Hand Tracking via Efficient Database Indexing

Miro Enev
UCDS Cognitive Science Department
9500 Gilman Dr., La Jolla CA
menev@ucsd.edu

Abstract:
Tracking the pose of a moving hand from a monocular perspective is a difficult problem. In the current study the tracking problem is recast as efficient database indexing. Recognition is hastened by approximating chamfer distances between hand poses which have been embedded into a common vector space. Using this embedding, a small number of candidate matches for each query can be quickly identified using simple vector comparisons. These candidates are then processed with an expensive metric (chamfer distance) to determine the optimal match. This approach does not suffer from the traditional weakness of tracking methods, and lends itself naturally to real time solutions.

1 Introduction

Hand tracking in image sequences is a difficult problem due to the highly articulated and self-occluding nature of the hand. The hand's intricate joint structure (Figures 1 & 2) enables 21 degrees of freedom (DOF). An additional 6 DOF are required to capture orientation and location information, and in sum 27 parameters are needed to define pose and location information.

Figure 1

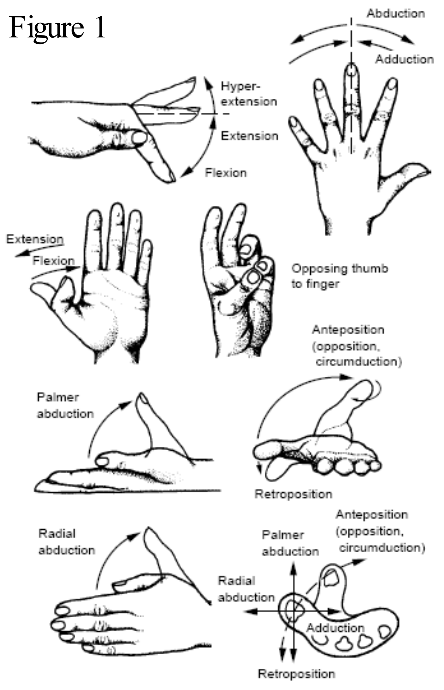


Figure 2

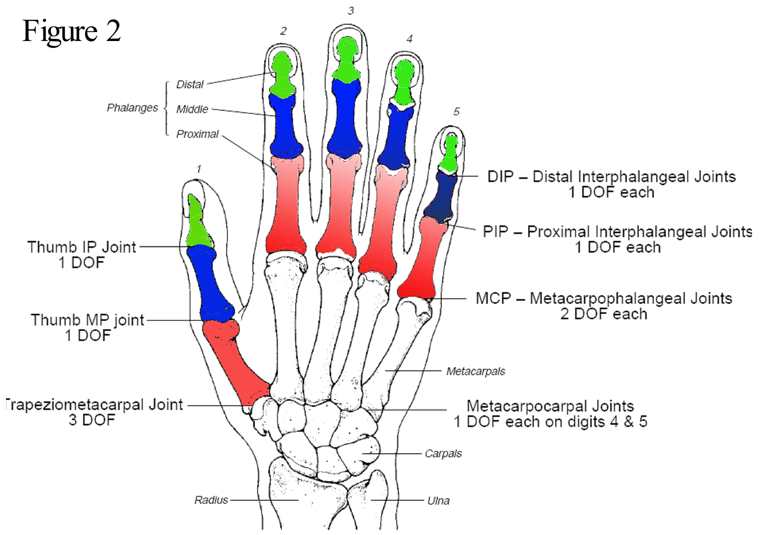
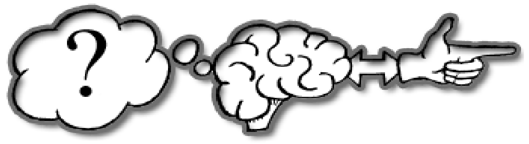

1
2
3
4
5
Phalanges
Distal
Middle
Proximal
DIP – Distal Interphalangeal Joints 1 DOF each
PIP – Proximal Interphalangeal Joints 1 DOF each
MCP – Metacarpophalangeal Joints 2 DOF each
Thumb IP Joint 1 DOF
Thumb MP joint 1 DOF
Trapeziometacarpal Joint 3 DOF
Metacarpals
Metacarpocarpal Joints 1 DOF each on digits 4 & 5
Carpals
Radius
Ulna

Figure 3



In addition to the high dimensionality, a significant challenge to hand tracking, is our lack of a good model for the transition dynamics of these parameters in successive poses. Thus even if all DOF are specified for a frame of a video sequence, predictions of subsequent frames are very limited. Essentially, our lack of knowledge about the brain's control policies leads to fickle dynamical models (Figure 3).

