Spring 2015

American Sign Language alphabet recognition using Microsoft Kinect

Cao Dong

Department:

AMERICAN SIGN LANGUAGE ALPHABET RECOGNITION USING MICROSOFT KINECT

by

CAO DONG

A THESIS

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

2015

Approved by

Ming C. Leu, Advisor

Zhaozheng Yin

Fuwen (Frank) Liou

ABSTRACT

American Sign Language (ASL) fingerspelling recognition using marker-less vision sensors is a challenging task due to the complexity of ASL signs, self-occlusion of the hand, and limited resolution of the sensors. This thesis describes a new method for ASL fingerspelling recognition using a low-cost vision camera, which is Microsoft's Kinect. A segmented hand configuration is first obtained by using a depth contrast feature based per-pixel classification algorithm. Then, a hierarchical mode-finding method is developed and implemented to localize hand joint positions under kinematic constraints. Finally, a Random Decision Forest (RDF) classifier is built to recognize ASL signs according to the joint angles. To validate the performance of this method, a dataset containing 75,000 samples of 24 static ASL alphabet signs is used. The system is able to achieve a mean accuracy of 92%. We have also used a publicly available dataset from Surrey University to evaluate our method. The results have shown that our method can achieve higher accuracy in recognizing ASL alphabet signs in comparison to the previous benchmarks.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude towards my advisor, Dr. Ming C. Leu, for his support, advice and guidance throughout my graduate studies. I would like to thank Dr. Zhaozheng Yin for his constant help and patient advice in this research project. I would also like to thank Dr. Frank Liou for spending his time and effort as my committee members and helping me through the course of studies.

I would like to thank Dr. Randy Moss and Jason Hagerty, who have helped me greatly during my course study, project work and research work. I would like to thank Dr. Douglas Bristow, Dr. N. Balakrishnan and Dr. Serhat Hosder for their help through the course of studies. I would like to thank all my research mates in VR&AM lab for their help during my graduate study.

Finally, I am deeply indebted to my parents for their support, encouragement and all my friends who stood by me during the entire duration of my studies at Missouri University of Science and Technology.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

American Sign Language (ASL) is a complete sign language system that is widely used by deaf individuals in the United States and the English-speaking part of Canada. ASL speakers can communicate with each other conveniently hand gestures. However, communicating with deaf people is still a problem for non-sign-language speakers. There are some professional interpreters that can serve deaf people by real-time sign language interpreting, but the cost is usually high. Moreover, such interpreters are often not available. Therefore, an automatic ASL recognition system is highly desirable.

Researchers have been working on sign language recognition systems using different kinds of devices for decades. Sensor-based devices, such as cyber-glove [1, 2], can be used to obtain hand gesture information precisely. However, these devices were difficult to use outside of laboratories because of unnatural user experience, difficulties in setting up the system, and high costs. Vision-based devices can provide natural user experience, but the gesture recognition accuracy is always limited. The recent availability of low-cost, high-performance sensing devices, such as the Microsoft Kinect, has made vision-based ASL recognition potentially. As a result, ASL and other hand gesture recognition using such devices has raised high interests in the past few years [6, 7].

The most common approach to recognize hand gestures using vision-based sensors is to extract low-level features from RGB or depth images using image feature transform, and then employ statistical classifiers to classify gestures according to the features. A series of feature extraction methods have been developed and implemented, such as Scale-invariant Feature Transform (SIFT) [8, 9], Histogram of Oriented Gradients (HOG) [4, 5, 10], Wavelet Moments [11], and Gabor Filters (GF) [12, 13].

Typical classifiers include Artificial Neural Networks (ANN), Support Vector Machines (SVM), and Decision Trees (DT). These methods are robust in recognizing a small number of simple hand gestures. For example, in [9], 96.23% accuracy was reported in recognizing six custom signs using SIFT-based bag-of-features and a SVM classifier. However, when it comes to ASL sings, which are complex and large in number, these methods are usually not able to achieve desirable accuracies. In [12], a Gabor Filter based method was implemented to recognize 24 static ASL alphabet signs, resulting in only 75% mean accuracy and high confusion rates between similar signs such as "r" and "u" (17% confusion rate).

Other methods have also been developed and implemented to estimate hand poses and recognize hand gestures. Lasonas et al. [16] developed a model-based approach that can recover hand pose by matching a 3D hand model to the hand's image. Yeo et al. [18] proposed a contour shape analysis method that can recognize 9 simple custom hand gestures with 86.66% accuracy. Qin et al. [19] attempted to recognize 8 direction-pointing gestures using convex shape decomposition method based on the Radius Morse function, which achieved 91.2% accuracy. Ren et al. [21] proposed a part-based hand gesture recognition method that parsed fingers according to the contour shape of the hand. There were 14 hand gestures containing 10 digits and 4 elementary arithmetic symbols recognized with 93.2% accuracy. Dominio et al. [20] combined multiple depth-based descriptors for hand gesture recognition. The descriptors included the hand region's edge distance and elevation, the curvature of the hand's contour, and the displacement of the samples in the palm region. An SVM classifier was employed to classify gestures and achieved 93.8% accuracy in an experiment to reconize 12 static

ASL alphabet and digit signs. Still, these above methods can only recognize a small number (less than 15) of simple gestures (custom signs, ASL digits, or a small portion of ASL alphabet signs).

Shotton et al. [22] proposed a significant approach that segmented the human body pixel by pixel into different parts using depth contrast features and a Random Decision Forest (RDF) classifier. This method was successfully implemented in the Kinect system to estimate human body poses. Keskin et al. [24] adapted Shotton's method to segment hand into parts, and successfully recognized 10 ASL digit signs by mapping joint coordinates to know hand gestures, resulted in 99.96% accuracy. Liang et al. [25] improved the per-pixel based hand parsing method by employing a distance-adaptive feature candidates selecting scheme and super-pixel partition-based Markov Random Fields (MRF). The improved algorithm achieved 17% (89% vs 72%) higher accuracy in per-pixel classification.

The recent achievements [22, 24, 25] based on the per-pixel classification algorithm have shown a high potential of recognizing a large number of complex hand gestures, such as the ASL alphabets. Comparing to the low-level image features [8, 9, 10, 11, 12, 13], the depth comparison features contain more informative descriptions of both the 2D shape and the depth gradients in the context of each pixel. The RDF model is also robust in multi-class classification on a large dataset. However, the existing per-pixel classification algorithm can only segment a hand's region into parts but does not have the capability to recognize hand gestures. Thus, we developed a method to recognize hand gestures according to the pixels' classification.

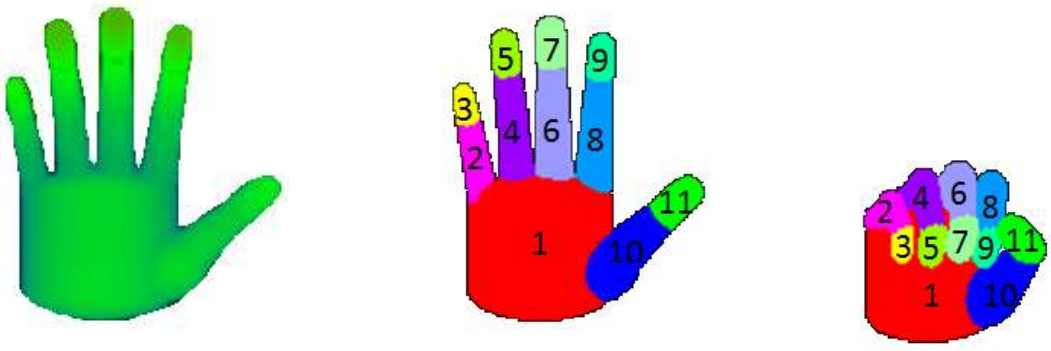
This thesis study focused on the method of recognizing complex hand gestures using pixels' classifications information. We implemented the depth comparison features [22] with distance-adaptive scheme (DAS) [25] and the RDF classifier to segment the hand's region into parts. In addition, we proposed a novel hierarchical mode finding method to localize joints under kinematic constraints. Then, a hand gesture recognition method using high-level features of joint angles was developed, which achieved high recognition accuracy to recognize 24 ASL alphabets (except dynamic gestures "j" and "z"). We have also evaluated our method using the public dataset [12].

The paper is presented as follows. Section 2 introduces the process of hand parts segmentation. Section 3 explains the methodology of joint localization and gesture recognition. Section 4 presents and discusses the experimental results. Section 5 draws the conclusions of the study.

2. HAND PART SEGMENTATION

The per-pixel classification method [22] was adapted to segment the hand into parts. The input of this process was the depth image of the hand region (Figure 2.1a), and the output was the classification label of each pixel (Figure 2.1b and Figure 2.1c). The hand was segmented into 11 parts: the palm, 5 lower finger sections and 5 fingertips. The proximal phalanges and the intermediate phalanges are merged for the following reasons:

- The limited resolution of the Kinect sensor made it difficult to segment proximal and intermediate phalanges accurately.
- The positions of the proximal and intermediate phalanges were constrained by the corresponding distal phalange's (fingertip's) position and the palm's position, so it is not necessary to segment them.
- A smaller number of parts will result in higher computational efficiency.



(a) (b) (c)

Figure 2.1. Illurstration of hand part segmentation tasks. (a) Input hand depth image. (b) Labeled hand parts of an open palm. (c) Labeled hand parts of a closed palm.

The hand parts segmentation process contains both off-line training and online classification; see Figure 2.2. The data acquisition and pre-processing is briefly described in Section 2.1. The method of generating training data is explained in Section 2.2. The feature used for per-pixel classification is introduced in Section 2.3. The classifier's training and classifying process is described in Section 2.4.

