A REAL TIME SYSTEM FOR DYNAMIC HAND GESTURE RECOGNITION WITH A DEPTH SENSOR

*A. Kurakin**

Moscow Institute of Physics and Technology
Department of Applied Math and Control
Moscow, Russia

Z. Zhang, Z. Liu

Microsoft Research
Redmond, WA, USA

ABSTRACT

Recent advances in depth sensing provide exciting opportunities for the development of new methods for human activity understanding. Yet, little work has been done in the area of hand gesture recognition which has many practical applications. In this paper we propose a real-time system for *dynamic* hand gesture recognition. It is fully automatic and robust to variations in speed and style as well as in hand orientations. Our approach is based on action graph, which shares similar robust properties with standard HMM but requires less training data by allowing states shared among different gestures. To deal with hand orientations, we have developed a new technique for hand segmentation and orientation normalization. The proposed system is evaluated on a challenging dataset of twelve dynamic American Sign Language (ASL) gestures.

Index Terms— Gesture recognition, depth camera

1. INTRODUCTION

Hand gestures are commonly used for human-human communications. We do not even think about it when we use hand gestures because they feel so natural to us. It is a natural question whether the hand gesture could become one of the main modalities for human-computer interactions. Driven by this elusive goal, tremendous amount of research activities have been devoted to hand gesture recognition. So far, most of the research work has been focused on recognition from images and videos due to the popularity of digital cameras. The recent progress in depth sensors such as Microsoft's Kinect device has generated a new level of excitement in gesture recognition. With the depth information, a skeleton tracking system has been developed by Microsoft. Many game developers have leveraged the skeleton tracking system to design games with body gestures as the main interaction mode.

The skeleton tracking system addresses the problem of body part segmentation which is very useful for body gesture recognition, but it does not handle hand gestures which typically involve palm and finger motions. Hand gestures are more difficult to recognize than body gestures because the motions are more subtle and there are serious occlusions between the fingers.

There exist several working hand gesture recognition systems [1, 2]. However [1] is rule-based and therefore a lot of human efforts are required to add new gestures, [2] work with static postures rather than dynamic gestures. In contrast to the previous systems, our system is purely data-driven, fully automatic, and is robust to variations in lighting conditions, hand orientation, performing speed and style. To validate our system, we have captured a dataset using a Microsoft Kinect device. The dataset contains 12 dynamic American Sign Language (ASL) gestures (fig. 4) performed by ten subjects. Each subject performs each gesture three times. There are large variations in style, speed and hand orientation for each gesture. Our system achieves 87.7% recognition accuracy on this challenging dataset. Furthermore, we have developed a real time system. Any user can perform hand gestures in front of the Kinect device, and our system performs gesture recognition automatically.

2. RELATED WORK

Hand gesture recognition has been an active research field for a long time, and many different approaches can be found in the literature. Like many other recognition tasks, a hand gesture recognition system typically consists of two components: front-end for feature extraction and back-end for classification.

As the back-end classifier, given that the input of a hand gesture is typically a sequence of image observations over time, some researchers (e.g., [3]) have used Hidden Markov Models (HMM) [4], while other researchers have used a time-delayed neural network [5] or finite state machines [6].

We use a method called action graph as our back-end classifier. This method was proposed by Li *et al*. [7] for human action recognition from videos. Compared to HMM, action graph has the advantage that it requires less training data and is can be easily expanded to recognize a new action type with-

*The first author performed the work while at Microsoft Research.

out retraining. Recently, action graph was also used for action recognition from 3D depth sequences [8].

Many different types of visual features have been proposed for hand gesture recognition. One type of features is based on the skeleton structure of the fingers. One can use the positions of the joints or the rotation angles at the joints as the visual features. This type of features requires articulated finger tracking. Various techniques have been proposed for finger tracking [9, 10, 11, 12, 13]. Despite the great progress, finger tracking is still a challenging problem due to the occlusions between the fingers.

To avoid relying on finger tracking, an alternative approach is to use some form of geometric descriptors which may not have clear anatomical meanings. This approach does not rely on finger tracking, but it usually requires to segment out the hand region.

Many hand segmentation techniques make the assumption that the hand is the closest object to the camera. With this assumption, thresholding [14, 1] or region growing [15] techniques can be used to extract hand regions. Other researchers have proposed to use skin color along with depth data for hand segmentation [16].