Despite the aforementioned challenges, there are solutions in the hand tracking literature that yield notable results [1, 5-8]. The successful approaches for hand tracking rely primarily on efficient database indexing methods, and a few introduce simple dynamical models for transitions in the state space of poses. Although the database indexing techniques have been quite fruitful, the dynamical models in the literature are either basic or non-existent. Any attempts at complex dynamical models suffer from accumulating estimation errors that quickly diverge from the ground truth [9]. In practice only simplistic linear approximations between poses in parameter space seem to be of any worth; although these are usually coupled with a high frame rate camera system whose rapid production of new state observations makes the computations of a transition model obsolete.

The emphasis in the current study is on efficient matching of a query hand image to a large database of candidate poses. This strategy recasts tracking as a sequential recognition problem, and circumvents traditional problems (eg. costly initialization and recovery from errors in previous estimates). Similar ideas have been pursued by by several authors - Stenger [6] uses a heirarchical decomposition of the state space, while Sudderth [5] uses a KD-tree representation of edge features to speed up chamfer comparisons.

In the current work I rely on an embedding method developed in Athitsos et al 2003 [1] to speed up the matching process. In this line of reasoning, query images are not compared to all of the elements in the pose database, but only matched against a subset of landmarks, or reference images. The term embedding refers to the fact that a query pose is re-represented as a signature vector of distances relative to each landmark - thus embedded in the landmark reference space. The intuition here is that similar signatures vectors are expected to arise from similar poses (Figure 4). To find the best candidate match to an incoming query, the pre-computed signature vectors of all of the poses in the database are compared to the query signature vector. This requires simple operations (L-2 norm) over vectors with dimensionality equivalent to the number of landmarks, and is computational inexpensive relative to computing distance metrics between poses in the original feature space. In practice, choosing a small subset of landmarks (randomly among the poses in the database) yields satisfactory retrieval results. The speed and accuracy of the embedded approach makes indexing the database at every frame of an image sequence viable for real time applications.

