

國立交通大學

資訊科學與工程研究所

碩士論文

利用深度影像即時估計手指動作

Real-Time Finger Motion Estimation from Depth Images

研究生：王星寒

指導教授：林奕成

中華民國 101 年 9 月

利用深度影像即時估計手指動作

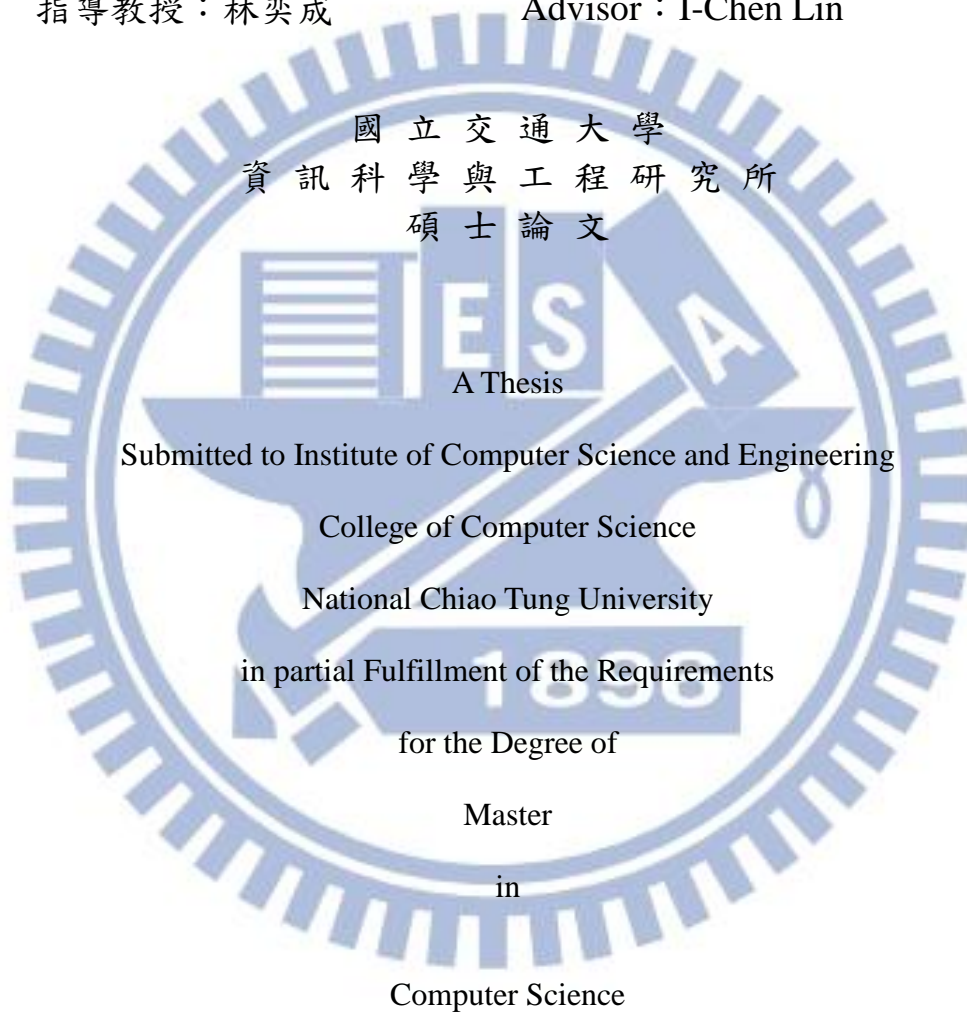
Real-Time Finger Motion Estimation from Depth Images

研究生：王星寒

Student : Hsing-Han Wang

指導教授：林奕成

Advisor : I-Chen Lin



September 2012

Hsinchu, Taiwan, Republic of China

中華民國 101 年 9 月

利用深度影像即時估計手指動作

學生：王星寒

指導教授：林奕成

國立交通大學
資訊科學與工程研究所

摘要

我們提出了一個新方法能從深度圖像有效率地估計手部關節。遵從數據驅動的策略，它結合了手勢辨識與搜尋其他時間點的過程，我們採取了一個辨識物件的方法把困難的估計手指動作問題對應到一個較簡單的累計分類問題。使用深度相機讓我們的新分類器不會隨著手的形狀、服裝、飾品等而改變。我們的框架還提供手部估計問題更多的靈活性和應用性。特別是它允許玩家在廣闊的空間中以全身和手指動作來操作系統。

該系統以足以進行互動的速率運作在消費級個人電腦上。我們的評估顯示它在實際的測試集中具有較高的兼容性和穩定性。我們展示了一個利用我們的估計系統來同時使用身體和手指動作的有效率的 3D 遊戲劇本。

關鍵字：電腦動畫、互動介面、深度影像、KINECT。

Real-Time Finger Motion Estimation from Depth Images

Student: Hsing-Han Wang

Advisor: I-Chen Lin

Institute of Computer Science and Engineering

National Chiao Tung University

Abstract

We propose a new method to efficiently estimate hand articulation configurations from a depth image sequence. Following a data-driven strategy that combines gesture reconstruction with temporal retrieval process, we take an object recognition approach that maps difficult finger motion estimation problem into a simpler histogram classification problem. Using a depth camera allows our novel classifier to estimate finger motion invariant to hand shape, clothing, accessories, etc. Our framework further provides more flexibility and applicability in hand estimation issue. In particular, it allows players to operate the system in extensive space with both full-body and finger motion.

The system runs at interactive frame rates on consumer-level PCs. Our evaluation shows its high compatibility and stability on real test sets. We demonstrate efficient 3D game scenarios using both body and finger motions by our estimation system.

Keywords: Computer animation, interactive interfaces, depth image, KINECT.

Acknowledgement

在許多人的幫助下，這篇論文順利地完成了，於此謹以文字表達我的感謝。首先，我要感謝我的家人，沒有他們的支持與鼓勵，勢必得花更多的時間與精力才能完成這篇論文。其次我得感謝我的指導教授，在老師的指導下，我才能順著興趣做自己喜歡的研究，並有所成果。最後我要感謝同窗學長與同學，在研究中產生的許多細微問題，都是依靠大家的幫助與建議才得以解決。



Content

摘要.....	i
Abstract.....	ii
Acknowledgement.....	iii
Content.....	iv
List of Figures.....	v
1. Introduction.....	1
2. Related Work.....	3
3. Data Collection and Normalization.....	6
3.1. Depth imaging.....	7
3.2. Hand model.....	8
3.3. Normalization.....	9
4. Gesture Inference.....	11
4.1. Feature extraction.....	13
4.2. Candidate generation.....	15
4.3. Feature space lookup.....	18
4.4. Hypothesis voting.....	20
4.5. Motion blending.....	21
5. Experiments.....	22
5.1. Test data.....	23
5.2. Estimation accuracy.....	27
6. Discussion.....	31
References.....	32

List of Figures

Figure 1.1: XBOX 360 KINECT [Microsoft].....	2
Figure 3.1: Data recording scenario.....	6
Figure 3.2: Our depth imaging space.....	7
Figure 3.3: Noisy input depth stream.....	8
Figure 3.4: FSM controller.....	10
Figure 3.5: Normalization process.....	10
Figure 4.1: Overview of our gesture estimation framework.....	12
Figure 4.2: Independent blocks for feature extraction.....	14
Figure 4.3: A feature vector (point histogram).....	14
Figure 4.4: Spatial candidate generation.....	15
Figure 4.5: Temporal candidate's central position (blue).....	16
Figure 4.6: Capture with different window size.....	17
Figure 4.7: Normalization of occlusion case.....	19
Figure 4.8: Feature space lookup.....	19
Figure 5.1: A screenshot of our system.....	22
Figure 5.2: Our used gestures.....	23
Figure 5.3: Recognition accuracy (User 1).....	27
Figure 5.4: Recognition accuracy (User 2).....	27
Figure 5.5: The distribution of the best-matched candidate for static OK sign.....	28
Figure 5.6: The distribution of the best-matched candidate for dynamic OK sign.....	28
Figure 5.7: The distribution of the best-matched candidate for static Five sign.....	29
Figure 5.8: The distribution of the best-matched candidate for dynamic Five sign.....	29
Figure 5.9: Performance comparison.....	30
Figure 5.10: Accuracy convergence of search depth.....	30

1. Introduction

In recent years, depth cameras, like Microsoft Kinect and ASUS Xtion, have become consumer-level hardware. It means that depth-image-based motion recognition system will be easily accessible in day life. Additionally, depth cameras offer several advantages compared with RGB cameras, including capability in low illumination conditions, giving a calibrated scale estimate, color and texture invariance, and resolving silhouette ambiguities in pose. They also greatly simplify the task of background subtraction.

While full-body motion expresses emotion through large movement like hand waving, finger motion implies further information with subtle motion detail. Combining the result of both full-body and finger motion recognition help to bring virtual characters to life and do more delicate operations. We can find applications including human-computer interaction, gaming, telepresence, and even health-care.

Recent state-of-the-arts show that real-time human pose recognition is feasible. However, finger motion recognition and hand tracking with depth sensors is still challenging and with defects. In other words, even the best existing systems still exhibit limitations. For example, obtrusive equipment like retro-reflective markers [HRM12, PY06] or special glove [WP09] is used in these frameworks. However, the cost and setup of hardware is burdensome. Other systems achieve high accuracy of tracking bare hands from depth or color images [OKA12, RYZ11, WPP11], but they are highly influenced by operating space and so are not robust. In this thesis, we focus on finger motion estimation from full-body depth images and introduce a new user-input system that captures subtle full-body command.

To tackle this problem, we project the depth images into 3D space as a point cloud and separate the space to independent blocks. Evaluating each block separately

extracts a low dimension feature of the depth image. To estimate gesture hypotheses, we extract feature from the input data and compare with the 3D gesture features in database. However, within a feature, there are still unpredictable differences in the contextual appearance. In order to address this problem, a realistic and highly varied data set is generated from humans with many shapes and sizes. For further accuracy, we generate spatial and temporal candidates dealt with the noise of hand position estimation and provide a set of parameter which avoids unreasonable transformation between gestures. Finally, the classifier combines Kalman filter to facilitate real-time tracking with stable results even for fast and complex motions.

Our system performs up to 15ms per frame and the operating region is valid by $2m \times 2m \times 1.8m$ space. Our main contribution is to treat gesture estimation as database lookup procedure, and we use a novel feature representation designed to recognize pose at high flexibility and low computational cost.



Figure 1.1: XBOX 360 KINECT [Microsoft].

2. Related Work

Many optical motion capture systems with markers have been applied to finger motion capture, and they acquired satisfactory results. Park et al. used LEDs to design an interactive system [PY06]. Ludovic et al. solved the synchronization problem of body and finger motion from reduced marker sets [HRM12]. However, they still needed obtrusive markers and complicated cameras setup.

On the other hand, bare-hand tracking is still a challenging topic now. Edge detection and silhouettes are the most common features used to recognize the pose of the hand [SMF04, STT06, DDH06] but their performance is still far from real-time. Shakhnarovich et al. proposed an upper body pose estimation system which searches a database of synthetic poses [SVD03]. Athitsos et al. developed fast and approximate nearest-neighbor techniques to estimate 3D hand pose [AS03, AAS04]. Ren et al. built a database of silhouette features for controlling animated human characters [RSH05]. Wang and Popović used a color glove to map the hand configuration to a database of hand poses [WP09]. Later, they introduced data-driven bare hand tracking system for efficient 3D mechanical assembly of computer aided design (CAD) models using [WPP11].

After the launch of inexpensive depth cameras, like Kinect, we can handle a full range of body shapes and sizes at interactive rates on consumer hardware. Ren et al. focused on a specific gesture set. They employed Finger-Earth Mover's Distance to measure hand shape dissimilarity such that the classifier can tackle parts of challenging cases for hand gesture recognition [RYZ11]. However, it still requires a black belt on the user's wrist for hand segmentation and is easily affected by users' silhouette. Oikonomidis et al. use Particle Swarm optimization to track the full articulation of two strongly interacting hands observed by an RGB-D sensor [OKA12].

Since they used expensive evolutionary optimization method, the system ran at 4fps and was still far from real-time. This precludes their use for interactive applications.

