國立交通大學

資訊科學與工程研究所

碩士論文

使用有限的訓練資料與分層式搜尋之
手部動作追蹤技術

Hierarchical-searching-based hand tracking with limited training data

研 究 生：潘亦廣

指導教授：林奕成 博士

中華民國一百年九月

使用有限的訓練資料與分層式搜尋之手部動作追蹤技術

Hierarchical-searching-based hand tracking with limited training data

研究生： 潘亦廣　　Student： Iek-Kuong Pun
指導教授： 林奕成教授　　Advisor： Dr. I-Chen Lin

國立交通大學
資訊科學與工程研究所
碩士論文

A Thesis
Submitted to Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Computer Science

September 2011

Hsinchu, Taiwan, Republic of China

中華民國一百年九月

Hierarchical-searching-based hand tracking with limited training data

Student: Iek-Kuong Pun Advisor: Dr. I-Chen Lin

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

In this thesis, we consider tracking an articulated hand without using markers. Our hand tracking method performs nearest-neighbor-based search in a 3D hand model large database. For robustly and efficiently, we choose to capture a small real hand images database for each user as an intermediate dataset. And use Hierarchical-searching and temporal consistency to efficiently search in the large database and disambiguate the result. Our prototype system can estimate hand pose including rigid and non-rigid out-of-image-plane rotation, slow and fast gesture charging when rotation, and recover after the hand left the camera in real time. We believe it can be a more intuitive way for advance human computer interaction.

Keywords: hand tracking, interactive interfaces, Pose Estimation

Acknowledgement

首先一定要感謝我的指導教授林奕成博士,沒有架子的老師與我們的關係亦師亦友,細心指導與教導我們專業知識的同時,私底下亦是人生中一位不可缺少的朋友,也容忍了我們的懶惰與任性,我真心的感謝老師對我們的付出與關懷

另外也要感謝CAIG與GPL實驗室的學長學弟與同學們,在這裏的歡樂氣氛是其他實驗室所沒有的,你們的存在讓單調的研究生活增添了許多歡笑,我永遠都會記得我們一起熬夜趕作業,趕論文,辦會議(還有聖誕PARTY打電動打桌遊跟狩獵之類的)的時光

最後一定要感謝我的女朋友,容忍了我決定攻讀碩士的任性決定,對我不離不棄,讓我可以完成夢想的第一步

雖然結果不是十分完美,不過我相信兩年的碩士經歷豐富了我的思想與人生態度,希望以後可以活用所學,為人類社會作出貢獻

Content

Chapter 1

Introduction

In recent years, significant revolutions in graphics input and human-computer interaction have occurred. Various new consumer-level devices that can capture the motion of human body and hand gesture, such as data glove, the Microsoft Kinect and the Wii Remote. However, all of these new devices require specific equipment like markers, infrared cameras or projectors. These devices are difficult to supplant the conventional input device. On the other hand visible-light cameras become essential components in modern mobile equipment. Therefore, bare-hand vision-based tracking are still an attractive research topic. Moreover, tracking a non-rigid articulated object like human hand through normal single camera is one of the fundamental and important problems in computer vision and robot learning.

For the existing techniques, marker-less 3D reconstruction of hand pose based on a single image is an extremely difficult problem. First, the hand is an articulated object with more than 20 DOF. There are a large number of parameters to be estimated. That is difficult to operate in real-time with current consumer-level hardware. Second, the hand is an articulated object but the color of all parts is close to each other. Furthermore, since the views in cameras are projected results, the shapes in camera view are of large variety and with considerable self-occlusions. It is difficult to extract robust feature. Finally, resembling the full body motion; motion of the fingers is non-linear such that it is difficult to model the hand motion distribution. On the other hand, as the research by Ali Erol *et al*[EBN*07], the hand pose change fast and abrupt with a speed reaching up to 5 m/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 thesis, we consider tracking an articulated hand without using markers and focus on the motion estimated. Our motion model includes 20 degrees of freedom for the joint angles and 6 for orientation and location. To deal with the large self-occlusion and high dimensionality problem, we propose a data-driven technique using multiple feature and hierarchical approximate nearest neighbor search to efficiently approximate arbitrary hand motion distributions. As described by many color marker-based research, the bare-hand feature is insufficient to reliably constrain the hand pose. We choose to capture a small real hand image database for each user as a nature "color glove". And for the high dimensionality data searching, we use non-Euclidean distance measures such as chamfer distance for robustness. The majority of efficient database indexing methods are designed for Euclidean distance measures or metric distance measures. We propose using hierarchical method with multiple features to combine these database indexing methods and non-Euclidean distance measures matching. Our prototype system can estimate hand pose in an interactive rate and provides an intuitive interface for advanced human computer interaction.

Chapter 2

Related Work

This chapter gives a short overview of the current techniques in optical hand tracking that can estimate the 3-D position and joint configuration of the hand. From a methodological point of view, the work on optical hand tracking could be divided into two groups: *model-based tracking* and *single-frame pose estimation.* From the view point of feature selection, it could be divided into *bare-hand tracking* and *marker-based tracking.* Since hand tracking continues to be a very active research area, there are many earlier review papers have been published, for example: Ali Erol *et al.* [EBN*07]. In this thesis, we focus on some state-of-the-art on bare hand with both model-based and single frame that related to our method.

2.1 Video tracking

Video tracking is a process of locating a moving object in consecutive video frames. To track a rigid object moves on 2D image sequences, there are several algorithms have been proposed. For example: Mean-shift tracking [KH75] is an iterative localization algorithms based on the maximization of a similarity measure. However, hand is an articulated object and the motion of it can include 3D position and orientation. *Filtering* involves incorporating prior information about the object dynamics and hypotheses. With robust features, may allow the tracking of complex objects with high degree-of freedom. For linear functions subjected to Gaussian noise, Kalman filter is one of the best filtering algorithms. However, motion of the fingers is non-linear. Particle filtering [AFJ01] is for implementing recursive Bayesian filters using Monte Carlo simulations for non-linear and non-Gaussian motion. But the problem of it is the requirement on the number of samples. For a high DOF non-linear motion, the sampling region and transition prior is often difficult to be determined, may require a extremely large number of simples. Otherwise, it results in a poor outcome.

2.2 Model-based hand tracking

Model-based approaches use an articulated 3D hand model for tracking. At each frame of the image sequence, they project the model into the image and search in the configuration space to find the best parameters that minimize the error functions. In most cases it is assumed that the model configuration at the previous frame is known. So in the first frame, manual initialization procedure is required.

B. Stenger *et al.* [SMC01] used a model that based on generalized cylinders and presented a method based on unscented Kalman filter. Then, they presented another method [STTC06] based on hierarchical Bayesian filter. In this method, a large number of templates are uniformly generated from their cylinder based model. Then, they used chamfer distance for matching. Since the hierarchical filtering is efficient, they successfully tracked restricted rigid motion and low degrees-of-freedom articulated motion but still far from real-time and high DOF motion.

Martin de La Gorce *et al.* [GFP11] built a detailed generative model that incorporated a polygonal mesh to accurately model shape, synthesize the model projection on-line, the hand texture, and the illuminant are dynamically estimated through Shape from Shading. The model provides state-of-the-art pose estimate on complicated background, high DOF and occlusion sequence, but their method is also too complicated and takes too much execution time on synthesize novel model projection. These methods showed the on-line synthesis and uniform sampling can handle complicated situation but cannot reach real-time or interactive performance off-the-shelf hardware.

2.3 Single-frame pose estimation

In the single-frame 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 single-frame 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[PA08] generated a large image database included 80,640 images, i.e., 20 hand shapes × 84 viewpoints × 48 image plane rotations. All database images were generated using projection of 3D model. The authors proposed an embedding-based and hash table-based indexing methods for hand shape recognition. Javier *et al.* [RKK09] proposed a nearest neighbor search in a large database included 100,000 computer-generated images with different grasp types, different viewpoints and different illuminations. They used time continuity enforcement in joint space to disambiguate the ambiguity. All of these efficient methods can match every incoming image to the large number of database images at interactive times. However, most of them can only recognize a few gestures or motions, which limited their applications on human-computer interaction.

