e^{\log(\tilde{P_i}^{(t)}(y_i=j))/T(t)}}{\sum_{k\in\mathcal{C}} e^{\log(\tilde{P_i}^{(t)}(y_i=k))/T(t)}} \tag{9}$$

In our experiments the temperature parameter T decays exponentially. This step not only does help in avoiding local minima, but it also creates two desirable effects. It forces $\tilde{P_i}^{(t)}(y_i = j)$ to be binary (either 1 or 0) at low temperatures, as a consequence each point will tend to be mapped by only one specialized function at the end of optimization. Moreover, it makes $\tilde{P_i}^{(t)}(y_i = k)$ $(k = 1..m)$ be fairly even at high temperatures, making the optimization less dependent on initialization.

Note that there is no closed form solution for the M-step as described above. In practice we have decided to perform two or three iterations per M-step. Another source of randomness added to the process so far described consists in choosing data points randomly uniformly distributed when performing the M-step. These two variants of the M-step have been justified in the sense of a partial M-step [22].

5.4 Feedback Matching

Once the model parameters have been estimated each specialized function maps (with different levels of accuracy) the whole input space. Therefore, the following question arises: during reconstruction, given a point in input space, how do we choose the mapping function ϕ_k that should be used to map this point?

Fig. 2(b) illustrates the inference process. When generating an estimate $\hat{\mathbf{h}}$ of body pose given an input $\mathbf{x}$ (the gray point with a dark contour in the lower plane), SMA's generate a series of output hypotheses $\mathcal{H} = \{\mathbf{h}\}_k$ obtained using $\mathbf{h}_k = \phi_k(\mathbf{x})$, with $k \in \mathcal{C}$ (illustrated by each of the points pointed by the arrows). Given the set $\mathcal{H}$, we define the most accurate hypothesis to be that one that minimizes the function $F(\zeta(\mathbf{h}_j),\mathbf{x},\mathcal{Z})$, over j, in this paper we use:

$$i = \arg\min_j (\zeta(\mathbf{h}_j) - \mathbf{x})^\top \Sigma_\Upsilon^{-1} (\zeta(\mathbf{h}_j) - \mathbf{x}), \tag{10}$$

and make $\hat{\mathbf{h}} = \mathbf{h}_i$, where Σ_Υ is the covariance matrix of the elements in the set Υ (*i.e.,*the input vectors in our training set) and i is the assigned label. In Fig. 2(b) we can see that each of the points in the output space is mapped back to the input space, once in this space, these points can be compared (using a given cost function *e.g.,*Eq. 10) to the initial input observation. The form of the cost function could vary, using Eq. 10 is the same as assumming that $P(\mathbf{h}|\mathbf{x}) = \mathcal{N}(\mathbf{h}; \mathbf{x}, \Sigma_\Upsilon)$.

6 Hand Detection and Segmentation

Some of our test data consists of video sequences collected with a color digital camera. In those sequences the background is static, there is only one person present, and the person is facing towards the camera. Our system tracks both hands of the user automatically, using a skin color tracker.

In the first frame of the sequence, the tracker needs to be initialized, by locating in the image the objects that we want to track. That could be done by applying a skin detector system, like the one described in [15]. However, using that detector, clothes are labeled as skin, sometimes, because of their color.

We can locate and segment the hands more accurately using the fact that their color is very similar to the color of the face. The position of the face can be found reliably using a face detector system [29]. For each pixel in the detected face we compute a measure of how skin-like the pixel color is. That measure is based on histograms of skin and non-skin color distributions, computed from a database of thousands of images in which regions were labeled as skin and non-skin. Those labeled images were frames from commercially available DVD movies.

We select the top 50% of the pixels in the face, for which the measure of skin similarity is the highest. For each of those pixels we compute its rg color ($r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B}$), and we find the mean rg color of all selected pixels. Then, for each pixel in the image, we calculate the distance of its rg color from the mean rg color. We label as skin all pixels for which that distance is less than a threshold. The threshold we use is 17, for RGB values between 0 and 255. The objects we want to track are the three largest connected components of the skin pixels. One of them overlaps with the face, and the other two are considered to be the hands. We initialize the skin tracker with the position of the face and hand regions, and the tracker locates the face and hands in the rest of the frames in each sequence.

