3D Tracking = Classification + Interpolation

Carlo Tomasi, Slav Petrov, and Arvind Sastry
Department of Computer Science
Duke University — Durham, NC 27708

Abstract

Hand gestures are examples of fast and complex motions. Computers fail to track these in fast video, but sleight of hand fools humans as well: what happens too quickly we just cannot see. We show a 3D tracker for these types of motions that relies on the recognition of familiar configurations in 2D images (classification), and fills the gaps in-between (interpolation). We illustrate this idea with experiments on hand motions similar to finger spelling. The penalty for a recognition failure is often small: if two configurations are confused, they are often similar to each other, and the illusion works well enough, for instance, to drive a graphics animation of the moving hand. We contribute advances in both feature design and classifier training: our image features are invariant to image scale, translation, and rotation, and we propose a classification method that combines VQPCA with discrimination trees.

1. Introduction

Tracking people and their hands is useful in user-computer interfaces, communication over expensive channels, analysis and capture of human motion for medical, artistic, or scientific purposes, translation and interpretation of sign languages and gestures, and for many other tasks. All these applications would benefit from a vision system that without excessive intrusion computes the three-dimensional (3D) trajectories of limbs, heads, and fingers. While specialized sensors may be used to advantage, it would be most convenient if these trajectories could be computed from the data recorded by a single video camera. The past few years have seen important strides toward these goals (a mere sample is [21, 12, 3, 17, 16]).

Most of this work, however, makes the fundamental assumption that the motions of interest are slow relative to the frame rate of the camera that records the video, so that image features can be followed from frame to frame. Unfortunately, this assumption is all too often violated. Figure 1 illustrates this point by showing three consecutive frames of a hand closing at a natural speed, and recorded at 30 frames per second: the entire motion occurs in less than 66 milliseconds. Higher frame-rate cameras, a natural solution, are more expensive and require more light because of the shorter exposure times. Freeing tracking algorithms from the slow-motion assumption seems a more desirable alternative, and this paper shows a way to do this.

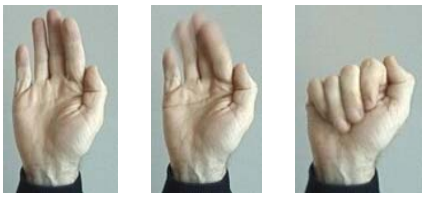


Figure 1: Three consecutive frames at 30 frames per second.

The motions of a single hand are used as an example running throughout this paper. We start with a database of known hand poses and configurations, for which both sample images and 3D configuration parameters are known. In doing this, we make the key conjecture that *only a relatively small subset of all possible configurations are of genuine interest.* For instance, only a few dozen configurations play a fundamental role in finger spelling, and several researchers have designed classification systems accordingly (see [19, 18] for surveys). This paper posits that these same configurations, or perhaps only a few more, are sufficient for *tracking* a hand doing finger spelling in three dimensions, that is, for inferring intermediate 3D configurations as well.

At first, the proposal may seem trivial: recognize finger-spelling configurations, and then somehow interpolate between them. However, several nontrivial problems remain, and ample opportunities lie ahead:

- Associating 3D pose and configuration parameters to 2D images requires reasoning on data from different sources, as shown in section 2.
- Interpolation in the 20-dimensional space of 3D finger motions is a problem of configuration-space path planning with collision avoidance, which can fortunately be simplified for hand tracking, as shown in section 3.
- The classification of hand configurations is hard, and prior work on the interpretation of finger spelling [7, 19] has been only partially successful. We propose

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE

invariant image features for this problem in section 4, and explore classification methods in section 5.

- Even with large amounts of training data, classification cannot expected to be perfect. In section 6, we discuss smoothing techniques that can reduce the effects of misclassification.

Section 7 shows experiments that support the feasibility of this approach to tracking. In the conclusion, we propose a method that combines real images and computer renderings to populate a rich and realistic database for classifier training.

2. The Tracking Paradigm

To illustrate our proposal, suppose that we wish to track a hand so that we can create a rendering of the input sequence with a computer graphics program. This is useful, for instance, when the rendered video is created at the output of an expensive or narrow-band communication channel: sending the commands that describe hand motions is less expensive than sending the video itself. Other applications include motion capture for animation, or the use of gestures to interact with a computer or other electronic device. In many of these applications, we may want to render the output from a different point of view, or with a different lighting than the input. Then, a three-dimensional model of the shapes and motions is necessary. Most graphics animation packages (we use [14]) have complete, detailed models of the shape and articulations of a hand, and these models can be animated by specifying pose parameters and joint angles for each frame of video to be rendered.