Human motion reconstruction from multiple cameras has produced an numerous literatures. Bregler and Malik used twists motions and exponential maps to produce motion estimation even with complex self-occlusion [BM98]. Rosales and Sclaroff reconstructed human poses from low-level visual features [RS00]. Ioffe and Forsyth grouped parallel edges as candidate body segments and pruned the search of such segments combinations [IF01]. Mori and Malik matched shape context with multiple 2D exemplars [MM03]. Ramanan and Forsyth clustered candidate body segments found by pairs of parallel lines, finding all individuals in each frame [RF03]. Agarwal and Triggs reconstructed poses by learning a regression against shape vectors extracted from image silhouettes [AT04]. In [SBR04], Sigal et al. implemented Eigen-feature detectors for head, upper arms, lower legs and shoulders as desired. Felzenszwalb and Huttenlocher employed pictorial structures for efficiently finding the best match to an image [FH05]. Navaratnam et al. showed marginal distribution sampling of unlabeled data to improve pose fitting [NFC07]. Okada and Stenger built a search tree on a hierarchy of body shape to capture human motion [OS08]. Based on a local mixture of Gaussian Processes, Urtasun and Darrel proposed a regression scheme to inference human poses [UD08]. In [TU08], Tu used auto-context to label body parts, but it did not define localized joints and took about 40 seconds per frame. Randomized decision forests was built in [RRR08] on classes defined by human action patterns and camera viewpoints. Bourdev and Malik introduced ‘poselets’ that used tightly clustered in both 3D pose and 2D image appearance by using SVM classifier [BM09].

While real-time estimation of full-body motions from monocular intensity image sequences is still an open problem, the gradually popular depth cameras spur further

opportunity for human pose reconstruction. It allows more reliable 3D pose estimation from a single viewpoint. Grest et al. estimated the body poses of a known size and starting position using an Iterated Closest Point (ICP) algorithm [GWK05]. Based on MRFs, Angelov et al. segmented puppets in 3D scan data into body parts and background [ATC05]. In [ZF07], Zhu and Fujimura used a linear programming relaxation to solve body component identification for coarse upper body parts, but they required a T-pose initialization to size the model. Bleiweiss et al. use 3D model fitting to track human skeletons [BEK09]. Siddiqui and Medioni used a data-driven Markov chain Monte Carlo (MCMC) model to find optimal pose and showed significant improvement over ICP [SM10]. Kalogerakis et al. labeled and segmented 3D meshes into different parts [KHS10], but they did not deal with occlusions and the results were sensitive to training sets. Ganapathi et al. showed that data-driven evidence is crucial for tracking self-occlusion [GPT10]. Plagemann et al. [PGK10] built an interest point detector for 3D meshes, finding geodesic extrema, and localizing body parts. Their method generated both a location and orientation estimate of each part, but the use of interest points limits the choice of parts, such that left or right is unable to be recognized. Shotton et al. segmented the different human body parts using a random forest classifier implemented in the Kinect system [SFC11]. The segmentation was used to generate joint positions of a skeleton. Baak et al. combined generative and discriminative methods to estimate full-body pose at interactive frame rates [BMB11]. Girshicky et al. extended Hough forests and directly predict the positions of body joint [GSK11].

3. Data Collection and Normalization

Extracting useful information from incomplete and noisy data is often a key issue of human pose estimation from cameras. While multi-view tracking systems require solving challenging non-linear optimization to eliminate ambiguity [WPP11], single-view pose estimation is hampered by the huge color and texture variability, including hair, skin and clothing. In recent years, depth cameras capture 2.5D scene geometry [KBK10] at interactive frame rates, and therefore significantly reduce this difficulty. However, intensive noises in the depth data disturb the estimated skeleton configurations.

Fig. 3.1 illustrates the process that how we collect depth data. In this chapter, we review depth imaging, introduce our hand model configuration and show how we normalize real depth capture data to fit our framework.

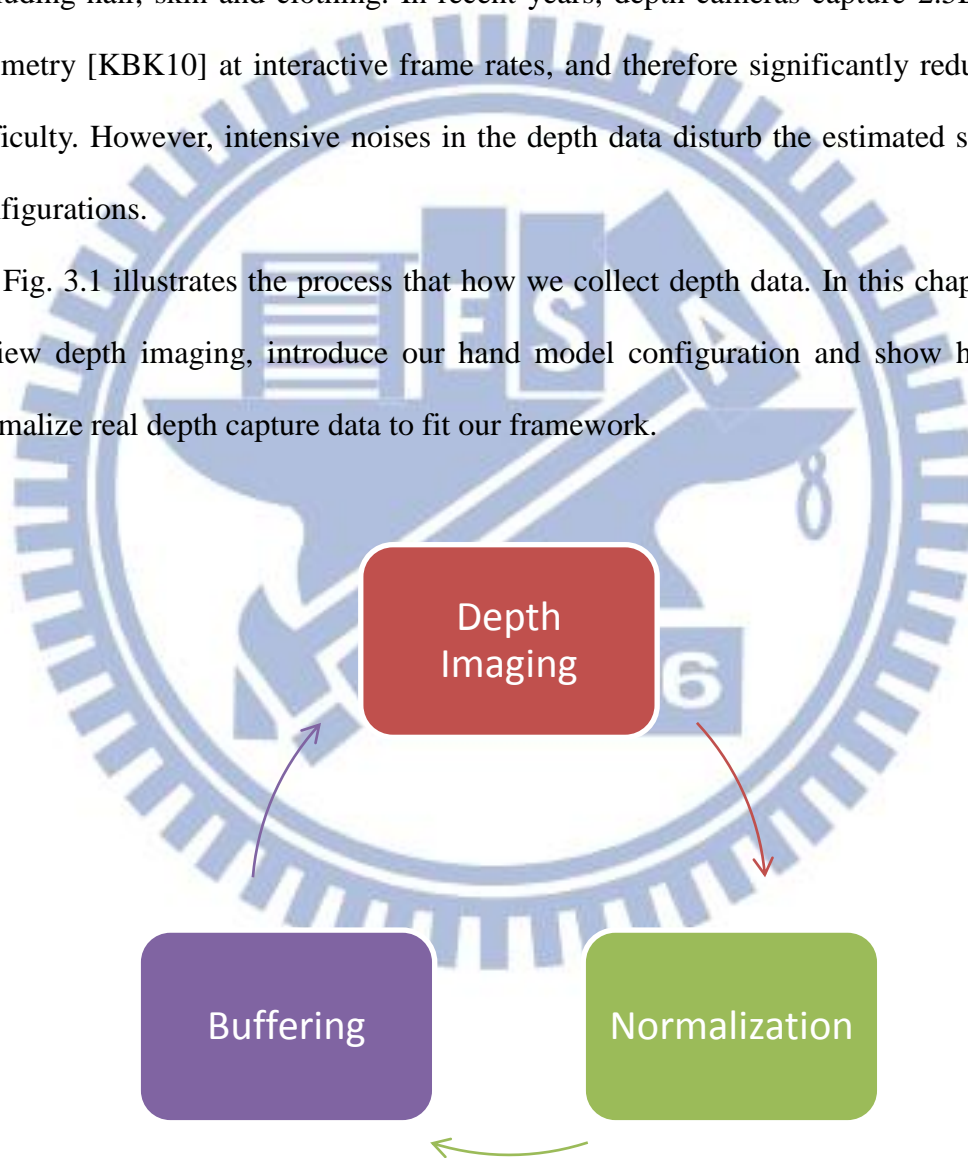


Figure 3.1: Data recording scenario.

3.1. Depth imaging

We choose Microsoft Kinect as the depth camera in our system. The depth camera captures depth data at 30 frames per seconds with about 1cm distance resolution. Opposed to color cameras, pixels in a depth image indicate the calibrated distance between camera and corresponding 3D positions by projection and receive of infrared light. Several advantages are benefited from the above property, including working in low illumination conditions, being color and texture independent, giving a calibrated distance estimate, and reducing shape ambiguities. The most important issue for our framework is its simplicity in the task of background detection.

Unfortunately, there are still several challenges, including exhibiting low resolution, generating intensive random noise and performing a systematic bias [KBK10]. The defects are even obvious under fast motions and show unstable results at some thin body parts, leading to jiggling motion estimation.

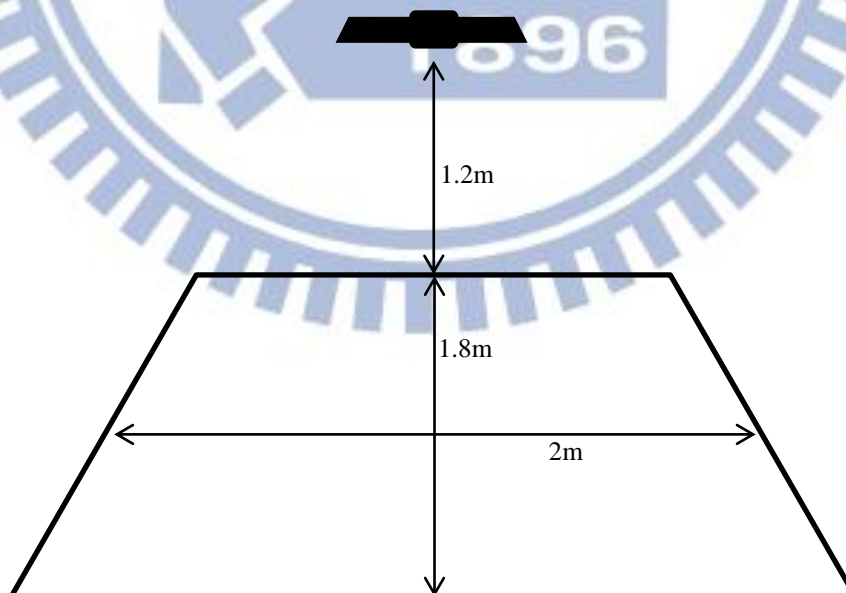


Figure 3.2: Our depth imaging space.

3.2. Hand model

A human hand has 27 DOFs and they thus create an enormous variety of gestures which are difficult to simulate. However, there are only a few gestures are widely used and intuitive in message passing. Based on this idea, we built our gesture database mostly on meaningful and memorable gestures, like paper sign, scissor sign, forward pitch, etc. We take these gestures as analog of instruction to indicate corresponding motions.

Unrestricted 3D body postures actions are the ultimate goal of interaction but it is computationally expensive to consider all varieties. Additionally, changes in gesture from one frame to the next are often small and insignificant. Inspired by biological structure, we thus remove nearly identical 3D motions to a single representation pose by the vector from elbow to wrist, so that only small key gestures are necessary. In our implementation, we captured 150 depth frames per gesture that form a cluster in database, and they provided acceptable result against the noisy depth stream.

In conclusion, the reduced gesture database constitutes a suitable representation for real hand gestures. Each element retains most of its detailed discriminative ability even under noisy input data.



Figure 3.3: Noisy input depth stream.

3.3. Normalization

In the proposed framework, we use a depth frame database. It contains a set of gestures that people would perform in an entertainment scenario, and is used to help local optimization.

We apply Microsoft Kinect [KINECT] to capture depth images and get joint position estimation. For training data generation, we capture hand depth images within 1.7m ~ 1.9m from the sensor. There are several states in our depth stream capture process. Fig 3.4 illustrates our data collection process as a finite state machine. At first, we use a simple gesture signal for data collection initialization. Here the system enters a countdown state which gives user time for gesture preparation. Then system starts capturing when countdown is over. We here record depth stream into memory buffer eliminating huge IO operations. Whenever user is about to stop capturing, we also prepare a gesture signal for the stop operation. The gesture signal must be rarely used in data collection process. This helps us to avoid unpredictable state transformation in capture scenario. Afterward users can divide recorded depth stream with specified frame number.

For each recorded frame, we firstly use estimated hand positions to clip depth data of hands. Secondly, we apply a filter that shifts depth values of hand frames to 0 ~ 255 range. Based on idea referred in Sec. 3.2, we thirdly rotate hand frames by the vector from elbow to wrist. Finally, we obtain 64x64 pixels hand depth frames with upward wrist configuration as illustrated in Fig. 3.5.