After the hand region is segmented out, one can use the hand silhouette as a shape descriptor [15, 1, 2]. Another approach is to divide the hand region into cells and use the cell occupancy information as features [14]. In [16], downsampled hand image with dimensionality reduction is used as features.

Our hand segmentation approach is related to [1] that first detects human body and then searches for the hand region inside it, as well as to [14] that uses Otsu thresholding technique to segment hand.

3. SYSTEM OVERVIEW

Our gesture recognition pipeline consists of the following steps:

1. **Segmentation.** For each input depth map, we first perform hand segmentation to obtain a set of hypothesized hand regions. Sometimes this set might contain erroneously segmented regions.
2. **Tracking and filtering.** At the tracking stage, we find correspondence between hypothesized hand regions at current and previous frames, and find the true hand region among all the hypotheses for the current frame.
3. **Orientation normalization.** After obtaining the hand region, we determine its position and orientation. We then rotate the hand in such a way that the palm will be approximately parallel to the image plane.
4. **Feature extraction.** We use normalized hand depth map to extract visual features. We developed two types of visual features: cell occupancy features and silhouette features.
5. **Classification.** The visual feature vector obtained from each frame is fed to an action graph for gesture classification.

3.1. Segmentation and tracking

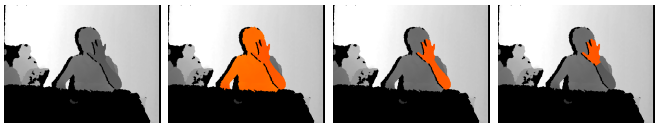
For segmentation, we assume there is only a single person in front of the depth camera, and this person occupies a significant portion of the camera's field of view. Furthermore, we assume that the hand is closer to the camera than the arm and the body. These assumptions are reasonable for many practical scenarios.

We first divide the depth map into a number of blobs using a connected-component labeling algorithm [17]. Two adjacent pixels are connected if the difference between their depth values is less than a pre-specified threshold. The resulting connected components are called blobs.

After we obtain the blobs, we find the largest blob and identify the other blobs which are close to the largest blob. We then use Otsu's method to separate the arm region from the body. Below is an outline of the algorithm.

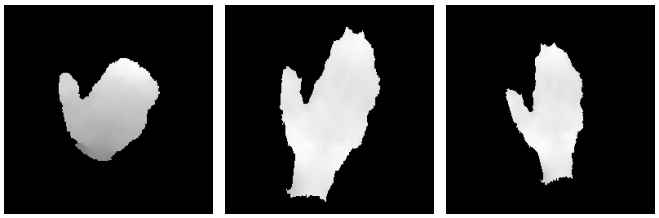
1. **Find human body.** Find the biggest blob, and denote it as $MaxBodyBlob$. Let B denote the set of blobs which consists of $MaxBodyBlob$ and the other blobs whose distances from $MaxBodyBlob$ are less than a threshold. The union of the blobs in B is considered as the human body.
2. **Find hypothesized hand/arm regions.**
   (a) Calculate a depth threshold t based on Otsu's method for all the points in the blobs of B.
   (b) Apply threshold t to points in B and select all the points which are closer to the camera than t. These points form several connected components $H^1, \ldots, H^n$ each of which is a hypothesized hand region.
   (c) The true hand region is determined through blob tracking. We establish correspondences between blobs $(H^1, \ldots, H^n)$ in the current frame and those in the previous frames. The blob with the longest track is chosen as the true hand region.
3. **Refine the hand region.** The obtained hand region might contain portions of the arm. We use the geometric size and aspect ratio to determine whether the region needs refinement. If so, we identify the wrist area which is the thinnest part of the arm, and remove the points beyond the wrist.

Fig. 1 shows an example of hand segmentation. The initial depth map is shown in (a). The segmented human body obtained at Step 1 is shown in (b) in orange color. The hand region obtained at Step 2 is shown in (c). The refined hand region is shown in (d).



(a) (b) (c) (d)

Fig. 1. Segmentation. (a) Initial depth image. (b) Segmented human body. (c) Initial hand region. (d) Refined hand region.



(a) (b) (c)

Fig. 2. Orientation and scale normalization. (a) Before normalization. (b) After in-depth normalization. (c) After in-depth, in-plane, and scale normalization.