In this context, by "tracking" we mean processing the input video to determine the 3D parameters and angles that are necessary to drive such an animation. More specifically, the *pose* of a hand is described by the six Degrees of Freedom (DoF) of the wrist. A *configuration* is the set of finger joint angles, for which twenty DoF turn out to be adequate.

When tracking, four of these DoF can be measured rather directly in the image: two DoF for translations in the image plane, one for the rotation in the image, and one for translation in depth. The latter can be approximately inferred from image size through a weak-perspective [1] model. The remaining twenty-two DoF describe out-of-plane rotations (two DoF) and finger configuration, and are the main source of complexity for the problem.

We propose to handle these twenty-two degrees of freedom by recognition. In a nutshell, we first build a database that has several samples of each hand configuration for every out-of-plane rotation of interest. With each configuration (and therefore for a whole set of views and samples for the same configuration), we store the set of joint angles and pose parameters that describe that configuration, or at least a similar one, obtained manually through the same graphics package used for rendering. During tracking, we compare video frames to these samples, and whenever we see a "familiar" view we retrieve its configuration. For this approach to work, we need to (i) verify that it is indeed feasible to build such a database, (ii) specify a method for handling the "unfamiliar" frames in-between, and (iii) develop a classifier that retrieves the appropriate configuration for familiar frames.

Of course, it is hardly possible to enumerate all combinations of twenty-two parameters, even when quantized to a few values each. However, we make the key conjecture that *most of these combinations are either unlikely to occur, or they are uninteresting,* in the sense that they can be harmlessly replaced with other plausible configurations. Although only exact motion capture, or the talent of an artist, may provide the illusion of life [24], perhaps this illusion can be approximated by ensuring plausible motions.

A database of sample views is adequate when all unfamiliar configurations are uninteresting. To keep databases small, different application domains will usually have different databases. For instance, in this paper we illustrate our idea of recognition-based tracking through the example of finger spelling. Then, the configurations for each of the letters of the alphabet are the most interesting ones, and the database should contain sample views at least for these. In the conclusion, we return to the question of adequate databases for other applications.

It is important to notice that the *interpretation* of finger spelling is not our goal, but merely a means to our end, which is *tracking* a 3D model from 2D video. In fact, we will show that a tracker can sometimes work even with occasional interpretation failures.

In our experiments, we find hand-region candidates in each frame of the input video as large connected components [23] on the output of a skin-color detector [11]. This will also find faces and other regions. Some non-hand regions are filtered away by their size. The remaining regions are compared to the sample views in the database, so that regions that are not hands are unlikely to be consistently recognized as hands. However, we keep our illustrative experiments simple by placing a neutral background behind our hands.

In the remainder of this section, we summarize the sources of information for the computation of the various aspects of hand pose and configuration. In section 3, we discuss the treatment of unfamiliar configurations. The two sections thereafter address feature design and classification methods, respectively.

2.1. Four Easy Parameters

Four of the twenty-six hand pose and configuration parameters are computed as follows:

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE

In-plane position: Gärtner's Miniball code [8] finds the circum-circle for the hand region, and the displacements of the bottom point of this circle from frame to frame provide the two in-plane translation parameters of the hand pose. This works for both familiar and unfamiliar views, because no database image is involved. The computation of the circum-circle would be overkill for this computation, but we also use this circle for other purposes, described later on.

Position in depth: For familiar views, the radius of the circum-circle is divided by the radius of the circum-circle in the corresponding database view to provide a scale parameter. The latter is used in a weak-perspective camera model [1] to compute the approximate distance between camera and hand.

In-plane rotation: For each familiar view, as well as every database view, we compute the major and minor axes of an ellipse that approximate the hand region. Rotation in the plane of frame k is then computed as $\omega_k = \omega_{k-1} + \rho(\alpha - \omega_{k-1})$. In this expression, ρ is an increasing function of the ratio between the lengths of the major and minor axis of the database view, and α is the angular difference between the major axis in the image and the major axis in the matching database view. When the database view is close to circular, ρ goes to zero, and the rotation ω_{k-1} from the previous frame is preserved. For more elongated ellipses, ρ saturates to 1, and $\omega_k = \alpha$.