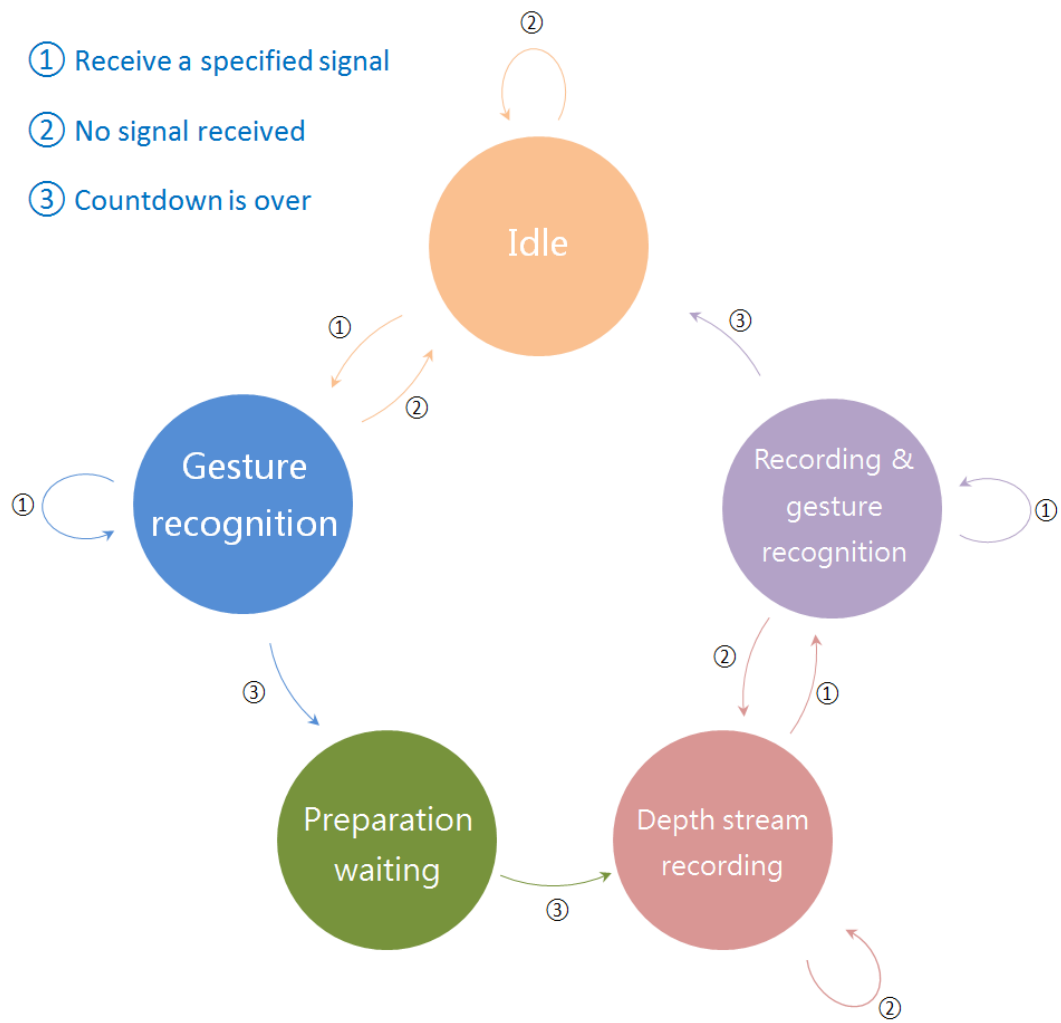


Figure 3.4: FSM controller.

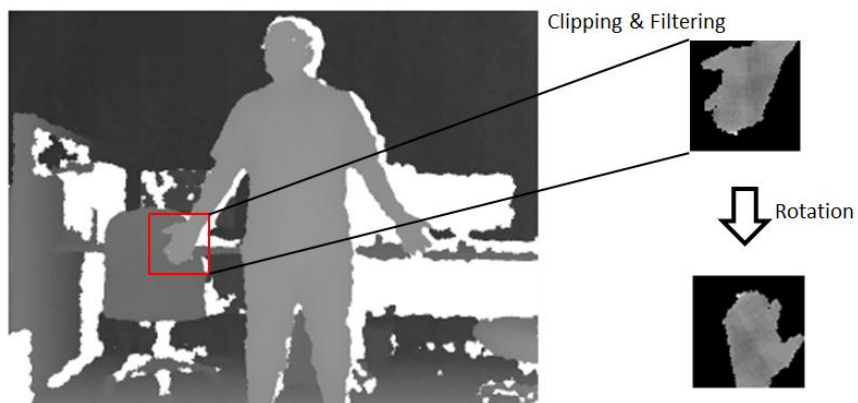


Figure 3.5: Normalization process.

4. Gesture Inference

Our framework has two phases, data preprocessing phase and real-time estimation phase, as illustrated in Fig. 4.1. In the data preprocessing phase, we obtain the image set of depth frame first. For each frame, the next step is to compute the point cloud $M_I \subset R^3$ from the depth image I . We extract a low-dimensional feature vector from M_I , and represent the 3D point distribution of point cloud by 3D position histogram (Sec. 4.1). Afterward, with respective gesture configuration, the features become a discriminative feature space.

In the real-time estimation phase, there are six stages to estimate the final pose of our framework. The first stage is to acquire depth frame I_t and estimated positions of full-body joints X_t at time t . Using previous estimated positions X_{t-1} , the second stage generate a set of candidates, including spatial candidates and temporal candidates (Sec. 4.2). In third stage, we apply feature extraction (as explained in Sec. 4.1) to the candidates and transform possible depth images into feature vectors F . Let G_{t-1} be the final gesture estimate of previous frame $t-1$. The fourth stage is to find K nearest neighbors for each candidate in F by querying the feature database (Sec. 4.3), and then obtain spatial gesture hypotheses G_t^S and temporal hypothesis G_t^T . Based on a voting scheme, the fifth stage uses a distance function to find optimum estimate G_t^* from the set of retrieved gestures (Sec. 4.4). Finally, we employ Kalman filter with G_{t-1} and G_t^* for motion blending (Sec. 4.5).

Data preprocessing



Real-time estimation

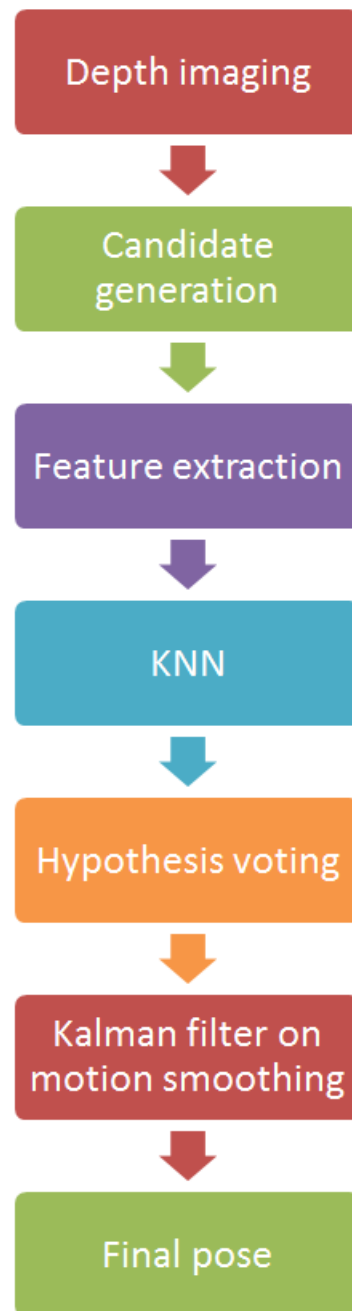


Figure 4.1: Overview of our gesture estimation framework.

4.1. Feature extraction

We employ block histogram feature, inspired by the point cloud concept in [BMB11]. In this section, we detail the feature extraction process of our framework.

For each depth frame I , we project depth values into 3D space and obtain a 3D point cloud $M_I \subseteq R^3$. Let a 3D subspace $S \subseteq R^3$ be a bounding box, comprising all points of M_I . Afterward, we divide X and Y ranges into 4 partitions and divide Z range into 8 partitions. Thus, we can get 128 independent blocks B^{128} in S (Fig 4.2).

$$B^{128} = (b^1, \dots, b^{128}) \subseteq S \quad (1)$$

For each block, we compute sum of point number in such subspace and then get a 128-dimension vector F^{128} , *i.e.*

$$F^{128} = (f^1, \dots, f^{128}) \in N \quad (2)$$

, where

$$f^i = \|m^i\| \quad (3)$$

$$m^i = \{p | p \in b^i, p \in M_I\} \quad (4)$$

Here f^i represent the number of point in block i . Since all hand depth image blocks are normalized, we prune the last 32 bins, which represent point distribution at the bottom blocks. They are easily affected by wrist rotation.

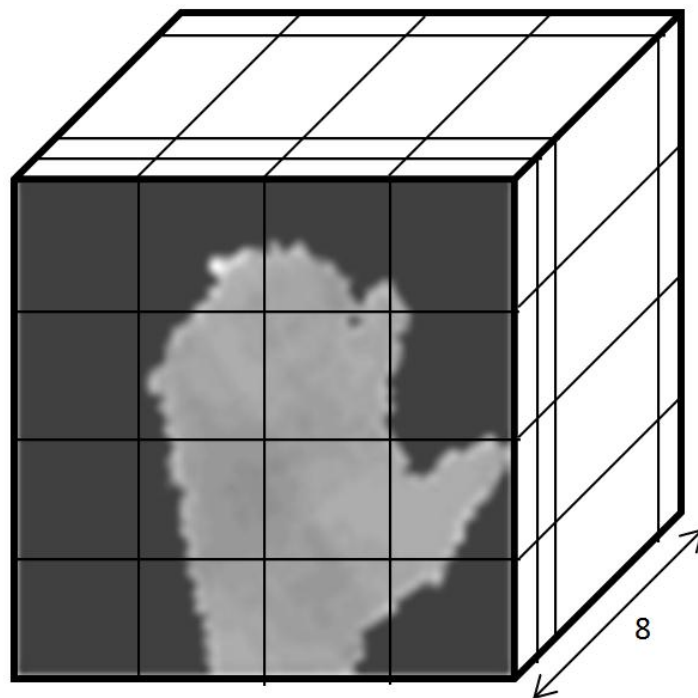


Figure 4.2: Independent blocks for feature extraction.

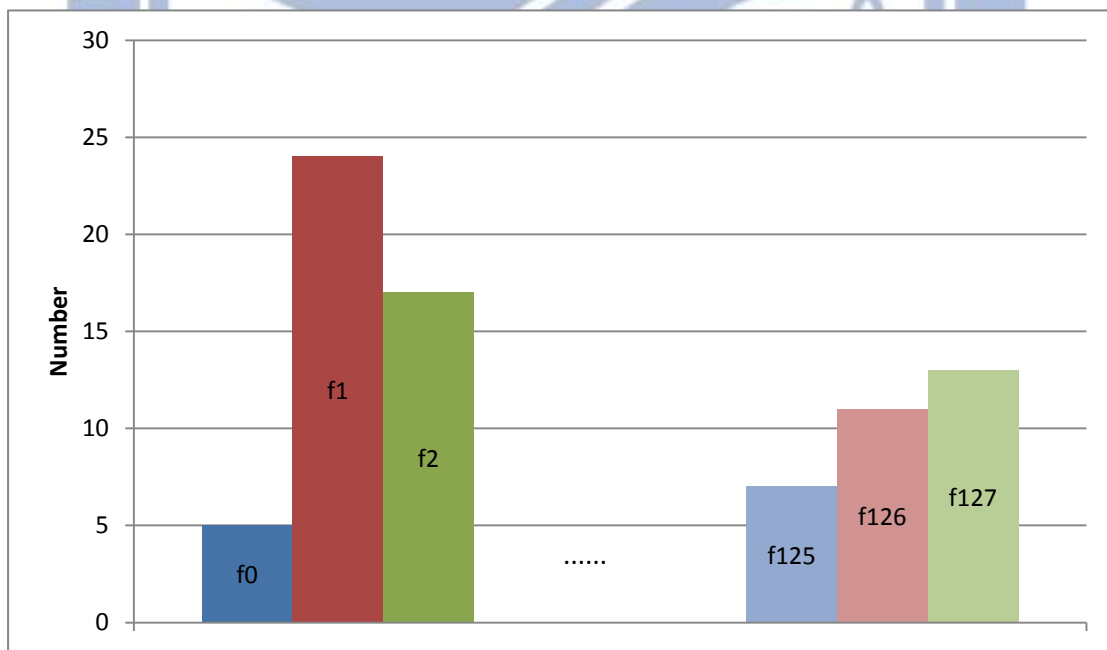


Figure 4.3: A feature vector (point histogram).

4.2. Candidate generation

The novel full-body reconstruction approaches [SFC11, BMB11] produce state-of-the-art results on several data sets. However, there are still some drawbacks, e.g. jiggling position estimate for limbs joints (elbow, wrist, hand, etc.).

