Markerless 3D Hand Posture Estimation from Monocular Video by Two-level Searching

Iek-Kuong Pun, I-Chen Lin, Tsung-Hsien Tang
Department of Computer Science,
National Chiao Tung University, Hsinchu, Taiwan
Email:louispun.csie94@nctu.edu.tw, ichenlin@cs.nctu.edu.tw, te13579@gmail.com

***Abstract*—In this paper, a markerless 3D hand tracking system for monocular RGB video is presented. We propose a novel two-level approach to efficiently grasp the personal characteristics and high varieties of hand postures. Our system first searches the approximate nearest neighbors in a small personalized real-hand image set, and retrieves more details from a large synthetic 3D hand posture database. Temporal consistency property is also utilized for disambiguating and noise reduction. Our prototype system can approximate hand poses including rigid and non-rigid out-of-image-plane rotation, slow and fast gesture changing during rotation. It can also recover from a short-term missing hand situation in an interactive rate.**

Keywords-hand tracking; 3D gesture; advanced interface

I. Introduction

In recent years, significant revolutions in graphics input and human-computer interaction have occurred. Various new consumer-level devices are produced to capture the motion of human bodies and hand gestures, such as data glove, Microsoft Kinect [1], or ASUS Xtion Pro [2]. However, these devices require additional specific sensors, e.g. infrared cameras or projectors. On the other hand, visible-light cameras are now essential components in most modern mobile and desktop equipment. Therefore, it can have more pervasive usages to track the human hand gestures from a monocular camera.

Based on a single-view camera, markerless 3D reconstruction of hand poses is a highly difficult problem. First, a human hand is an articulated object with more than 20 degrees of freedom (DOFs). Without sufficient prior information or constraints, it is difficult to estimate the optimal postures from such high dimensional space in real time. Second, our hands are composed of multiple articulated parts, but textures and colors of different articulated parts are similar. Furthermore, since the captured images results from perspective projection, the shapes in a camera view are of large variety, and with considerable self-occlusions. Invariant tracking features [3], [4] that are commonly used in computer vision are not fit for this problem. Resembling the full body motion, motion of the fingers is non-linear and its high variety makes it difficult to be well approximated by a simplified model. As addressed by A. Erol *et al*. [5], the hand pose changes fast and abrupt with a speed reaching up to 5m/s for translation and 300°/s for wrist rotation. It combines with non-linear motion, introduces extra difficulties for tracking algorithms, especially for temporal-filter-based methods.

In this paper, we aim at tracking an articulated hand without using markers. Our motion model includes 20 DOFs for the joint angles and 6 DOFs for global hand orientation and location. To deal with the serious self-occlusion and high-dimensionality approximation, we propose a data-driven technique using multiple features, hierarchical approximate nearest neighbor(ANN) search and Bayesian-filtering-based pose reconstruction to efficiently estimate arbitrary hand motion.

Instead of using the markers or "color glove" in related articles, we choose to capture a small real hand image database for a user as a training process. This small database records user specific information, and is helpful of rapid initial approximation. We further use a large 3D posture database to deal with the high variety of detailed motions. The initial estimated posture in the small dataset can be further refined with the detailed data searching. For robust searching in the high dimensional samples, we choose non-Euclidean distance measures such as chamfer distance.

Our experiments demonstrate that the proposed prototype system can estimate hand poses in an interactive rate and can provide an intuitive and advanced human computer interface.

II. Related Work

This section gives a short overview of general optic tracking and estimating the 3-D position and joint configuration of hands. From a methodological point of view, the work on optical hand estimation could be divided into two groups: *model-based tracking* and *indexing-based posture estimation*. From the viewpoint of feature selection, it could be divided into *bare-hand tracking* and *marker-based tracking*. These decades, hand tracking continues to be an active research area. Many pioneer or classic papers can be found in the review by A. Erol *et al*. [5]. Since our system takes bare-hand inputs and utilizes synthetic models and image data, the following subsections focus on the state-of-the-arts about model-based and indexing-based methods for a

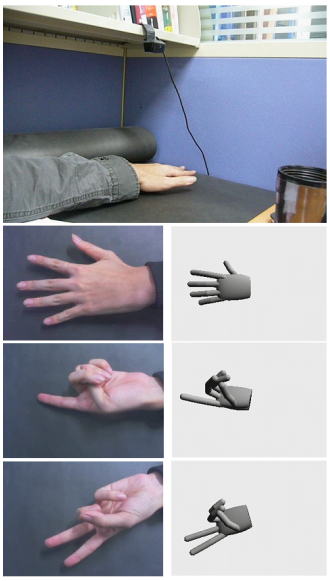

Figure 1. The proposed interactive-rate method for 3D hand motion estimation. Top: the experiment environment; the left and right columns are the input real images and the estimated hand articulations, respectively.

markerless hand.