2.2. The View/Configuration Database

The remaining twenty-two parameters describe out-of-plane rotation (two parameters) and finger joint angles. For finger spelling, the forearm is never moved to point towards the viewer, so in our experiments we only account for out-of-plane rotations around a vertical axis. A database of sample views handles the single DoF for out-of-plane rotation and the twenty parameters for finger configuration.

Specifically, two of the twenty-six letters of the English alphabet (j and z) are motions of hands in configurations that, if the hand is kept stationary, would represent other letters (i and x). Thus, there are twenty-four distinct configurations of interest. We will add a few to these in section 3. We then use a graphics animation package [14] to determine typical joint angles for each interesting configuration (These parameters come in a convenient library that ships with the product. It would be a matter of about an hour to compute them from scratch.)

If we now discretize the out-of-plane rotations into fifteen values, approximately covering an 80-degree range, we obtain $24 \times 15 = 360$ different combinations of configurations and views. For each of these, we need sample images for a real hand. In our experiments, we considered different scenarios. In the simplest case, we asked someone to sign the twenty-four main signs (all letters except j and z) in front of a video camera, and to rotate their wrist several full swings. We observed that after two swings the rotation velocity (as measured by the number of frames) becomes consistently about the same across configurations (about 15 frames for the same subject), so we merely used fifteen frames out of the third swing for each sign as our sample views, and assigned each frame an angle obtained by splitting ± 40 degrees into 14 equal intervals. This yields a small database, with only one sample per view and configuration. Of course, one cannot expect any classifier to achieve good generalization with a training sample this small. Nevertheless, we achieved acceptable results in a limited number of experiments by keeping lighting and the imaging setup constant across video takes.

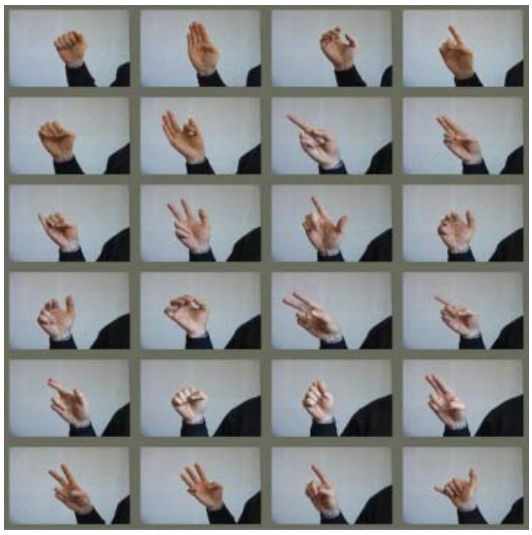
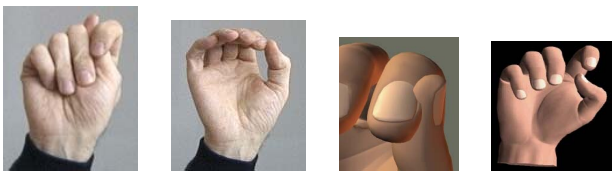


Figure 2: One view for each of the twenty-four configurations of finger spelling. Our smaller database contains fifteen views per configuration.

We also built a database of a more realistic size by capturing about 7,000 images of a single person signing the twenty-four signs several times with small variations in the details of the hand configuration for each sign. More on this database, and on the upgrade to user-independence, in the conclusion. Other than these remarks, the experiments in this paper are carried out with the small database. Figure 2 shows one view for each interesting configuration.

A video frame with a familiar configuration is then classified into one of these 360 categories as explained in section 5. This yields an out-of-plane rotation angle and a set of finger joint angles. Together, these parameters complete the set of parameters for familiar frames. Frames that fail classification are labelled as "unfamiliar," and are handled as discussed next.

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE



(a) (b) (c) (d)

Figure 3: Straight interpolation between 't' (a) and 'o' (b) would cause fingers to compenetrate (c) in the rendering. The disengagement position (d) extricates the thumb first.

3. Interpolation of 3D Configurations

In contrast with "pure tracking" approaches [10, 12, 22] that track hand configuration in each frame, we determine the configurations in frames between familiar views by interpolation. This is because motions often occur too quickly in video, and hand configurations are often too complex for pure tracking to work. However, pure tracking still has a useful role to play, discussed in the conclusion.