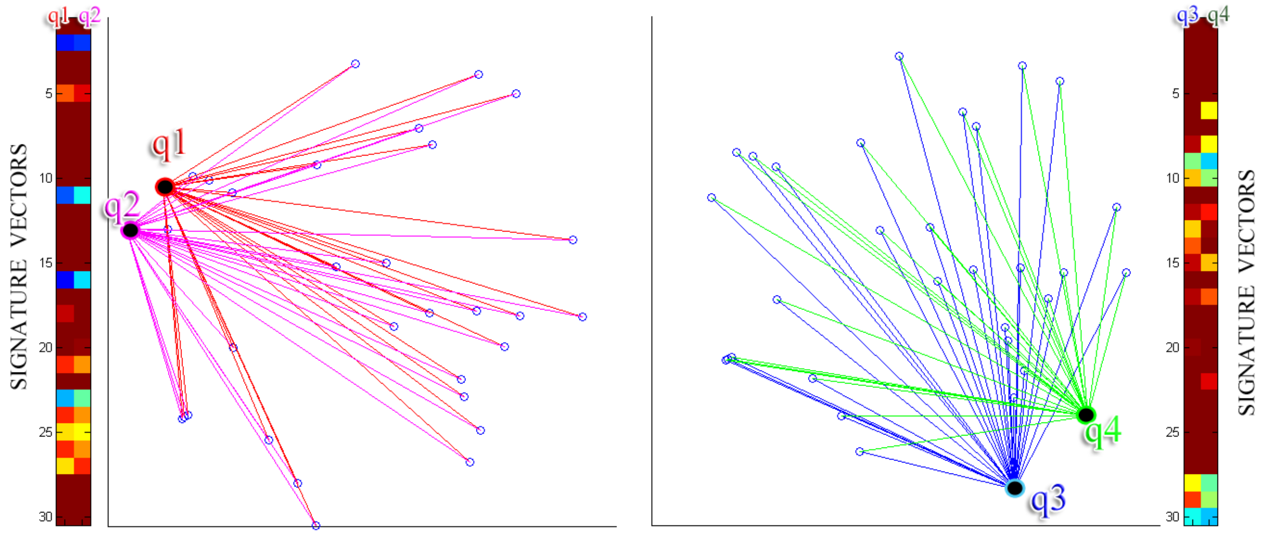
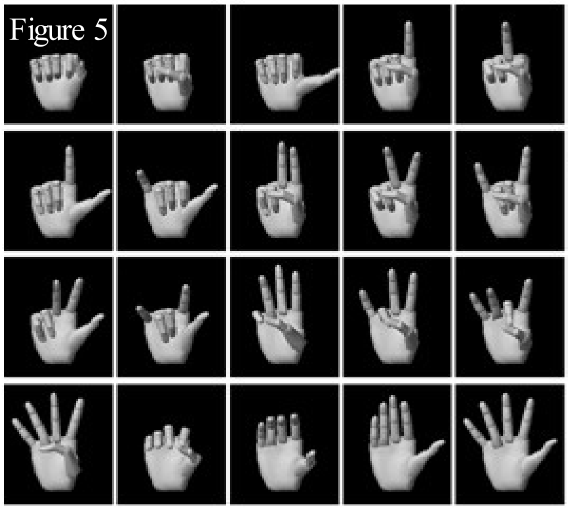


Figure 4: Here we can see that the signature vectors of nearby query points q1 and q2 are similar to each other. This is also the case for the vectors for q3 and q4. However the signature vectors for q1 vs q3 are dissimilar.

2 Framework

In order to find a good match to an incoming query a number of streps are taken. Initially, the query image is centered, rescaled, and reduced to an edge features representation. The query edge image then has its signature vector computed (embedding), and this is compared against the signature vectors of all of the poses in the database. The 20 nearest signature vectors (based on L-2 distance) to the query are then located. For each of the 20 candidate poses, 40 rotational transforms are performed at 9 degree iterations. Among these images the pose with the least chamfer distance relative to the query edge image is considered the best match. These steps are explained in more detail below.

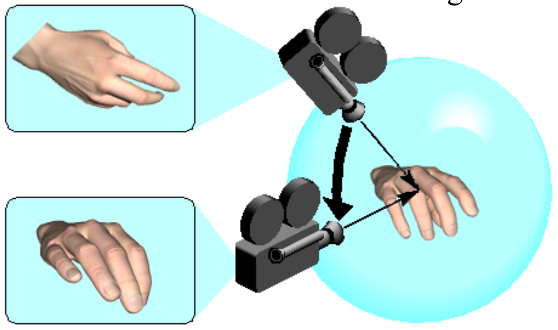
Figure 5



2.1 Database

The synthetic pose database was provided by Vassilis Athitsos. It is based around 20 American Sign Language (ASL) hand shapes (Figure 5) that were rendered in Poser. Each image has 24 degrees of freedom (location is fixed) and is rendered from 84 different viewpoints. The different viewpoints define the coordinates of a camera at the edge of a perspective sphere (Figure 6) which is pointed at the centrally located hand. The viewpoints range in a uniform fashion to cover the sphere.

Figure 6



To reduce the dimensionality of the data, only edge features of the poses were saved. Furthermore, all images are rescaled to 256x256, and the edge image is centered so that its minimum enclosing circle has center (128,128) and radius 120.

In the original paper [1] the authors increased the coverage of the 3D space of orientations by re-rendering each database pose an 48 additional times (a series of 7.5 degree rotations 48*7.5 = 360). Their database contains 80640 poses intended to offer an approximately uniform and dense sampling of the space of 3D orientations for the selected hand shapes.