Is a database of 100,000 entries insufficient for tracking more motion for HCI? Robert and Jovan [WP09] proposed using a color glove to improve the robustness of matching function and use the color information to improve their pose estimate result by using inverse kinematics to penalize differences between the result of the pose estimate and the original image. However the database that they used is still in a size of 100,000 entries and it is generated from a 3D model. They successfully track many common hand gestures, sign language alphabet and random jiggling of the fingers. Therefore their system enables interactive-time control character and rigid bodies in 3D space. This color-glove tracking using example search inspire a way for our interactive posture estimation, our problem become if we can efficiently and correctly search a database of about 100,000 bare hand images, we can probably track a bare hand motion for HCI at interactive times.

Chapter 3

Overview

Our goal is to track bare hand’s 3D positions and gesture from a single-view image sequence. Using 3D hand model is difficult to directly match the user’s hand shape. Nevertheless, capturing 100,000 real examples is intractable. In our work, we propose a hybrid method. First, we utilize a small labeled real hand image database from easy, short and simple real user training, to find a few approximate nearest-neighbors groups that are near the query image. Then we apply our Bayesian-filtering-based pose reconstruction on these groups in the large database that is generated from 3D model to fully estimate the true pose.

In chapter 4, we describe the construction of the databases and a means 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 trained from a set of hand-labeled images. For the matching function, the chamfer distance [BTBW77] is used to compute the similarity between the input hand image and database images. This method is a well-known method to measure the distance between two edge images.

In chapter 5, we propose our Approximate Nearest Neighbor (ANN) search that using 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 the K-nearest neighbor images on their Corresponding 26 DOF hand configuration data. By discovering the approximate nearest neighbors, we can apply our Bayesian-filtering-based pose reconstruction on the large 3D model database, efficiently sample the closest entries in the large database not only from the temporal continuity but the observations.

Our experiment result and further discussion are presented in chapter 6 and 7.

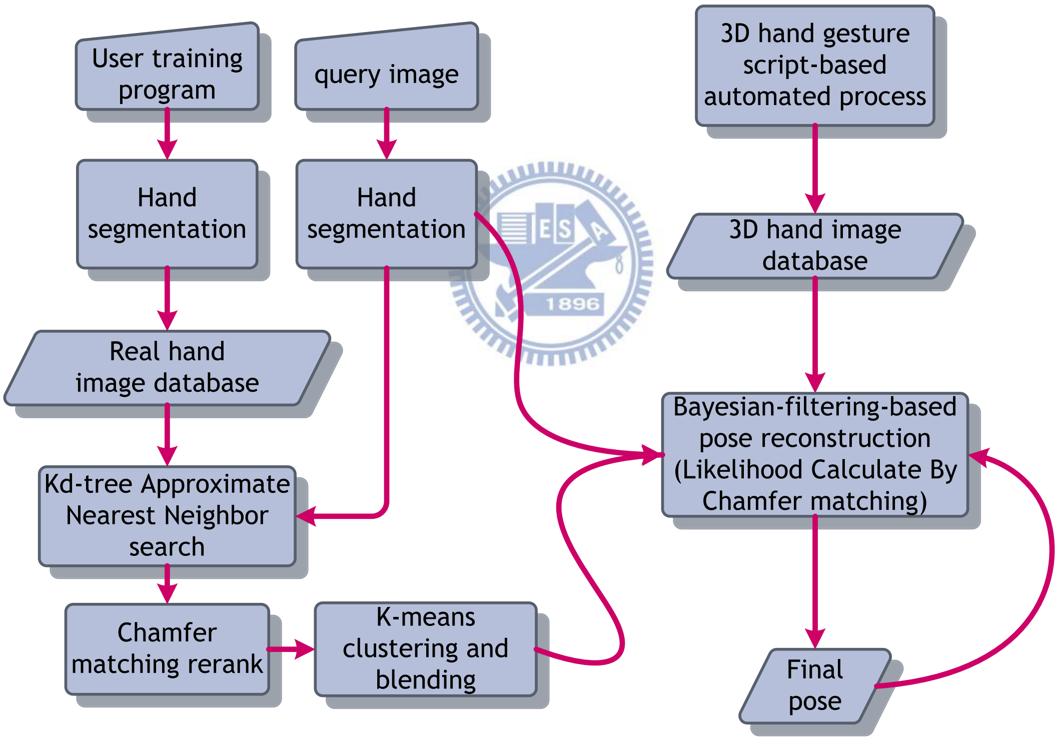

User training program
query image
3D hand gesture script-based automated process
Hand segmentation
Hand segmentation
3D hand image database
Real hand image database
Bayesian-filtering-based pose reconstruction (Likelihood Calculate By Chamfer matching)
Kd-tree Approximate Nearest Neighbor search
Chamfer matching rerank
K-means clustering and blending
Final pose

Figure 3.1 Flow chart of our system

Chapter 4

Hand image recognition

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

4.1 Image segmentation and normalization

From both the query sequence and database training data, we classify each pixel either as background or foreground by using a Gaussian mixture model trained from a set of hand-labeled images. At first, we capture a few hand images of training subject's hand with the specified background. Then we manually label the region of the hand and pre-cluster each pixel either as hand pixel or background as the initial mixture distribution. At this stage, 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.

Consider the set of **N** pixels $\boldsymbol{x_1}, \boldsymbol{x_2},\ldots,\boldsymbol{x_n}$ from HSV space drawn from a Gaussian mixture:

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

where **K** is the number of mixtures, $\boldsymbol{\mathcal{N}}$ is the normal distribution density with the mean μ_k and covariance matrix Σ_k, π_k is the weight of the k-th mixture. Given the number of mixtures **K** and the samples $\boldsymbol{x_1,\ x_{2,\dots,}x_n,}$ the EM algorithm finds the maximum-likelihood estimates (MLE) of the all these mixture parameters.

$$\max_{\pi_k,\mu_k,\Sigma_k} \sum_{i=1}^{N} \log \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

EM algorithm includes two steps: Expectation-step (E-step) and Maximization-step (M-step). At E-step, we find a probability of each sample to belong to mixture **k** using the currently available mixture parameter estimates. Then at the M-step, the mixture parameter estimates are refined using the computed probabilities. The algorithm is repeated these two steps until model parameters converge. To clarify our target, we use a simple black color background, but the method can be easily extended to complex background with more training images.

Now for each pixel of query sequence and database training data, we can efficiently cluster the hand pixels. Since there are still a few noises, we assume 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. Finally, we normalize the segmented hand image into a 64X64 tiny image. (See Figure 4.1)

4.2 Database sampling

Ideally, the database for pose estimation should be a large database that uniformly samples all natural hand configurations and all possible 3D orientations. Unfortunately, since hand configuration has 20 DOF, searching a database including all the configurations require extremely high computation cost. In our system, we aim at a natural way of human-computer interaction. Therefore, our 3D model database