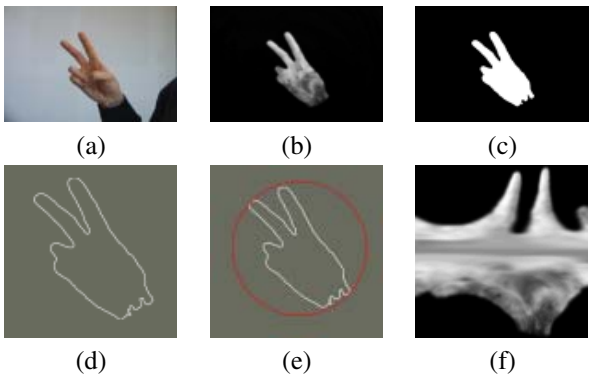
In this section, we discuss our methods for interpolating configurations between familiar frames. We assume that a moderate amount of latency is unimportant, so that the output can be buffered in the intervals between familiar frames. The method outlined in the previous section provides 3D pose and configuration parameters for familiar frames. Two such frames can be seen as the start and end frame of the unfamiliar video segment between them, and the 3D parameters for the latter frames can be obtained by interpolation.

A few difficulties arise in this process. First, straightforward interpolation would lead to undesired finger collisions. Figure 3 illustrates this for the transition from the letter 't' to the letter 'o'.

The general solution to this problem is to plan a collision-free path [15] in the configuration space of the hand. Fortunately, for hands the solution is simpler: Let us call the order of the fingers in the open hand the *natural order*. Some configurations violate this order. The 't' configuration is one example, 'r' is another (index and middle finger are crossed). Fortunately, the list is short. For each "unnatural" configuration, we define a *disengagement* configuration, which essentially extricates the violating finger from its unnatural position. For instance, figure 3 (d) shows the disengagement position for 't'.

The interpolator then merely inserts disengagement positions before and after unnatural configurations. If $\delta(t)$ is the disengagement configuration of 't' and the letters 't' and 'o' occur at frames $f(t)$ and $f(o)$, we set $f(\delta(t)) = \mathrm{round}((f(t) + f(o))/2)$, and interpolation occurs from 't' to '$\delta(t)$' to 'o'. In our experiments, we found linear interpolation to be generally acceptable.

Another small problem occurs when the hand in the input video transitions smoothly between letters, but the classifier transitions abruptly. For instance, a transition between a 'd' and an 'a' may occur over five frames, but the first two of these are still classified as 'a', and the last three as 'd'. This would cause an awkward jump in the rendering. To prevent this, the interpolator forces every transition to take at least $f_{\min}$ frames (we set this to 4) by relabelling frames that are within $f_{\min}/2$ on either side of a sudden transition as "unfamiliar," so that a smooth interpolation takes place.



(a) (b) (c) (d) (e) (f)

Figure 4: Steps for computing features.

Of course, the tracking paradigm we have sketched in the previous section hinges on our ability to develop a classifier with satisfactory performance. In the next two sections, we explain what features and classifier we use.

4. Features for Classification

Features for classification should be invariant to in-plane translation, rotation, and scaling. While several authors [19, 5] have used the hand's silhouette, distinctions between hand configurations in which fingers are essentially closed into a fist require looking inside the hand as well. There are at least four such configurations: 'e', 's', 't', and 'a', and even an 'o' viewed at an angle is essentially a closed hand. Even when one or more fingers are extended, the distinctions between letters are often related to the position of the fingers that are curved into the palm. These distinctions cannot be inferred reliably from the hand's silhouette.

To achieve invariance, and at the same time acquire information about the inside of the hand region as well, we subject database and video images to the processing steps illustrated in figure 4.

The image is smoothed with a 2D Gaussian kernel of standard deviation 2 pixels in order to eliminate color dithering artifacts. As mentioned above, the skin detector of [11] produces the likelihood image in figure 4 (b). Thresholding the likelihood image yields the silhouette of the hand (c). Because of reflections from the sleeve and poor lighting from below, the wrist has a blue tinge and is not classified as skin. Morphological closing [9] solves this problem as it

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE

connects outlying pixels to the rest of the hand, completing the silhouette.

A gray level version of the input image is masked with the silhouette, and the contour (d) of this is computed in order to find the circum-circle [8]. Normalizing the radius of this circle to one, and referring the image to the circle's center provides translation and scale invariance. The influence of lighting is reduced by thresholding the image with its median value, and smoothing with a six pixel-wide Gaussian kernel in order to decrease the importance of the exact location of the edges.