To overcome above problem, we introduce a candidate generation method which is based on spatial and temporal relationships. Here a candidate means an image block with 2D central position (u, v) . We define the candidates C_{Total} which consist of $C_{Original}$, $C_{Spatial}$ and $C_{Temporal}$. To generate candidate $C_{Original}$, we obtain the central position $H_x H_y$ from hand joint position of full-body joint estimation.

In spatial part, the candidates $C_{Spatial}$ capture image blocks surround $H_x H_y$. Their central position (u, v) are

$$(u, v) = (H_x + \gamma \cos \theta, H_y + \gamma \sin \theta) \quad (5)$$

$$\theta = 45^\circ, 90^\circ, \dots, 315^\circ, 360^\circ \quad (6)$$

, parameter γ describe an offset from 2D hand joint position.

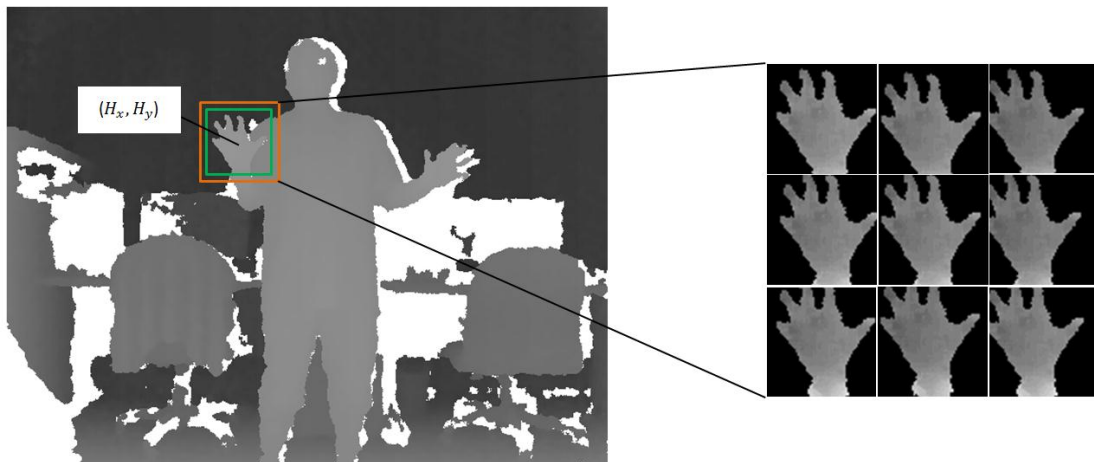


Figure 4.4: Spatial candidate generation.

In temporal part, we use 2D position-velocity model of Kalman filter on previous position and new position estimate to evaluate central position (u,v) of candidate $C_{Temporal}$.

$$\begin{bmatrix} u(n) \\ v(n) \\ \dot{u}(n) \\ \dot{v}(n) \end{bmatrix} = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u(n-1) \\ v(n-1) \\ \dot{u}(n-1) \\ \dot{v}(n-1) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ w_u(n-1) \\ w_v(n-1) \end{bmatrix} \quad (7)$$

$$\begin{bmatrix} z_u(n) \\ z_v(n) \end{bmatrix} = \begin{bmatrix} C_u & 0 & 0 & 0 \\ 0 & C_v & 0 & 0 \end{bmatrix} \begin{bmatrix} u(n) \\ v(n) \\ \dot{u}(n) \\ \dot{v}(n) \end{bmatrix} + \begin{bmatrix} m_u(n) \\ m_v(n) \end{bmatrix} \quad (8)$$

The function $w(n)$ is the change in velocity and the function $m(n)$ means the measurement error.

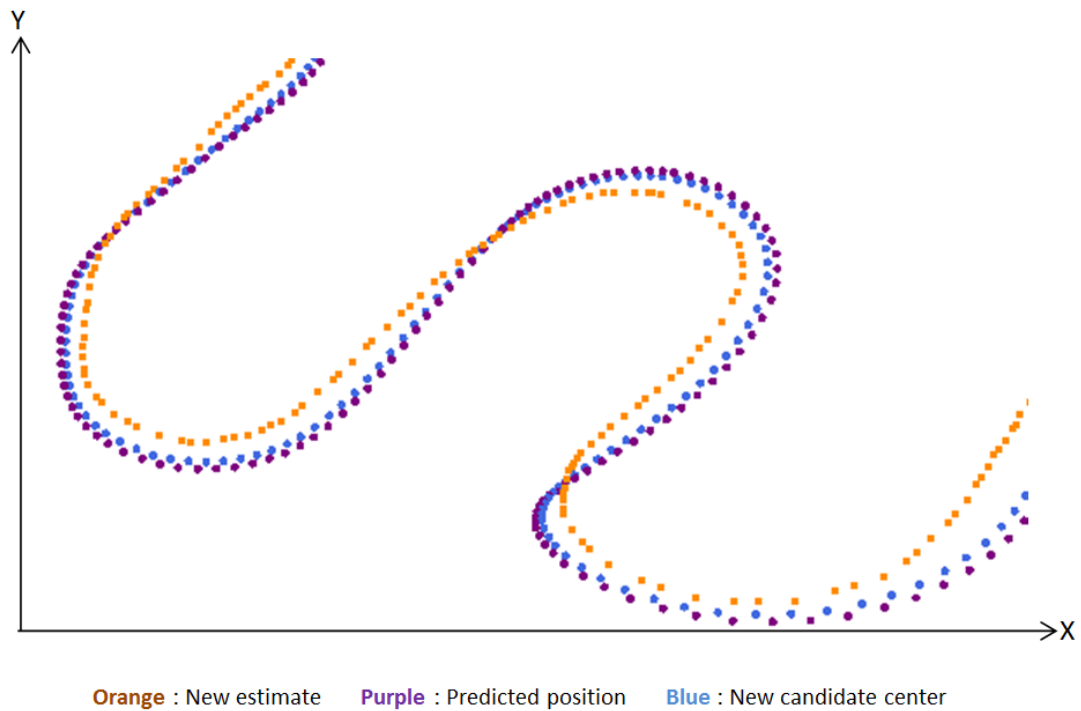


Figure 4.5: Temporal candidate's central position (blue).

These candidates help us to find correct hand position for each depth frame. However, the distance between camera and hand obviously affects hand size on depth image. To address this problem, as the distance change we use different window size to capture hand depth image block and then resize the images with unified resolution.

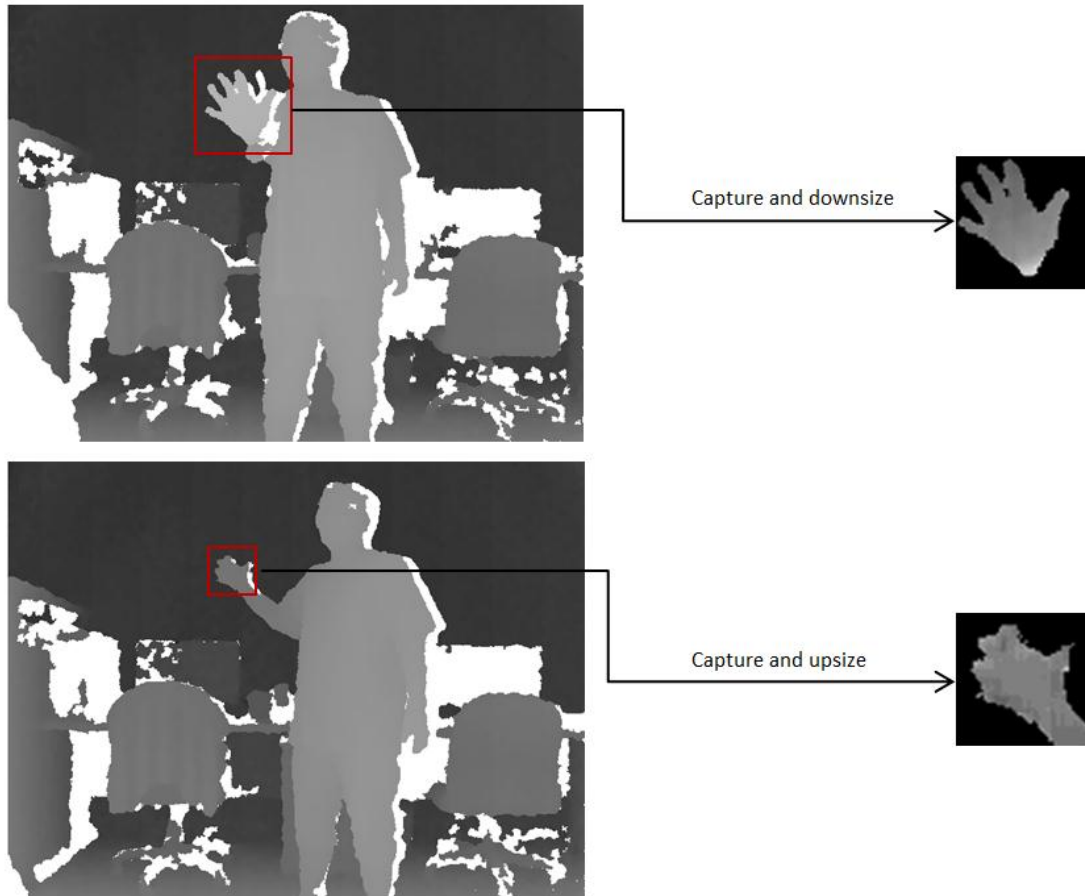


Figure 4.6: Capture with different window size.

4.3. Feature space lookup

As described in Sec. 4.1, we compute features using point histogram. In data preprocessing phase, we capture 150 depth frames per gesture for more flexible result and therefore we obtain several sets of features. These features provide a feature space of our classifier.

In real-time estimation phase, the candidates (Sec. 4.2) often provide correct real hand position against the jiggling problem. Unfortunately, we find that even though the full-body joint estimation gives almost correct hand position estimate, but it provides unstable elbow position estimate in the self-occlusion case. Therefore, the rotation step in the normalization process often inherits incorrect information and creates unpredictable result.

To address this problem, we calculate the angle between normal of XY-plane and the vector from elbow to wrist. If the angle is less than threshold λ , we rotate the captured image by the vector from wrist to hand since the self-occlusion occurred. This helps the classifier to do reliable normalization of all candidates (Fig. 4.7).

Finally, we calculate features F of all normalized candidates and query the feature space for K nearest neighbors (Fig. 4.8). Thus, we obtain classification result ϕ of each feature vector.

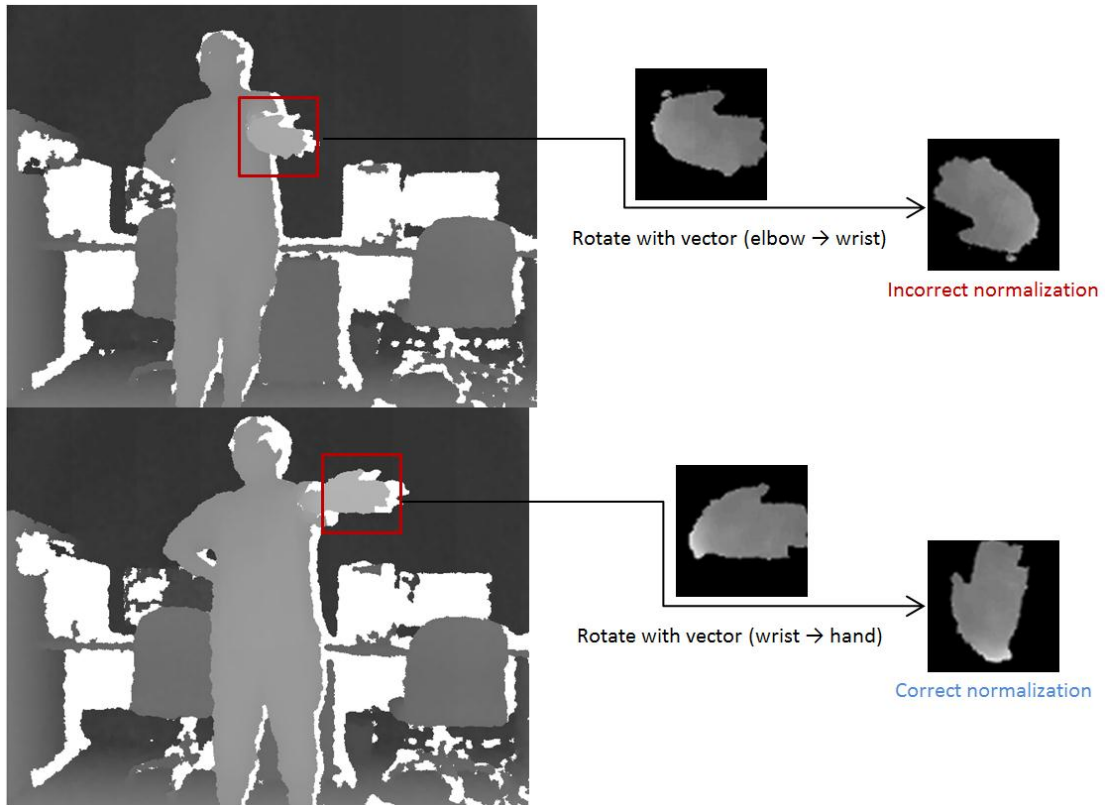


Figure 4.7: Normalization of occlusion case.