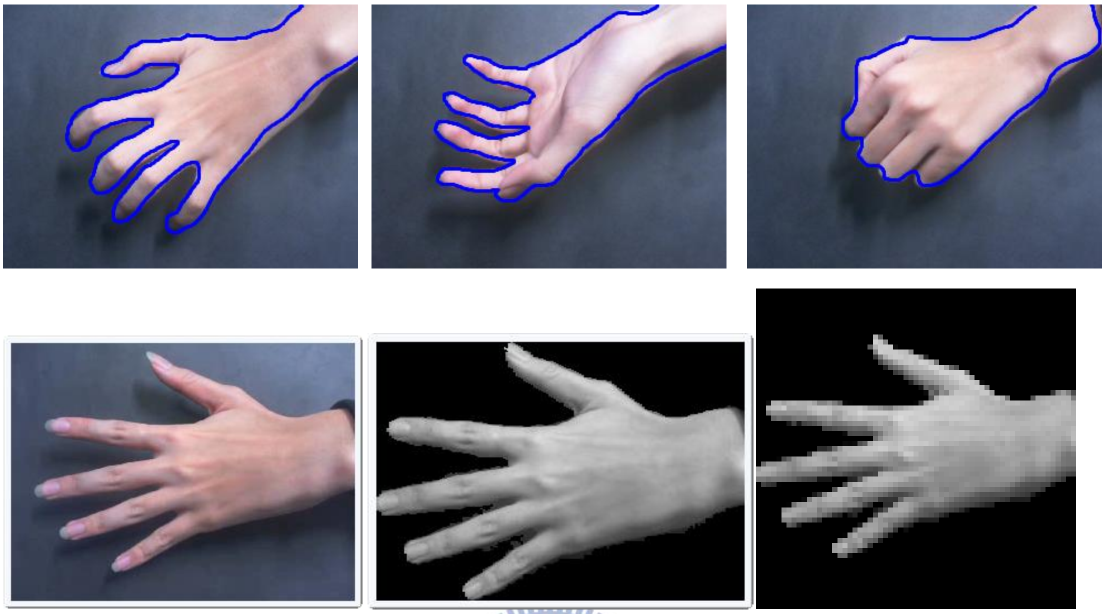


Figure 4.1 Examples of our image segmentation input and output.
Top: 3 samples from our training images, the region of the hand has labeled manually.
Bottom: left: the raw query image. Middle: segmented hand. Right: normalized 64X64 tiny image

includes 50 various hand shapes. These configurations span the sign language alphabet, common hand gestures, and random jiggling of these gestures. We used the graphics software Poser Pro 2010 [SS10] to generate synthetic images from different 3D rotations. Since the range of the hand Extension and the performance considerations, 3D rotations are limited to a hemisphere. In our experiment, the resolution of y and z-axis rotation is 15 degrees, and the x-axis Extension is limited to -40 degrees to 40 degrees and is 20 degrees, resulting in a total of 13 X 13 X 5 X 50 = 42500 images. It takes a few hours to generate these synthetic images. Image generation can be designed as a script-based automated process.

For our hybrid method, we require to capture another small real hand image database. We use an iterative and greedy sampling algorithm to select a few important configurations from the 3D model database. We define a distance metric between two configurations using the root mean square (RMS) error between the joint angles of the corresponding hand configurations. Then each sample configuration is selected to be furthest from any of the previous selected samples. The selected configurations can cover the most dispersed configurations of the 3D model database. (See Figure 4.2)

At the training state, a user only needs to pose these gestures with different x-axis rotations. We capture a video that the user turns the hand a hemisphere along the y-axis. Then we apply a simple video segmentation that based on image RMS to approximate the real y-axis rotations. Therefore, the training program can be short and simple. The result is a small real image database that efficiently covers the space of the 3D model database.

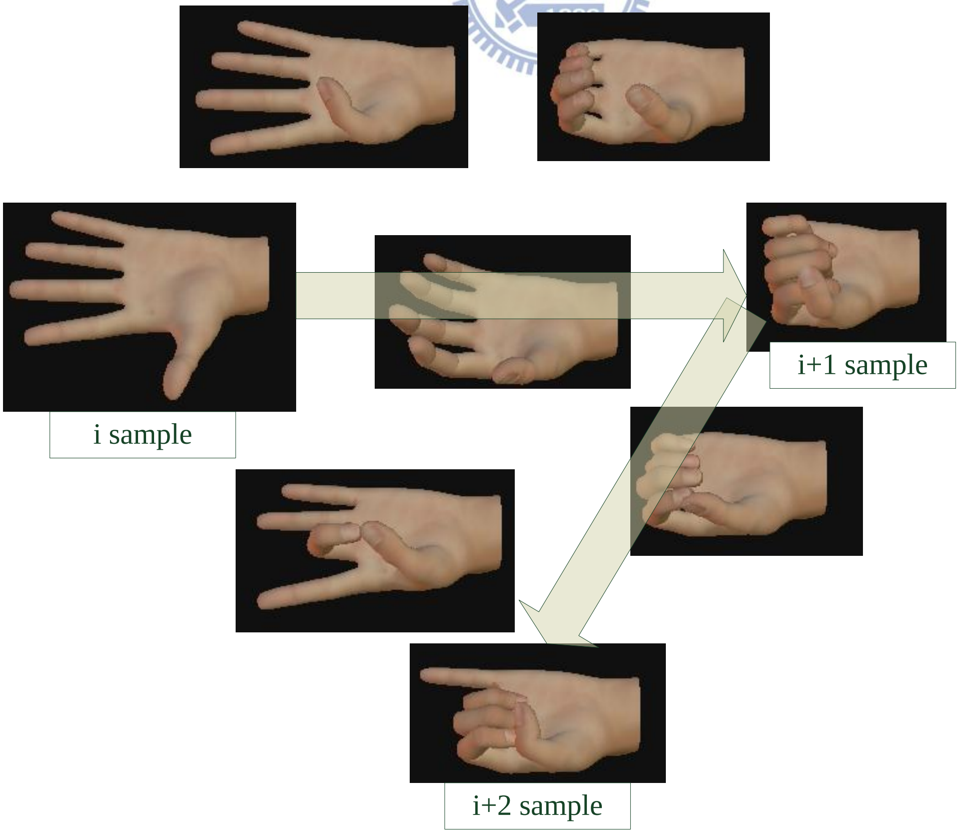


Figure 4.2 an example of the small database configurations selection, the number of

selection will affect the performance of our hybrid knn search.

4.3 The Chamfer distance

Now given a normalized query image, we extract the database images that are the closest to the query. In our system we measure distance between edge images, because edge images tend to be more stable than 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, ***X*** and ***Y***, the chamfer distance ***D(X, Y)*** is

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

where $\| \boldsymbol{x}-\boldsymbol{y} \|$ denotes the distance between two pixel locations $\boldsymbol{x}$ and ***y***, ***D(X, Y)*** penalizes for 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 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 $\boldsymbol{u}$ contains the value $\mathbf{min}_{\boldsymbol{y}\in\boldsymbol{Y}} \| \boldsymbol{u}-\boldsymbol{y} \|$**.** The chamfer distance for a pair edge images can be computed by correlating their edge points with their corresponding DT images. (See Figure 4.3)

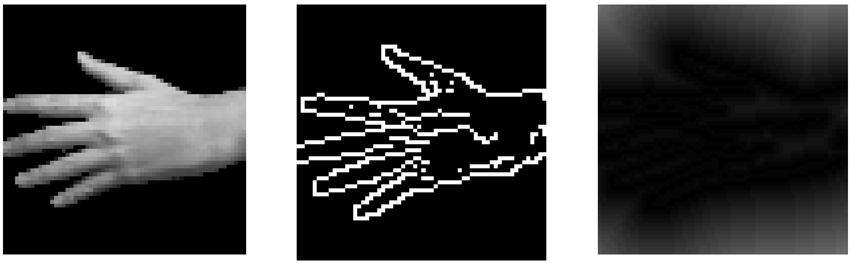


Figure 4.3 from left to right: normalized input image, edge image and DT images

Chamfer matching is a robust method for edge image matching. However, if we use brute force search on a database included 100,000 entries, it takes a few seconds to match the input image with the database is clearly too long for an interactive application. It can be shown empirically that KNN method is inherently ambiguous (one-to-many), substantially different poses can give rise to the similar edge image.

In the next chapter, we proposed our hierarchical and Bayesian-filtering-based hybrid method to efficiently disambiguate the KNN result.