A. *General object tracking*

To track a nearly rigid-body object moving in 2D image sequences, several algorithms have been proposed. For example, Mean-shift tracking [6] is an iterative localization algorithms based on the maximization of a similarity measure. However, a hand consists of multiple articulated components, and the hand motions include not only global palm translation and orientation, but also local articulated rotations.

Filtering-based methods incorporate prior assumptions about the object dynamics and hypotheses. For linear functions subjected to Gaussian noise, Kalman filter is one of the popular filtering algorithms. Particle filtering [7] utilizes sequential Bayesian filters with Monte Carlo simulations for non-linear and non-Gaussian motion. But, it requires a number of sampling points. For a high DOF non-linear motion, the sampling region and transition prior is often difficult to be determined. To improve the tracking result, it may have to use considerable numbers of sample points with a high video capture rate.

B. *Model-based Hand Tracking*

Model-based approaches use an articulated 3D hand model for tracking. At each frame of the image sequence, model-based methods search within the configuration space to find the best parameters that minimize the differences between the projected hand model and the image frame. To quickly search in configuration space, these systems usually assume that the configuration at the previous frame is known. Therefore, manual initialization procedure is usually required in the first frame.

B. Stenger *et al*. [8] used a model based on generalized cylinders and presented tracking based on unscented Kalman filter. Then, they presented another method [9] based on hierarchical Bayesian filter. In this method, a large number of templates are evenly generated from their cylinder based model. Then, they used chamfer distance for matching. Their hierarchical filtering improved the computational performance and successfully tracked restricted rigid motion and low degrees-of-freedom articulated motion but still far from real-time and high DOF motions.

Martin de La Gorce *et al*. [10] built a delicate model that incorporated a polygonal mesh to accurately fit shape and synthesized the hand projection with texture. The illumination was dynamically estimated through shape-from-shading techniques. The model provides state-of-the-art pose estimate on complicated background, high DOFs and occlusion sequence. However, their method is also complicated and takes considerable time on synthesis of novel projection images. The above-mentioned methods showed that the synthetic data and uniform sampling templates can handle complicated situations but they cannot reach real-time or interactive performance with off-the-shelf hardware.

Oikonomidis et al. [11] present a model-based method for tracking the full articulation of two interacting hands observed by an RGB-D sensor. They formulate two-hands tracking as an optimization problem in a 54-dimensional parameter space. To solve this problem, they use Particle Swarm optimization.

C. *Indexing-based Pose Estimation*

In the indexing-based pose estimation approach, a set of hand features is labeled with a particular hand pose, and a classifier is learnt from this training data. Recently, the boundaries between model-based and indexing-based methods are blurred. In several papers, training data are generated from 3D models but did not search over the entire configuration space.

Michalis and Vassilis [12] generated a large projected 3D hand image database and proposed an embedding-based and hash table-based indexing methods for hand shape recognition. Romero *et al*. [13] proposed a nearest neighbor search in a database with different grasp types, different viewpoints and different illuminations. They used time continuity enforcement in joint space to disambiguate the ambiguity. These methods can match every incoming image to the large number of database images at interactive times. However, the recognized gestures or motions of these methods are limited, which restrict their applications.

Wang and Popović [14] proposed using a color glove to improve the robustness of data matching. The color patches on the glove can further be regarded as alternative markers for pose refinement. Their database generated from a 3D

model contains 100,000 entries. They successfully track many commonly used hand gestures, sign language alphabet and random jiggling of the fingers. Moreover, their system can provide an interactive-time control for object interaction in 3D space. The high performance of color-glove tracking by searching inspires our bare-hand posture estimation. Our goal is to find an efficient and robust method to estimate 3D postures for a database with about 100,000 bare hand images.

III. OVERVIEW

Our goal is to track bare hand's 3D positions and motions from a single-view image sequence. It is difficult to directly match the user's hand shape with an approximate 3D hand model. On the other hand, capturing 100,000 real examples is intractable as well. In our work, we propose a hybrid method. First, we require a user to do a short and simple training data collection. We utilize the user-dependent and computer-labeled real hand image data set to find a few approximate nearest-neighbors (ANN) groups that are near the query image. Then, we use these groups as searching seeds, and perform our Bayesian-filtering-based pose reconstruction in a large database, generated from a 3D model, to further estimate the actual pose. Fig. 2 shows the flow chart of the proposed system.