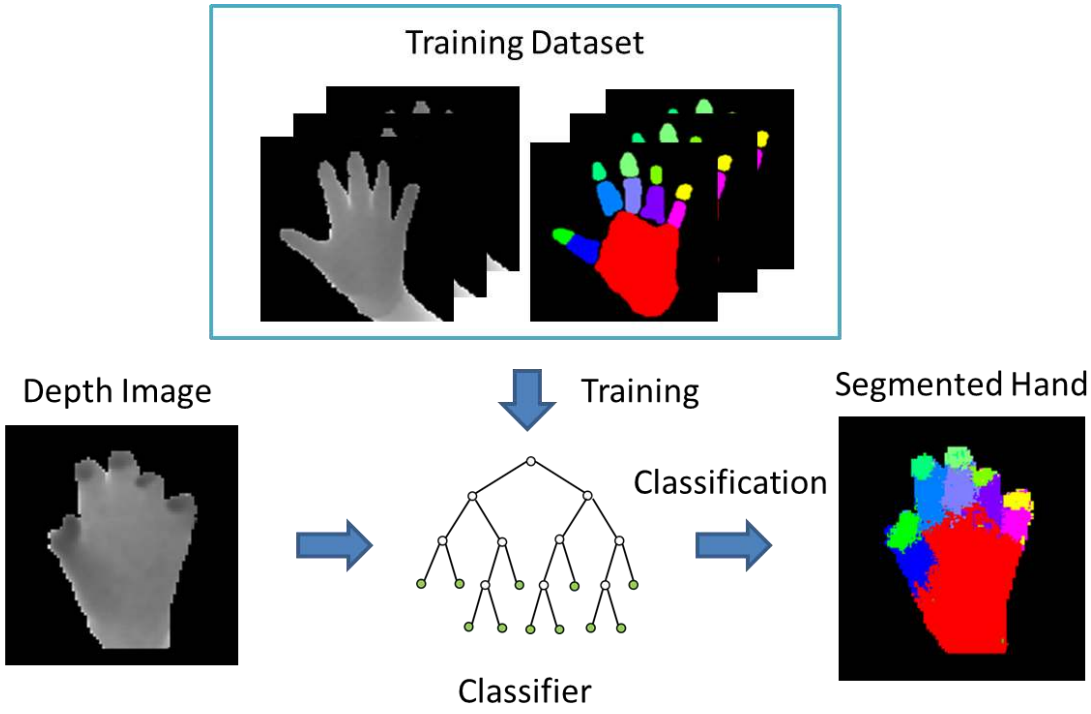


Figure 2.2. Hand part segmentation. The training dataset contains depth images and the ground truth configurations of the hand's parts. The classifier trained using the training dataset can segment the input depth image into hand parts pixel by pixel.

2.1. DATA ACQUISITION AND PRE-PROCESSING

The raw data obtained from the Kinect sensor via the Natural User Interface (NUI) contained 512×424 depth data, 1920×1080 RGB data, and 26-joint body skeleton data.

The hand region in the depth image was described using spatial thresholds in X-axis direction $[T_{x\_min}, T_{x\_max}]$, Y-axis direction $[T_{y\_min}, T_{y\_max}]$ and Z (depth)-axis direction $[T_{Depth\_min}, T_{Depth\_max}]$. As illustrated in Figure 2.3, the Kinect depth sensor located at position $\boldsymbol{S}$ has angles of view α (horizontal) and β (vertical). The resolution of the depth image is R_x by R_y pixels. The position of the "hand" joint $H(x, y, D)$ in the depth image can be obtained from the Kinect skeleton data (Figure 2.4a). Thus, the spatial thresholds are described as:

$$[T_{x\_min}, T_{x\_max}] = [x - \frac{d_x}{2}\frac{\frac{R_x}{2}}{Dtan\frac{\alpha}{2}}, x + \frac{d_x}{2}\frac{R_x/2}{Dtan\frac{\alpha}{2}}] \quad (1)$$

$$[T_{y\_min}, T_{y\_max}] = [y - \frac{d_y}{2}\frac{R_y/2}{Dtan\frac{\beta}{2}}, y + \frac{d_y}{2}\frac{R_y/2}{Dtan\frac{\beta}{2}}] \quad (2)$$

$$[T_{Depth\_min}, T_{Depth\_max}] = [D - \frac{d_z}{2}, D + \frac{d_z}{2}] \quad (3)$$

where d_x, d_y and $d_z$are constant dimensions (in millimeters) of the hand's region. The hand's region in the depth image is shown in Figure 2.4b. The hand's region in the color image can also be obtained by mapping the hand's region on top of the color image (Figure 2.4c).

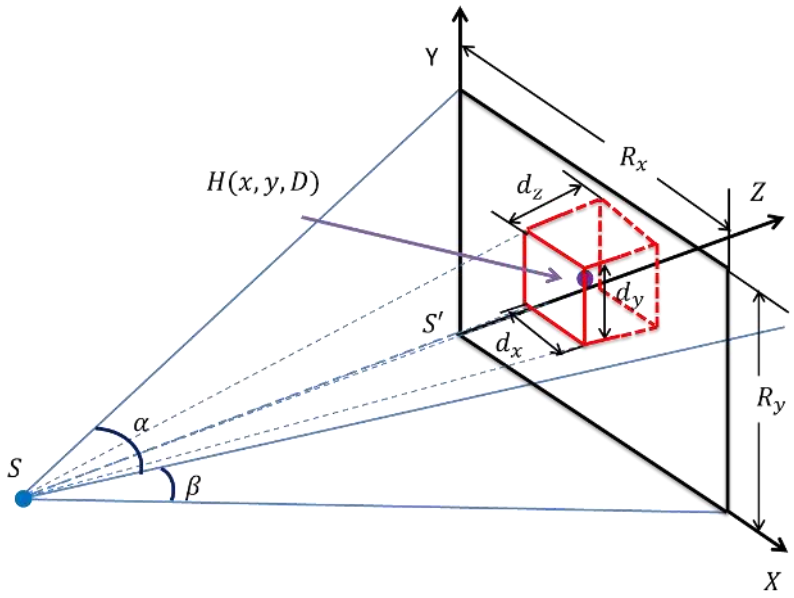
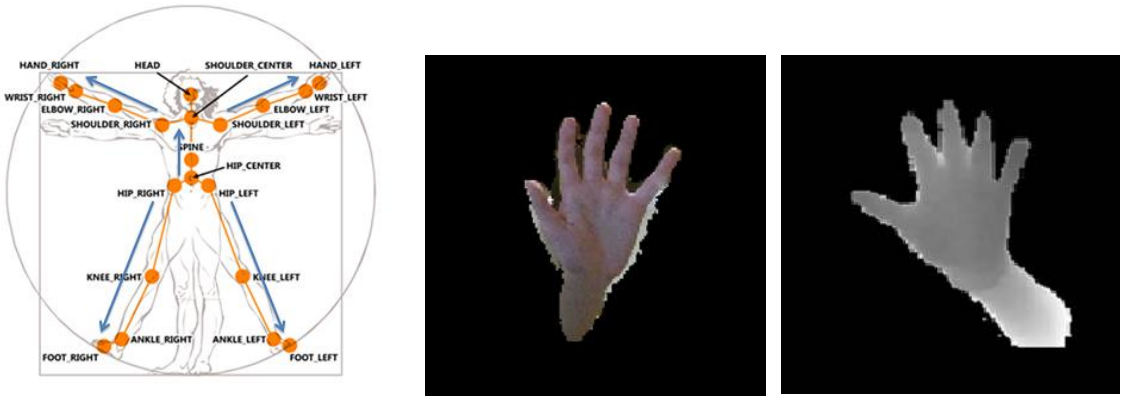


Figure 2.3. Illustration of the hand region segmentation: the $d_x \times d_y \times d_z$ hand region at $\boldsymbol{H}(x, y, D)$ was segmented from the $R_x \times R_y$ depth image obtained using a depth sensor located at the position $\boldsymbol{S}$.



(a) (b) (c)

Figure 2.4. Illustration of data obtained using Kinect. (a) Kinect body skeleton (b) RGB color image of the hand region (c) Depth image of the hand region

2.2.TRAINING DATASET

The dataset to train the per-pixel classifier includes two parts: the hand depth image and the ground truth classification for each pixel. The depth image of the hand region can be obtained directly from the process discussed in Section 2.1. Obtaining the ground truth classification for each pixel, however, is not trivial. Segmenting each depth image manually would be a massive work. Generating synthetic data [22, 24] requires building a high-quality 3D hand model, and simulating the distortion and noise for synthetic data is necessary and challenging. Therefore, a color glove was designed in order to generate realistic training data conveniently (Figure 2.5).

Figure 2.5. Color glove

The glove was painted using 11 different colors according to the configuration of hand parts. The glove can fit the human hand's surface perfectly because it is made from an elastic material. In this way, not only RGB images with colored hand parts but also precise human hand depth images can be obtained using a Kinect sensor. The RGB

images were then processed in a hue-saturation-value color space to segment the hand parts according to colors. Therefore, the dataset for hand parsing can be generated efficiently by performing various hand gestures wearing the glove (Figure 2.6).

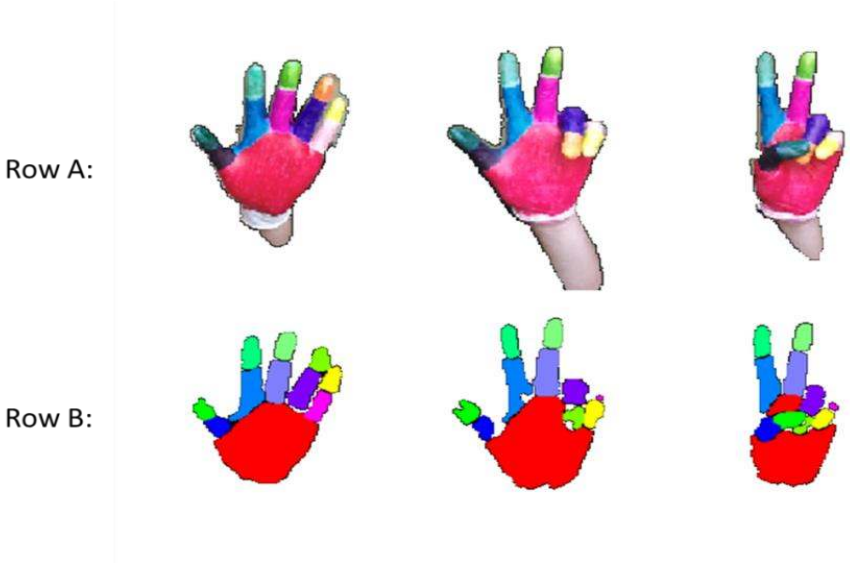


Figure 2.6. Illustration of hand part segmentation training data. Row A: Raw color images of hand wearing the color glove. Row B: segmented hand parts represented using different colors.