The skin tracker models skin color distribution as a histogram in HSV space. It can handle distributions that change from one frame to the next, because of varying illumination or motion with respect to light sources. The changes in skin color that occur in a new frame are modeled as the results of translating, rotating and scaling the current histogram. Furthermore, the evolution of the histogram is modeled as a second-order Markov process. The tracker is initialized in the first frame, by being told which regions to track, and it estimates the initial color distribution. In the next 8-30 frames, in addition to tracking and adapting the skin color histogram, it also learns the parameters of the Markov process. After the learning stage, it uses those parameters to predict the color distribution in every new frame, while still updating the Markov model, based on the actual histogram that is observed in the new frame. The learning and tracking stage are explained in detail in [32].

Our simple hand detection and tracking algorithm would not work at any frame where the hands overlap with each other or with the face. In our video sequences we took care to avoid such situations. Our system could be made more general by including modules to predict occlusion of an object by another and to detect when those objects are separated again. A similar approach has been successfully applied in the domain of multiple person tracking with occlusion handling [26].

7 Experimental Results

The described approach was tested in experiments with training data consisting of approximately 30 sequences obtained through the use of a Cyberglove. Input-output pairs were generated using computer graphics by rendering from 86 viewpoints roughly uniformly distributed on the view sphere. The output consisted of 24 joint angles of a human hand linearly encoded by nine real values using Principal Component Analysis (PCA).

The input consisted of seven real-valued Hu moments [14] computed on synthetically generated silhouettes of the hand. Hu moments are functions of central image moments. They are invariant to translation, scaling, and rotation on the image plane. These invariances ease the observation process (*e.g.,*we do not need to be concern about where and how large the hand appears on the image). However, rotation invariance makes hand rotation parallel to the image plane unobservable. For the real experiments observation inputs were obtained tracking skin color distributions [32].

Approximately 300,000 images were generated synthetically. Of these, 8,000 were used for training and the rest for testing. We used cross-validation for early stopping the training procedure and avoid overfitting. In the experiments shown, the number of specialized functions was set to 30. Each of these functions was a one hidden layer, feed-forward network with 5 hidden neurons. The annealing schedule was $1/k$ where k was the iteration number in the EM algorithm. Other experiments were performed to test the convergence and fitting properties of the model, due to space limitations these results will not be presented in this paper.

7.1 Quantitative Experiments

Fig. 3 shows example hand configuration estimates obtained in representative test frames (not in the training set). Synthetic images were used in this experiment, because ground-truth data was available for quantitative performance evaluation. As can be seen in the figure, self-occluding configurations are obviously harder, but still the estimate is close to ground-truth given that no human intervention nor pose initialization was required.

In order to provide quantitative measures of performance, test data was used to generate viewpoint dependent error measures. Fig. 4 shows the mean squared error and its variance per viewpoint at the equator 4(a) and at different latitudes 4(b). Note in 4(a) that for views on the equator the error is smaller for longitudes closer to π radians, this corresponds to a view of the palm (from different latitudes). These performance differences are most likely due to that at side-view angles there is an increased amount of self-occlusion and also because the projections involve fewer pixels, reducing the samples used to calculate image moments. In 4(b) we can observe that reconstruction errors increase at the poles of the view sphere, where there is also little information projected to the image plane. While the MSE result is encouraging, the variance suggests that certain hand poses are not accurately recovered (we discover they mostly correspond to complex hand configurations coming from the American Sign Language part of our data).

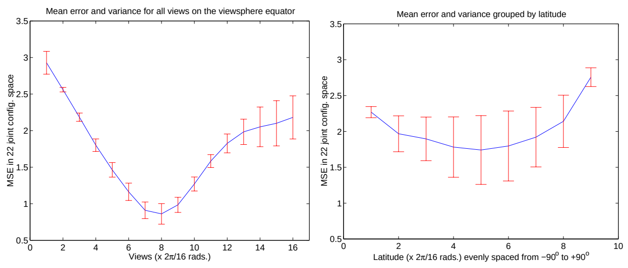


Figure 4: Quantitative experimental results. Mean square error in the reconstruction is shown in (a) taken at the equator of the view sphere, varying the longitude and (b) at different latitudes, averaging over all the longitudes. Longitude π and latitude 0 radians represents a view towards the palm of the hand.