In section IV, we describe the construction of the databases and a method of robust matching. At first, for both the construction of the database and processing the query image, we need to apply hand image segmentation to all images. We classify each pixel either as background or hand using Gaussian Mixture Models(GMM) trained from a set of hand-labeled images. For the matching function, the chamfer distance [15] is used to compute the similarity between the input hand image and database images. This method is a well-designed method to measure the distance between two edge images.

In section V, we propose our Approximate Nearest Neighbor(ANN) search for both two databases. For the small database, a standard Approximate Nearest Neighbor quick search method, Kd-tree is employed. Then, we apply the k-means method to derive the cluster of nearest-neighbors images according to their hand configurations(26 DOFs). After evaluating the configurations of approximate nearest neighbors, we can apply our Bayesian-filtering-based pose reconstruction on the large 3D posture database. Therefore, the proposed method can take advantage of personalized real images and the large pre-generated database. Our experiment results and further discussion are presented in the last two sections.

IV. HAND IMAGE RECOGNITION

The core of our approach is to efficiently search the most similar image sample from databases for a given bare-hand image. To accomplish this, an input image is first transformed into a normalized query, and then compared to entries in the database according to a robust distance metric.

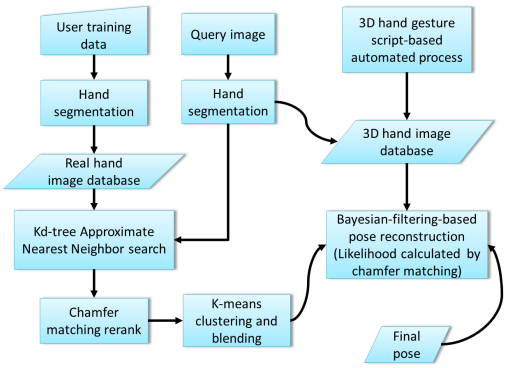

Figure 2. Flow chart of the proposed system. It can be divided into two groups: hand image recognition and fast bare-hand pose estimation.

A. *Image Segmentation and Normalization for Image Data*

From both the query sequences and training database, we classify each pixel either as the background or foreground by using a Gaussian mixture model(GMM) trained from a set of hand-labeled images. At first of GMM training, we capture a few hand images of a training subject's hand with the specified background. We manually label the region of the hand and pre-cluster each pixel either as hand pixel or background as the initial mixture distribution. We use Expectation-Maximization(EM) algorithm to estimate the parameters of the multivariate probability density function in the form of a Gaussian mixture distribution with K mixtures. To focus on the hand tracking, we use a black background to ease the segmentation. The segmentation process can be extended to a complex background with more training images or other segmentation methods.

For each pixel of query sequence and training data, we can now rapidly retrieve the hand pixels by GMM. Since there are still a few noises or outliers, we assume a user's hand is the biggest hand-like color object in the image, and find the biggest connected components of the hand pixels as the hand segmentation result. At last, we normalize the segmented hand image into a 64×64 tiny image as shown in Fig. 3.

B. *Gathering Database Samples*

Ideally, the database for pose estimation should be a large database, where samples are real hands with all possible hand configurations, palm translations and orientations. However, since hand configuration has 20 DOFs, it requires considerable computations to search a database including all the configurations. In our system, we aim at a natural way of human-computer interaction. Therefore, our 3D model database includes 50 typical hand gestures. These configurations span the sign language alphabet, common

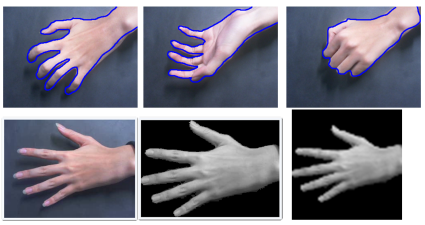

Figure 3. Examples of our image segmentation input and output. Top: 3 samples from our training images, the hand region is labeled manually. In the bottom, left: the raw query image; middle: segmented hand; right: normalized 64×64-pixel tiny image.