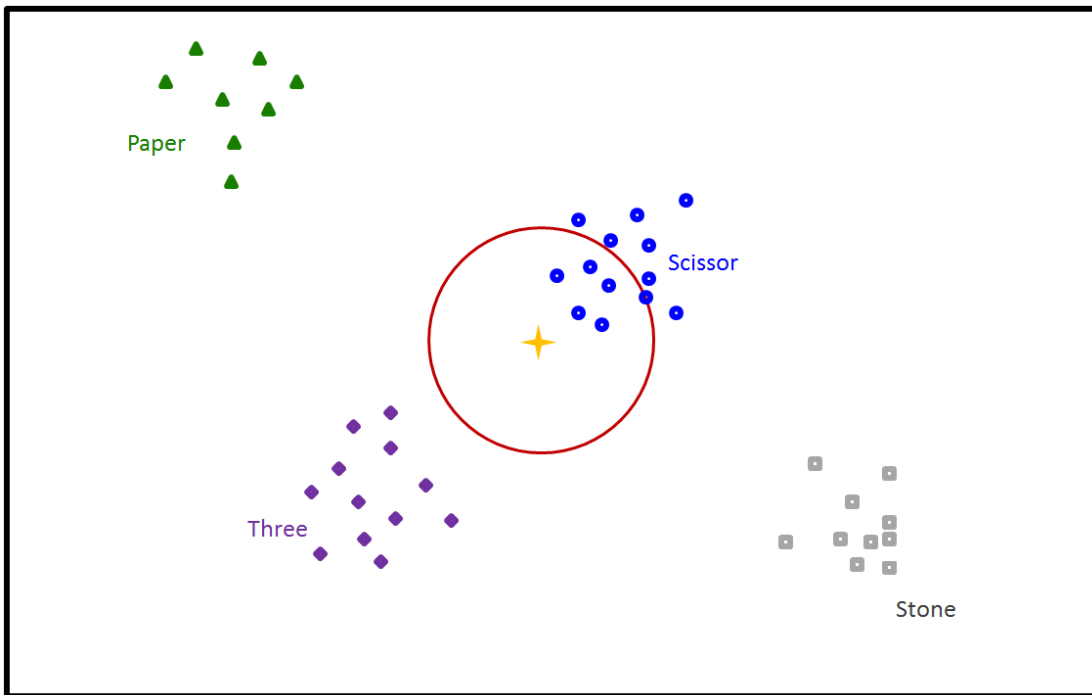


Figure 4.8: Feature space lookup.

4.4. Hypothesis voting

In this section we show how the classification results of candidates are used to find optimized gesture estimate. Firstly, we compute distance values D^* between gestures in data preprocessing phase, *i.e.*

$$D^* = \{d_{ij} | i \in G, j \in G\} \quad (9)$$

$$d_{ij} = \sum_{k \in A} |Angle_{ik} - Angle_{jk}| \quad (10)$$

, where symbol G means total captured gesture, and symbol A describes all hand articulations. These values provide main probability p_{ij} for gesture transformation.

$$p_{ij} = \begin{cases} \varepsilon, i = j \\ \frac{\frac{1}{d_{ij}^2}}{\frac{\sum_{x \neq y} \frac{1}{d_{xy}^2}}{2}}, i \neq j \end{cases} \quad (11)$$

, where ε is a constant for transformation between the same gesture.

In the real-time estimation phase, we compute optimized gesture estimate ϕ^*

$$\phi^* = \operatorname{argmax}_{\phi} V^*(\phi, \delta) \quad (12)$$

$$V^*(\phi, \delta) = \begin{cases} p_{\phi\delta}, \phi \in C_{Original} \\ \lambda * p_{\phi\delta}, \phi \in C_{Spatial} \cup C_{Temporal} \end{cases} \quad (13)$$

, where ϕ mean classification results of all candidates C_{Total} , δ is previous final estimate and λ gives $C_{Spatial}$ and $C_{Temporal}$ lower probability.

4.5. Motion blending

We use a practical set of all gestures to construct the feature space. In order to create finger motion which is not in the gesture set, we propose a modified motion blending method.

Based on optimized estimate ϕ^* (Sec. 4.4), we assume that the gesture with feature ϕ^* is the fundamental of final pose. Rotating the hand articulations by negative angle of normalization produces an appearance G_t^* which is similar to real user's gesture. However, such configuration gives visually discontinuous motion.

To address this problem, we use 1D position-velocity model of Kalman filter on each hand articulation θ of previous final pose G_{t-1} and new gesture configuration G_t^* , *i.e.*

$$\begin{bmatrix} \theta(n) \\ 0 \\ \dot{\theta}(n) \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta(n-1) \\ 0 \\ \dot{\theta}(n-1) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ w_\theta(n-1) \\ 0 \end{bmatrix} \quad (14)$$

$$\begin{bmatrix} z_\theta(n) \\ 0 \end{bmatrix} = \begin{bmatrix} C_\theta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta(n) \\ 0 \\ \dot{\theta}(n) \\ 0 \end{bmatrix} + \begin{bmatrix} m_\theta(n) \\ 0 \end{bmatrix} \quad (15)$$

The function $w(n)$ means the change in velocity and the function $m(n)$ describes the measurement error. A final confidence estimate G_t is given as a combination of smoothly variable articulation θ_t .

5. Experiments

In this chapter we describe the experiments used to verify our method. Firstly, we introduce the detail of our gesture configuration and collection. Afterward, we perform the experiments, involving normalization indispensability and different parameter setting. We demonstrate results on several entertainment scenarios.

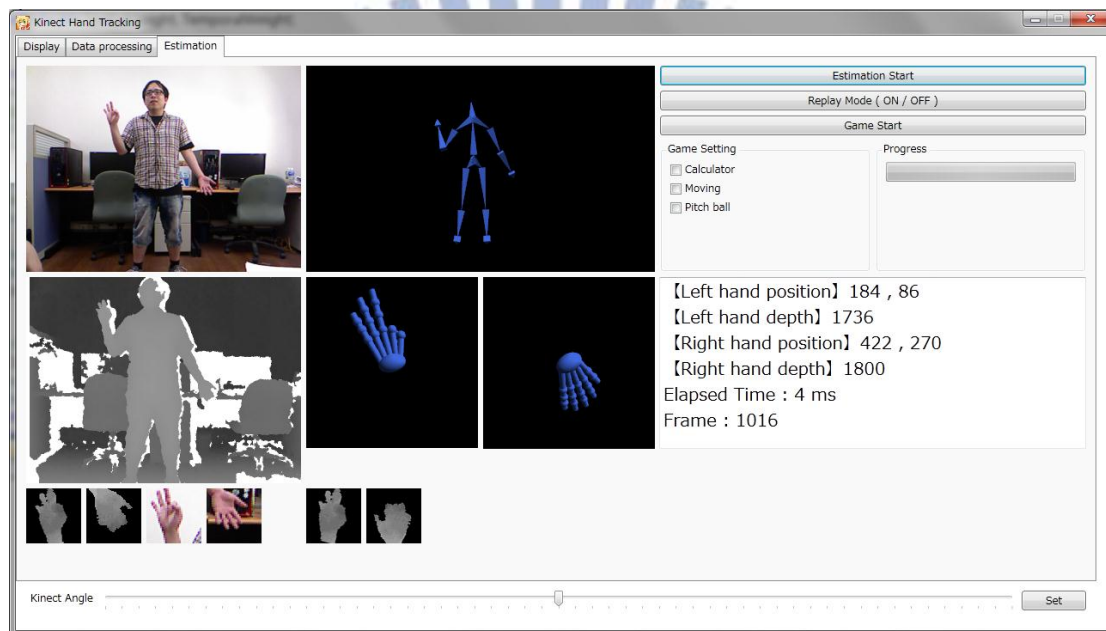
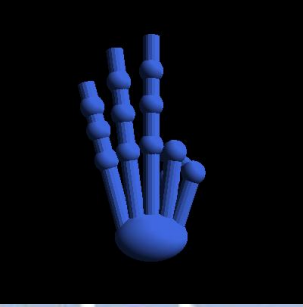
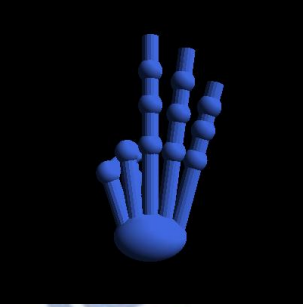
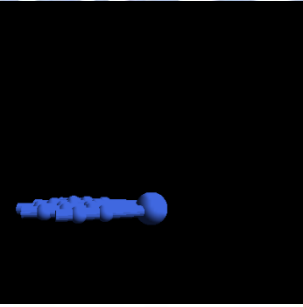
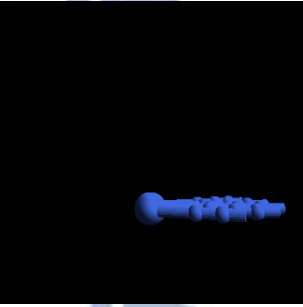
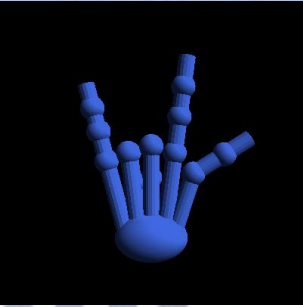
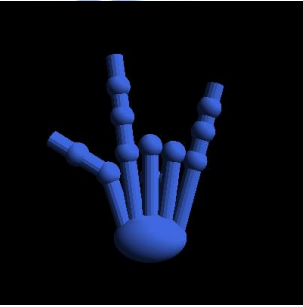
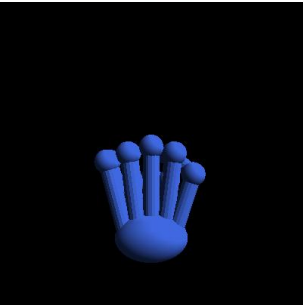
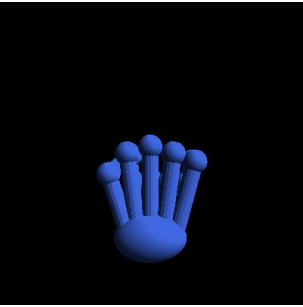
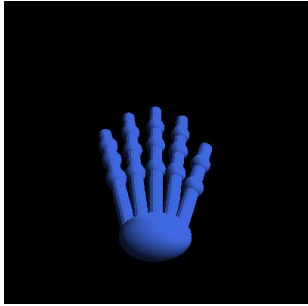
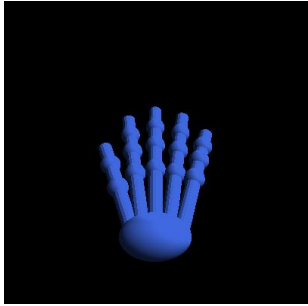
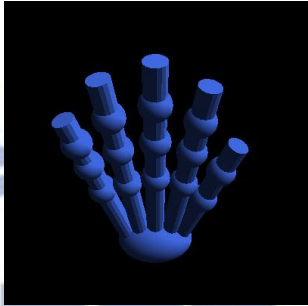
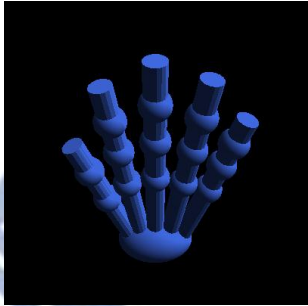
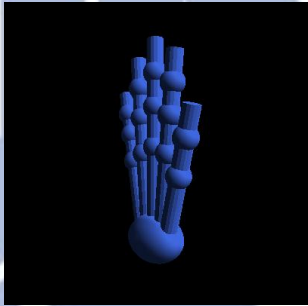
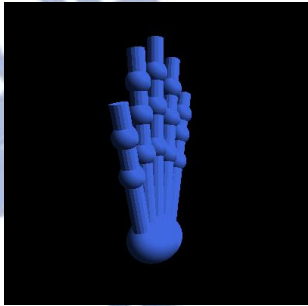
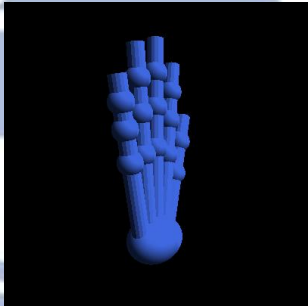
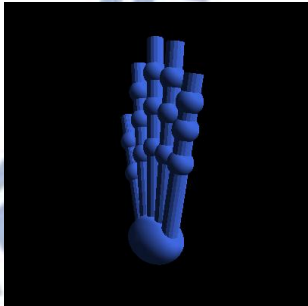
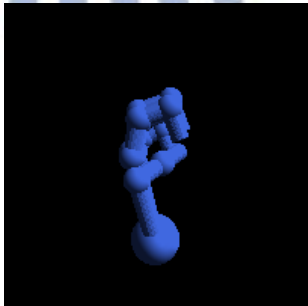
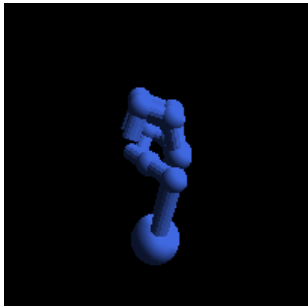