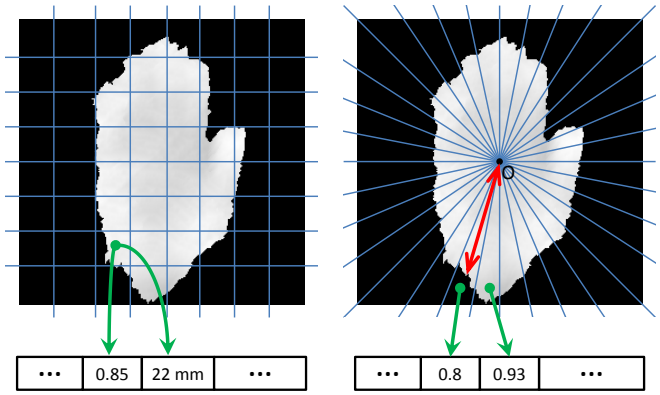
3.2. Normalization

This step performs scale and orientation normalization so that the extracted feature descriptors are scale and rotation invariant. For orientation normalization, we first estimate the hand orientation parameters, and then rotate the hand point cloud in such a way that the palm plane will be parallel to the image plane and the hand will point upward.

The normalization algorithm consists of three steps as described below:

1. **In-depth normalization.** Fit a plane P to the hand point cloud, and compute a rotation that will rotate P to be parallel to the image plane. This step is very useful when the visible surface of the hand is approximately planar. If not, such normalization could lead to an overstretched image with holes, and we do not perform in-depth normalization in this case. See fig. 2a and 2b.
2. **In-plane normalization.** We project all the points onto P and compute the principal direction. We then compute an in-plane rotation matrix so that the principal direction points upward after rotation. See fig. 2b and 2c.
3. **Scale normalization.** The hand region on P is scaled to fit into a predefined rectangle.

After the normalization procedure, we obtain the rotation parameters and a depth map of the normalized hand mesh (see fig. 2c). This image is called $HandImg$, which, along with the rotation parameters, will be used at the feature generation stage.



(a) (b)

Fig. 3. Feature extraction. (a) For cell occupancy feature, we compute the occupied area of each cell as well as the average depth for the non-empty cells. (b) For silhouette feature, we divide the entire image into fan-like sectors. For each sector, we compute the average distance from the part of the contour inside this sector to the origin O.

3.3. Feature extraction

We extract a feature descriptor for each frame. Let i denote the index of a frame. Its feature vector F_i has the following form:

$$F_i = \{\vec{v}_i, \vec{r}_i, \vec{s}_i\} \quad (1)$$

where $\vec{v}_i$ is the velocity of the hand center, $\vec{r}_i$ is the rotation parameters of the hand (quaternion in our implementation), and $\vec{s}_i$ is a shape descriptor.

Let $\vec{x}_i$ denote the center of gravity of the hand at frame i. The velocity $\vec{v}_i$ is computed as $\vec{v}_i = \vec{x}_i - \vec{x}_{i-1}$.

We have developed two different shape descriptors:

1. **Cell occupancy feature.** We divide the hand image $HandImg$ into a uniform grid (4×4, 8×8 or 16×16). For each cell of the grid we calculate its occupancy (area of the cell occupied by hand mesh) and the average depth after normalization. We scale values of the average depth into [0,1] range. Then we combine the occupancy and the average depth of each cell into a vector $\vec{s}_i$. See fig. 3a.
2. **Silhouette feature.** Use the center of $HandImg$ as an origin, and divide the whole image into a number of fan-like sectors. For each sector, compute the average distance from the hand contour in the sector to the origin. Concatenate these distances for all the sectors into a vector $\vec{s}_i$. See fig. 3b.

Due to the large dimensionality of both shape descriptors, we perform dimensionality reduction using the Principal Component Analysis (PCA). We choose the first 10-30 PCA coefficients as the new shape descriptor.

3.4. Action graph

For the back-end classifier, we use an approach called action graph, which was proposed by Li *et al*. [7] for video-based action recognition. Compared to HMM, action graph has the advantage that it requires less training data and allows different actions to share the states.

An action graph is represented as a quadruplet

$$\Gamma = (\Omega, \Lambda, G, \Psi) \tag{2}$$

where $\Omega = \{\omega_1, \ldots, \omega_M\}$ is the set of key postures, $\Lambda = \{p(x|\omega_1), \ldots, p(x|\omega_M)\}$ is the observation likelihood model, $G = (\Omega, A, A_1, \ldots, A_L)$ is the set of transition matrices (a global transition matrix, and a transition matrix for each gesture type), and Ψ is a set of gesture labels.

The training procedure for the action graph consists of key posture learning and transition matrix learning.