2.3. FEATURE EXTRACTION

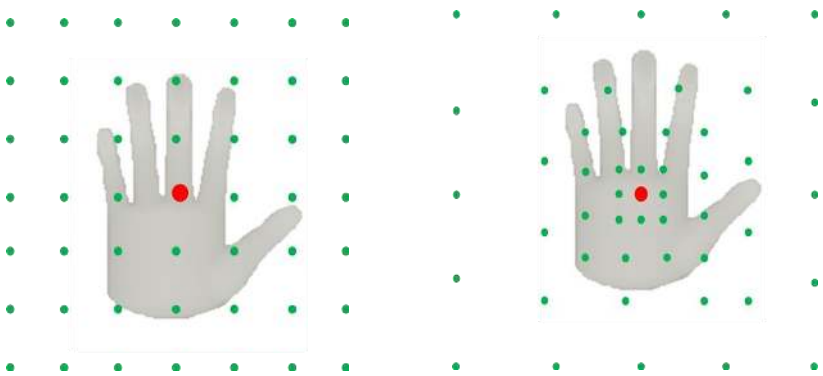
The depth comparison feature [22] were employed to describe the context information of each pixel in the hand depth image. The classification of each pixel can therefore be determined by the context information.

For each pixel $\boldsymbol{x}$ in the depth image I, a feature value was described as:

$$f_n(I, \boldsymbol{x}) = I(\boldsymbol{x} + \boldsymbol{v_n}) - I(\boldsymbol{x}) \quad (4)$$

where the feature f_n was calculated using the depth value contrast between the pixel $\boldsymbol{x}$ and the offset pixel $(\boldsymbol{x} + \boldsymbol{v}_n)$. A set of features were extracted for each pixel according to a certain feature selecting scheme that contains a set of offset vectors $\{\boldsymbol{v}_n\}$. A large number of features insure a comprehensive description of the pixel's context, but it also may result in considerable computational costs.

In order to improve the efficiency of feature usage, the distance adaptive scheme (DAS) was employed [25]. The hand region pixels are usually clustered in a relatively small area of the whole depth image. Thus, depth value contrasts between hand pixels and background pixels which are at far away will typically provide very little useful information. The contrasts between closer pixels can, however, provide important information. Therefore, a feature selecting scheme was generated randomly using a Gaussian distribution kernel to focus on context pixels in a closer area.



(A) (B)

Figure 2.7. Illustration of feature-selecting schemes generated through (A) the use of an evenly distribute scheme (EDS) and (B) a distance adaptive scheme (DAS).

Figure 2.7 illustrates two feature selecting schemes which were generated using an evenly distribute scheme (EDS) and a distance adaptive scheme (DAS), respectively. The distance adaptive context points were more focused in the hand region. As a result, DAS features were more likely to contain detailed information in hand region than EDS features.

2.4. PER-PIXEL CLASSIFIER

Labeling pixels according to their corresponding hand part is a typical multi-class classification task. A number of statistical machine learning models can be used, including the Artificial Neural Networks (ANN), Support Vector Machine (SVM), Decision Tree (DT) and Random Decision Forest (RDF) [26, 27]. The RDF has been proven effective [22] for human body segmentation using depth contrast features. It is robust to outliers, can avoid over-fitting situations in multi-class tasks, and is highly efficient in large database processing. Therefore, RDF was selected as the machine learning model in this study (Figure 2.8).

The RDF classifier consists of a set of independent decision trees. At each split node of a decision tree, a feature subset is used to determine the split by comparing the feature values to corresponding thresholds. At each leaf node, the prediction is given as a set of classification probabilities $P(c|\boldsymbol{f}(I,x))$ for each class c (Figure 2.8a). The final prediction of the forest is obtained by a voting process of all trees (Figure 2.8b). In the training of each tree, the dataset is randomly separated into two subsets. Approximately 2/3 of the data are used for training while the rest are used for error estimation. The

training subset is used to collect a statistical histogram of classifications at each leaf node. The histogram can therefore be used to estimate the classification probabilities $P(c|\boldsymbol{f}(I,x))$ for data samples which reach the leaf node. The thresholds are optimized to find the best split that can minimize the errors in error estimation.

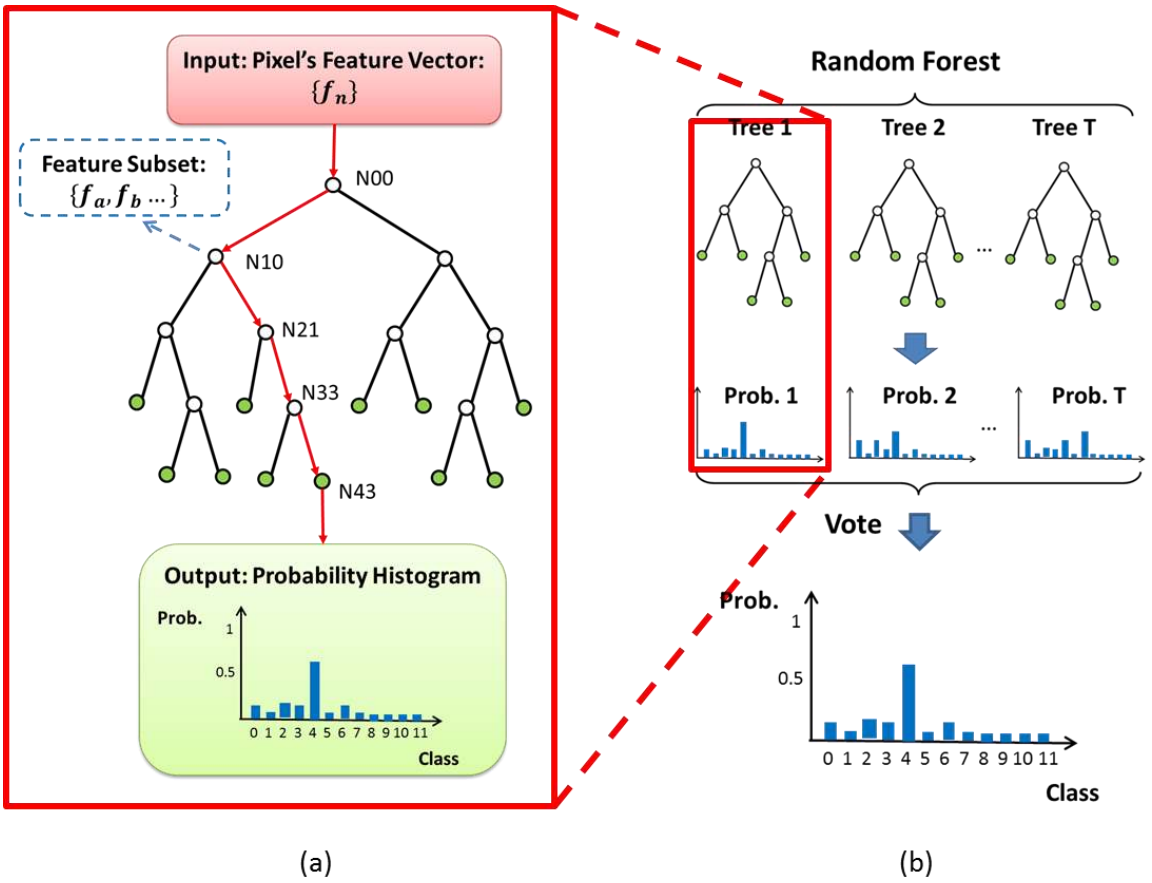


Figure 2.8. Illustration of the RDF classifier. (a) Single-tree classification: The pixel that has feature vector $\{f_n\}$ is pushed down the tree from N_{00} to N_{43}. A feature subset (e.g., $\{f_a, f_b \ldots\}$) is used to determine the split at each split node (e.g., N_{10}). At each leaf node (e.g., N_{43}), a histogram illustrates the probabilities of all classes. (b) Multi-tree voting: The output probability histogram is determined by a voting process that involves all of the trees.

Each pixel of the hand's depth image is assigned a set of probabilities $P(c|\boldsymbol{f}(I,\boldsymbol{x}))$ of all classes using the RDF classifier. The probability distribution maps of several different classes are illustrated in Figure 2.9a, b, and c. A sample of hand part segmentation result is illustrated in Figure 2.9d, where each pixel is colored according to the class that has the highest probability.

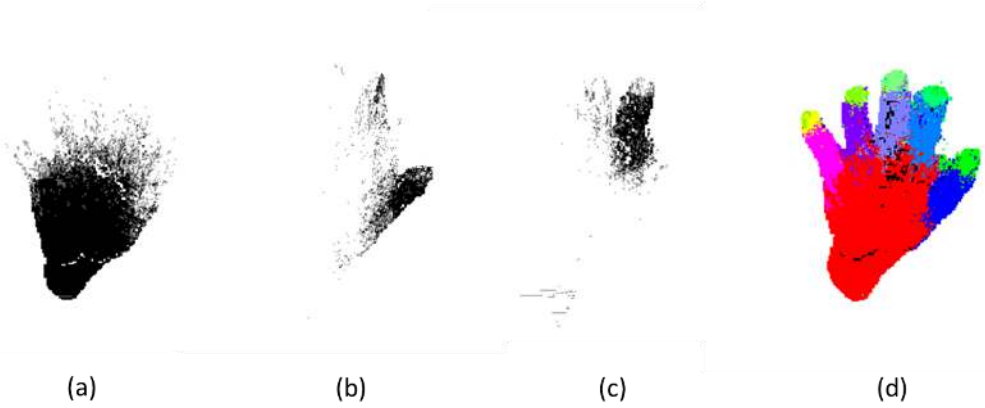


Figure 2.9. Per-pixel classification results. (a), (b) and (c) Probability distribution maps of "palm," "thumb finger," and "middle finger" respectively (Darker represents higher probability). (d) Pixel classifications on a hand depth image.

3. GESTURE RECOGNITION

The RDF-based per-pixel classification process classifies each pixel by assigning classification probabilities $P(c|\boldsymbol{f}(I,\boldsymbol{x}))$ for classes representing different hand parts. In [24], the joint positions are obtained by the mean-shift local mode-seeking algorithm [28] performed on the probability distribution maps of the classes $\{c\}$. The hand gestures are then recognized by mapping the estimated joint coordinates to know hand gestures. However, both noise and misclassifications in the probability distribution maps make it difficult to localize joint positions accurately. Moreover, the joint coordinates not only can be determined by different gestures but also can be significantly affected by the hand's size and rotational direction. Thus, joint coordinates are not suitable descriptions of the hand gestures. In addition, lacking constraints can result in unjustified joint positions that make the joint position information unreliable.