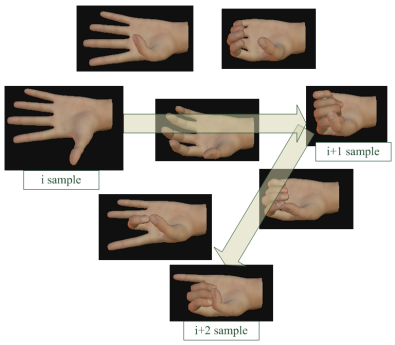

Figure 4. An example of the small database posture selection from the large synthetic database. We sequentially find the sample furthest to existing ones.

hand gestures, and random jiggling of these gestures. We used the graphics software Poser Pro 2010 [16] to generate synthetic images from various 3D palm rotations. Since we take a fixed camera view points, 3D rotations of the synthetic hand are limited within a hemisphere. In our experiment, the resolution (interval) of y (forearm axis) and z-axis (orthogonal to palm) rotation is 15 degrees, and the x-axis (wrist-axis) rotation is limited from -40 degrees to 40 degrees and with interval 20 degrees. These sampling intervals result in a total of $13 \times 13 \times 5 \times 50 = 42500$ images. It only has to be computed once (during the data gathering stage), and took a few hours to generate these synthetic images.

In our two-level method, we need to capture another small real hand image database for a user. To grasp the hand posture variety, the postures for this small dataset are selected by an iterative and greedy algorithm from the large synthetic database. We define a distance metric between two configurations as the root mean square (RMS) error of their joint angles. The selection procedure is to sequentially find a sample configuration that is furthest from any of the previous selected samples. The selected configurations can cover the most dispersed configurations of the 3D model database as shown in Fig. 4. Since the z-rotation is parallel to image plane and can simply be generated by image rotation, in our implementation, we only take samples with the 0-degree z-rotation into the posture selection pool.

With the selected few samples configurations, a user only needs to pose these selected gestures with a few palm rotation samples at the user training stage. In front of the video camera, the user poses designated gestures and turns the hand along the y-axis and x-axis respectively. Then, we apply a simple video segmentation based on image RMS differences to estimate the real rotation angles. The z-axis rotation data are then generated by simple image rotation. Therefore, the overall training process is simple and short. Our small image database now spans the space of the 3D model database and can be used for quick first-level searching.

C. *The Chamfer Distance*

Given a normalized query image, we intend to extract the closest database images that to the query. In our system, we measure distance between edge images, because edge images tend to be more stable than color or gray intensity images with respect to different lighting conditions. We use the chamfer distance to compute the similarity between two edge images. Edge images are represented as sets of points, corresponding to edge pixel locations. Given two edge images, $\mathbf{X}$ and $\mathbf{Y}$, the chamfer distance $\mathbf{D}(\mathbf{X}, \mathbf{Y})$ is:

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

, where $\|\mathbf{x}-\mathbf{y}\|$ denotes the location distance between two pixel $\mathbf{x}$ and $\mathbf{y}$, $\mathbf{D}(\mathbf{X}, \mathbf{Y})$ penalizes points in either edge image that are far from any point in the other edge image. It can be computed efficiently by using a distance transform (DT) of the edge image. This transformation takes the set of edge pixels as input and assigns each location the distance to its nearest edge pixels. For example, the distance transform value at location $\mathbf{u}$ contains the value $\mathbf{min}_{\mathbf{y}\in\mathbf{Y}}\|\mathbf{u}-\mathbf{y}\|$. The chamfer distance for a pair edge images can be computed by correlating their edge points with their corresponding DT images.

Chamfer matching is a robust method for edge image matching. However, if we use brute-force search on a database comprising 100,000 entries, it takes a few seconds to match the input image for the database.

The computation is too intensive for an interactive application. Besides, in our cases, the common data-driven method, K-Nearest-Neighbors (KNN), is inherently ambiguous (one-to-many), since substantially different poses can give rise to the similar edge images. In the next chapter, we proposed our hierarchical and Bayesian-filtering-based hybrid method to efficiently disambiguate the KNN result.

(a)

(b)

Figure 5. Example of KNN ambiguity: (a) Top: query image, middle: from left to right, top 10 NN image after chamfer matching, bottom: top 10∼20, (b) using different query image. Fortunately, most of incorrect poses are far away from the approximate correct poses in configuration space.

V. FAST BARE-HAND POSE ESTIMATION

A. *Approximate Nearest Neighbor Search*

Estimating the exact nearest neighbors of high dimensions within limited time is a difficult task. Alternatively, we can use an approximation which is highly close but does not guarantee the nearest neighbor in every case, such as Kd-trees, locality sensitive hashing(LSH) and best bin first. However, most of these methods cannot be applied to an arbitrary non-Euclidean distance measure like chamfer distance. From our knowledge, no existing method can directly apply Kd-tree or LSH for the chamfer distance.