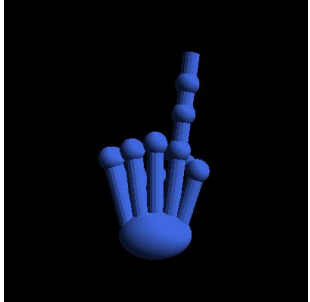
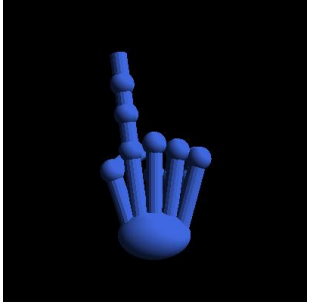
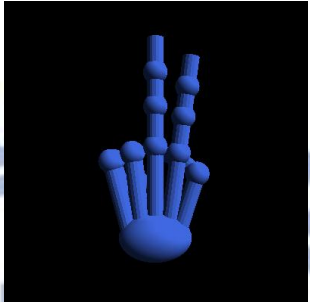
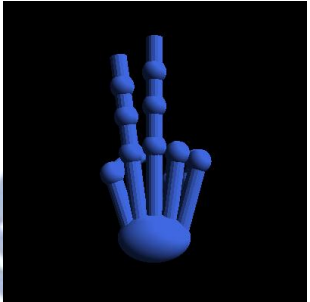
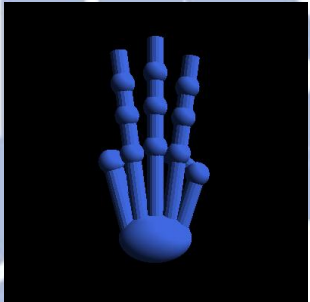
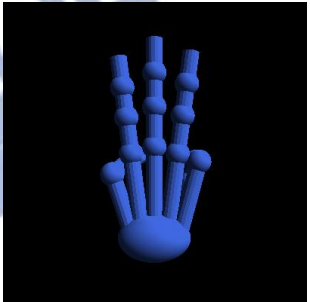
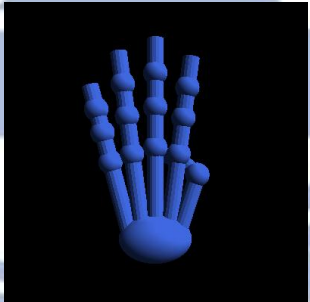
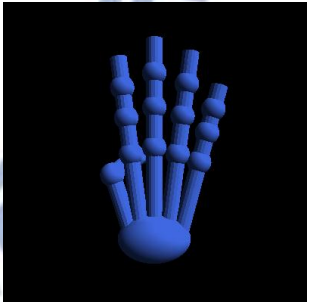
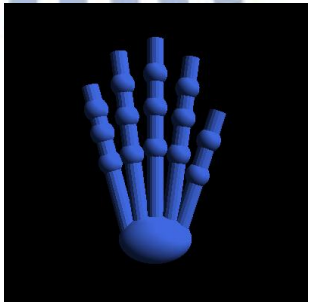
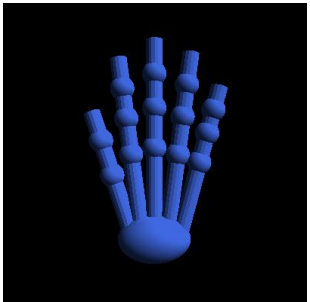
Figure 5.1: A screenshot of our system. The bottom six images show the hand depth images, hand color images and normalized hand depth images respectively. The blue skeleton shows the full-body joints estimated by Microsoft Kinect SDK, and the two blue hands show our finger motion estimation results.

5.1. Test data

We collect a new gesture dataset using a Kinect sensor. As illustrated below, our dataset is composed of 18 common gestures.

Class	Gestures	Articulation configuration (Left / Right)	
Emotion	OK		
	No idea		
	Shining		
	Stone		

Direction	Forward		
	Backward		
	Inside		
	Outside		
Number	Zero		

	One		
	Two		
	Three		
	Four		
	Five		

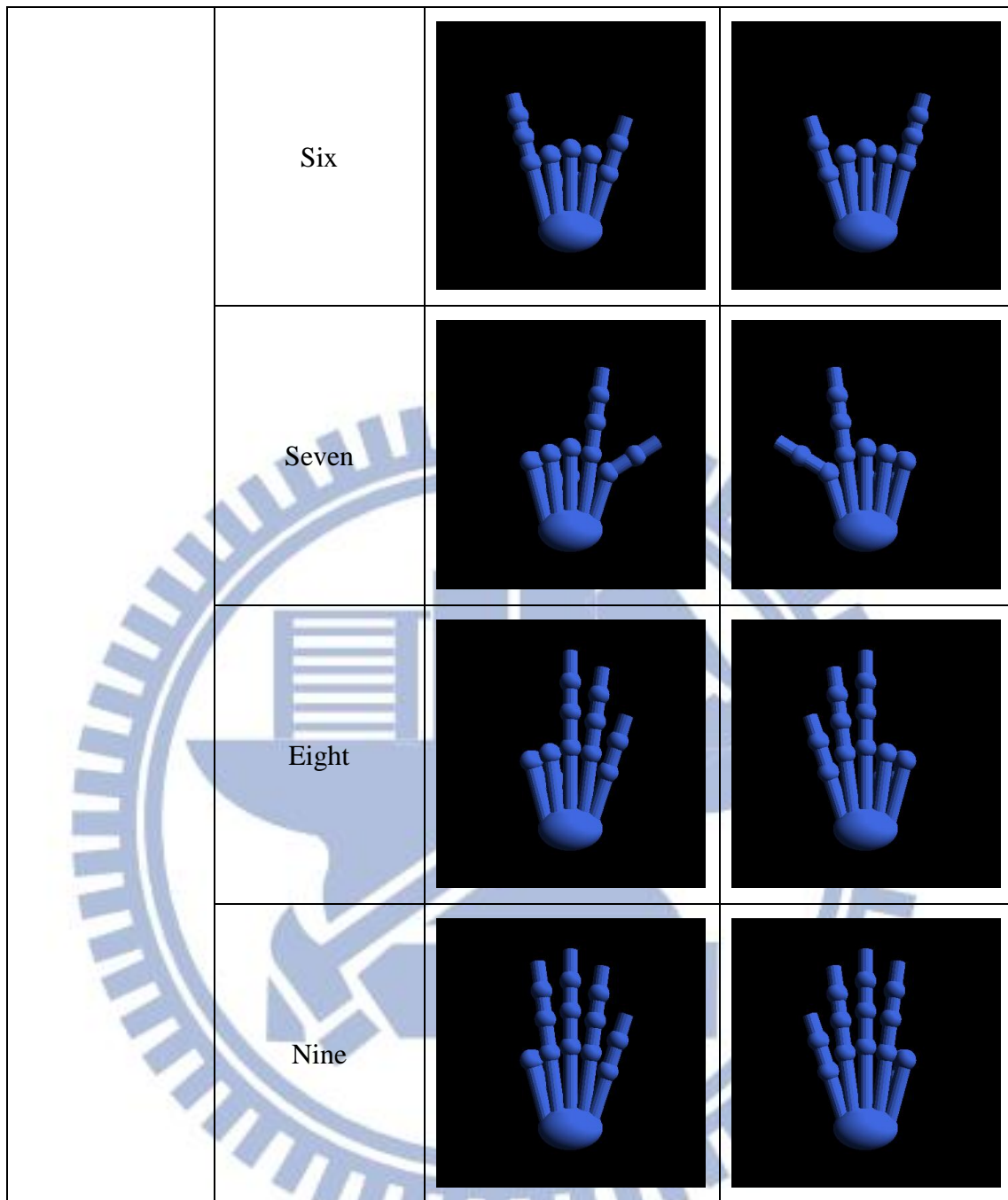


Figure 5.2: Our used gestures. We capture 150 depth frames per gesture and all depth frames in the dataset is normalized.

5.2. Estimation accuracy

In this section, we investigate the effect of several system features on estimation accuracy. By the way, we implemented a KD-tree structure on the feature space for faster KNN searching. Thus, we also verify the performance progress and the effect of search depth to accuracy.

Recognition accuracy. We test all 18 gestures through about 30 seconds estimation per gesture to verify the recognition accuracy. In particular, all test cases were operated 1.8m from the depth camera. This shows the ability against the noisy input and the application use. In Fig. 5.3 we show the recognition success rate respectively.

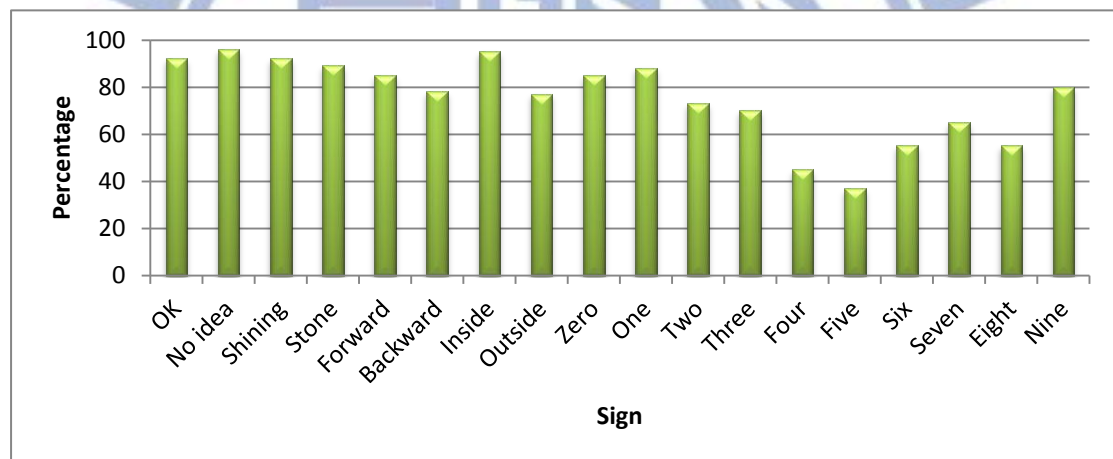


Figure 5.3: Recognition accuracy (User 1).