In this section, the approach to recognize hand gestures that can overcome the above problems is introduced. The noisy forearm region is cut precisely in Section 3.1. In Section 3.2, the mean-shift mode finding algorithm is improved by adapting the searching window size with the target hand part size. A confidence function is also employed to evaluate the reliability of the hand part localization. In Section 3.3, the method to constrain joint locations based on the hierarchical kinematic structure of the hand is proposed. Thus, the joint localization algorithm is more robust to outlier clusters in the probability distribution maps. In Section 3.4, the joint angle features are used to describe the hand gestures, thus the feature is invariant to the hand's size and rotational directions.

3.1. FOREARM CUTTING

As discussed in Section 2.1, the data preparation process segments the hand region from the background using spatial and depth thresholds, whereas the forearm region usually cannot be partitioned with the hand region precisely. The remaining forearm might introduce error in the hand parsing process (Figure 3.1 first row).

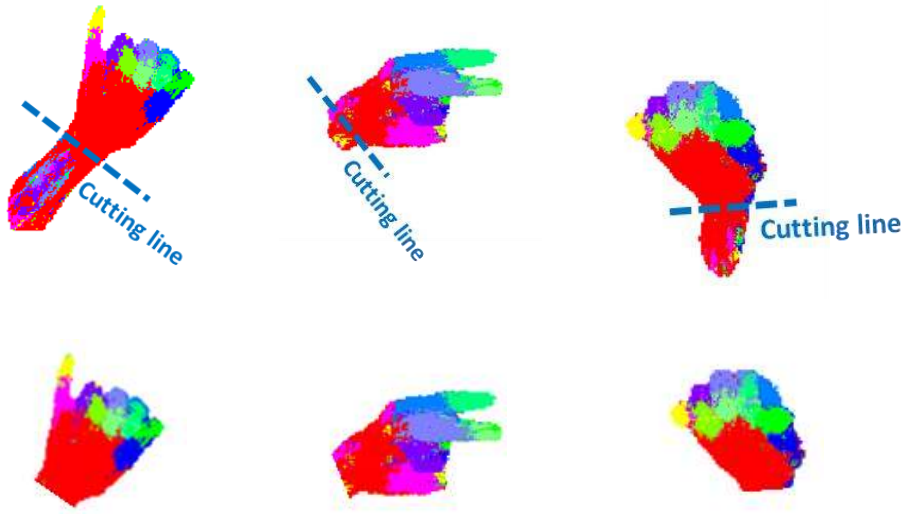


Figure 3.1. Illustration of forearm cutting. First row: noisy hand part segmentation with forearm region. Second row: hand region after forearm cutting process

Therefore, a forearm cutting algorithm was developed to solve this problem. The cutting line between the palm region and the forearm region can be determined using three main factors: the position of the palm, the relative direction of the forearm to the palm, and the distance from the palm center to the cutting line.

The cutting line is obtained using the palm region classification probability distribution map $\{P(c|\boldsymbol{f}(I,\boldsymbol{x})\}$ (Figure 3.2a). A 2D Gaussian filter with a large standard

deviation is applied to blur the probability distribution map, so that the pixel with the maximum intensity can be considered as the palm center $\boldsymbol{O}$ (Figure 3.2b). There could be multiple local maximum points. In this case, $\boldsymbol{O}$ is defined as the maximum point that is closest to the "hand" joint given by the Kinect skeleton. Then, a circle with a center $\boldsymbol{O}$ and a certain radius r is used to represent the palm region (Figure 3.2c). Next, the number M of pixels outside the palm region circle is counted at each direction (from 0 to 2π). Thus, the largest blob (blue) above the threshold T can be considered as the forearm region, and the peak position "$\boldsymbol{P}$" is used to determine the direction of the arm (Figure 3.2d). Finally, the cutting line l that is vertical to $\boldsymbol{OP}$ and tangent to circle $\boldsymbol{O}$ can be determined (Figure 3.2e).

Therefore, the forearm region can be partitioned from the hand region using the cutting line (Figure 3.1 second row).

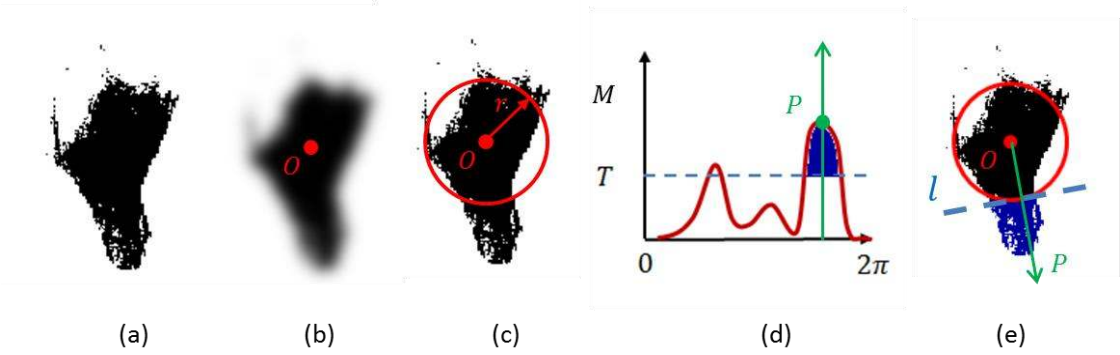


Figure 3.2. Illustration of forearm region cutting method. (a) Probability distribution map of the "palm" region. (b) The Gaussian-blurred probability distribution map with the maximum intensity at $\boldsymbol{O}$. (c) Hand region represented using the "palm circle" $\boldsymbol{O}$. (d) Histogram of pixels outside the "palm circle". (e) The cutting line is vertical to $\boldsymbol{OP}$ and tangent to the "palm circle" $\boldsymbol{O}$.

3.2. JOINT LOCALIZATION

The hand part segmentation process assigns the classification probabilities $P\{c|\boldsymbol{f}(I,\boldsymbol{x})\}$ of each pixel $\boldsymbol{x}$ for each class (hand part) c. Typically, a multi-modal probability distribution map would be obtained for each hand part (e.g., the probability distribution map of the "thumb" finger in Figure 3.3a) from the per-pixel classification algorithm. Thus, the global mass center of the probability distribution map is not suitable to represent the joint position. Therefore, the mean-shift local mode-seeking algorithm [28] was adapted to estimate the joint positions. The mean function can be described as:

$$\boldsymbol{m}(\boldsymbol{x}) = \frac{\sum_{i=1}^{N} K(\boldsymbol{x_i} - \boldsymbol{x})\boldsymbol{x}_i}{\sum_{i=1}^{N} K(\boldsymbol{x}_i - \boldsymbol{x})} \tag{5}$$

where $\{\boldsymbol{x}_i\}_{i\epsilon[1,N]}$ is the set of neighborhood pixels, and N is the number of pixels in the searching window. The algorithm starts with an initial estimate $\boldsymbol{x}$, and sets $\boldsymbol{x} \leftarrow \boldsymbol{m}(\boldsymbol{x})$ iteratively until $\boldsymbol{m}(\boldsymbol{x})$ converges. A weighted Gaussian kernel K is used as follows:

$$K(\boldsymbol{x} - \boldsymbol{x}_i) = I(\boldsymbol{x}_i)^2 w_{ic} e^{-\sigma\|\boldsymbol{x}-\boldsymbol{x}_i\|} \tag{6}$$

$$w_{ic} = P(c|\boldsymbol{f}(I,\boldsymbol{x}_i)) \tag{7}$$

where σ is a constant parameter to determine the bandwidth of the Gaussian function, w_{ic} is the weight of the pixel $\boldsymbol{x}_i$ in the image I. $I(\boldsymbol{x}_i)^2$ was used to estimate the pixel area in the world coordinate system, which is related to the depth of the pixel.

In order to find the global mode, the dimension-adaptive method is used. The searching window is initialized at the center of the probability distribution map with a

large size $N_0 = a_0 \times b_0$ (Figure 3.3b). Then, the window shrinks in each iteration (Figure 3.3c,d) until the size is approximately similar to the size of the hand part (Figure 3.3 e). The final window size $N_k = a_k \times b_k$ and the shrinking rates $\frac{a_k}{a_{k-1}}$ and $\frac{b_k}{b_{k-1}}$ are constant parameters determined by the size of each hand part.

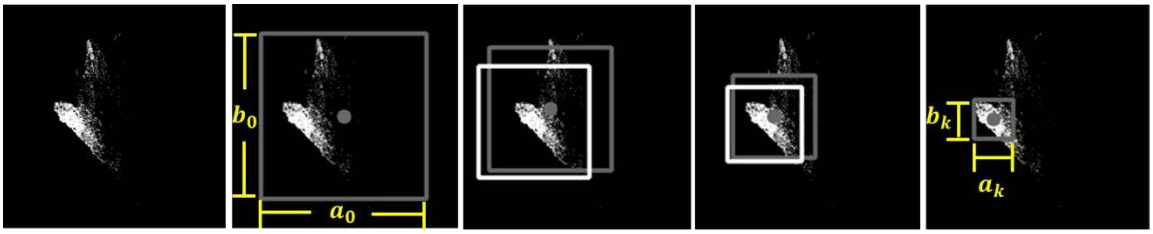


Figure 3.3. Mean-shift based joint localization process. (a) Probability distribution map of a hand part. (b) Initial searching window $a_0 \times b_0$. (c), (d) Mean-shift and window shrinking process. (e) Final window $a_k \times b_k$ that localized the global mode.

In some cases, some hand joints may be invisible or unreliably classified. Therefore, a confidence score S_c of the hand part c is given by averaging all the pixel weights w_{ic} in the final searching window. Joints that have poor scores will be considered as "missing" joints. The location of a "missing" joint is assigned by the location of its parent joint. Specifically, the locations of missing fingertips are assigned to the locations of their corresponding fingers; and the locations of missing fingers are assigned to the location of the palm center.