In the current work the database size is 20*84=1680 poses and the 48 pose rotations are omitted due to the running time of MATLAB implementation. The running time for computing the embeddings with 50 landmarks in a database of 1680 poses was roughly 10 hours. To compensate, rotations are computed on a need basis for each query.

2.2 Selecting landmarks

In Athitsos et al. [1] the authors conclude that random selection of landmark poses for the embedding outperforms more elaborate methods in terms of accuracy. The only weakness of the random selection process is the slightly increased variance of the performance. However, as long as enough random landmark poses are selected, the embedding scheme performs well. In practice 200 landmarks were sufficient to produce strong results with a 80640 pose database. Inspired by these results I also adopt a random selection policy for landmark selection. In particular, I chose to use 50 landmarks with a database of 1680 poses. Although I have decreased the total number of poses by a factor of 48, I have increased the ratio of landmarks to poses by a factor of 10.

2.3 Computing embeddings for the database

Using the 50 reference poses (landmarks), I compute signature vectors for each pose in the database using the chamfer distance metric (described below). The (i,j) entry in the signature vector for a particular pose indicates the similarity between the i-th reference image and the number of the pose in the databse (j). The vectors for all of the database poses are concatenated into a matrix of embeddings.

2.4 Chamfer Distances

Previous simulations [2] with several different similarity measures (e.g. chamfer distance, geometric moments, edge orientation histograms, finger matching, and line matching) show that the chamfer distance is the most accurate, but also the most computationally expensive metric for shape matching. Given two edge images X and Y, the equation for the chamfer distance C(X,Y) is as follows:

$$C(X,Y)=\frac{1}{|X|}\sum_{x\in X}\min_{y\in Y}\|x-y\|+\frac{1}{|Y|}\sum_{y\in Y}\min_{x\in X}\|y-x\|$$

The high computational cost of this method is due to the non-linear minimization term which requires that for each pixel in image X a nearest neighbor search is performed among all the pixels in image Y. This operation is repeated for all the pixels in image Y since the nearest neighbors of X->Y are not the same as those of Y->X. This is illustrated in the lower row of Figure 7 which uses blue lines to connect nearest neighbors from pose pose{1} to pose{100} and red lines to connect neighbors going from pose{100} to pose{1}. In summary, the non-linear min operator drives the complexity of the chamfer calculation to O(Xn log Yn) - where Xn denotes the number of pixels in X, and Yn the number of pixels in Y.