Chapter 5

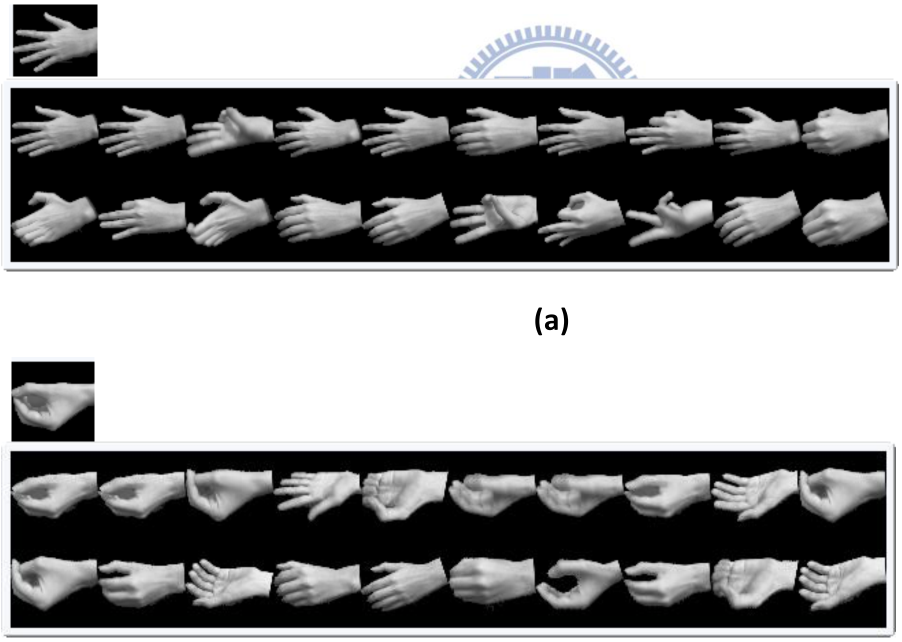
Fast bare-hand pose estimation

5.1 Approximate Nearest Neighbor Search

Computing the exact nearest neighbors in high dimensions in limited time is a difficult task. In those cases, we can use an algorithm which doesn't guarantee to return the actual nearest neighbor in every case, such as Kd-trees, Locality Sensitive Hashing and best bin first. However, most of these methods cannot be applied to an arbitrary non-Euclidean distance measure like chamfer distance. And there is no existing method that allows applying Kd-tree or LSH to the chamfer distance.

Since the simple Euclidean distance is insufficiently for the distance metric between a real hand image and a model image. In contrast, using chamfer distance on a small database is time-consuming to achieve human-computer interaction. We use a hierarchical method to address this contradiction. First, we simplify a small real-image database that uniformly covers the space of the 3D model database as we described in Section 4.2. We apply a simple Kd-tree on this small database with simple Euclidean distance. Then, we re-evaluate the weight of the top K-NN using the chamfer distance to approximate the exhaustive chamfer distance search. Since the size of the small database we used including less than 10,000 images, we observed that we can only apply chamfer matching on less than 1,000 nearest neighbor to get a sufficiency result.

Although we can blend these nearest neighbor poses now, 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 the 3D rotations and configurations of these real hand images are only approximate, and the result of KNN is inherently ambiguous. (See Figure 5.1) The blending pose can be pulled away from all nearest neighbors. In contrast, as B. Stenger[STTC06] described, hierarchical searching, at higher levels partition may not yield accurate approximations to the true likelihood distribution, but are used to discard inadequate hypotheses. More analysis and processing to the KNN result are required.



(a)

(b)

Figure 5.1 Example of KNN ambiguous: (a) Top: query image, middle: from left to right, top 10 NN image after chamfer matching, bottom: top 20, (b) using different query image. Fortunately, most of incorrect poses are far away from the approximate correct poses

5.2 Pose Clustering and Weight Blending

Ideally, we would like to apply a full chamfer matching to all 3D model images that around all the real KNN result states. However, it is inefficient when a search consists of many redundant samples, so called over-complete. We observed that the 3D rotation and configurations of the true neighborhood poses are always close, and the ambiguous poses are far away from them. So we can apply the k-means method to derive the cluster of each pose. Since hand configuration has 20 DOF and the rotation only has 3 DOF. We apply a simple Principal Component Analysis (PCA) to reduce the dimensional of configurations, to keep the cluster balance between rotation and configuration.

After clustering, we can blend the poses that are in the same cluster to acquire a few independent and non-overlapping regions of hypothesis. Again, an ambiguous pose in the middle of two clusters can still affect the blending accuracy. We propose using the temporal smoothness to disambiguate. Let $Q_t = \{q^t_{1,1}, \dots, q^t_{N,1}, \dots, q^t_{N,M}\}$ be the set of joint angle configurations in time ***t*** which has ***M*** cluster and each cluster has ***N*** pose, q^{t-1} be the previous estimated pose. For each member of Q_t, we set their weights as a simple Exponential function:

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

, 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 ***m*** which has ***N*** poses, let $q^t_{1,m}, \dots, q^t_{N,m}$ be the members this cluster, we normalize these weights independently such that:

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

And the blending pose of the cluster is computed as:

$$\mathrm{q}_{\mathrm{m}}^{\mathrm{t}} = \sum_{\mathrm{n}=1}^{\mathrm{N}} \omega_{\mathrm{n,m}} \, \mathrm{q}_{\mathrm{n,m}}^{\mathrm{t}}$$

Since the hierarchical likelihood distribution problem we described in section 5.1, we choose to use the temporal information to disambiguate but apply a temporal filter at this stage. We observed that the hand motion can move quickly, and dynamics of hand motion is difficult to model at the full state space. Our real image database is small enough to allow us to apply real time ANN search and attach importance to observation than temporal information. It allows us having more chance to track hand motion that changes in different speed and better accuracy.

5.3 3D model Database sampling and Hand Pose Reconstruction

Now, we are received a few regions that may with high probabilities in the model database. In this section, we introduce our pose reconstruction method based on these candidate regions. In general, tracking can formulate as a Bayesian inference problem. Given the configuration at time $\boldsymbol{t}$ is represented as $\boldsymbol{q}_{\boldsymbol{t}}$ and the observation is $\boldsymbol{z}_{\boldsymbol{1}:\boldsymbol{t}}$, the state estimation probability $p(\boldsymbol{q}_{\boldsymbol{t}}|\boldsymbol{z}_{\boldsymbol{1}:\boldsymbol{t}})$ with the following Bayesian formulation:

$$p(\boldsymbol{q}_{\boldsymbol{t}}|\boldsymbol{z}_{\boldsymbol{1}:\boldsymbol{t}}) \propto p(\boldsymbol{z}_{\boldsymbol{t}}|\boldsymbol{q}_{\boldsymbol{t}}) \int p(\boldsymbol{q}_{\boldsymbol{t}}|\boldsymbol{q}_{\boldsymbol{t-1}}) p(\boldsymbol{q}_{\boldsymbol{t-1}}|\boldsymbol{z}_{\boldsymbol{1}:\boldsymbol{t-1}}) d\boldsymbol{q}_{\boldsymbol{t-1}}$$

where $p(\boldsymbol{z}_{\boldsymbol{t}}|\boldsymbol{q}_{\boldsymbol{t}})$ denotes the likelihood function that relates observations $\boldsymbol{z}_{\boldsymbol{t}}$ in the image to the unknown state $\boldsymbol{q}_{\boldsymbol{t}}$, and $p(\boldsymbol{q}_{\boldsymbol{t}}|\boldsymbol{q}_{\boldsymbol{t-1}})$ 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 $\boldsymbol{N}$ number of samples at each time $\boldsymbol{t}$.