Rotation invariance is now achieved by a method inspired by the literature on the Mellin transform [4], although we handle scale and translation separately as explained above. Equidistant points on concentric circles are sampled (so the circles closer to the center have very few samples). This has the same effect (but is faster) than transforming the image to polar coordinates (f), and then sampling the resulting image along horizontal lines, with a linearly decreasing number of samples per row. The Fourier transform of the values on each circle is calculated, and only the magnitude is retained. Since the magnitude is translation-invariant, this achieves rotation invariance on each of the circles. The magnitudes from all the circles are collected into a 1050 dimensional vector that we use as our feature for classification.

These vectors are a rather detailed representation of each hand. In fact, except for sampling, and for discarding the phase of the Fourier transforms, all steps are reversible. Of course, the large size of these features subjects them to the curse of dimensionality [2]. Our classifier, described in the next section, addresses this issue.

5. Classification

Classifying hands is hard because of the vast differences in hand shapes and colors, articulation habits, details of gesture configurations, lighting conditions, backgrounds, viewpoint, and imaging parameters. Because of all these factors, a classifier can work reliably only if a very large database of samples is available for learning, or if features somehow capture what is relevant to class membership, and discard what is not. Both these options are hard, the first logistically, the second conceptually. In the previous section, we have designed features that achieve a substantial degree of invariance. In this section, we discuss a classification method that allowed us to achieve reasonable classification rates with a small number of training samples. This classifier works under restrictive assumptions: single-user, restricted lighting, and a fairly disciplined way to sign. However, performance is surprisingly good given the small number of training samples. We consider directions for more general scenarios in our conclusions.

Principal Component Analysis (PCA) has been proposed by many (e.g. [26]; see also [25] for a good introduction) as a way to reduce the dimensionality of feature space, so as to retain as far as possible only the aspects of a feature that are relevant to classification. In our experiments, we have devised a variation on the theme of Vector Quantization PCA (VQPCA) first proposed in [13]. VQPCA generalizes PCA by computing a set of linear subspaces, each of which accounts well for a portion of the feature space. In [25], VQPCA is further improved into a mixture of PCAs model. We retained the simpler features of VQPCA, but adapted it to reflect the small number of training samples we are working with in our experiments.

VQPCA applies the EM algorithm [6] to cluster data into k groups. However, instead of minimizing distance from a cluster center, VQPCA clusters so as to minimize distance from a subspace, as defined by the normalized projection residual. If $\mathbf{f}$ the initial feature and $\mathbf{p}$ is its projection, the normalized projection residual is defined as $\sqrt{(\mathbf{f}^T\mathbf{f} - \mathbf{p}^T\mathbf{p})/\mathbf{f}^T\mathbf{f}}$. In using VQPCA, we found it hard to assign reasonable values to the number and initial values of the subspaces so as to obtain consistent results. Instead, we constructed a binary tree of subspaces by recursively calling VQPCA with $k = 2$, first on the entire database, and then on the two groups generated by each call. We stopped the split of a group whenever the projection residual for all its members fell below a threshold. A softened version of this scheme assigns a feature to *both* children when both projection residuals for that feature are small.

Classification starts at the root of the tree, and at every node the new feature is assigned to the child with the better projection residual. At the leaf, the feature is classified by nearest neighbor. In this way, the number of clusters depends on the distribution of residuals, and need not be predefined. Also, subspace initialization with $k = 2$ is less critical than with higher values of k.

6. Handling Classification Errors

When two different features are confused with each other, classification can fail to retrieve the correct sign, the correct view, or both. Confusion can occur because two 3D configurations are similar to each other. In this case, misclassification is not necessarily fatal: the rendered video will have the wrong configuration, but because this looks similar to the correct one the effect may not be important. We will see an example of this in section 7, where the letter 'o' was confused with the letter 'c'. The only difference between the two is that thumb and index touch each other in the 'o', but not in the 'c'.

A second type of misclassification occurs when the correct sign is recognized, but the wrong view of it is retrieved. This is likely to cause glitches in the rendered mo-

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE

Figure 5: The signs for 'v', 'w', 'm', and 'n' (left to right).

tion. To address this problem, we used a simple exponential smoother for the vector $\mathbf{q}_k$ of geometric parameters to produce the smooth version $\mathbf{s}_k = (1-\rho)\mathbf{q}_k + \rho\mathbf{s}_{k-1}$. This smooths the viewing angles as well as translation, scale, and in-plane rotation.