7.2 Experiments with Real Sequences

In the next set of experiments, we tested the system against real segmented visual data. The sequences were segmented to yield blobs that corresponded to hands in each frame, as described in Sec. 6. The resulting reconstruction for several relatively complex gesture sequences is shown in Fig. 5. Note that given blob images, recovering 3D hand pose is a difficult task even for a human observer. This difficulty is increased by performing inference from blob moments, obviously with an inferior descriptive power. Methods for addressing this issue will be covered further in Sec.8.

7.3 3D Reconstruction Reliability

It should be noted that SMA's can provide a measure of reconstruction reliability by using the log-probabilities computed in Eq. 10. Ambiguous inputs can be discovered by looking at the relative scores given by Eq. 10 (another option is to look at the entropy of $P(\mathbf{h}|\mathbf{x})$). This is extremely important because even though the forward maps are designed to handle ambiguities, the inference process clearly still suffers from ambiguities. Therefore, it can be impossible to recover some configurations with enough reliability. As an example, in Fig. 5, the configurations 5-6 have low reliability score, even though we obtain good estimates. Some of the competing hypotheses include estimates that are also consistent (in terms of the visual features used) with the input presented, and some of these consistent hypothesis are far from the true 3D reconstruction. Thus, it was very likely to choose one of the bad estimates instead of the good ones shown in configurations 5-6.

8 Discussion and Conclusions

In this paper we addressed the problem of recovering 3D hand pose from a monocular color sequence. The main contributions of our work are:

1. A single observed frame can be used for estimation[2]. As a consequence, no manual initialization is required. Furthermore, the sequence can start with the hand in any position and orientation.

2. No limitation is imposed in the camera viewpoints allowed.

3. The system does regression rather than classification, thereby providing a continuum of pose estimates rather than recognizing a finite number of classes.

4. A novel non-linear supervised learning framework is adapted to the pose estimation problem. This framework allows us, among other things, to avoid the pitfalls of explicit tracking and to measure reliability of estimates during inference.

5. Reconstruction can be accomplished at near frame rate.

The main advantage of using SMA's in this domain over other function estimation paradigms is that it allows modeling of the ambiguous input-output relationships that arise. For instance, different hand configurations can generate the same visual features, due to self-occlusion. Different visual features can be related to a single hand configuration, due to inaccurate observations or variations in hand morphology. SMA's splits (partitions) the problem into simpler mapping problems. This allows for modeling different parts of the output space independently, as well as computation of multiple possible configurations in ambiguous situations. However, so far we choose one estimate only. This is an interesting aspect not fully addressed in this paper, Sec. 7.3 extends a little on this topic.

In our current implementation, temporal context is not used for improving the output estimates during mapping, but only for segmentation. The hand pose is re-estimated at every frame given the segmented data. We expect that using previous estimates in computing the current hand shape will improve accuracy, and we plan to extend our approach to allow this. However frame independence allows a very attractive inference time of $O(M)$, with M specialized functions.

Our algorithm could be used as a front end in several gesture recognition applications that take the hand configuration as input. Current systems rely almost exclusively on non-vision techniques to obtain such data, such as CyberGloves [7, 18, 19, 30] and color markers [11]. An automated computer vision technique like ours imposes no restrictions on users. It can also be used in domains where we do not have control of the data collection, and therefore we cannot require the use of more sophisticated input devices.

In future work, we plan to experiment with sets of features that are richer and more descriptive than binary silhouettes; *e.g.*,orientation histograms, or other texture features. Using stereo should further increase the accuracy of the system, by providing more shape constraints than a single 2D image does. Finally, more sophisticated models of temporal dependencies, like linear Gaussian Models in general [11, 34, 37], could be used in the feedback matching to guide the choice of best reconstruction.

Even though we have a useful estimate of confidence, given by Eq. 10, we are looking at alternatives for decreasing the error variance. 3D Hand pose reconstruction from a single image is a very difficult task, and at present no fully-general solution to the problem exists. Our results show that it is possible to approach this problem using a combination of vision and statistical learning tools. We consider this an important step considering the complexity of the task and the low descriptive power of the features currently employed.