To learn the key postures we first cluster all the feature vectors of all the gesture types in the training data using a K-means clustering procedure. For each cluster, we fit a Gaussian distribution and estimate the observation likelihood model $p(x|\omega_m)$.

We compute elements $p(j|i)$ of transition matrices as $N_{i\to j}/N_i$, where $N_{i\to j}$ is the number of transitions from state i to j, and N_i is the number of times state i was observed. Due to the small amount of training data, the transition matrices are usually very sparse which leads to poor recognition rate. To overcome this difficulty we add a small regularization value ξ to the transition matrix, and normalize them so that $\sum_j p(j|i) = 1$. Experiments show that ξ should be chosen in the range $[0.0001, 0.01]$, so we use $\xi = 0.001$.

As in [7] we implemented 5 decoding schemes: *Action-Specific Viterbi Decoding* (ASVD), *Uni-Gram* and *Bi-Gram Global Viterbi Decoding* (UGVD and BGVD), *Uni-Gram* and *Bi-Gram Maximum Likelihood Decoding* (UMLD and BMLD). Due to lack of space we refer reader to the original paper for their detailed description.

4. EXPERIMENTS

Since there are no publicly released dataset for 3D dynamic gesture recognition, we collected a dataset of twelve American Sign Language (ASL) gestures including Bathroom, Blue, Finish, Green, Hungry, Milk, Past, Pig, Store, Where, Letter J, Letter Z (see fig. 4 and 5).

Each gesture is performed 3 times by each of the 10 participants. We use recordings of 9 persons for training and the remaining person for testing. Such experiment is repeated for five random subdivisions of the data. The overall accuracy is calculated as the average ratio of the correctly recognized gestures over the total number of test sequences.

In addition to the off-line experiment, we have also implemented an online real time system. The unoptimized version of the system runs at 10 FPS on a regular desktop machine.

Table 1 summarizes best two results for each decoding scheme. We observe that the system works the best when the dimensionality is set to 15, and the number of key postures ranges between 70 and 90.

Silhouette feature usually works better than cell occupancy features. This is probably because the most discriminative information about the hand shape is encoded in the outline of the hand.

In out experiments Uni-gram Decoding usually shows better accuracy than Bi-Gram Decoding, and Maximum Likelihood Decoding is better than Global Viterbi Decoding. Also the accuracy of ASVD is comparable to Bi-Gram schemes but always worse than Uni-Gram schemes. These results differ from [7], where Uni-Gram schemes were worse. This is probably because in our dataset some gestures could be distinguished by key shapes without having to rely on the exact sequence of key-postures, and decision of Uni-Gram decoding schemes based mostly on the used key postures, not their sequence.

Scale and orientation normalization increases the accuracy by $5 - 27\%$ for silhouette feature, and up to 10% for cell occupancy features.

Decoding scheme	Features	# Clusters	# Dims	Accuracy
ASVD	Silh.	90	15	0.777
	Silh.	70	15	0.766
BGVD	Silh.	70	15	0.8
	Silh.	90	20	0.791
BMLD	Cell	70	25	0.805
	Silh.	90	15	0.797
UGVD	Silh.	50	20	0.877
	Silh.	70	15	0.83
UMLD	Silh.	50	20	0.858
	Silh.	90	25	0.852

Table 1. The top two results for each decoding scheme.

5. CONCLUSION

We presented a real time dynamic hand gesture recognition system using a commodity depth camera. Our system is purely data-driven thus can be used to recognize any other gestures. Furthermore, it is fully automatic and robust to variations in hand orientation, performing style and speed. To evaluate the performance of our technique, we collected a small but challenging dataset of 12 dynamic ASL gestures, and studied influence of different parameters on the overall recognition accuracy (the dataset will be released to the public). This is the first data-driven system that is capable of automatic hand gesture recognition.

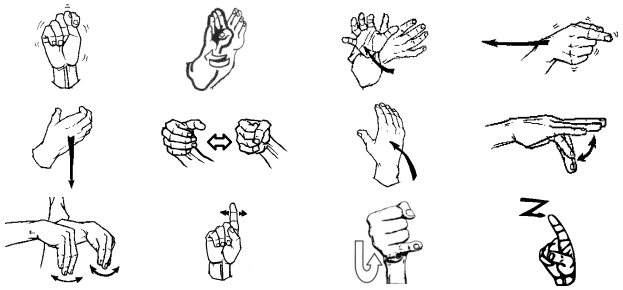


Fig. 4. Gestures from our database. Left to right, top to bottom: bathroom, blue, finish, green, hungry, milk, past, pig, store, where, letter J, letter Z.