Since the simple Euclidean distance is less reliable as the distance metric between a real-hand image and a model-synthesized image. In contrast, using chamfer distance on a whole database is time-consuming for interactive applications. We use a hierarchical method to address these difficulties.

First, we use a small real-image database that evenly spans the space of the 3D model database as in subsection IV-B. We apply a simple Kd-tree on this small database with Euclidean distance. In this step, a large number of nearest neighbors are quickly retrieved. To improve the matching, we re-evaluate the weight of the top K nearest neighbors (KNN) using the chamfer distance to approximate the exhaustive chamfer distance search. Our small real-image database includes around 5,000 images, we found that we can use the above pre-filtering to reduce the candidates for chamfer matching to fewer than 1,000 and still get satisfactory results.

One direct thought after finding the nearest postures is to blend these poses, and then use this blended pose as a distribution center to approximate the best pose likelihood distribution using a *Monte Carlo* sampling or uniform searching. However, since we match only the projected 2D images, the 3D rotations and configurations of these real hand images are only approximate, and the nearest postures are inherently ambiguous as shown in Fig. 5. The blended pose can be pulled away from all nearest neighbors. As described by B. Stenger [9], hierarchical searching, at higher levels partition may not yield accurate approximations to the true likelihood distribution, but are used to discard inadequate hypotheses. Therefore, we have to do more analysis and processing to these KNN results.

B. *Pose Clustering and Weight Blending*

To match the 3D model dataset, ideally, we would like to apply a full chamfer matching to all 3D model images that around all the real-image KNN results. However, it is inefficient when each search of one NN result consists of redundant samples with other searches. We observed that the 3D rotation and configurations of the true neighborhood poses are always close, and the misleading poses are far away from them. So we can apply the k-means method to further cluster the retrieved NN into groups. Since hand configuration has 20 DOFs and the rotation only has 3 DOFs. We also apply a Principal Component Analysis (PCA) to reduce the dimensions of joint configurations and keep the features balance between rotations and configurations.

After clustering, we can blend the poses within the same cluster to acquire a few independent and non-overlapping regions of hypothesis. Again, an ambiguous or misleading pose in the middle of two cluster centers can still affect the blending accuracy. We propose using the temporal smoothness for disambiguation. Let $Q_t = \left\{q^t_{1,1}, ..., q^t_{N,1}, ..., q^t_{N,M}\right\}$ be the set of joint angle configurations in time $\mathbf{t}$; $\mathbf{M}$ indicate the cluster index and each cluster has $\mathbf{N}$ poses. q^{t-1}represent the previous estimated pose. For each member of Q_t , we set their weights as a simple exponential distance function:

$$\omega_{n,m} = e^{-\frac{(q^t_{n,m} - q^{t-1})^2}{2\sigma^2}} \tag{2}$$

, where σ^2 is the variance of the distance from each entry pose $q^t_{n,m}$ to the previous estimated pose q^{t-1}.

For a cluster $\mathbf{m}$ which has $\mathbf{N}$ poses, let $q^t_{1,m}, ..., q^t_{N,m}$ be the members in this cluster, we normalize these weights as such that:

$$\sum_{n=1}^{N} \omega_{n,m} = 1 \tag{3}$$

And the blending pose of the cluster is computed as:

$$q^t_m = \sum_{n=1}^{N} \omega_{n,m} q^t_{n,m} \tag{4}$$

Based on the temporal information, we can reduce and disambiguate the problem addressed in subsection V-A.

While applying the estimation with only the real small database, we found it is still difficult to track quick and high-variety hand motion. Hence, our next step is to use the retrieved cluster to estimate more detailed postures in the higher-variety dataset.

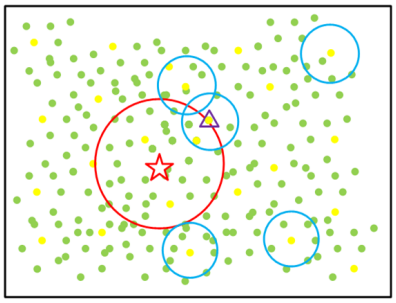

Figure 6. A 2D example of our data samples in the searching problem. Yellow points represent our real image data in the small database and green points represent the synthetic data. The red star represents the previous best pose; the triangle represents the current best pose; the red circle represents the neighbors of the previous frame; the blue small circles represent the high posterior regions by ANN search on the real image database, in most situations, the regions of the red circle is difficult to be determined.