Certainly, 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 always modeling 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 region. In contrast, choosing a particular motion model can also be restrictive. 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. (See figure 5.3)

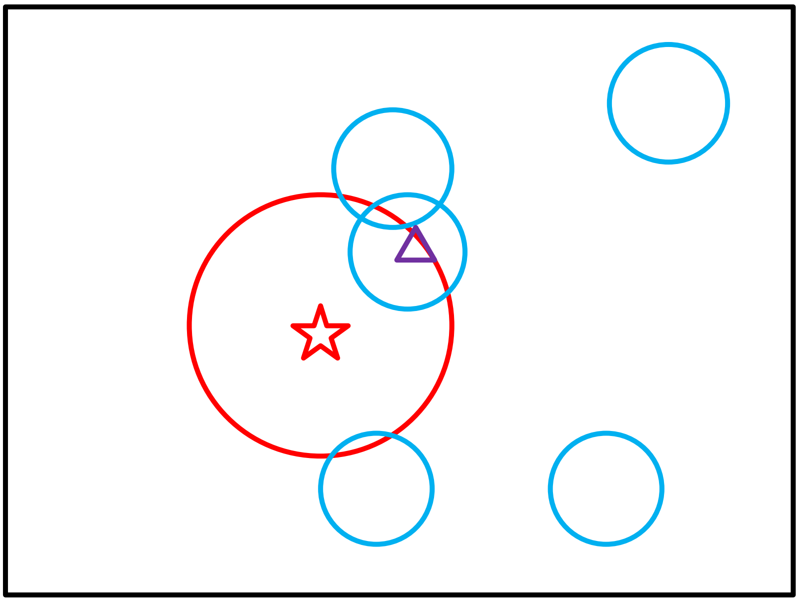


Figure 5.2 an 2D example of our sampling method, the red star represents the previous best pose, the triangle represents the current best pose, the red circle represents the neighbors of the previous frame, and the blue small circles represent the high posterior regions by ANN search on the real image database, in most situation the regions of the red circle is difficult to decide

As the same as in real database, the likelihood function is based on the chamfer distance. However, the edge features are often ambiguous due to clutter and it can be different for same configuration at different illuminations. This is hard to model even for synthetic hand images. For this reason, we use silhouette as our features instead.

Silhouette provides less information about the full hand configuration. Since our importance regions far away from each other and silhouette can be synthesized easily, silhouette is useful than edge for chamfer distance in this stage.

To reconstruct the final hand pose, similar to the particle filter, we can approximate the distribution $p(\boldsymbol{q_t}|\boldsymbol{z_{1:t}})$ by a weighted set of **Q** samples, for **L=1,…,P** draw samples from the importance function:

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

where $\boldsymbol{r}$ denotes the observation in real database, $p(\boldsymbol{q_t}|\boldsymbol{r_t})$ represents the proposal distribution based on ANN search, and Wdenotes the importance weights. Certainly we normalize these weights such that $\sum \omega_L = 1$, then we can use the weight blending we described in section 5.2 to get a final pose efficiently. However, there is still significant jitter since the pose blending without a temporal smoothing, and many simples have small weights. In the following, we only consider those samples having high weights and combine with the temporal smoothness term; we can formulate the motion reconstruction as an energy minimization problem. A data prior term enforces plausible reconstruction results and a smoothness term measures the smoothness of the synthesized motion:

$$q^* = \underset{q}{argmin}(\,\omega_{prior}\,E_{prior}\,(q) + \omega_{smoot\,h}E_{smoot\,h}\,(q))$$

where the two weights ω_{prior} and $\omega_{smoot\,h}$ are user-defined constants. For a set of poses $Q_L^t = \{q_1^t, \dots, q_P^t\}$ with corresponding weights W_L^t, we assume the poses in the local region are a simple distribution that can model by a kernel function. We use a kernel based approach proposed by [TZK*11], the data prior term E_{prior} :

$$E_{prior}\,(q) = \sum_{\mathrm{L}=1}^{\mathrm{P}} W_L^t\,K(|q_L^t - q|)$$

Where K() is kernel function. As Tautges described [TZK*11], a kernel based representation is well suited to approximate arbitrary 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:

$$E_{smoot\ h}(q) = K(|q - q^{t-1*}|)$$

Where q^{t-1*} are the best poses in the previous two frames. We initialize the optimization with the weight blending pose and optimize using the Levenberg-Marquardt algorithm [Lourakis. 04].

Chapter 6

Experiments and Results

Our experiments perform on a desktop with Intel® Core™ i5-760 Processor, 4GB main memory. In our experiments, the entire following test sequences from a single camera at 10 frames/sec and the user is same with the training subject. And for all of these sequences, we use the same database and same parameters. 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.

To measure the accuracy of our approach, we perform the evaluation by applying Root Mean Square (RMS) to finger end point 2D position. However, this is difficult to acquire the ground truth 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.