The X and Y coordinates x_c, y_c of the joint $\boldsymbol{J}_c$ in the world coordinate system can be obtained by transforming the center position of the final searching window from the

image coordinate system to the world coordinate system. The Z coordinate is defined using an average value in the final searching window W_c as

$$z_c = \frac{\sum_{x \epsilon W_c} I(\boldsymbol{x})\, u(I(\boldsymbol{x}))}{\sum_{x \epsilon W_c} u(I(\boldsymbol{x}))} \tag{8}$$

$$u(\boldsymbol{x}) = \begin{cases} 1, & I(\boldsymbol{x}) \in [m - \varepsilon, m + \varepsilon] \\ 0, & otherwise \end{cases} \tag{9}$$

$$m = Median(\{I(\boldsymbol{x}) | \boldsymbol{x} \in W_c) \tag{10}$$

where I is the depth image, $\boldsymbol{x}$ is the pixel's position vector, and ε is a constant threshold value. The function $u(\boldsymbol{x})$ is used to determine if the depth of the pixel $\boldsymbol{x}$ is valid, where the depth values which are larger than $m + \epsilon$ of smaller than $m + \epsilon$ are considered as noise.

3.3. KINEMATIC CONSTRAINTS

As discussed in Section 3.2, the joint positions can be obtained using the mode-seeking algorithm. However, sometimes the mode-seeking process cannot localize the correct joint position because the pixels of neighborhood hand parts are likely to be misclassified. For example in Figure 3.3a, besides the global mode located at the "thumb" position, there is another significant cluster located at the "index finger" position that is possibly to be recognized as the "thumb". Some other examples of joint localization errors are shown in Figure 3.4.

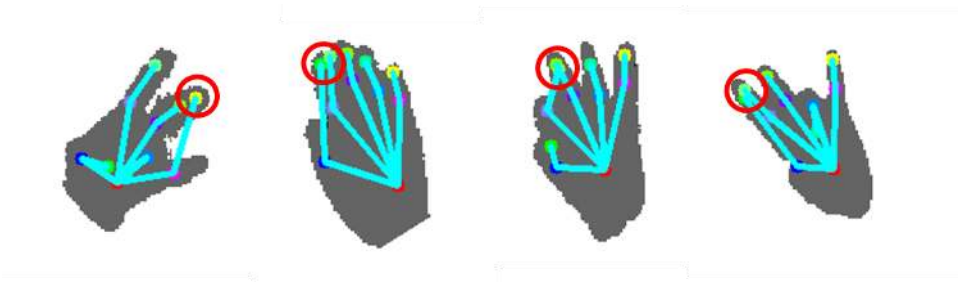


Figure 3.4. Joint localization errors

A kinematic constraining method is developed to solve this problem. The concept is employing kinematic probability $P(c|\boldsymbol{x}_i)$ to penalize the weights w_{ic} of pixels that cannot fit the kinematic structure of the hand, i.e., Equation 7 becomes:

$$w_{ic} = P(c|\boldsymbol{x}_i) \cdot P(c|\boldsymbol{f}(I, \boldsymbol{x}_i)) \tag{11}$$

The kinematic probability distribution $\{P(c|\boldsymbol{x}_i)\}$ is obtained from the training dataset generated by the color glove, which contains a large number of segmented hand images (e.g., Figure 2.6 Row B) for different gestures. The probability distribution map is obtained by

$$P(c|\boldsymbol{x}_i) \propto \frac{\sum_{j=1}^{M} \delta(L(\boldsymbol{x}_{i,j}) = c)}{M} \tag{12}$$

where $L(\boldsymbol{x}_{i,j})$ is the class label of pixel x_i in the training image j, and M is the number of training images. The statistical distribution $\frac{\sum_{j=1}^{M} \delta(L(\boldsymbol{x}_{i,j})=c)}{M}$ is obtained by counting the number of pixels $\boldsymbol{x}_{i,j}$ from all M images which belong to class c (Figure 3.5a). The

kinematic probability distribution map $P(c|\boldsymbol{x}_i)$ (Figure 3.5b and c) is obtained by smoothing the statistical distribution.

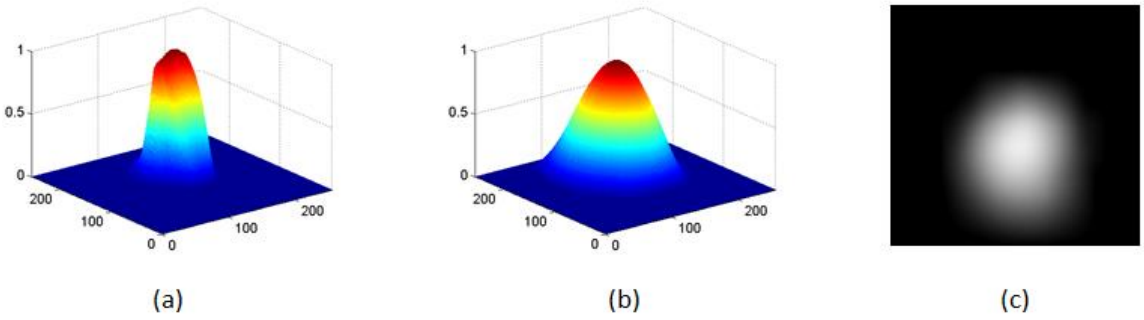


Figure 3.5. Generate kinematic probabilities. (a) Statistical distribution. (b) Kinematic probability distribution . (c) Kinematic probability distribution shown in a grey level image.

The kinematic probability distribution maps are generated hierarchically (Figure 3.6). Firstly, the "palm" joint is localized under the constraints by the kinematic probabilities $p(C_{palm}|\boldsymbol{x}_i)$ (Figure 3.6a). Secondly, the lower fingers' kinematic probabilities $\{P(c_{finger1}|\boldsymbol{x}_i) \ldots P(c_{finger5}|\boldsymbol{x}_i)\}$ are obtained on the reference coordinate system of the palm, where the origin is the center of the palm, and the x-axis and y-axis are taken to be horizontal-right and upright respectively. Then the lower fingers can be localized (Figure 3.6b). Thirdly, the kinematic probabilities of five fingertips $\{P(c_{tip1}|\boldsymbol{x}_i) \ldots P(c_{tip5}|\boldsymbol{x}_i)\}$ are obtained on the reference coordinates system of the five fingers respectively (Figure 3.7), where the origins are the lower finger joints, the y-axis are along the directions from the palm to the lower finger joints, the x-axis is

perpendicular to the y-axis (Figure 3.6c). Thus, the fingertips can be localized (Figure 3.6d).

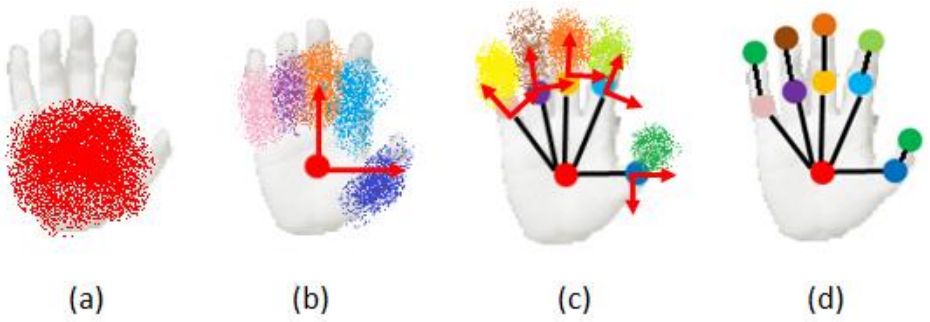


Figure 3.6. Hierarchical kinematic constraints. (a) Kinematic probabilities of the palm region. (b) Localize palm and obtain kinematic probabilities of the lower fingers. (c) Localize the lower fingers and obtain kinematic probabilities of the fingertips. (d) Localize fingertips.

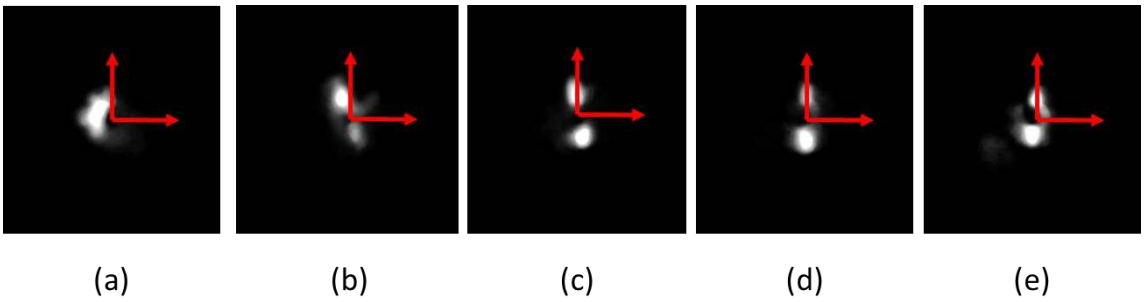


Figure 3.7. Kinematic probability distribution maps of the fingertips, where the reference coordinates are shown in red. (a) Thumb fingertip. (b) Index fingertip. (c) Middle fingertip. (d) Ring fingertip. (e) Pinky fingertip.

Using the method above, the hand joints can be constrained in a smaller region that is kinematically possible. Especially for the fingertip joints, which are highly

constrained by the positions and directions of their parent lower finger joints, the hierarchical constraints can effectively improve the joint localization accuracy (Figure 3.8). As the kinematic probabilities $P(c|\boldsymbol{x}_i)$ are obtained by smoothing the statistics distribution, the smoothing methods and parameters can affect the results. In general, the smoother the kinematic probability distribution maps are, the more likely the joints may be localized at wrong positions. However, less smooth kinematic probability distribution maps result in poor gesture classification accuracy because the kinematic probabilities $P(c|\boldsymbol{x}_i)$ could overwhelm the effects of the classification probabilities $P(c|f(I, \boldsymbol{x}_i)$.