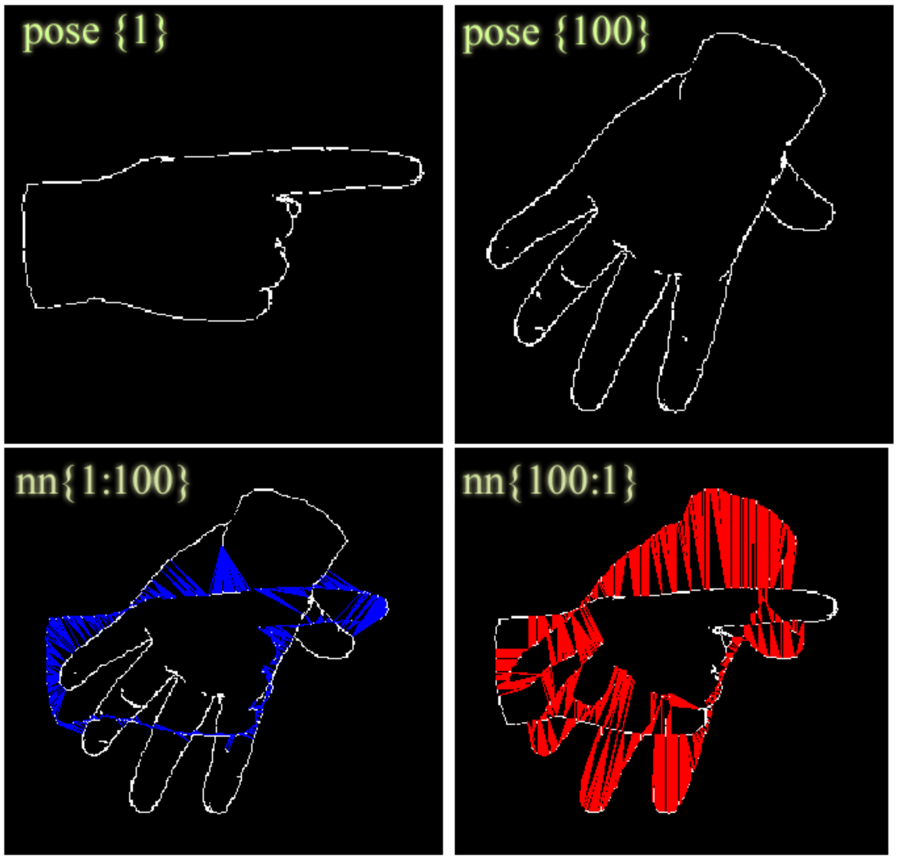


Figure 7: Sample chamfer computation between two poses. Chamfer distance = 42.124

2.5 Pre-processing the query image

Once the signature vectors of the database have been computed we are ready to handle a query. Query images are photographs of a hand against a controlled (black) background. These are RGB color images that include the fingers, palm, and a small section of the forearm.

The aim of the pre-processing is to reduce the amount of data while preserving the important structural properties of the image. To this end, the incoming RGB query images are converted to grayscale intensity values. This is a valid operation since the color channel does not carry any relevant pose information (given the controlled background). The shape is then centered (if not already centered) and scaled to 256x256 pixels using bicubic interpolation (Figure 8).

Next, edge features are extracted using a Sobel operator. The Sobel edge detector is a fast algorithm that uses a pair of (3x3) convolution masks to estimate the vertical and horizontal gradient of the image. Pixels with high Sobel scores (above a threshold of 0.04) are considered to be edges.

Sobel score $|G| = \sqrt{Gx^2 + Gy^2}$ where

Gx

-1	0	1
-2	0	2
-1	0	1

Gy

1	2	1
0	0	0
-1	-2	-1

The Sobel edge detection was chosen after experimental simulations against other edge algorithms (e.g Canny method). It was the most robust method for extracting the palm outline and the finger delineations.

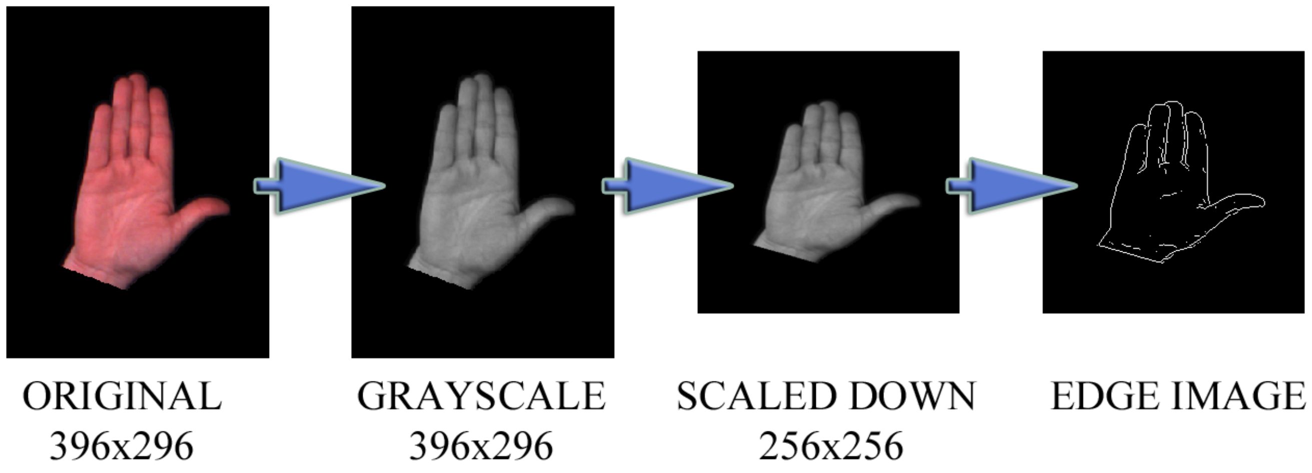


Figure 8: Preprocessing a query image.