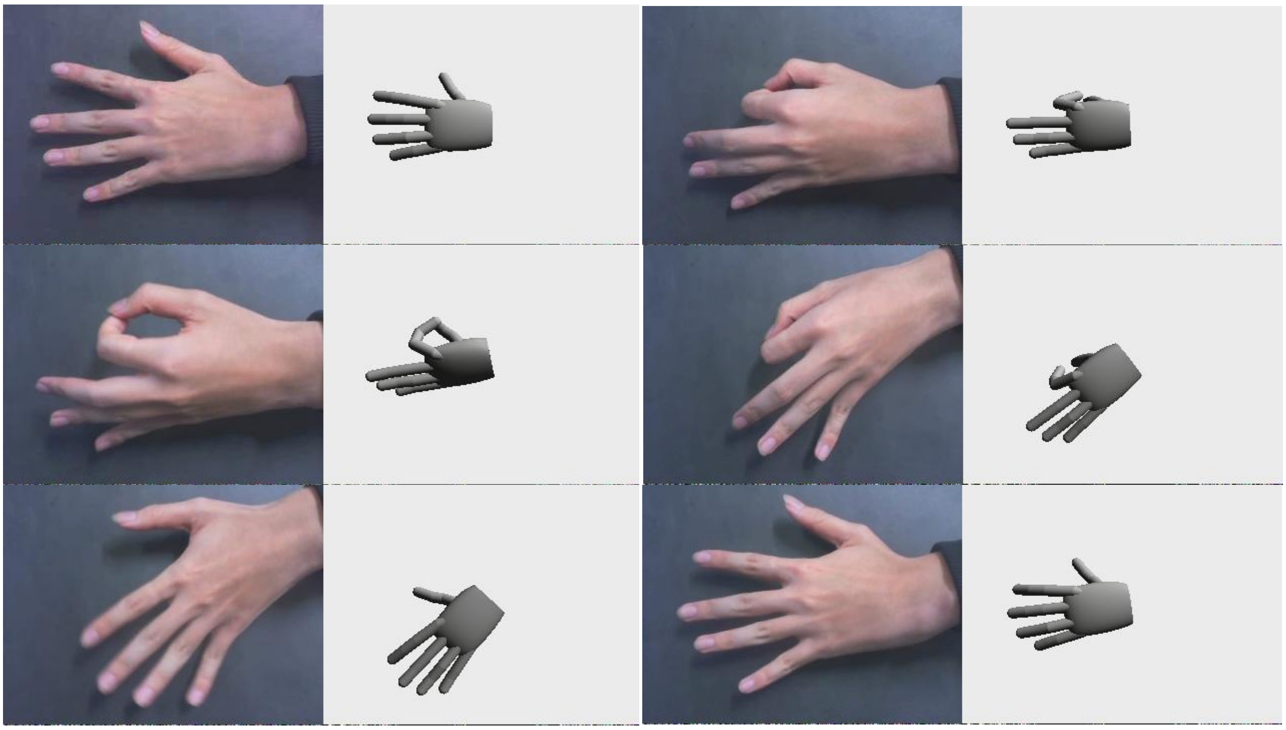


Figure 6.1 sequence ok_shoot

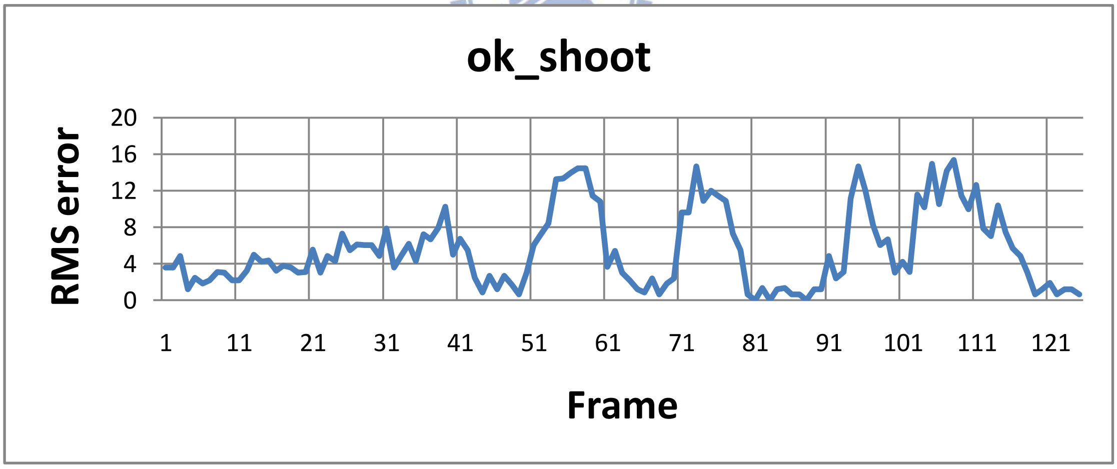


Figure 6.2 sequence ok_shoot error performance

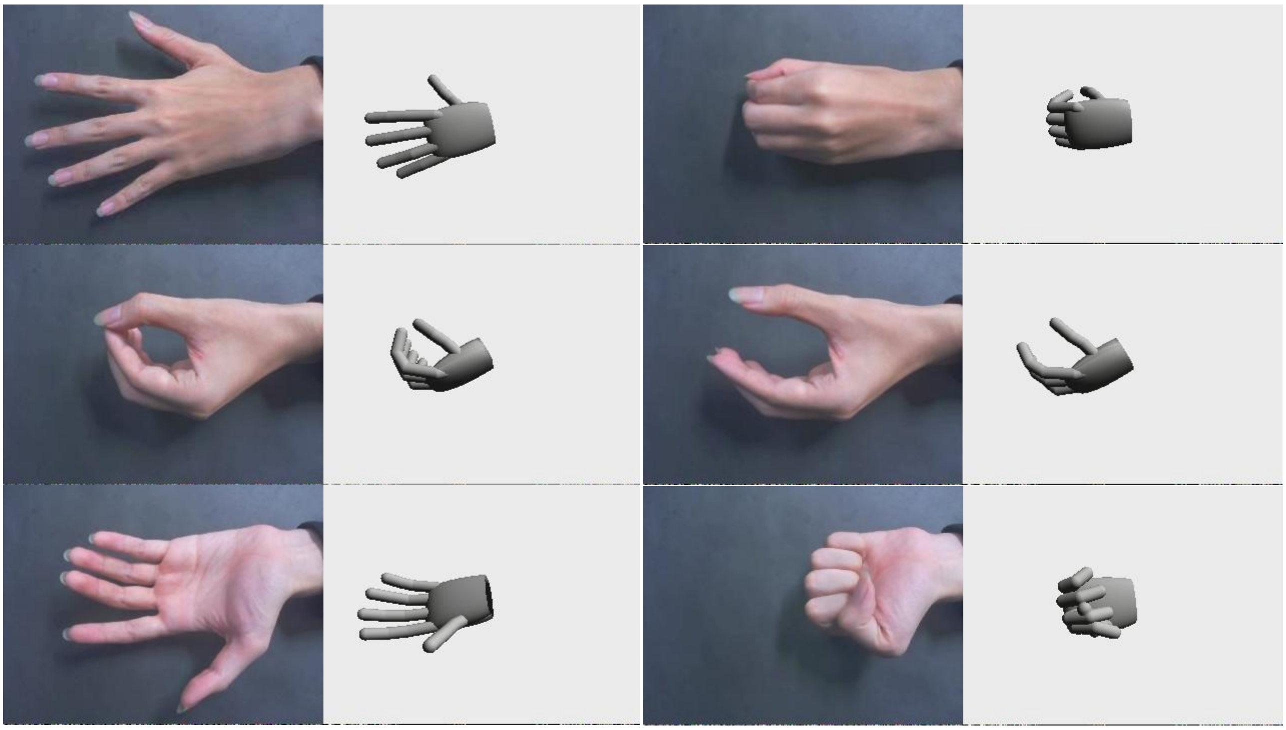


Figure 6.3 sequence grasp_and_rotation

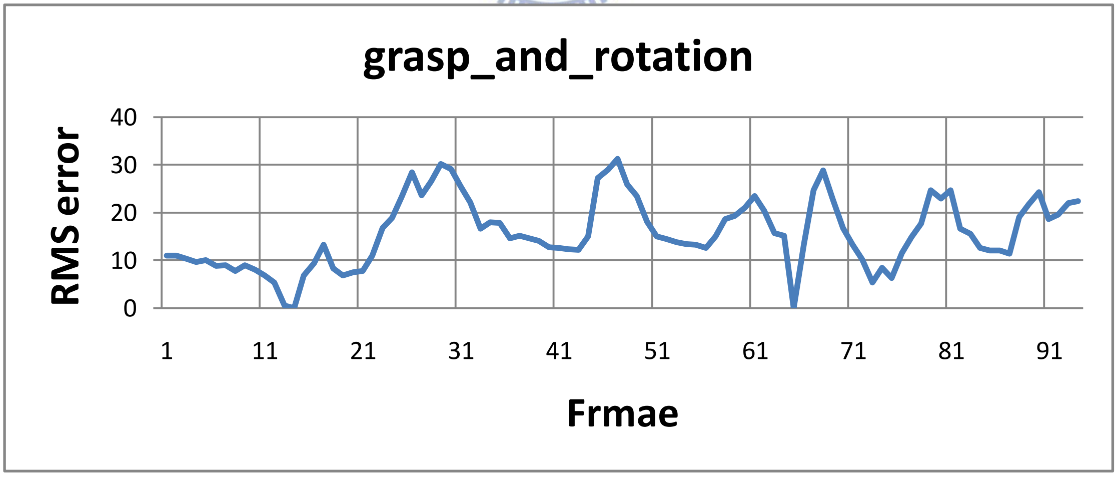


Figure 6.4 sequence grasp_and_rotation error performance

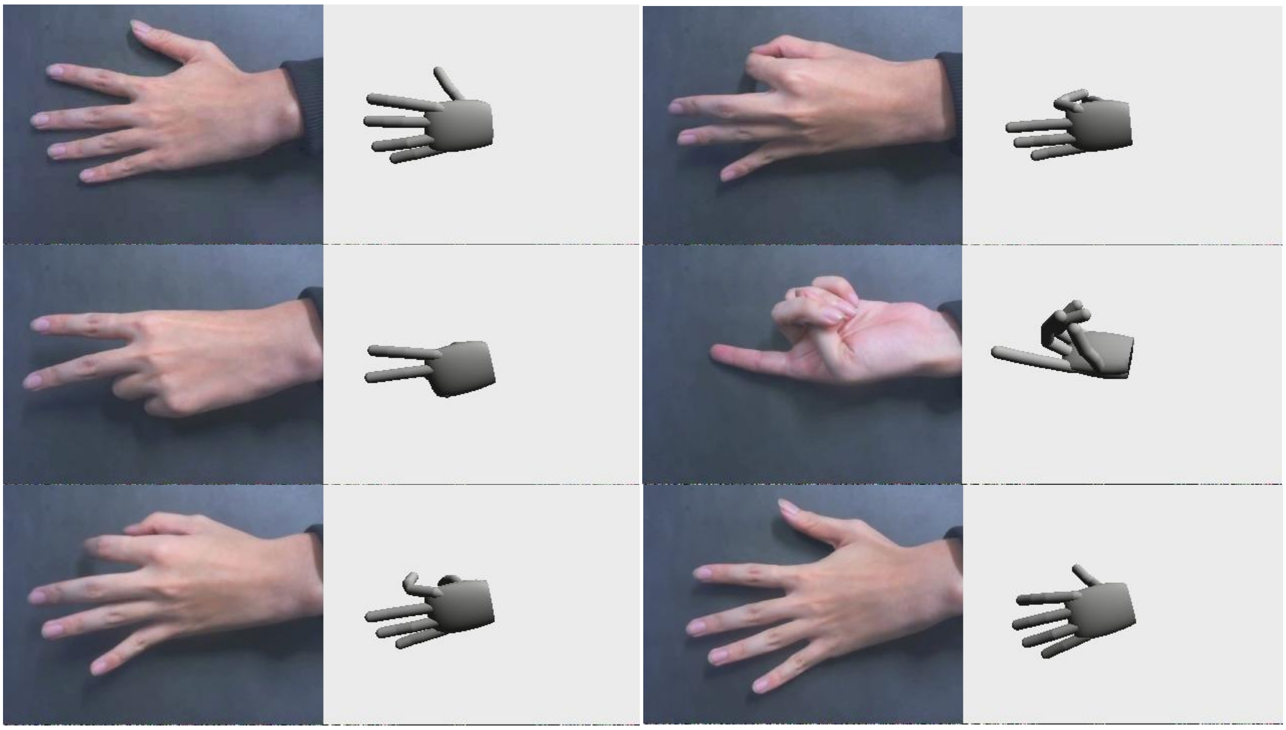


Figure 6.5 sequence fast_gesture

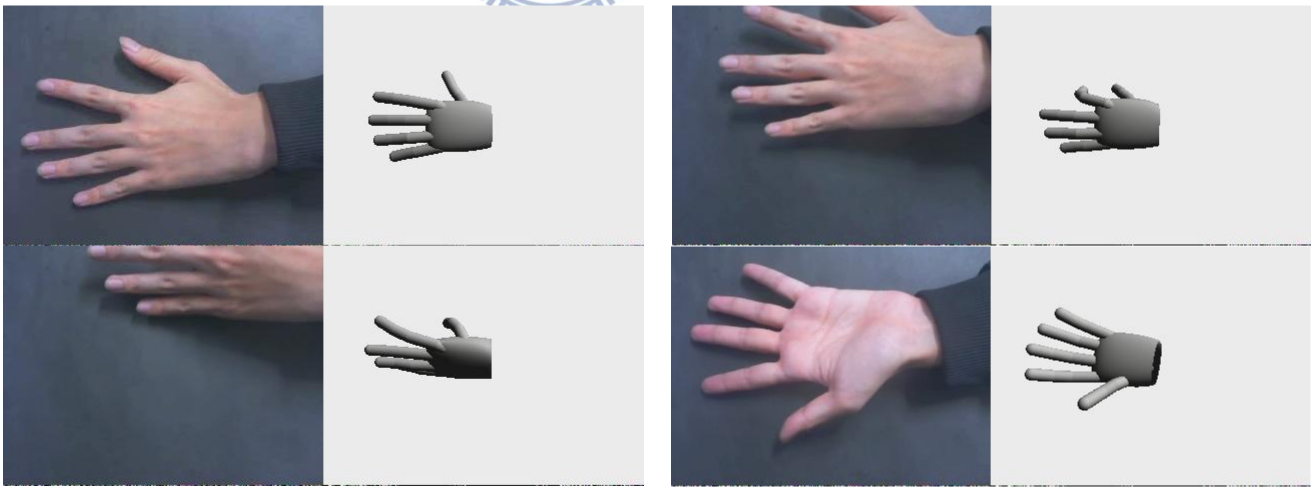


Figure 6.6 sequence recovery

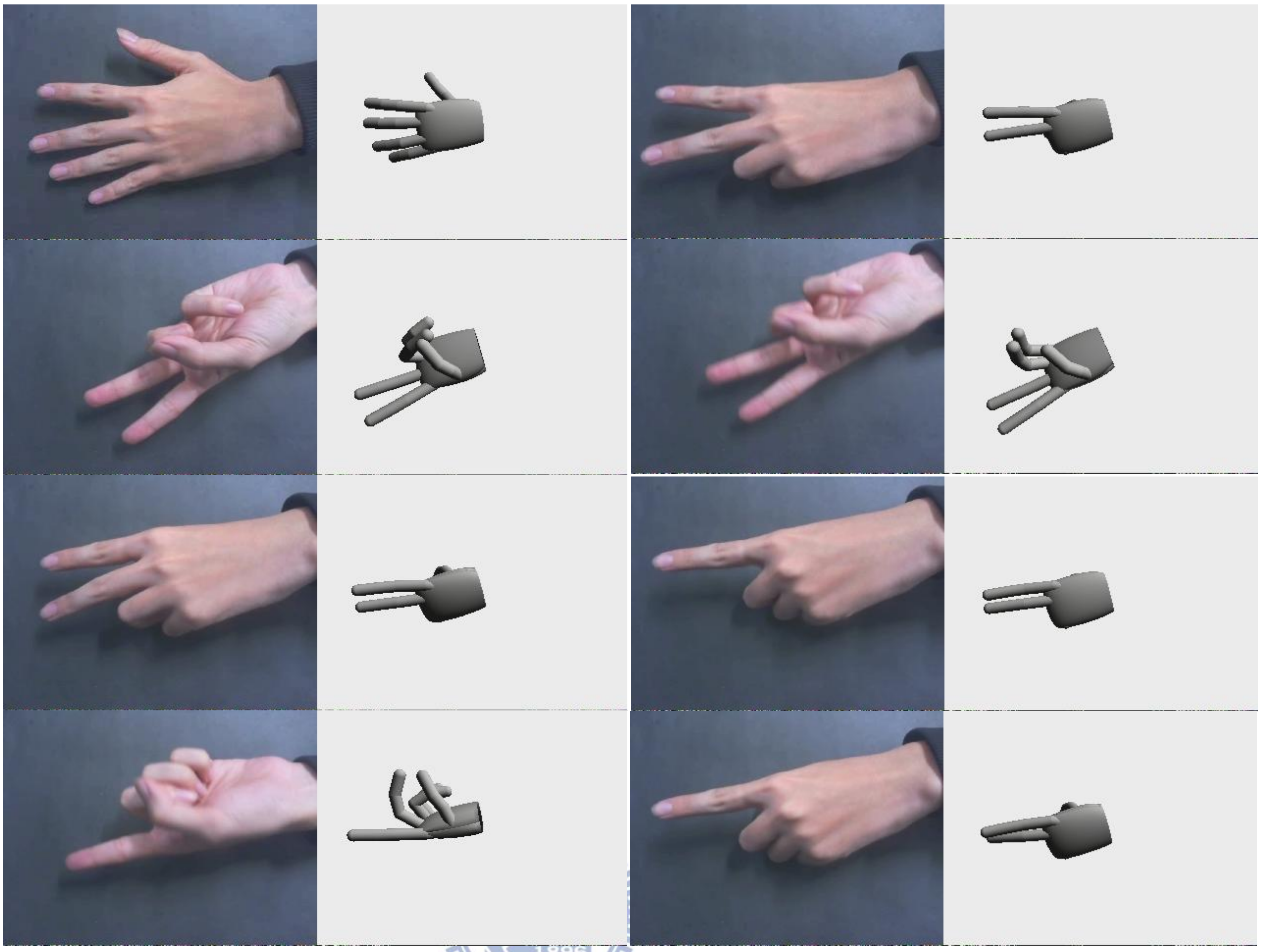


Figure 6.7 sequence pose_occlusion

Discussion

In experiments, we show that our system succeed tracking most of these motions. And our system can run in real-time performances, which are around 10 frames per second (10 FPS) rate. Figure 6.1 shows the results from a sequence that includes two gestures and rotation. Figure 6.3 shows the results from a sequence that includes a grasp motion and rotation. Figure 6.2 and Figure 6.4 show their error performance 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.

In Figure 6.5, we test a fast gesture change with rotation. We observed that at the last change, the gesture change is too fast since the tracking cannot succeed fully recovering the “OK” gesture. It is because the distance between the “1” gesture and “OK” gesture is far, the temporal coherence may affect the likelihood of “1” gesture, and must take more frames to recover this gesture.

Since our ANN search on the whole real image database every time, our system is able to recover after the hand left the camera. Figure 6.6 shows that.

In Figure 6.7, we show the limitation of our bare-hand features. In the right 2 image, the hand shake and a small error occur. And at the bottom, we observed that that the “2” gesture cannot change to “1”gesture. It is because we have 2 similar gestures “H” and “R” in our database, the edge and silhouette are not a robust feature to measure the different between different fingers. Figure 6.8 shows these there gestures.