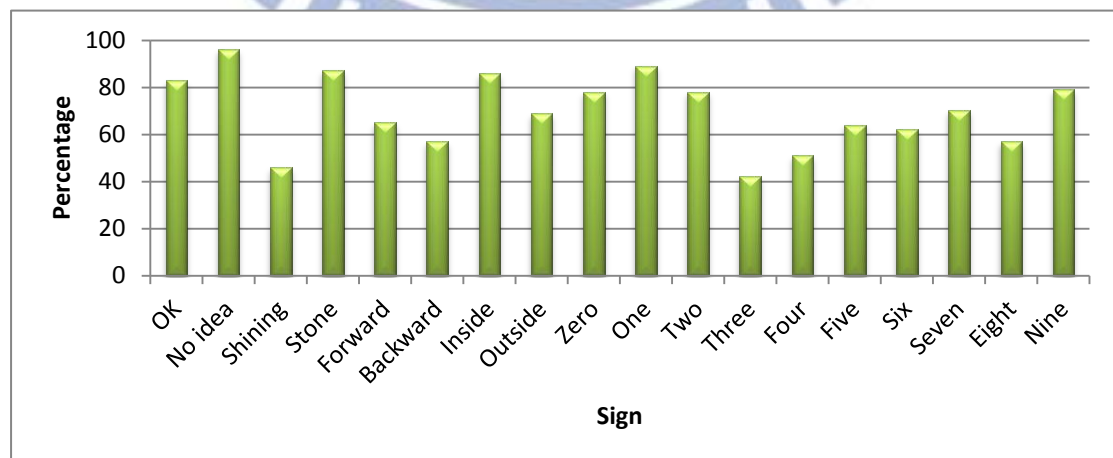


Figure 5.4: Recognition accuracy (User 2).

Candidates' contribution. Here we show the necessity of spatial candidate and temporal candidate respectively. When candidate $C_{Original}$ generated a failed estimation, the spatial candidate $C_{Spatial}$ and the temporal candidate $C_{Temporal}$ sometimes provide other estimation which helps the system to get correct result.

Fig. 5.5 and Fig. 5.6 show the contribution rate of OK sign when user is at static state or dynamic state respectively. In the same way, Fig. 5.7 and Fig. 5.8 show the contribution rate of Five sign.

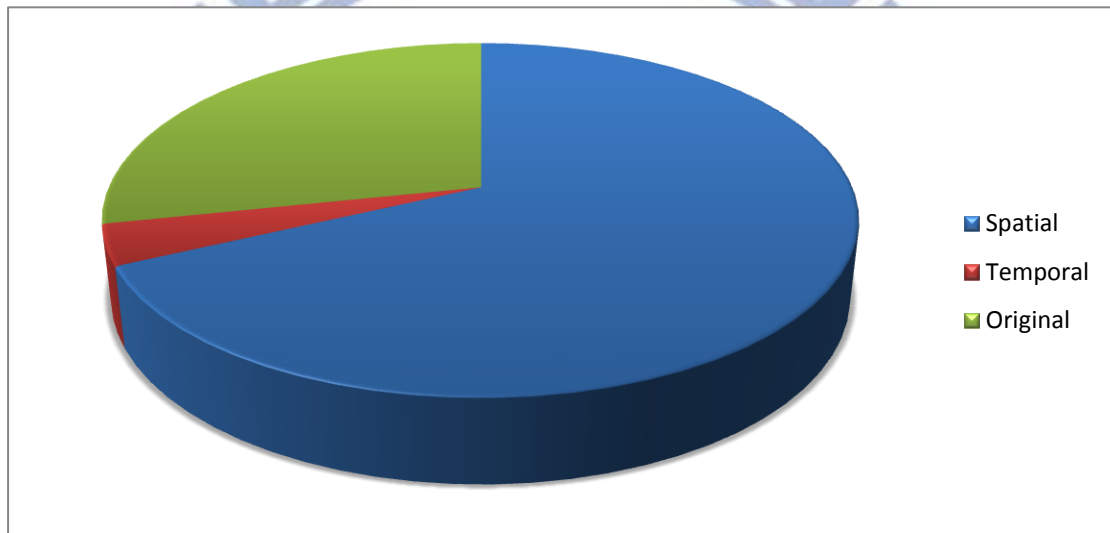


Figure 5.5: The distribution of the best-matched candidate for static OK sign.

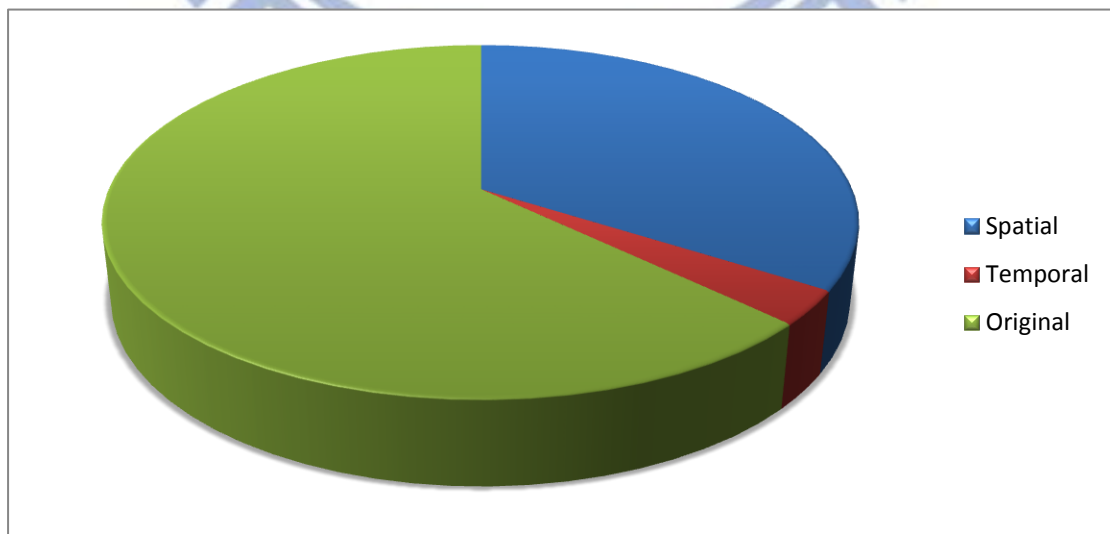


Figure 5.6: The distribution of the best-matched candidate for dynamic OK sign.

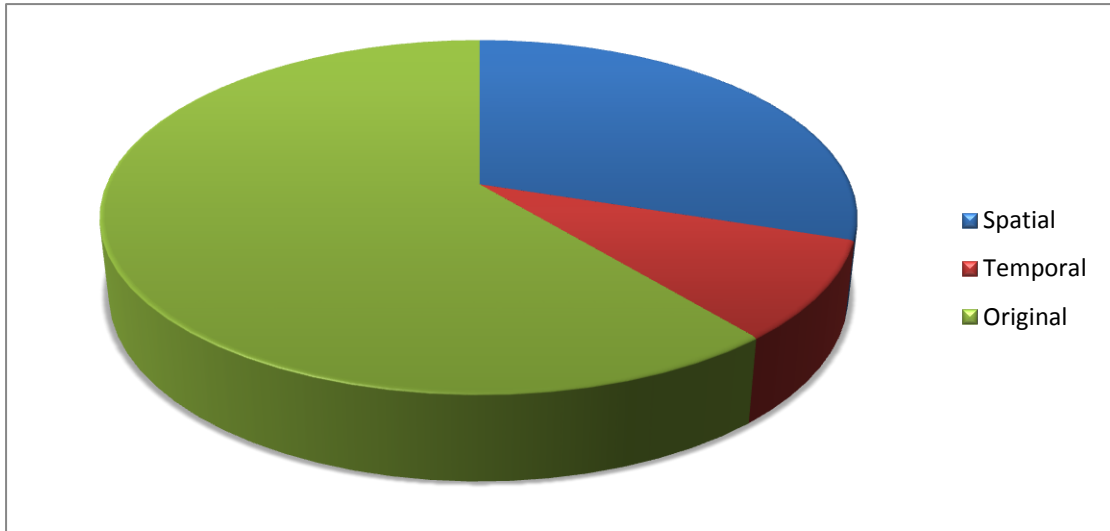


Figure 5.7: The distribution of the best-matched candidate for static Five sign.

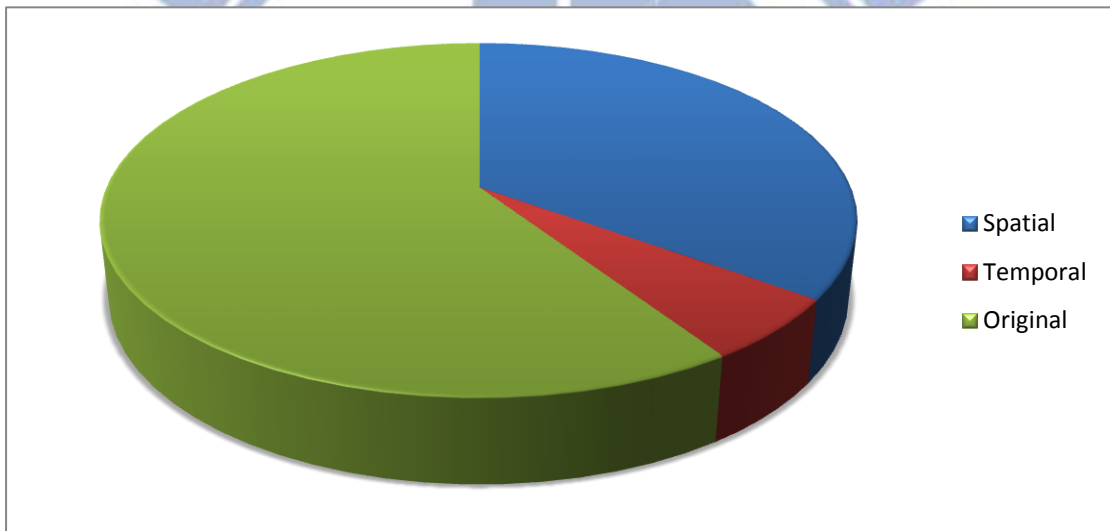


Figure 5.8: The distribution of the best-matched candidate for dynamic Five sign.

Performance. Since linear search usually generate slower performance, we implement a KD-tree structure on the feature space for faster KNN searching. Fig. 5.9 illustrates the searching time variation against the search depth. In Fig. 5.10, we show the accuracy convergence of search depth.

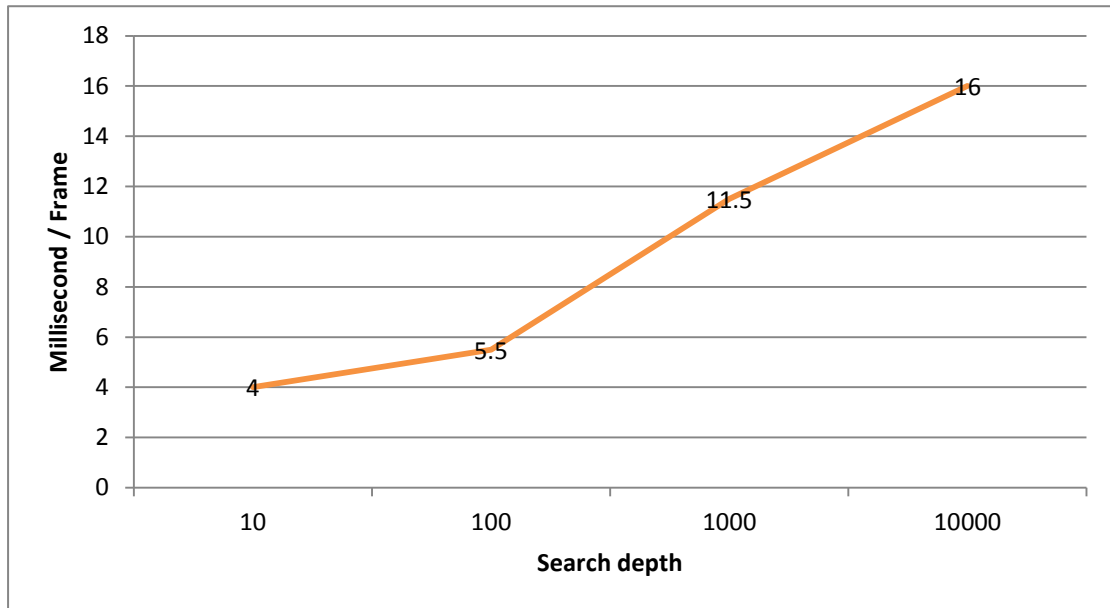


Figure 5.9: Performance comparison.