2.6 Computing a signature vector for the query

The edge pixels of the query image are used to compute its signature vector (embedding). In the current setting, this operation requires 50 expensive online chamfer computations. To expedite this procedure I developed a down-sampled version of the chamfer computation which reduced the set of edge pixels to around 500. I settled on this number because it did not significantly reduce the accuracy of the method. The down-sampled approach drastically reduces the cost of the metric comparison between the query and the landmarks since the chamfer computation now requires fewer nearest neighbor searches and each search is shortened.

2.7 Finding candidate poses

Using a euclidean distance (L-2 norm) metric I find the 20 poses with a characteristic signature most similar to that of the query. This is illustrated in Figure 9 with the query image at the top left position.

2.8 Refining among the most likely candidates

The 20 most likely candidate poses are each rotated 40 times yielding a total of 800 potential matches. The approximate chamfer distances (down-sampled) to these candidate poses are computed and the best one is considered the optimal match. The rotation step improves the accuracy of the indexing procedure and it is intended to make up for the reduced size of the database. Figure 9 illustrates several rotation variants of a candidate pose.

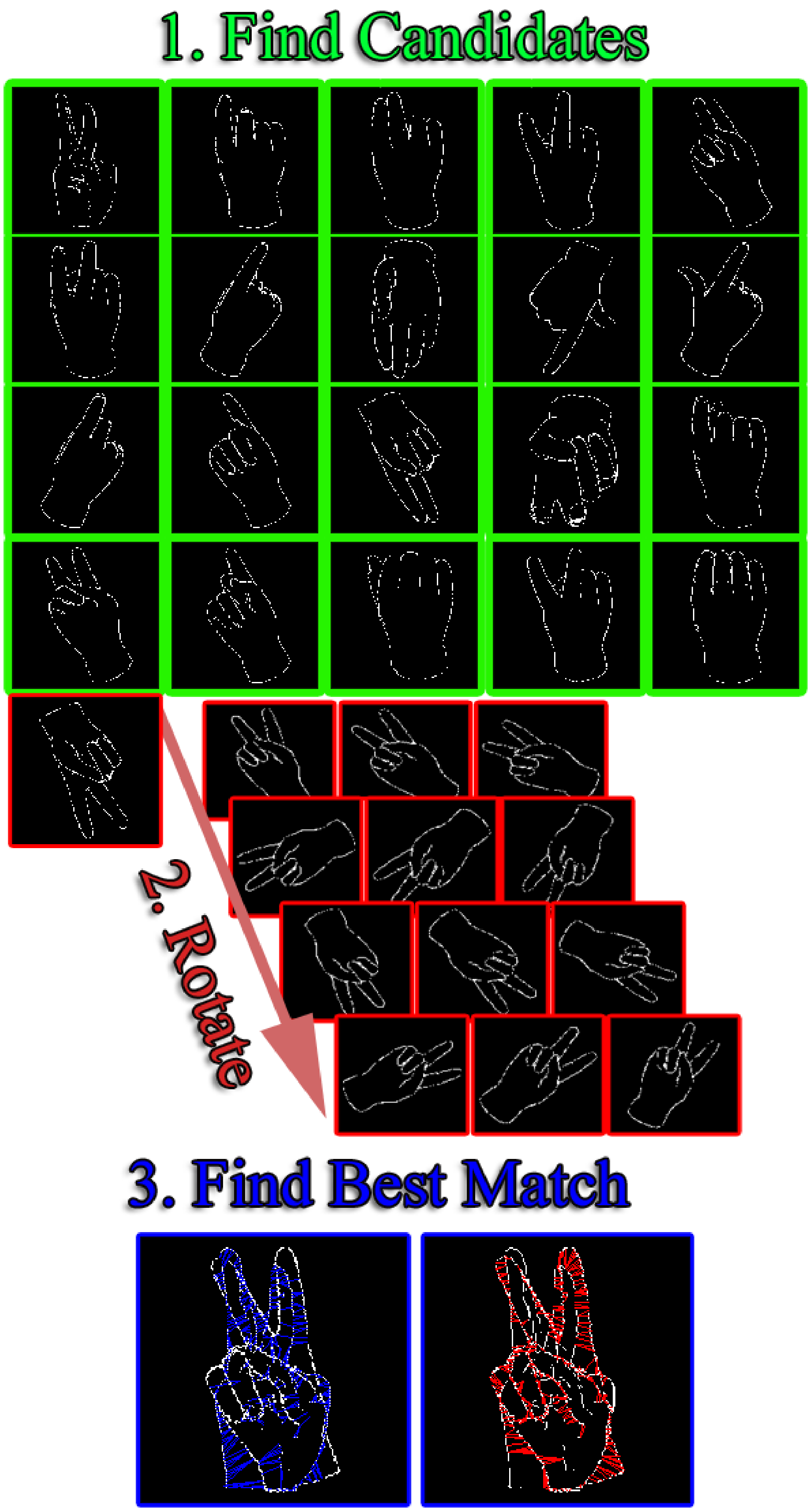
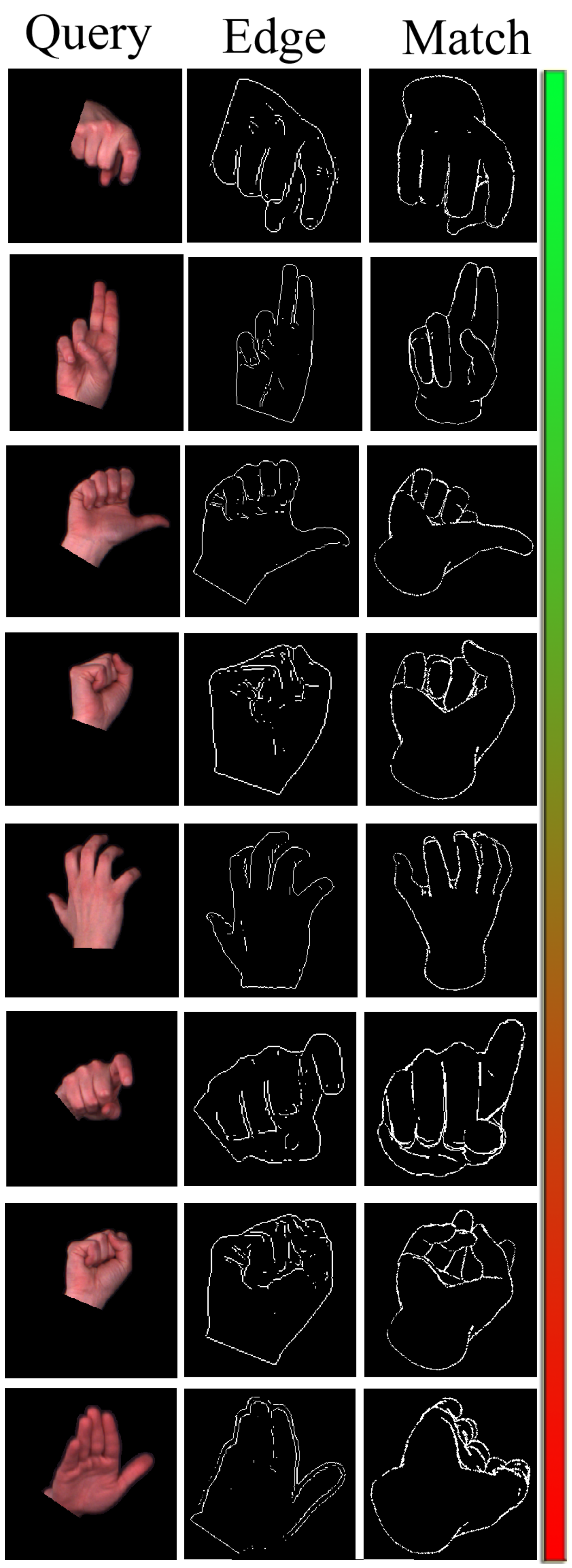


Figure 9: The steps involved in finding an optimal match to a given query.