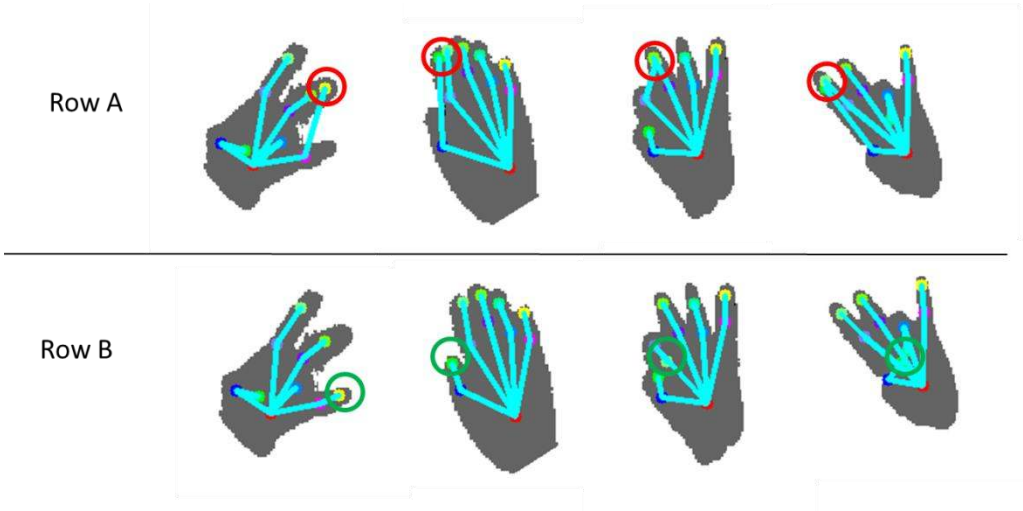


Figure 3.8. Joint localization results. (Row A) Localized joints without constraints (Row B) Localized joints with constraints

3.4. GESTURE RECOGNITION

The 3D joint positions $\{\boldsymbol{J}_{c_1}, \boldsymbol{J}_{c_2} \ldots \boldsymbol{J}_{c_{11}}\}$ in the world coordinate system can be obtained by using the joint localization method discussed in Section 3.2. Thus, the hand gesture can be described using a joint angle feature vector (see Figure 3.9).

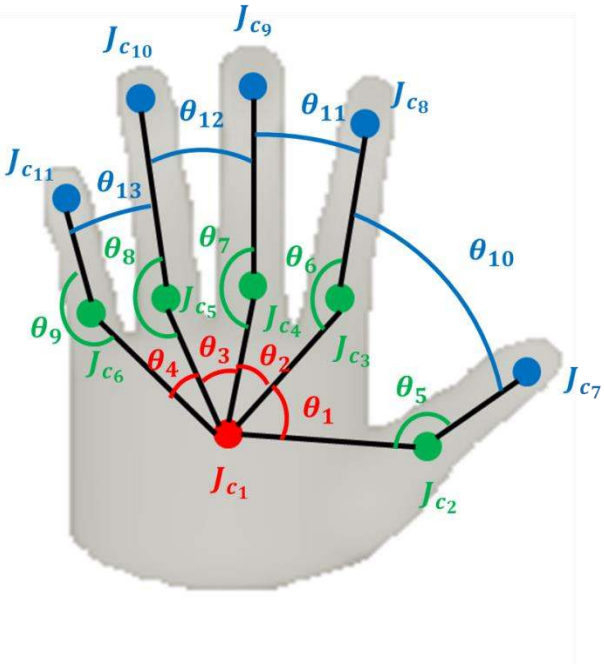


Figure 3.9. Joint angle features

The feature vector contains the angles between neighborhood lower fingers $\{\theta_1, \theta_2, \theta_3, \theta_4\}$, the angles between each pair of lower and upper fingers $\{\theta_5, \theta_6, \theta_7, \theta_8, \theta_9\}$, and the angles between neighborhood upper fingers $\{\theta_{10}, \theta_{11}, \theta_{12}, \theta_{13}\}$. Using the feature vector $\{\theta_1, \theta_2 \ldots \theta_{11}\}$ as the input, the hand gesture as the ground truth, a hand gesture classifier can be trained to recognize pre-defined hand

gestures. The 24 ASL alphabet signs recognized using our RDF gesture classifier is shown in Figure 3.10 as examples.

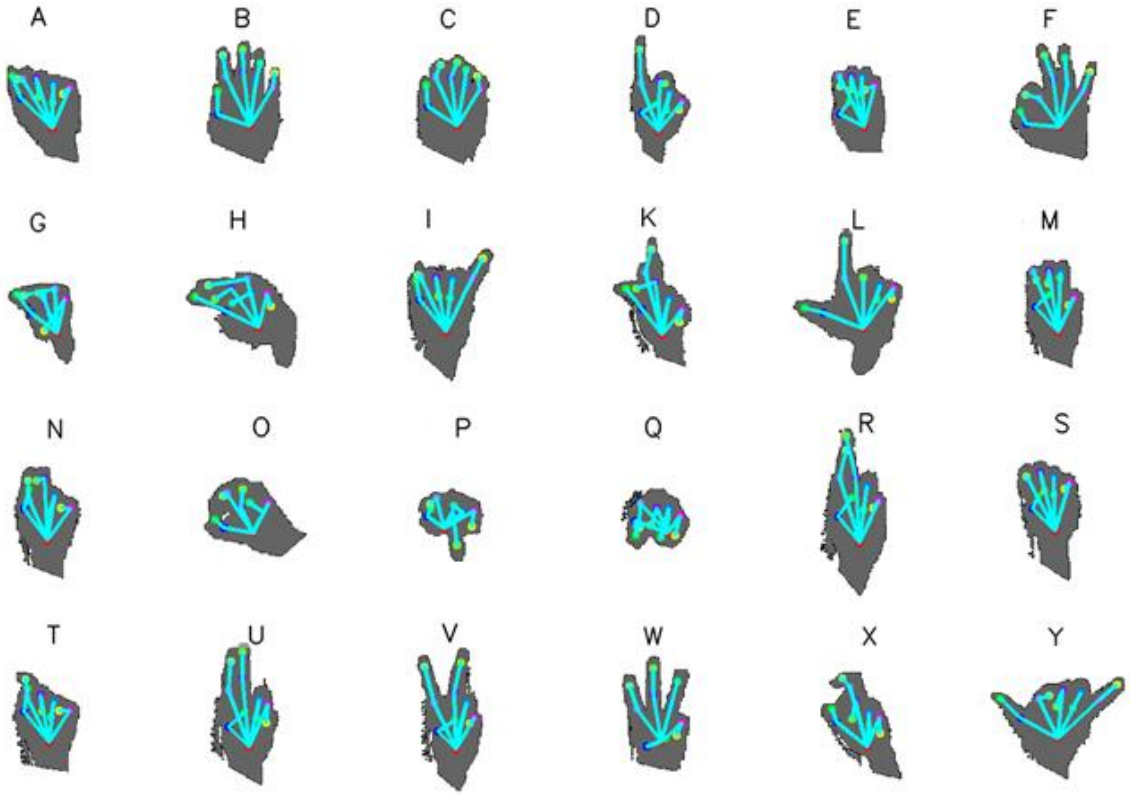


Figure 3.10. Examples of ASL alphabet recognition. For each hand gesture, the localized joints and the "skeleton" are shown using different colors on the grey background of the hand's silhouette.

4. EXPERIMENTAL RESULTS

In order to evaluate the developed method, we have done three parts of experiments as described. First, the RDF-based per-pixel classification results are shown in Section 4.1. We tested the per-pixel classifier using the dataset generated using the color glove. Second, in Section 4.2, we compared our method with the method developed by Keskin et al. [24] on our dataset. Third, we used the public dataset from Surrey University [12] to compare our method with other benchmark methods under the same conditions in Section 4.3.

4.1. PER-PIXEL CLASSIFICATION

Our training dataset contains 3,000 images generated using the color glove, of which 2,000 images were picked randomly for training and the rest were used for validation. The resolution of the training image was normalized to 256×256. For each pixel, 100 depth comparison features were extracted. In order to reduce the unnecessary memory storage cost, only pixels with valid depth value were used for training, so the total number of training vectors was about 10 million. Still, it cost 28.3 gigabytes RAM to perform the training process. Three random trees that had 20 levels each were generated. The training process took about 3 hours using a workstation with the E5 processor and 32 GB RAM. We tested the EDS and DAS feature selecting methods on different datasets varying in size. The accuracy corresponding to the training sample amount is shown in Figure 4.1.

The accuracy on the 10 million dataset experiment using distance adaptive feature selecting scheme (DAS) was 88.96%, comparing to 81.34% using the evenly distribute feature selecting scheme (EDS). These results shows that the adaption of DAS have significantly improved the accuracy of per-pixel classification.

In addition, according to the graphic curve, the classification accuracy could still increase if the size of training database were expanded. However, because of the limitations of the resource and time, and considering the high accuracy we have already achieved, there were no further attempts done to increase the size of the dataset.

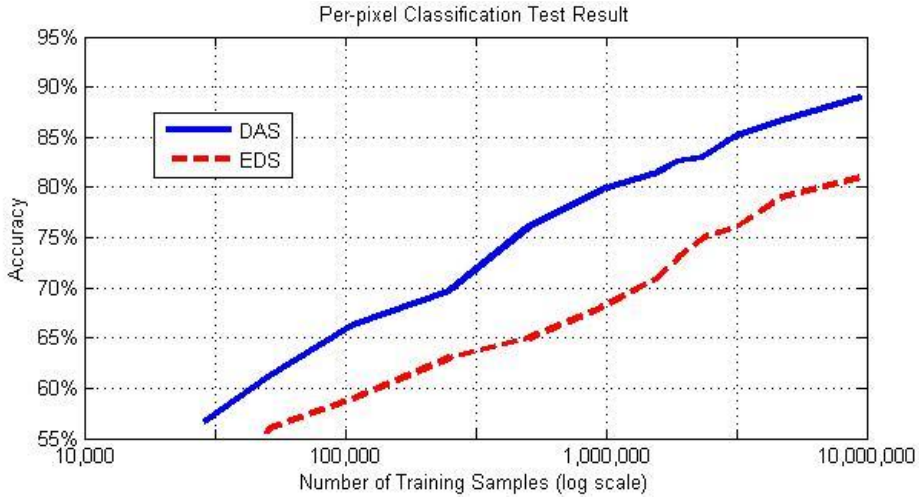


Figure 4.1. Classification accuracy corresponding to database size (log scale)

4.2. ASL FINGERSPELLING RECOGNITION

In order to evaluate the performance of the developed system for ASL sign recognition, 72,000 depth images of a hand were generated using the Kinect, of which

48,000 of the data were used for training and the rest 24,000 were used for testing. The gestures included 24 alphabet gestures (excluding the dynamic signs “j” and “z”). The signed alphabets followed the standard from the ASL University website [14] (Figure 4.2) with a variety in distances and view angles from the Kinect sensor.