[2]We insist that in applications where highly correlated frames can be observed, it is imperative to use this temporal information. However, the ability of our framework to estimate hand pose given only a single frame affords automatic initialization, faster estimation, and could be used as a bootstrap mechanism in more complex systems.

References

[1] R. Bowden, T. Mitchell, and M.Sarhadi. Non-linear statistical models for the 3d reconstruction of human body pose and motion from monocular image sequences. *Image Vision Comp.*, 18(9:729-737), 2000.

[2] M. Brand. Shadow puppetry. In *ICCV*, 1999.

[3] R. Cutler and M. Turk. View-based interpretation of real-time optical flow for gesture recognition. In *Face and Gesture Recognition*, pages 416–421, 1998.

[4] T.J. Darrell, I.A. Essa, and A.P. Pentland. Task-specific gesture analysis in real-time using interpolated views. *PAMI*, 18(12), 1996.

[5] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood estimation from incomplete data. *Journal of the Royal Statistical Society (B)*, 39(1), 1977.

[6] J.H. Friedman. Multivatiate adaptive regression splines. *The Annals of Statistics*, 19,1-141, 1991.

[7] M. Fröhlich and I. Wachsmuth. Gesture recognition of the upper limbs : From signal to symbol. In I. Wachsmuth and M. Fröhlich, editors, *Gesture and Sign Language in Human-Computer Interaction, Gesture Workshop*, pages 173–184, Bielefeld, Germany, 1997.

[8] R. Grzeszczuk, G. Bradski, M.H. Chu, and Jean-Yves Bouguet. Stereo based gesture recognition invariant to 3d pose and lighting. In *CVPR*, volume 1, pages 826–833, 2000.

[9] R. Hamdan, F. Heitz, and L. Thoraval. Gesture localization and recognition using probabilistic visual learning. In *CVPR*, volume 2, pages 98–103, 1999.

[10] T. Heap and D. Hogg. Towards 3d hand tracking using a deformable model. In *Face and Gesture Recognition*, pages 140–145, 1996.

[11] H. Hienz, K. Kraiss, and B. Bauer. Continuous sign language recognition using hidden markov models. In *Intl. Conf. on Multimodal Interfaces*, volume 4, pages 10–15, 1999.

[12] G. Hinton, B. Sallans, and Z. Ghahramani. A hierarchical community of experts. *Learning in Graphical Models, M. Jordan (editor)*, 1998.

[13] N. Howe, M. Leventon, and B. Freeman. Bayesian reconstruction of 3d human motion from single-camera video. In *NIPS*, 1999.

[14] M. K. Hu. Visual pattern recognition by moment invariants. *IRE Trans. Inform. Theory*, IT(8), 1962.

[15] M.J. Jones and J.M. Rehg. Statistical color models with application to skin detection. In *CVPR*, pages I:274–280, 1999.

[16] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6, 181-214, 1994.

[17] M. Kohler. Special topics of gesture recognition applied in intelligent home environments. In *Proceedings of the Gesture Workshop*, pages 285–296, 1997.

[18] R. Liang and M. Ouhyoung. A real-time continuous gesture recognition system for sign language. In *Face and Gesture Recognition*, pages 558–567, 1998.

[19] J. Ma, W. Gao, and C. Wang J. Wu. A continuous chinese sign language recognition system. In *Face and Gesture Recognition*, pages 428–433, 2000.

[20] J.P. MacCormick and M. Isard. Partitioned sampling, articulated objects, and interface-quality hand tracking. In *ECCV*, 2000.

[21] J. Martin, V. Devin, , and J.L. Crowley. Active hand tracking. In *Face and Gesture Recognition*, pages 573–578, 1998.

[22] R. Neal and G. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. *Learning in Graphical Models, M. Jordan (editor)*, 1998.

[23] A. Nishikawa, A. Ohnishi, and F. Miyazaki. Description and recognition of human gestures based on the transition of curvature from motion images. In *Face and Gesture Recognition*, pages 552–557, 1998.

[24] E-J. Ong and S. Gong. Tracking hybrid 2d-3d human models through multiple views. In *ICCV Workshop on Modelling People, Corfu, Greece*, 1999.

[25] J.M. Rehg. *Visual Analysis of High DOF Articulated Objects with Application to Hand Tracking*. PhD thesis, Electrical and Computer Eng., Carnegie Mellon University, 1995.