C. *3D Model Database sampling and Hand Pose Reconstruction*

Now, we have a few clusters with corresponding high-probability regions in the model database. In this subsection, we introduce our pose reconstruction method based on these candidate regions. In general, our tracking can formulate as a Bayesian inference problem. Given the configuration at time $\mathbf{t}$ is represented as $\mathbf{q_t}$ and the observation is $\mathbf{z_{1:t}}$, the state estimation probability $p(\mathbf{q_t}|\mathbf{z_{1:t}})$ with the following Bayesian formulation:

$$p(\mathbf{q_t}|\mathbf{z_{1:t}}) \propto p(\mathbf{z_t}|\mathbf{q_t}) \int p(\mathbf{q_t}|\mathbf{q_{t-1}})p(\mathbf{q_{t-1}}|\mathbf{z_{1:t-1}})d\mathbf{q_{t-1}} \quad (5)$$

, where $p(\mathbf{z_t}|\mathbf{q_t})$ denotes the likelihood function that relates observations $\mathbf{z_t}$ in the image to the unknown state $\mathbf{q_t}$, and $p(\mathbf{q_t}|\mathbf{q_{t-1}})$ represents the transition prior that estimate by hand motion dynamics model based on the previous state. Therefore, the best hand configuration can be approximated by the Maximum a Posteriori (MAP) estimate over the $\mathbf{N}$ number of samples at each time $\mathbf{t}$.

Full searching or casually sampling can result in an unexpected result. In many temporal-filtering-based methods such as particle filters, hand motion dynamics is approximated by linear models like Gaussian model, and transition prior is often used as an importance function. Since hand motion is non-linear, linear transition prior can speed up the searching but also restrict the sampling regions. Therefore, we still use a Gaussian motion model, but the importance function is based on the high posterior regions by ANN search on the real image database, as shown in Fig.6.

The same as in real database, the likelihood function here is based on the chamfer distance, and we use silhouette as the applied features in this stage. To reconstruct the final hand pose, similar to the particle filter method, we approximate the distribution $p(\mathbf{q_t}|\mathbf{z_{1:t}})$ by a weighted set of samples $\mathbf{Q}$, with index $\mathbf{L} = \mathbf{1}, ..., \mathbf{P}$. We draw and weight samples based on the importance function:

$$W_L = \frac{p(\mathbf{z_t}|\mathbf{q_t})p(\mathbf{q_t}|\mathbf{q_{t-1}})}{p(\mathbf{q_t}|\mathbf{r_t})} \quad (6)$$

, where $\mathbf{r}$ denotes the observation in real database, $p(\mathbf{q_t}|\mathbf{r_t})$ represents the proposal distribution based on ANN search, and W denotes the importance weights. We normalize these weights such that $\sum \omega_L = 1$, and then we can use the weight blending in subsection V-B to estimate a final pose efficiently.

Due to the noise effects, many samples of small weights disturb the estimated configurations. The directly blended pose (weighted-blend samples) is still with obvious jitter. Therefore, we adopt two strategies to refine the motion. We omit samples with insufficient weights, and include the temporal smoothness term for configuration estimation. We formulate the motion reconstruction as an energy minimization problem, where the joint angles and global orientations are the variables. A data prior term enforces plausible reconstruction results and a smoothness term measures the smoothness of the synthesized motion:

$$q^* = arg_q min(\omega_{pri} E_{pri}(q) + \omega_{smooth} E_{smooth}(q)) \quad (7)$$

,where the two weights ω_{pri} and ω_{smooth} are user-defined constants.

For a set of poses $Q_L^t = q_1^t, ..., q_p^t$ with corresponding weights W_L^t, we assume the poses in the local region are of less variety and can be approximated by a kernel function. We use a kernel based approach proposed by [17], the data prior term E_{pri} .

$$E_{pri}(q) = \sum_{L=1}^{P} W_L^t K(\left|q_L^t - q\right|) \quad (8)$$

where $K()$ is kernel function. As Tautges described [17], a kernel based representation is well suited to approximate arbitrarily shaped probability density functions. For smoothness term, we assume that the pose at time t depends on the poses at time $t-1$ and $t-2$, and the smoothness term is:

$$E_{smooth}(q) = K(\left|q - q^{t-1*}\right|) \quad (9)$$

, where q^{t-1*} are the best poses in the previous two frames. We initialize the optimization with the weight blending pose and optimize the arguments using the Levenberg-Marquardt algorithm [18].