A third type of misclassification occurs when two different configurations look similar from different views. For instance, the letter 'v' has two outstretched fingers, and 'w' has three (see figure 5). When viewed from the side, the distinction is unclear. We marked these ambiguous side views in the database, and extrapolated interpretation from previous non-ambiguous frames into ambiguous ones. For instance, if a hand in the 'w' configuration rotates, frontal views of it are likely to be unambiguous. As a side view is reached, a possibly erroneous classification to a 'v' is overridden, because the side view is known to be ambiguous, and the previous, unambiguous 'w' interpretation prevails.

An even more serious misclassification confuses features corresponding to entirely different hand configurations: they are different, they look different, but they are confused. This is obviously a failure of feature design. These problems arise from the complexity of hand shapes and configurations: the same person will arrange her hand differently when signing the same letter twice. Letters like 'e', 's', 't', and 'a', in which the hand is closed into a fist are particularly prone to this problem. Also, 'm' and 'n' (figure 5), in which fingers point directly at the camera, create unpredictable 2D shapes, and are often confused with many other letters.

To minimize the effects of these errors, we slide a window on the string of classification results, and output an interpretation whenever a letter has the absolute majority in the window. When this is not the case, the frame is left as uninterpreted (*i.e.*, " unfamiliar").

7. Experiments

To illustrate the tracking framework proposed in this paper, we ran experiments on short videos with motions similar to finger spelling. This section summarizes findings and intermediate results for two of these sequences. The actual videos, both input and rendered output, are made available as supplements to this paper (and on a web site if this paper is published).

Figure 6: The signs for 'today'. Images are cropped for clarity.

```
Frames 1-50
D: TTTTTTTTTTTTTTTTTTTTTTT??????????OOOOOOOOOOOOOOO
R: TTTTTTTTTTTTTTTTTTTTTTTTDDDDFFMMYYNNCCCCCCCCNNCCAA
F: TTTTTTTTTTTTTTTTTTTTTTTT???????????CCCCCCCCCC????

Frames 51-100
D: OOOO????DDDDDDDDDDDDDDDDDDDDDDDDDDD??????AAAAAAAA
R: AAAANNNNNNUUDDDDDDDDDDDDDDDDDDDDDDDDBBDDAAAAAAAAAAA
F: ???????????DDDDDDDDDDDDDDDDDDDDDDDD??AAAAAAAAAAA

Frames 101-148
D: AAAAAAAAAAA????YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
R: AAAAAAAAAAAANNNNYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
F: AAAAAAAAAAA????YYYYYYYYYYYYYYYYYYYYYYYYYY???????
```

Figure 7: Interpretation results for the 148 frames of the 'today' sequence. Each group of three adjacent lines shows the Desired (D), Raw (R), and majority-filtered (F) interpretation.

In our experiments we used a database with 15 views for each of 24 signs. Sample views are shown in figure 2. The tree for the classifier uses 16-dimensional PCAs and VQPCA is initialized with random subset assignments. Node splitting is stopped when the maximum normalized residual falls below 0.1. The final tree, which is fairly well balanced, is seven levels deep, and there are approximately 20-30 elements at each leaf.

In one sequence, 148 frames long (5 seconds), the word 'today' is spelled. Figure 6 shows details of one representative frame for each letter.

Figure 7 shows the frames as interpreted by hand, after raw automatic classification, and after majority filtering of the raw results with a window size of 15. Several misclassifications occur during sign transitions. In addition 'o' is consistently misclassified as 'c', and 'n' replaces many letters, as explained in section 6. Majority voting cleans all the glitches but, of course, 'o' remains misclassified as 'c'.

View angles are close to the correct viewing angle, and exponential smoothing cleans the rendering, which can be seen in the supplementary video Today.avi next to the input sequence. In this rendering, only interpretation is at work, and we have turned off the tracking of in-plane rotation, translation, and scale.

The full tracking is shown at work in supplementary video Countdown.avi, which shows a hand counting down from three to one while rotating left to right, then waving at the camera, and finally moving towards the camera until it obstructs the entire view. This video, which is 333 frames long (11 seconds) also illustrates the point that finger spelling signs can be considered as a mere list of inter-

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE

esting hand configurations: the goal is tracking, not classification. These configurations can be replaced with others, or perhaps augmented with others that are of interest in a different domain. Figure 8 shows a few sample frames of this sequence. Motions for frames at the very end of the sequence are extrapolated rather than interpolated.