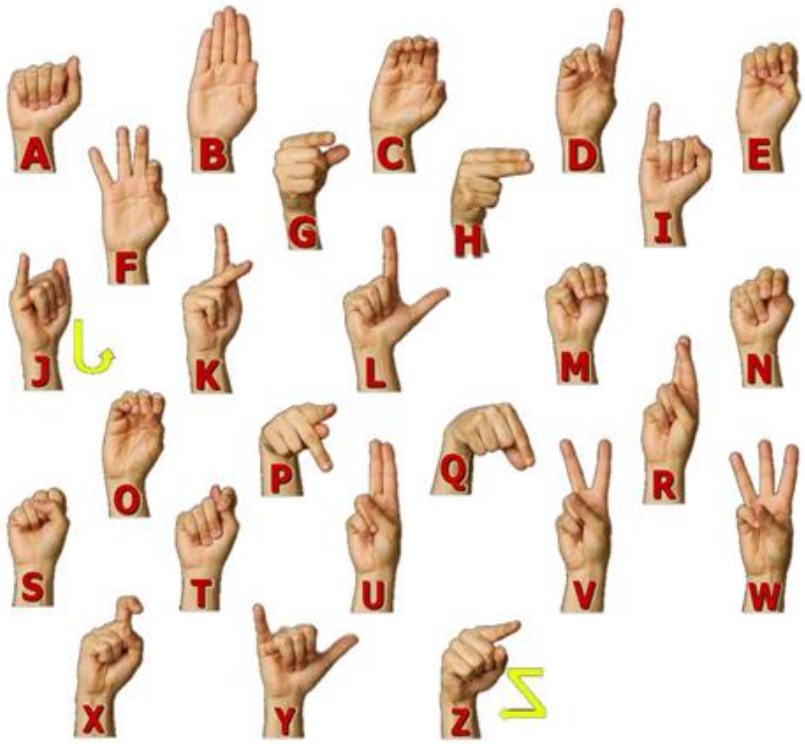


Figure 4.2. ASL Fingerspelling Alphabets

The experiments contained the cross validation results using multiple methods. Firstly, the joint was localized using the dimension-adaptive mean-shift method without constraints. The gestures were classified using a RDF classifier according to the feature vector of the 13 joint angles (RDF-A). Secondly, the hierarchical kinematic constrains

were implemented to evaluate the improvements (RDF-A+C). Thirdly, the method introduced in [24] was also implemented to compare with our method. The joints were localized using the basic mean-shift algorithm. The searching windows were initialized at different positions to obtain several local modes. Thus, the local mode with the highest maximum is regarded as the joint. The hand gestures were recognized by mapping the joint position coordinates to the known gestures (RDF-P). The results obtained using the above three methods above are shown in Figure 4.3.

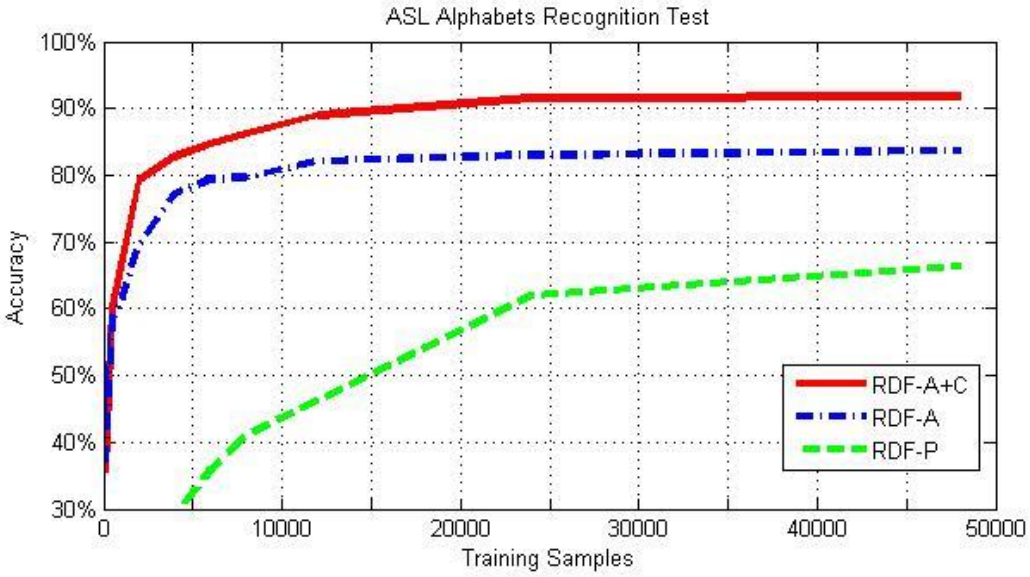


Figure 4.3. ASL alphabets recognition results

As the size of training dataset increases, the accuracy of alphabet recognition keeps increasing and then approaches a constant after the training samples exceed 1,000 images for each ASL sign (24,000 in total).

The accuracy using the RDF-P method was 66.32%, the accuracy using the RDF-A method was 83.65%, and the RDF-A+C method achieved an overall accuracy of 91.85%. The system can recognize hand gestures at 7 fps without parallel computing, while the per-pixel classification took about 90ms to process each frame. According to the experimental results in [22], it is clear that GPU aided parallel computing will significantly improve the performance of per-pixel classification. Therefore, the system has a high potential of meeting the requirements of real-time applications.

Table 4.1. Confusion matrix obtained using the RDF-A+C method. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class.

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	0.96								0.01			0.01						0.01	0.01					
B		0.91				0.02		0.01		0.02					0.01							0.01	0.01	
C		0.01	0.89	0.01	0.01			0.02							0.02	0.01	0.01	0.02						
D			0.01	0.91	0.01							0.01		0.01			0.01		0.01	0.01			0.01	
E				0.01	0.92							0.05						0.01						
F		0.02				0.95				0.01												0.01		
G		0.02					0.85	0.04		0.01		0.01		0.02	0.01	0.01		0.01	0.01					
H			0.01	0.01			0.05	0.87				0.01		0.01	0.01	0.01			0.01					
I	0.02								0.93									0.01						0.03
K		0.01	0.01				0.01			0.89					0.02	0.01	0.01			0.01		0.01	0.01	
L				0.01							0.95							0.01	0.01		0.01			
M	0.01				0.04		0.01					0.87	0.02			0.01		0.02						
N			0.01	0.01						0.01		0.02	0.88		0.01			0.01	0.04					
O			0.01				0.01	0.02						0.94	0.01									
P		0.01	0.02	0.01	0.01	0.01	0.01	0.03		0.02				0.02	0.80	0.02			0.01			0.01	0.01	
Q			0.01				0.01	0.02		0.02		0.01		0.01	0.02	0.89								
R	0.01			0.01						0.01							0.95			0.01				
S	0.01		0.01		0.01		0.01	0.01				0.01	0.02		0.01			0.91						
T	0.02						0.02			0.01	0.01		0.01					0.01	0.90					
U				0.01						0.01										0.96				
V				0.01																0.01	0.95	0.02		
W						0.01														0.01	0.01	0.96		
X		0.01		0.01																			0.95	
Y									0.02															0.96

4.3. EXPERIMENTS ON A PUBLIC DATASET

To evaluate and compare our method with previous method by other researchers, a public dataset [12] was used. The dataset contained 24 ASL fingerspelling signs (not including “J” and “X”) performed by 5 subjects, where 500 samples of each sign were recorded for every subject. The subjects were asked to sign facing the sensor and to move their hand around while keeping the hand shape fixed. This dataset was generated using Kinect V1 and had high variability.

Since the dataset contained some very noisy data, and the hand region was not segmented from the background, some pre-processing of the data was necessary. Firstly, the hand region was segmented from the background using depth thresholds, and smaller noisy regions were deleted. Secondly, the depth images were normalized using the depth value of the palm’s center, and then all depth images were converted to 256×256 in data size. Since the lenses of Kinect V1 and Kinect V2 are different, the normalize parameters was then adjusted so that the hand sizes in the public dataset and in our dataset are similar. The normalized parameters for the 5 subjects are the same to keep the variety of hand sizes. The dataset did not contain hand segmentation configurations, thus, we used the same per-pixel classifier trained using our dataset.

Then, following the evaluation method in [12], the hand pose classifier was trained using half of the public dataset, and the rest data were used for validation.

Figure 4.4 illustrates the comparison of the recognition accuracy for each alphabet between the results obtained using the Gabor filter-based hand shape feature and random forest classifier (GF+RF) [12] and the results obtained using the RDF-A+C method we developed. The recognition accuracy has been significantly improved by using our

method especially for complex and confusing gestures such as “m”, “e”, “n” ,”o” and “s”, “t”. The mean accuracy of RDF-A+C method was 88% versus 75% using the GF+RF method reported in [12].

We compared our results obtained using RDF-A+C with the results obtained by implementing the ensemble of shape function descriptor (ESF) and multi-layer random forest (MLRF) (ESF+MLRF) [29]. The RDF-P method is also compared on the public dataset. The comparison of mean accuracy achieved using the different methods is shown in Table 4.2. The confusion matrix using the RDF-A+C method on the public dataset is shown in Table 4.3.

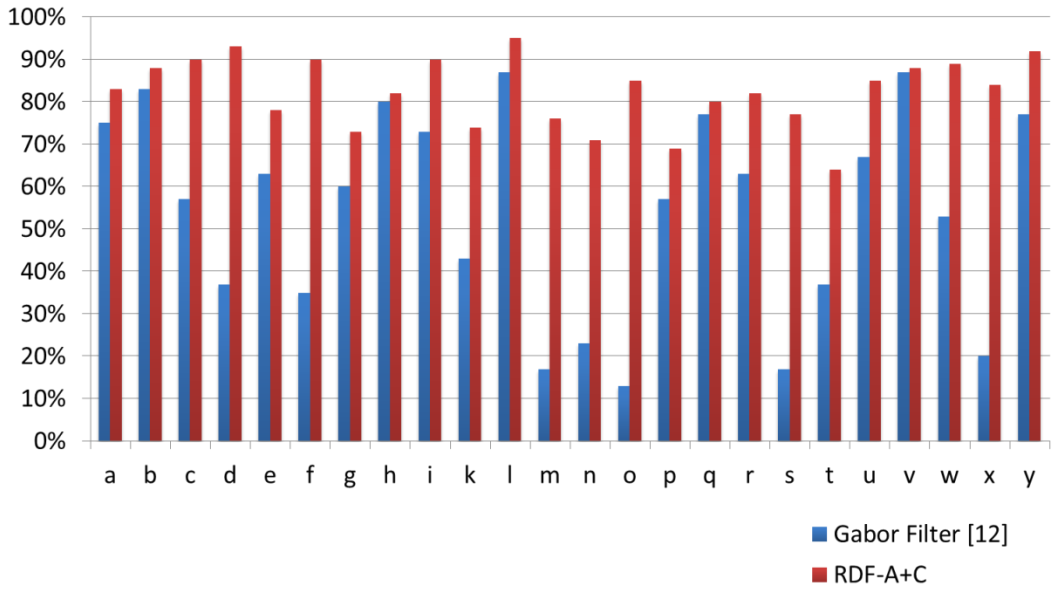


Figure 4.4. Comparison of the precision for each alphabet sign using the RDF-A+C method (Red) and the Gabor Filter-based method [12] (Blue).