3 Results and Discussion

In general, the accuracy of the matching procedure depends on several factors. Of primary concern is the number of original poses which were used to create the database. In this study I rely on a synthetic set of hand images which provide fairly low coverage of the parameter space of all possible poses. Despite this limitation reliable results are obtained for pose queries that are represented in the database. The quality of the result is based on a subjective judgment since the test data was unlabeled and there there was no absolute measure of accuracy. In Figure 10 queries and the best corresponding matches in the database are presented in descending order of subjective accuracy.

Perhaps more telling than the examples that were correctly matched, are queries that fell short of finding a good pairing (lower half of Figure 10). At a high level we see the these poor matches have local structural similarity that drives the chamfer cost down. This might be alleviated if we rely on deformation analysis methods (e.g. thin plate splines) that compare bending energies between two shapes to compute the signature vectors. The use of such methods is restricted given their computational cost, but since thin plate splines are resistant to affine transformations the database can be reduced (no-rotations are necessary).

4 Future Directions

In a way, the visual hand tracking project is part of a larger solution which would integrate the presented method with a dynamical motion capture tracking platform to power a generative model estimator of the hand pose. This hybrid system will be targeted to understand the interface between the hand and an object in natural manipulation tasks. In such a context, the embedded database matching will most likely serve as a refinement step over the estimations produced by a motion capture tracking system.

To advance along this track, I plan to create a new database with more natural poses and record all the relevant joint angles using a CyberGlove. Once a preliminary base set of hand pose parameters are recorded I can use a rendering tool to synthetically generate images from multiple viewpoints and rotations (while keeping the pose parameters updated via the same transformations applied to the image). Accessing the joint parameters for each pose will be essential for matching vision based estimates to the results of the motion capture tracker. By combining the vision and motion capture systems I hope to be able to develop a tracking system that outperforms the best available methods.

The current database indexing implementation is performed in MATLAB, and this significantly increases the running time of the queries relative to numbers reported in Athitsos [1]. In the future, I plan to approximated chamfer distances via the Borgefors algorithm [4] which relies on pre-computed distance transforms (using a 3x3 convolution matrix) for each image in the database. An efficient implementation of this algorithm is available in the OpenCV library and I plan to leverage this optimized system and write a distributed version of the database indexing method to exploit the parallel architecture of a 128 core GeForce 8800 video card. This will be particularly relevant given that multiple high speed cameras will be used in the object tracking task and a steady stream of hand images will be constantly available from multiple perspectives.

References:

[1] Vassilis Athitsos and Stan Sclaroff. Database Indexing Methods for 3D Hand Pose Estimation. Gesture Workshop, pages 288-299, 2003.

[2] Thomas B. Moeslund and Erik Granum. A Survey of Advances in Vision-Based Human Motion Capture and Analysis, Computer Vision and Image Understanding, Volume 81, Issue 3, Pages 231-268, 2006

[3] Vassilis Athitsos, Marios Hadjieleftheriou, George Kollios, and Stan Sclaroff. Query-Sensitive Embeddings, TODS 2007

[4] Gunilla Borgefors. Distance transformations in digital images. Computer Vision, Graphics, and Image Processing 34(3): 344-371, 1986

[5] Erik B. Sudderth , Michael I. Mandel , William T. Freeman , Alan S. Willsky, Visual Hand Tracking Using Nonparametric Belief Propagation, Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 12, p.189, 2004

[6] Stenger, B.D.R., Thayananthan, A., Torr, P.H.S., Cipolla, R., Model-Based Hand Tracking Using a Hierarchical Bayesian Filter, PAMI(28), No. 9, pp. 1372-1384, September 2006

[8] N. Shimada, K. Kimura, and Y. Shirai. Real-time 3-D hand posture estimation based on 2-D appearance retrieval using monocular camera. In Recognition, Analysis and Tracking of Faces and Gestures in Realtime Systems, pages 23–30, 2001.

[9] Todorov E (2007) Probabilistic inference of multi-joint movements, skeletal parameters and marker attachments from diverse sensor data. IEEE Transactions on Biomedical Engineering, 54: 1927-1939

[10] S. Belongie et al., Matching shapes, in: International Conference on Computer Vision, Vancouver, Canada, July 9-12, 2001.