VI. EXPERIMENTS AND RESULTS

Our experiments perform on a desktop with Intel ® Core™ i5-760 Processor and 4GB main memory. In our experiments, the test sequences were captured from a single camera at 10 frames per second and the testing user is the same as the training subject of the small image database.

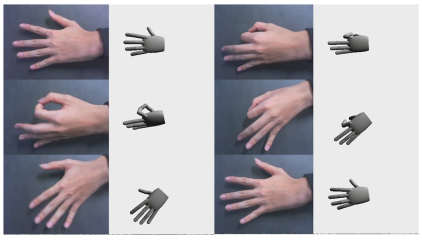

Figure 7. The sequence ok_shoot

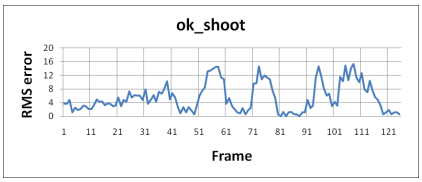

Figure 8. The RMS error (mm) of sequence ok_shoot

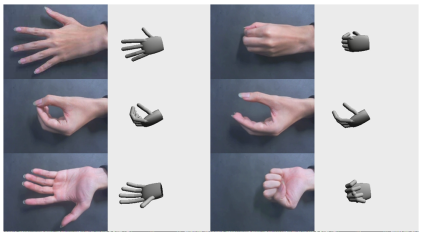

Figure 9. The sequence grasp_and_rotation

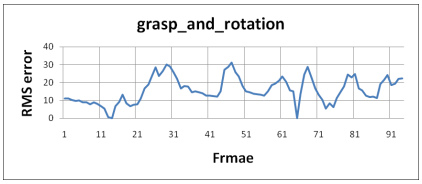

Figure 10. The RMS error (mm) of sequence grasp_and_rotation.

However, the variations of gestures, orientations, and translations are significantly different from the training sequence. Please refer to our demo video for details.

There are five test sequences that include rigid and non-rigid out-of-image-plane rotation, slow and fast gesture charging when rotation, and recover after the hand left the camera. For all of these testing sequences, we use the same database and same parameters without specific tuning.

To measure the accuracy of our approach, we perform the evaluation by applying Root Mean Square (RMS) to finger end point 2D positions, since it is difficult to acquire the ground truth 3D data from a single-view sequence. We manually label ground truth locations of the tip of the middle finger, and calculate the root mean square error for two sequences.

In experiments, we show that our system succeeds tracking most of these motions. And the system can perform in an interactive rate, which are around 10 frames per second (FPS). Fig. 7 shows the results from a sequence that includes two gestures and rotation. Fig. 10 shows the results from a sequence that includes a grasp motion and rotation. Fig. 8 and Fig. 10 show their errors and the mean RMS error is 5.4 and 15.7 mm, respectively. It can be observed that when the rotation and gesture change happen, the error increase but do not cause tracking to fail. Please refer to our demo video to observe the detailed hand motion estimation.

In Fig.11, we show the limitation of our bare-hand features. In the 2nd image of the right column, the hand shake and a small configuration error occurs. And at the following two images, we observed that that the ”2” gesture cannot change to ”1” gesture. It is because we have 2 similar gestures labelled ”H” and ”R” in our database, the noise-disturbed edge and silhouette features are not sufficient to measure the differences in this case.

Finally, we compare the accuracy between our hybrid method and full KNN search on real image database and synthetic database. We apply full KNN search on ”grasp_and_rotation” sequence. Fig. 12 shows the result. If we use only the real image database, the distribution of the database is too dispersed, and many inappropriate poses result in an unsatisfactory blending. If we use only the synthetic database, the direct comparing the appearance between the real and the synthetic model hand results in biased postures.

Here, we discuss the limitations of our system. Since we utilize only a single camera, one weakness of our system is the accuracy of z-translation. As shown in the video, jitter occurs during hand moving up and down. That is because we can only estimate the hand depth from relative scales, but the scale evaluation is easily contaminated by noises or imperfect hand region segmentation. Besides the movement jitter, we utilize the smooth term and temporal coherency to alleviate outlier gestures. It makes the estimated motion more stable, but may delay the response to user's rapid flip actions around a half second.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we present an approach to tracking an articulated hand without markers in nearly real time. We use hierarchical-searching to efficiently find the KNN results in a small real image and a large synthetic hand database. We further use the bayesian-filtering and temporal smoothness to disambiguate the results and estimate more reliable hand gestures.