[26] R. Rosales and S. Sclaroff. Improved tracking of multiple humans with trajectory prediction and occlusion modeling. In *IEEE CVPR Workshop on the Interpretation of Visual Motion*, 1998.

[27] R. Rosales and S. Sclaroff. Specialized mappings and the estimation of body pose from a single image. In *IEEE Human Motion Workshop. Austin, TX*, 2000.

[28] R. Rosales and Stan Sclaroff. Inferring body pose without tracking body parts. In *CVPR*, 2000.

[29] H.A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *CVPR*, pages 38–44, 1998.

[30] H. Sagawa and M. Takeuchi. A method for recognizing a sequence of sign language words represented in a japanese sign language sentence. In *Face and Gesture Recognition*, pages 434–439, 2000.

[31] N. Shimada, Y. Shirai, Y. Kuno, and J. Miura. Hand gesture estimation and model refinement using monocular camera - ambiguity limitation by inequality constraints. In *Face and Gesture Recognition*, pages 268–273, 1998.

[32] L. Sigal, S. Sclaroff, and V. Athitsos. Estimation and prediction of evolving color distributions for skin segmentation under varying illumination. In *CVPR*, 2000.

[33] Y. Song, Xiaoling Feng, and P. Perona. Towards detection of human motion. In *CVPR*, 2000.

[34] T. Starner and A. Pentland. Real-time american sign language recognition using desk and wearable computer based video. *PAMI*, 20(12):1371–1375, 1998.

[35] A. Utsumi and J. Ohya. Multiple-hand-gesture tracking using multiple cameras. In *CVPR*, volume 1, pages 473–478, 1999.

[36] Virtual Technologies, Inc., Palo Alto, CA. *VirtualHand Software Library Reference Manual*, August 1998.

[37] C. Vogler and D. Metaxas. Toward scalability in asl recognition: Breaking down signs into phonemes. In *Proceedings of the Gesture Workshop*, 1999.

[38] J. Weng and Y. Cui. Recognition of hand signs from complex backgrounds. In R. Cipolla and A. Pentland, editors, *Computer Vision for Human-Machine Interaction*. Cambridge University Press, 1997.

[39] Y. Wu and T.S. Huang. View-independent recognition of hand postures. In *CVPR*, volume 2, pages 88–94, 2000.

[40] M. Yang and N. Ahuja. Recognizing hand gesture using motion trajectories. In *CVPR*, volume 1, pages 466–472, 1999.