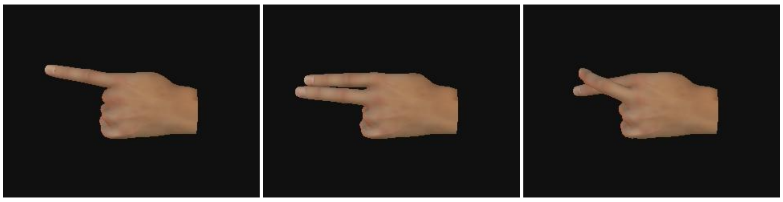


Figure 6.8 Form left to right: "1"　"H" and "R" gestures

Finally, the following experiment shows the comparison between our hybrid method and full KNN search on real image database and synthetic database. We apply our method and full KNN search on "grasp_and_rotation" sequence, figure 6.9 shows the result. If we use only the real image database, the distribution of the database is too dispersed, and many incorrect poses can make an unsatisfactory blending result. And if we use only the synthetic database, since the different between the real hand and the 3D model hand, will cause many pose occlusion.

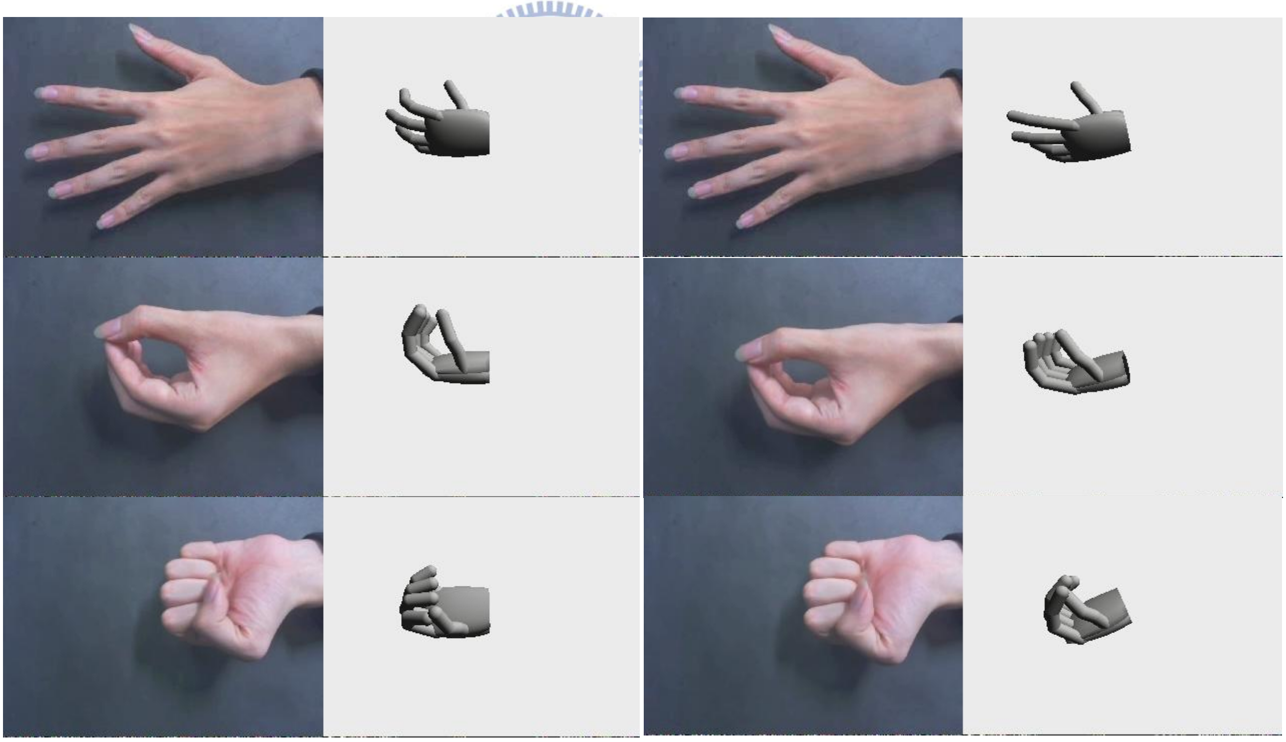


Figure 6.9 Left: full KNN search on real image database. **Right:** on synthetic database

Chapter 7

Conclusions and Future work

In this thesis, we present an approach to tracking an articulated hand without using markers in real time and the limitation of our work is that we need a small real-image database capture form each user. We use hierarchical-searching to efficiently find the KNN result in a large 3D synthetic hand database, and use the temporal consistency to disambiguate the KNN result. Our experiments show that we successfully estimate the 3-D position and joint configuration of the hand under many self-occlusion situations. And our system can provide data for human computer interaction.

There are many possible extensions to our system. The most critical issue in our future works is to use more robust similarity measures. The edge and silhouette feature is inherently ambiguous even we apply a temporal disambiguate. Especially for smooth- appearance gesture like fist or the different between N and S in the sign language alphabet. As Martin de La Gorce *et al.*'s described [GFP11], texture and shading is a crucial visual cue for hand, but it is difficult to extract shading feature in real time. If we can design a hybrid method, off-line synthesize a 3D model database with finding correct texture and shading from a little training data, may directly increase the robustness of the tracker. Second, our prototype system needs to capture real-image database from a novel user every time. We can collect more hand image from different people, and may allow user to use the most similar hand's data in database, instead of the specify user training.

Reference

[TZK*11] Jochen Tautges, Arno Zinke, Björn Krüger, Jan Baumann, Andreas Weber, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bernd Eberhardt Motion :
Motion Reconstruction Using Sparse Accelerometer Data
ACM Trans. Graph. (May 2011), 30:3(18:1-18:12)

[SS10] Smithmicro Software Poser Pro 2010
http://poser.smithmicro.com/poserpro.html

[Lourakis 04] Manolis Lourakis: Levmar
http://www.ics.forth.gr/~lourakis/levmar/ 2004

[SMC01] B. Stenger, P. R. S. Mendonça, R. Cipolla :
Model-Based Hand Tracking Using an Unscented Kalman Filter
British Machine Vision Conference, Vol. I, pages 63-72, Manchester, UK, September 2001.

[STTC06] B. Stenger, A. Thayananthan, P. H. S. Torr, R. Cipolla :
Model-Based Hand Tracking Using a Hierarchical Bayesian Filter
Pattern Analysis and Machine Intelligence, IEEE Transactions on , Vol. 28, No. 9, pages 1372-1384, September, 2006.

[EBN*07] Ali Erol, George Bebis , Mircea Nicolescu , Richard D. Boyle, Xander Twombly :
Vision-based hand pose estimation: A review
Computer Vision and Image Understanding
Volume 108 Issue 1-2, October, 2007

[GFP11] Martin de La Gorce, David J. Fleet and Nikos Paragios :
Model-Based 3D Hand Pose Estimation from Monocular Video
Pattern Analysis and Machine Intelligence, IEEE Transactions on , Volume: 33, Issue: 9, Pages: 1-15 , 2011

[PA08] Michalis Potamias and Vassilis Athitsos :
Nearest Neighbor Search Methods for Handshape Recognition.
Conference on Pervasive Technologies Related to Assistive Environments (PETRA), July 2008.

[RKK09] Javier Romero ,Hedvig Kjellstr¨om, Danica Kragic :
Monocular Real-Time 3D Articulated Hand Pose Estimation
Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on

[WP09] R. Y. Wang and J. Popovi´c. :
Real-time hand-tracking with a color glove.
ACM SIGGRAPH 2009 papers

[BTBW77] H. Barrow, J. Tenenbaum, R. Bolles, and H. Wolf.
Parametric correspondence and chamfer matching: Two new techniques for image matching.
Proceedings of the 5th international joint conference on Artificial intelligence - Volume 2

[KH75] Fukunaga, Keinosuke; Larry D. Hostetler.
"The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition".
IEEE Transactions on Information Theory (IEEE) **21** (1): 32–40. (January 1975)

[AFJ01] Doucet, A.; De Freitas, N.; Gordon, N.J.
Sequential Monte Carlo Methods in Practice. Springer. 2001