When viewing the rendered sequence for the first few times, the illusion works well: tracking follows the hand in all its motions, including depth; rotations are close to the ones in the input sequence; and the rendered gestures are also similar to the input ones. Upon closer inspection, however, an interesting difference can be noticed between input and output in the waving part of the motion, illustrated in particular in frame 221 (third from the bottom in figure 8). The input hand has the thumb to the side of the other fingers, while the thumb is bent into the palm in the rendered sequence. This occurs because the configuration in the finger spelling database that is closest to an open hand is that for the letter 'b', which happens to have the thumb on the palm. Reconstruction is not exact, and yet the difference is noticeable only after careful analysis or repeated presentation. This demonstrates two points: First, exact tracking may often be unnecessary. Second, our paradigm of tracking by classification and interpolation may yield a plausible analogy for why we ourselves are sometimes fooled by a nimble-fingered magician.

8. Conclusions and Future Work

The main point of this paper was to show that complex motions can be tracked in three dimension by a combination of 2D image classification and 3D motion interpolation. By far the hardest hurdle to overcome towards a fully reliable implementation of our proposal is the construction of a good classifier. This involves designing good features, possibly improving the learning and classification techniques used, and most importantly acquiring more data. In our work, we have collected a database of about 7,000 hand images, but we have not yet used this because of logistical difficulties in sorting out viewing angles and more important difficulties in controlling lighting conditions.

In this regard, we are working in two separate directions: the first is a more systematic and more carefully controlled series of image collection sessions, on which we hope to report at a later date.

The second direction involves using for database creation the same graphics animation software we used for rendering. In a nutshell, we ask a user to put her open hand in front of the camera (figure 9, middle). An optimization program then adjusts shape, position, and configuration parameters for a rendered hand to match the real one. For optimization, we are using the Nelder-Mead Simplex method [20], and we use the fraction of hand pixels that overlap in

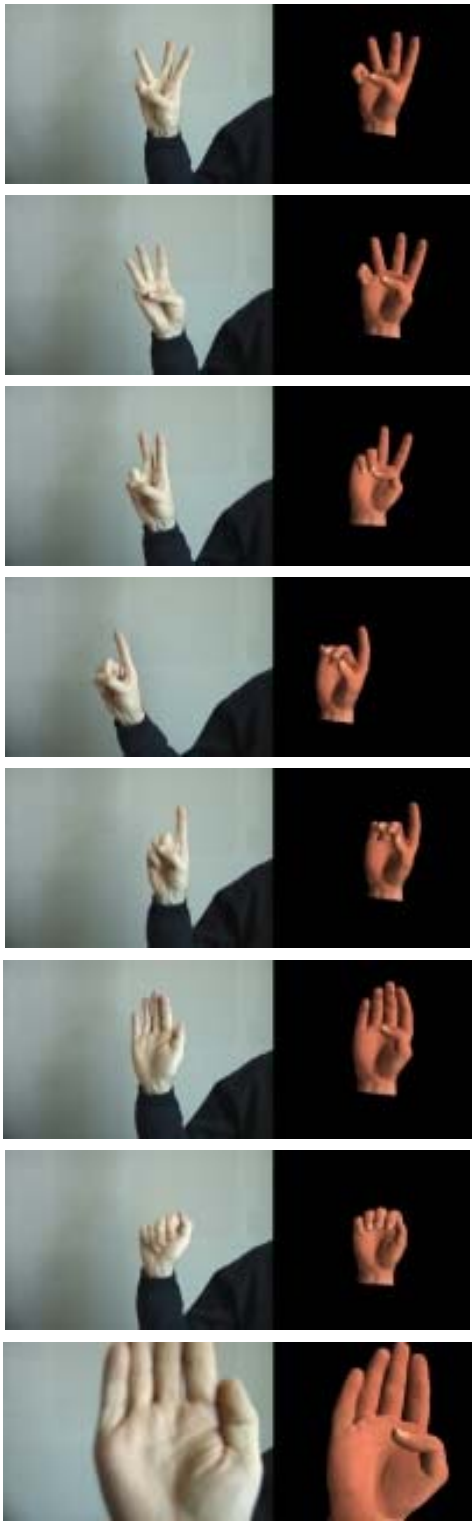


Figure 8: Frames 36, 59, 106, 146, 161, 221, 227, and 297 of the 'countdown' sequence.

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE

Figure 9: An optimizer transformed the shape of an initial rendered hand (left) to a new one (right) that is more similar to the real hand (middle).

the two images as our distance measure between real and rendered hand. Figure 9 shows the result of this optimization.