Table 4.2. Accuracy comparison on Surrey University's dataset

Method	Mean Accuracy
GF+RF [12]	75%
ESF+MLRF [29]	77%
RDF-P [24]	59%
RDF-A+C (our method)	88%

Table 4.3. Confusion matrix obtained using RDF-A+C method on the public dataset [12]. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class.

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
A	0.83				0.01		0.02	0.01				0.01	0.01			0.01		0.02	0.07					
B		0.88			0.01	0.03	0.02			0.01			0.01				0.01					0.02		
C			0.90		0.01	0.01	0.01	0.01	0.01	0.01			0.01											
D				0.93										0.02									0.02	
E	0.01	0.01	0.02		0.78	0.01	0.01	0.01		0.02		0.02	0.02	0.02			0.01	0.04	0.01				0.01	
F		0.05				0.90				0.01													0.01	
G	0.01	0.01			0.02		0.73	0.08	0.01	0.04		0.01	0.01	0.01	0.01	0.01			0.02				0.01	0.01
H			0.01				0.08	0.82	0.01	0.01			0.01		0.02								0.01	0.01
I	0.01							0.01	0.90					0.01				0.01	0.01					0.03
K		0.02	0.01		0.01	0.01	0.03	0.02		0.74	0.01	0.01	0.02		0.01		0.03		0.03	0.01	0.01	0.01	0.02	
L				0.01						0.01	0.95						0.01						0.01	
M	0.01	0.01			0.01		0.01			0.01		0.76	0.08		0.01	0.01		0.05	0.03					
N	0.01		0.01		0.01		0.01	0.01	0.01	0.01		0.08	0.71		0.01	0.02		0.02	0.08					
O			0.01	0.01	0.01		0.01	0.01	0.01				0.01	0.85	0.02	0.02		0.01	0.01					0.01
P	0.01		0.01		0.01		0.03	0.02	0.01	0.01		0.01	0.01	0.02	0.69	0.09	0.01	0.01	0.01		0.01		0.04	
Q	0.01				0.01		0.01	0.01				0.01	0.01	0.01	0.08	0.80		0.01	0.01				0.01	0.01
R		0.01		0.01	0.01					0.02							0.82			0.05	0.02	0.01	0.02	
S	0.02		0.01		0.03		0.01		0.01	0.01		0.05	0.02	0.01		0.01		0.77	0.04					
T	0.08		0.01		0.01		0.01	0.01	0.01	0.01		0.04	0.09		0.01	0.01		0.04	0.64				0.01	
U		0.01															0.06			0.85	0.06	0.01		
V		0.00								0.01							0.03			0.04	0.88	0.03	0.01	
W		0.03				0.01											0.01			0.01	0.04	0.89		
X				0.01	0.01		0.01	0.01		0.03		0.01		0.01	0.01	0.02	0.02		0.01				0.84	
Y									0.04							0.01								0.92

5. CONCLUSION

This paper describes a new method that we have developed for American Sign Language (ASL) recognition. By using data obtained from the depth-based Kinect sensor, the per-pixel classification algorithm was used to segment a human hand into 11 parts. We employed a latex color glove instead of a commonly used synthetic 3D hand model in order to generate realistic per-pixel training data. The joint positions were obtained using a dimension-adaptive mean-shift mode-finding algorithm. To improve the joint localization accuracy, we employed kinematic probabilities in the mode-finding algorithm to constrain joints within possible motion ranges. The assemblies of 13 key angles between the finger joints were used as the feature vectors to represent hand gestures. An RDF gesture classifier was implemented in the end to recognize ASL signs. The system achieved a mean accuracy of 92% on a new dataset containing 75,000 samples of 24 static alphabet signs after training with 50,000 samples of these alphabet signs. In comparison with a method developed at Surrey University using the same dataset that is publically available, our method was shown to have higher mean accuracy (88% vs. 75%) in recognizing ASL signs. Since ASL signs represent complex hand gestures, the capability of ASL fingerspelling implies that our method has a great potential of being applicable to other applications that involve use of hand gestures and their recognition by low-cost vision cameras, such as commanding industrial robots on a factory floor or remote communication with healthcare assistants from a hospital room.

BIBLIOGRAPHY

[1] C. Oz, & M. C. Leu, "Recognition of finger spelling of American sign language with artificial neural network using position/orientation sensors and data glove," Advances in Neural Networks, pp. 157-164, 2005.

[2] C. Oz, & M. C. Leu, "Linguistic Properties Based on American Sign Language Recognition with Artificial Neural Networks Using a Sensory Glove and Motion Tracker," Computational Intelligence and Bioinspired Systems, pp.1197-1205, 2005.

[3] R. Y. Wang, J. Popovic, "Real-Time Hand-Tracking with a Color Glove," ACM Transactions on Graphics (TOG), Vol. 28, August 2009.

[4] C. Nölker, H. Ritter, "Detection of Fingertips in Human Hand Movement Sequences," Gesture and Sign Language in Human-Computer Interaction, Vol.1371, pp. 209-218, 1998.

[5] C. Nölker, H. Ritter, "GREFIT: Visual Recognition of Hand Postures," Gesture and Sign Language in Human-Computer Interaction, Vol.1739, pp. 61-72, 1999.

[6] J. Suarez & Robin R., "Hand Gesture Recognition with Depth Images: A Review", RO-MAN, pp. 411-417, 2012.

[7] "Vision-based hand pose estimation: A review," Computer Vision and Image Understanding, Vol. 108, pp. 52-73, 2007.

[8] P. Gurjal, K. Kunnur, "Real Time Hand Gesture Recognition Using SIFT," International Journal of Electronics and Electrical Engineering, 2012

[9] N. H. Dardas, N. D. Georganas, "Real-Time Hand Gesture Detection and Recognition Using Bag-of-Features and Support Vector Machine Techniques," Instrumentation and Measurement, Vol. 60, pp. 3592-3607, 2011.

[10] C. R. Mihalache, B. Apstol, "Hand pose estimation using HOG features from RGB-D data," System Theory, Control and Computing (ICSTCC), pp. 356-361, 2013.

[11] Chen K., Guo X., Wu J., "Gesture recognition system based on wavelet moment," Applied Mechanics and Materials, Vol. 401-403, pp. 1377-1380, 2013.

[12] N. Pugeault & R. Bowden, "Spelling It Out: Real-Time ASL Fingerspelling Recognition," IEEE Workshop on Consumer Depth Cameras for Computer Vision, 2011.

[13] M. A. Amin, H. Yan, "Sign Language Finger Alphabet Recognition from Gabor-PCA Representation of Hand Gestures," Machine Learning and Cybernetics, Vol. 4, pp. 2218-2223, 2007.

[14] www.lifeprint.com. ASL University, Nov. 2014.

[15] H. Liang, J. Yuan, D. Thalmann, Z. Zhang, "Model-based hand pose estimation via spatial-temporal hand parsing and 3D fingertip localization," The Visual Computer, Vol. 29, pp. 837-848, 2013.

[16] L. Oikonomidis, N. Kyriazis & A. A. Argyros, "Markerless and Efficient 26-DOF Hand Pose Recovery," Computer Vision, pp. 744-757, 2011.

[17] X. Liu, K. Fujimura, "Hand Gesture Recognition Using Depth Data," Automatic Face and Gesture Recognition, pp. 529-534, 2004.

[18] H. S., Yeo B., G. Lee & H. Lim, "Hand tracking and gesture recognition system for human-computer interaction using low-cost hardware," Multimedia Tools and Applications, May 2013.

[19] S. Qin, X. Zhu, H. Yu, S. Ge, Y. Yang & Y. Jiang, "Real-Time Markerless Hand Gesture Recognition with Depth Camera," Advances in Multimedia Information Processing, pp. 186-197, 2012.

[20] F. Dominio, M. Donadeo, P. Zanuttigh, "Combining multiple depth-based descriptors for hand gesture recognition," Pattern Recognition Letters, pp. 101-111, 2014.

[21] Z. Ren, J. Yuan, J. Meng & Z. Zhang, "Robust Part-Based Hand Gesture Recognition Using Kinect Sensor," IEEE Transactions on Multimedia, Vol. 15, NO. 5, August 2013.

[22] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A.Blake, "Real-Time Human Pose Recognition in Parts from Single Depth Image," Communications of the ACM (CACM), 2011.

[23] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, A. Fitzgibbon, "Efficient Regression of General-Activity Human Pose from Depth Images," ICCV '11 Proceedings of the 2011 International Conference on Computer Vision, pp. 415-422, October 2011.

[24] C. Keskin, F. Kirac, Y. E. Kara, L. Akarun, "Real Time Hand Pose Estimation using Depth Sensors", Computer Vision Workshops, pp. 1228-1234, 2011.

[25] H. Liang, J. Yuan, D. Thalmann, "Parsing the Hand in Depth Images", Multimedia, Vol. 16, pp. 1241-1253, 2014.

[26] Breiman Leo, "Random Forests," Machine Learning, Vol. 45, pp. 5-32, 2001.

[27] https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm. Random Forests, October 2014.

[28] D. Comaniciu, P. Meer. "Mean shift: A robust approach toward feature space analysis," IEEE Trans. PAMI, 2002.

[29] A. Kuznetsova, L. L. Taixe, B. Rosenhahn, "Real-time sign language recognition using a consumer depth camera," Computer Vision Workshops, pp. 83-90, 2013.

VITA

The author of this thesis, Cao Dong was born in July 16, 1989 in Shandong, China. He received his Bachelor degree of Engineering in Automotive Engineering from Shandong University of Science and Technology, Shandong, China in July 2012. He joined the Master of Science program in Mechanical Engineering at Missouri University of Science and Technology, Rolla, Missouri in August 2012. The author received his Master of Science degree in Mechanical Engineering from Missouri University of Science and Technology in May 2015.