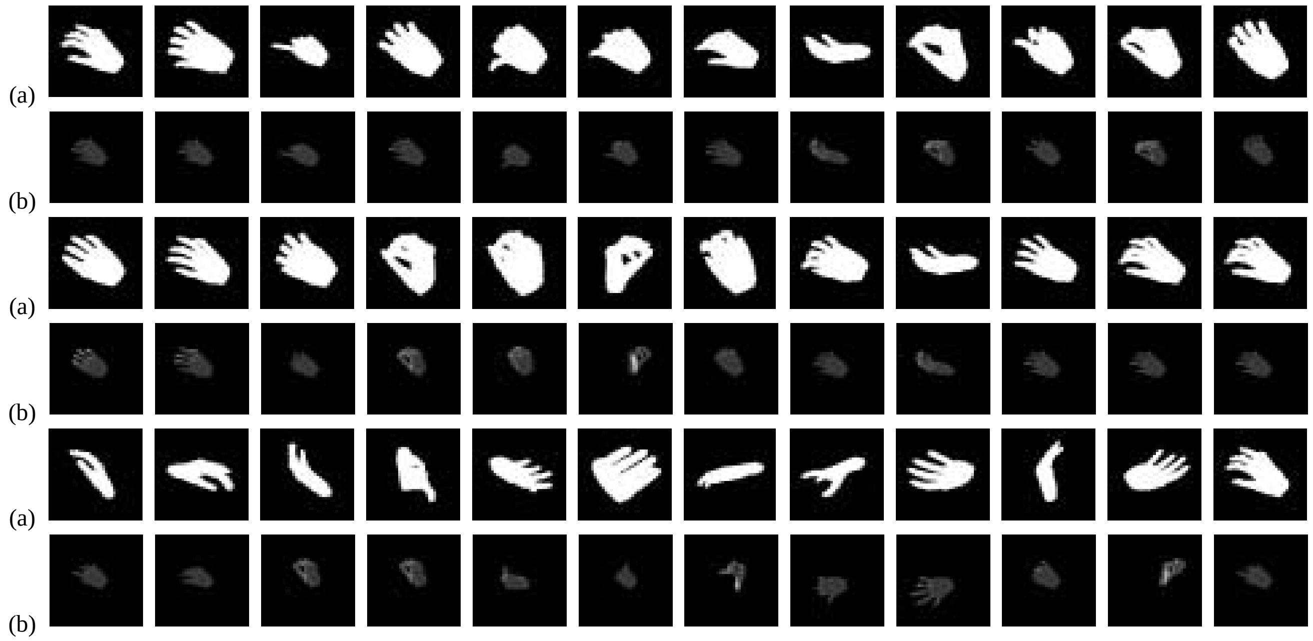


Figure 3: Example reconstruction of several synthetic test sequences. Each set (2 rows each) consists of (a)input images, (b)reconstruction. Because our approach can provide us with a reconstruction confidence, we used this to show high-medium-low confidence estimates (one pair of rows each).

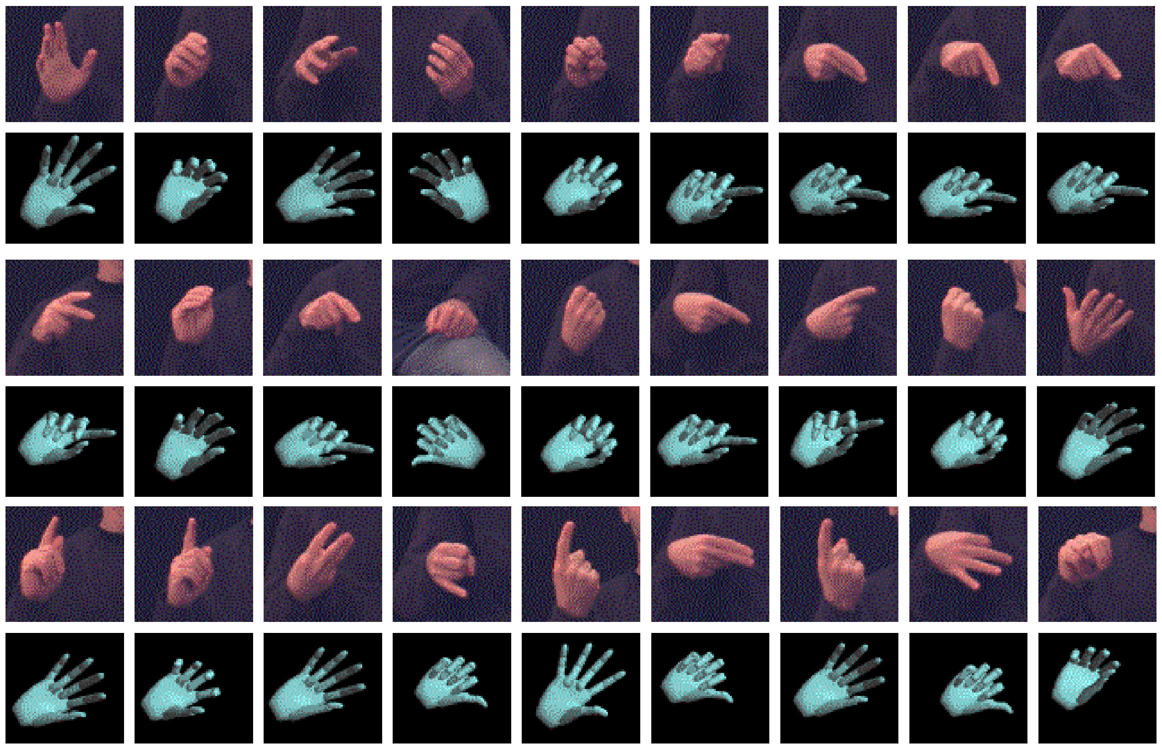


Figure 5: Reconstruction obtained from performing hand segmentation in a human subject. The two top pairs of rows show good reconstruction while the bottom pair show examples of bad performance. Reconstruction is shown from a fixed viewpoint (latitude 0-longitude π rads.).