Since our target is a highly ambiguous problem in a limited view, our system may not evaluate highly accurate configurations, such as those in marker-based systems. However, our experiments show that we successfully estimate the 3-D position and joint configuration of the hand under self-occlusion or rapid gesture chances. It can even recover from

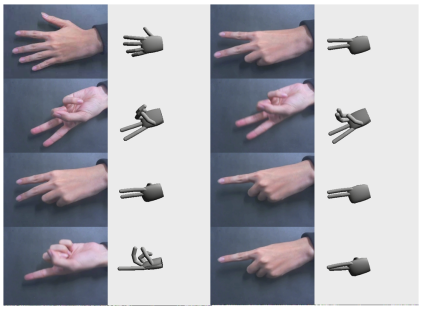

Figure 11. The sequence pose_occlusion

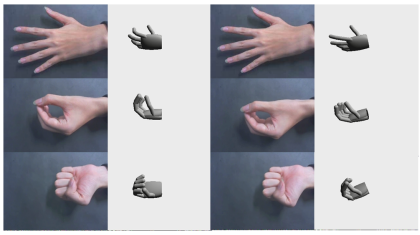

Figure 12. Comparison with full KNN search. **Left:** on real image database only. **Right:** on synthetic database only.

a short-term missing data situation. We think this prototype system can be extended for a user-friendly human computer interface.

A few possible extensions can be developed for our system. The most critical issue is to use more robust similarity measures besides the edge and silhouette features. As Martin de La Gorce *et al.*'s described [10], texture and shading is a crucial visual cue for hand. Including textures or shading information can increase the distinction but the computation cost as well. Second, our prototype system needs to capture real-image database for a novel user. User-independent methods can be helpful to skip the training process.

ACKNOWLEDGMENT

This project is partially supported by National Science Council, Taiwan under grant no. NSC 101-2221-E-009-154 and NSC 102-2218-E-009-003

REFERENCES

[1] Microsoft, "Kinect," http://www.xbox.com/kinect.

[2] ASUS, "Xtion pro," http://www.asus.com/Multimedia/Xtion_PRO.

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Intl. Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[5] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Comput. Vis. Image Underst.*, vol. 108, no. 1-2, pp. 52–73, Oct. 2007.

[6] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–575, May 2003.

[7] A. Doucet, N. D. Freitas, and N. Gordon, "Sequential monte carlo methods in practice," *Springer*, 2001.

[8] B. Stenger, P. R. S. Mendonca, and R. Cipolla, "Model-based hand tracking using an unscented kalman filter," in *In Proc. British Machine Vision Conference, volume I*, 2001, pp. 63–72.

[9] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Model-based hand tracking using a hierarchical bayesian filter," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1372–1384, Sep. 2006.

[10] M. de La Gorce, D. J. Fleet, and N. Paragios, "Model-based 3d hand pose estimation from monocular video," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 9, pp. 1793–1805, sep 2011.

[11] I. Oikonomidis, N. Kyriazis, and A. Argyros, ""Tracking the articulated motion of two strongly interacting hands"," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 1862–1869.

[12] M. Potamias and V. Athitsos, "Nearest neighbor search methods for handshape recognition," in *Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments*, ser. PETRA '08, 2008, pp. 30:1–30:8.

[13] J. Romero, H. Kjellstrm, and D. Kragic, "Monocular real-time 3d articulated hand pose estimation," in *IEEE-RAS International Conference on Humanoid Robots*, 2009, pp. 87–92.

[14] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM Transactions on Graphics*, vol. 28, no. 3, 2009.

[15] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf, "Parametric correspondence and chamfer matching: two new techniques for image matching," in *Proceedings of the 5th international joint conference on Artificial intelligence - Volume 2*, 1977, pp. 659–663.

[16] Smithmicro Software , "Poser pro 2010," http://poser.smithmicro.com/poserpro.html.

[17] J. Tautges, A. Zinke, B. Krüger, J. Baumann, A. Weber, T. Helten, M. Müller, H.-P. Seidel, and B. Eberhardt, "Motion reconstruction using sparse accelerometer data," *ACM Trans. Graph.*, vol. 30, no. 3, pp. 18:1–18:12, may 2011.

[18] M. Lourakis, "Levmar," 2004, http://www.ics.forth.gr/ lourakis/levmar/.