The plan is then to map the texture from the real hand onto the rendered one, and create a large database of images for that user by random perturbations of the basic configurations of interest. These perturbations must be designed with care for realism. In particular, fingers that tend to move together for a particular configuration must be perturbed in similar ways.

For each user, a database of any desired size could be then constructed automatically. This is particularly important when two-DoF out-of-plane rotations are allowed. Databases from different users can be merged towards a user-independent system.

It will also be interesting to see how well our tracker can cooperate with trackers based on differential methods [10, 12, 22]: our motions can be used as providing initialization for these methods, which can refine model parameters to match the input frames in greater detail.

Directions for further work include methods for finding hands in complex scenes, and for tracking two or more hands in the same video, or hands that manipulate objects.

Eventually we would like to understand how many basic configurations are needed for a generally useful hand tracker, and perhaps whether some of these ideas can be extended to tracking people in their entirety.

Acknowledgments

We thank R. Manduchi, S. B. Göktürk and B. Krishnapuram for several useful discussions, and J.-C. Latombe for suggesting the automatic creation of a database from rendered images. This paper is based upon work supported by the National Science Foundation under Grant No. 0222516.

References

[1] J. Aloimonos. Perspective approximations. *Image and Vision Computing*, 8(3):179–192, August 1990.

[2] R.E. Bellman. *Adaptive Control Processes*. Princeton U. P., 1961.

[3] C. Bregler. Learning and recognizing human dynamics in video sequences. In *1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 568–74, San Juan, Puerto Rico, 1997. IEEE Comput. Soc.

[4] D. Casasent and D. Psaltis. Position, rotation, and scale invariant optical correlations. *Applied Optics*, 15:1793–1799, 1976.

[5] James Coughlan, Alan Yuille, Camper English, and Dan Snow. Efficient deformable template detection and localization without user initialization. *Computer Vision and Image Understanding: CVIU*, 78(3):303–319, 2000.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data wia the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):1–22, 1977.

[7] R. Erenshteyn and P. Laskov. A multi-stage approach to fingerspelling and gesture recognition. In *Workshop on the Integration of Gesture in Language and Speech*, pages 185–194.

[8] B. Gärtner. Miniball, 2002. http://www.inf.ethz.ch/personal/gaertner/miniball.html.

[9] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision, Volume I*. Addison Wesley, Reading, MA, 1992.

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

[11] M. J. Jones and J. M. Rehg. Statistical color models with application to skin detection. In *IEEE Conf. on Computer Vision and Pattern Recognition*, volume 1, pages 274–280, 1999.

[12] I. A. Kakadiaris and D. Metaxas. Model-based estimation of 3D human motion with occlusion based on active multi-viewpoint selection. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition, CVPR*, pages 81–87, 18–20 1996.

[13] N. Kambhatla and T.K. Leen. Dimension reduction by local principal component analysis. *Neural Computation*, 9(7):1493–1516, 1997.

[14] Curious Labs. Poser, 2003.

[15] M. Lin, D. Manocha, J. Cohen, and S. Gottschalk. Collision detection: Algorithms and applications. In *Proc. of Algorithms for Robotics Motion and Manipulation*, pages 129–142, 1997.

[16] J. Martin, V. Devin, and J. Crowley. Active hand tracking, 1998.

[17] Jerome Martin and James L. Crowley. An appearance-based approach to gesture-recognition. In *ICIAP (2)*, pages 340–347, 1997.

[18] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding: CVIU*, 81(3):231–268, 2001.

[19] Vladimir Pavlovic, Rajeev Sharma, and Thomas S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):677–695, 1997.

[20] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, Cambridge, MA, 1988.

[21] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *Fifth International Conference on Computer Vision*, pages 612–17, 1995.

[22] B. Stenger, P. R. S. Mendonça, and R. Cipolla. Model-based hand tracking using an unscented Kalman filter. In *Proc. British Machine Vision Conference*, volume I, pages 63–72, Manchester, UK, 2001.

[23] S. Suzuki and K. Abe. Topological structural analysis of digital binary images by border following. *CVGIP*, 30(1):32–46, 1985.

[24] F. Thomas and O. Johnston. *The Illusion of Life*. Hyperion, 1995.

[25] Michael E. Tipping and Christopher M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

[26] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 1991.

Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set
0-7695-1950-4/03 $17.00 © 2003 IEEE