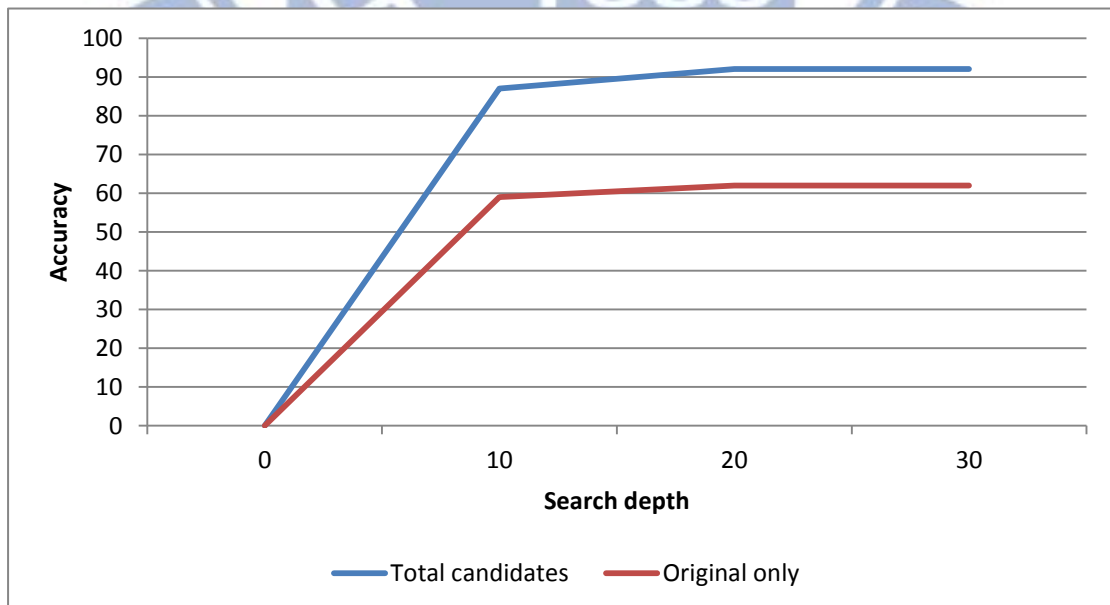
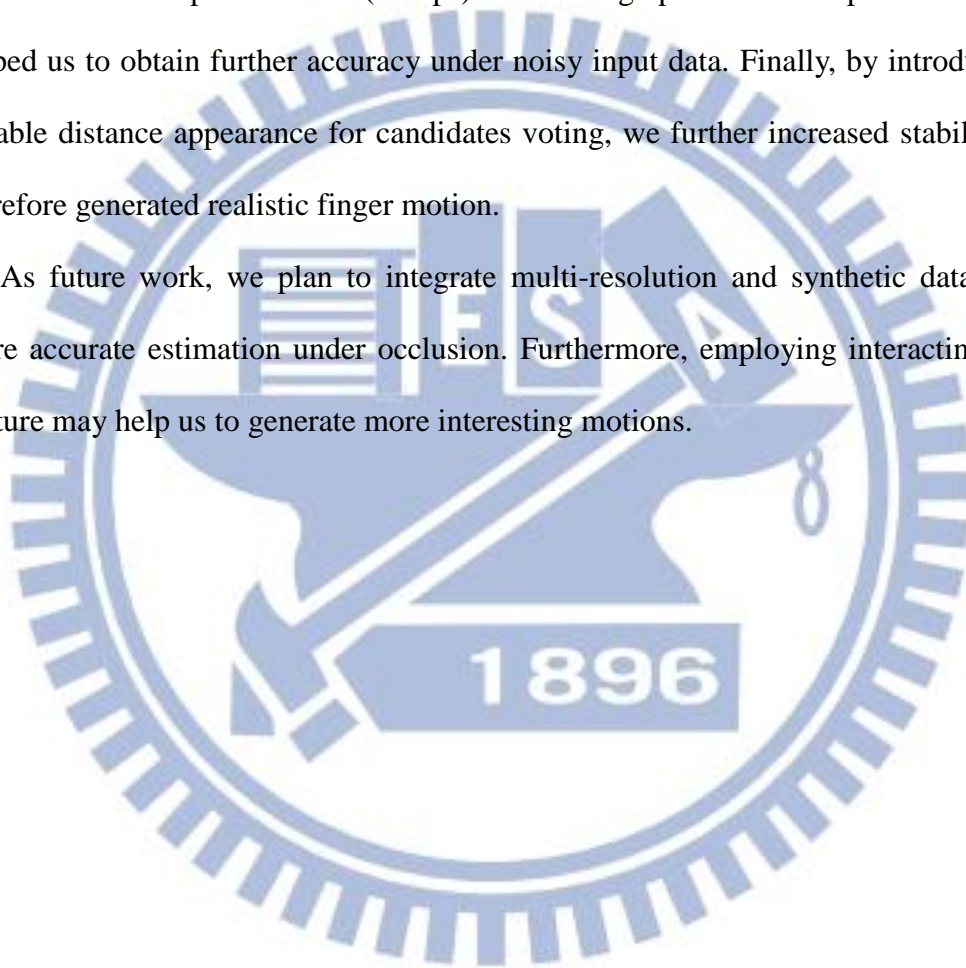


Figure 5.10: Accuracy convergence of search depth.

6. Discussion

In this thesis, we present how we obtain efficient finger motion estimation from noisy depth image. We introduced gesture recognition as a feature space lookup problem for finger motion estimation. Using a simple feature allowed us to generate a discriminative feature space using normalized depth images without overfitting, and enabled real-time performance (200fps). Generating spatial and temporal candidates helped us to obtain further accuracy under noisy input data. Finally, by introducing a reliable distance appearance for candidates voting, we further increased stability and therefore generated realistic finger motion.

As future work, we plan to integrate multi-resolution and synthetic dataset for more accurate estimation under occlusion. Furthermore, employing interacting hand gesture may help us to generate more interesting motions.



References

- [AAS04] V. Athitsos, J. Alon, S. Sclaroff, and G. Kollios. BoostMap: A method for efficient approximate similarity rankings. In Proc. Computer Vision and Pattern Recognition (CVPR), vol. 2, 268–275, 2004.
- [AS03] V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In Proc. Computer Vision and Pattern Recognition (CVPR), vol. 2, 432–439, 2003.
- [AT04] A. Agarwal and B. Triggs. 3D human pose from silhouettes by relevance vector regression. In Proc. Computer Vision and Pattern Recognition (CVPR), 2004.
- [ATC05] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, and A. Ng. Discriminative learning of markov random fields for segmentation of 3D scan data. In Proc. Computer Vision and Pattern Recognition (CVPR), 2005.
- [BEK09] A. Bleiweiss, E. Eilat, and G. Kutliroff. Markerless motion capture using a single depth sensor. In SIGGRAPH ASIA Sketches, 2009.
- [BM98] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In Proc. Computer Vision and Pattern Recognition (CVPR), 1998.
- [BM09] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3D human pose annotations. In Proc. International Conference on Computer Vision (ICCV), 2009.
- [BMB11] A. Baak, M. Müller, G. Bharaj, H. Seidel, and C. Theobalt. A Data-Driven Approach for Real-Time Full Body Pose Reconstruction from a Depth Camera. In Proc. International Conference on Computer Vision (ICCV), 2011.

- [DDH06] G. Dewaele, F. Devernay, R. Horaud, and F. Forbes. The alignment between 3-d data and articulated shapes with bending surfaces. In Proc. European Conference on Computer Vision (ECCV), 578–591, 2006.
- [FH05] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision (IJCV)*, 61(1):55–79, Jan. 2005.
- [GPT10] V. Ganapathi, C. Plagemann, S. Thrun, and D. Koller. Real time motion capture using a single time-of-flight camera. In *Computer Vision and Pattern Recognition (CVPR)*, 2010.
- [GSK11] R. Girshicky, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient Regression of General-Activity Human Poses from Depth Images. In *Proc. International Conference on Computer Vision (ICCV)*, 2011.
- [GWK05] D. Grest, J. Woetzel, and R. Koch. Nonlinear body pose estimation from depth images. In *Proc. Deutsche Arbeitsgemeinschaft für Mustererkennung (DAGM)*, 2005.
- [HRM12] L. Hoyet, K. Ryall, R. McDonnell, and C. O’Sullivan. Sleight of Hand: Perception of Finger Motion from Reduced Marker Sets. In *Proc. Interactive 3D Graphic (I3D)*, 2012.
- [IF01] S. Ioffe and D. Forsyth. Probabilistic methods for finding people. *International Journal of Computer Vision (IJCV)*, 43(1):45–68, 2001.
- [KBK10] A. Kolb, E. Barth, R. Koch, and R. Larsen. Time-of-flight sensors in computer graphics. In *proc. Computer Graphics Forum (CGF)*, 29(1):141–159, 2010.
- [KHS10] E. Kalogerakis, A. Hertzmann, and K. Singh. Learning 3D mesh segmentation and labeling. *ACM Trans. Graphics*, 29(3), 2010.

- [MM03] G. Mori and J. Malik. Estimating human body configurations using shape context matching. In Proc. International Conference on Computer Vision (ICCV), 2003.
- [NFC07] R. Navaratnam, A.W. Fitzgibbon, and R. Cipolla. The joint manifold model for semi-supervised multi-valued regression. In Proc. International Conference on Computer Vision (ICCV), 2007.
- [OKA12] I. Oikonomidis, N. Kyriazis, and A. Argyros. Tracking the Articulated Motion of Two Strongly Interacting Hands. In Proc. Computer Vision and Pattern Recognition (CVPR), 2012.
- [OS08] R. Okada and B. Stenger. A single camera motion capture system for human-computer interaction. The Institute of Electronics, Information and Communication Engineers (IEICE), E91-D:1855–1862, 2008.
- [PGK10] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun. Real-time identification and localization of body parts from depth images. In Proc. International Conference on Robotics and Automation (ICRA), 2010.
- [PY06] J. Park and Y. Yoon. LED-glove based interactions in multi-modal displays for teleconferencing. In International Conference on Artificial Reality and Telexistence (ICAT), 2006.
- [RF03] D. Ramanan and D. Forsyth. Finding and tracking people from the bottom up. In Proc. Computer Vision and Pattern Recognition (CVPR), 2003.
- [RRR08] G. Rogez, J. Rihan, S. Ramalingam, C. Orrite, and P. Torr. Randomized trees for human pose detection. In Proc. Computer Vision and Pattern Recognition (CVPR), 2008.
- [RS00] R. Rosales and S. Sclaroff. Inferring body pose without tracking body parts. In Computer Vision and Pattern Recognition (CVPR), pages 721–727, 2000.

- [RSH05] L. Ren, G. Shakhnarovich, J. Hodgins, H. Pfister, and P. Viola. Learning silhouette features for control of human motion. *ACM Transactions on Graphics* 24, 4, 1303–1331, 2005.
- [RYZ11] Z. Ren, J. Yuan, and Z. Zhang. Robust Hand Gesture Recognition Based on Finger-Earth Mover’s Distance with a Commodity Depth Camera. In *Multimedia*, 2011.
- [SBR04] L. Sigal, S. Bhatia, S. Roth, M. Black, and M. Isard. Tracking loose-limbed people. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2004.
- [SFC11] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from a single depth image. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2011.
- [SM10] M. Siddiqui and G. Medioni. Human pose estimation from a single view point, real-time range sensor. In *Computer Vision for Computer Games (CVCG) at Computer Vision and Pattern Recognition (CVPR)*, 2010.
- [SMF04] E. Sudderth, M. Mandel, W. Freeman, and A. Willsky. Distributed occlusion reasoning for tracking with nonparametric belief propagation. In *Neural Information Processing Systems (NIPS)*, 2004.
- [STT06] B. Stenger, A. Thayananthan, P. Torr, and R. Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, 28, 9, 1372–1384, 2006.
- [SVD03] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *Proc. International Conference on Computer Vision (ICCV)*, 750–757, 2003.
- [TU08] Z. Tu. Auto-context and its application to high-level vision tasks. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2008.

- [UD08] R. Urtasun and T. Darrell. Local probabilistic regression for activity-independent human pose inference. In Proc. Computer Vision and Pattern Recognition (CVPR), 2008.
- [WP09] R. Wang and J. Popović. Real-time hand-tracking with a color glove. ACM Trans. Graphics, 2009.
- [WPP11] R. Wang, S. Paris, and J. Popović. 6D Hands: Markerless Hand Tracking for Computer Aided Design. In User Interface Software and Technology (UIST), 2011.
- [ZF07] Y. Zhu and K. Fujimura. Constrained optimization for human pose estimation from depth sequences. In Proc. Asian Conference on Computer Vision (ACCV), 2007.
- [KINECT] Microsoft Kinect for Windows
<http://www.microsoft.com/en-us/kinectforwindows/>