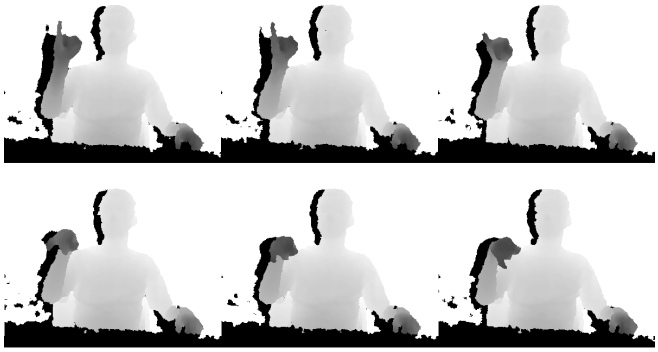


Fig. 5. Depth sequence for gesture J from our dataset.

6. REFERENCES

[1] Xia Liu and Kikuo Fujimura, "Hand Gesture Recognition using Depth Data," in *6th IEEE International Conf. on Automatic Face and Gesture Recognition*, 2004.

[2] Zhou Ren, Junsong Yuan, and Zhengyou Zhang, "Robust hand gesture recognition based on finger-earth movers distance with a commodity depth camera," in *ACM Intl. Conf. on Multimedia*, 2011.

[3] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using Hidden Markov Model," in *IEEE CVPR*, 1992.

[4] Lawrence R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Readings in speech recognition*. Morgan Kaufmann Publishers Inc., 1990.

[5] Ming-Hsuan Yang and Narendra Ahuja, "Recognizing hand gesture using motion trajectories," in *IEEE CVPR*, 1999.

[6] Pengyu Hong, Thomas S. Huang, and Matthew Turk, "Gesture modeling and recognition using finite state machines," in *4th IEEE Intl. Conf. on Automatic Face and Gesture Recognition*, 2000.

[7] Wanqing Li, Zhengyou Zhang, and Zicheng Liu, "Expandable data-driven graphical modeling of human actions based on salient postures," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 18, no. 11, pp. 1499–1510, 2008.

[8] Wanqing Li, Zhengyou Zhang, and Zicheng Liu, "Action recognition based on a bag of 3d points," in *IEEE Intl. Workshop on CVPR for Human Communicative Behavior Analysis*, 2010.

[9] Erik Sudderth, *Graphical Models for Visual Object Recognition and Tracking*, Ph.D. thesis, Massachusetts Institute of Technology, 2006.

[10] Tangli Liu, Wei Liang, Xinxiao Wu, and Lei Chen, "Tracking articulated hand underlying graphical model with depth cue," *Congress on Image and Signal Processing*, vol. 4, pp. 249–253, 2008.

[11] Ouissem Ben Henia, Mohamed Hariti, and Saida Bouakaz, "A two-step minimization algorithm for model-based hand tracking," in *18th International Conference on Computer Graphics, Visualization and Computer Vision (WSCG)*, Feb. 2010.

[12] Ouissem Ben Henia and Saida Bouakaz, "A new depth-based function for 3d hand motion tracking," in *International Conference on Computer Vision Theory and Applications(VISAPP)*, 2011.

[13] Sigurjón Árni Gudmundsson, Johannes R. Sveinsson, Montse Pardàs, Henrik Aanæs, and Rasmus Larsen, "Model-based hand gesture tracking in tof image sequences," in *6th International Conference on Articulated motion and deformable objects (AMDO)*, 2010.

[14] Poonam Suryanarayan, Anbumani Subramanian, and Dinesh Mandalapu, "Dynamic hand pose recognition using depth data," in *20th International Conf. on Pattern Recognition (ICPR)*, 2010.

[15] Eva Kollorz, Jochen Penne, Joachim Hornegger, and Alexander Barke, "Gesture recognition with a time-of-flight camera," *International Journal of Intelligent Systems Technologies and Applications*, vol. 5, November 2008.

[16] Michael Van den Bergh and Luc Van Gool, "Combining rgb and tof cameras for real-time 3d hand gesture interaction," in *IEEE Workshop on Applications of Computer Vision (WACV)*, 2011.

[17] L. Shapiro and G. Stockman, *Computer Vision*, Prentice Hall, 2002.