Loose Hand Gesture Recognition Based on Relational Features Using a Depth Sensor

Chen-Ming Chang and Din-Chang Tseng

***Abstract*—Hand gesture recognition (*HGR*) in real-time and with precision has become an important research topic. In this article, a loose hand gesture recognition (*LHGR*) system based on relational features using a depth sensor is implemented, which not only maintains an impressive accuracy in real-time processing but also enables the user to use loose gestures. *HGR* can usually be divided into three stages: hand detection, hand feature extraction, and gesture classification. However, the method we propose has been useful in improving all the stages of *HGR*. In the hand detection stage, we propose a *ROI* dynamic estimation method and a wrist-cutting method that conform to the characteristics of a human hand. In the feature extraction stage, we use the more reliable relational features which are constructed by local features, global features, and depth coding. In the gesture classification stage, we use three layers of classifiers including finger counting, finger name matching, and coding comparison; these layers are used to classify 16 kinds of hand gestures. In the end, the final output is adjusted by an adaptive decision. The average processing speed per frame is 38.6 ms. Using our method has resulted in an average accuracy of standard gestures of about 98.29%, and an average accuracy of loose gestures of about 88.32%. In summary, our *LHGR* system can robustly classify hand gestures and still achieve acceptable results for loose gestures.**

***Index Terms*—Computer vision, hand gesture recognition, human-computer interaction, image processing, kinect.**

I. Introduction

Human-computer interaction (*HCI*) has become one of the most popular research topics. Owing to the intuitive mode of operation and a high degree of freedom, hand gesture recognition (*HGR*) has always been a very popular focus area in *HCI*.

Based on different sensors, *HGR* systems can be categorized as vision-based [1]-[3] and glove-based [4], [5]. Vision-based *HGR* that uses optical sensors to capture 2D images is very sensitive to light; so, some studies have added various restrictions for a better hand segmentation. In contrast, glove-based *HGR* systems capture much more robust information from human hands. Nonetheless, users have to wear additional devices that cause inconvenience, involve extra costs, and give rise to inhibitions when gesturing. Fortunately, the lower-priced depth sensors provide a new opportunity for *HGR*. The common methods of depth sensing include stereo triangulation [6], structured light [7], and time-of-flight (*ToF*) [8]. Some research outcomes [9], [10] indicate that these methods have no absolute winner; so, a device that would be widely accepted should promote user interaction. For example, Microsoft has released the Kinect Software Development Kit (*SDK*) for programmers based on which Kinect has been used in many applications [11].

The past *HGR* studies can be divided into two types: one deals with the simplification of the complexity of hand gestures for real-time processing [3], [12]; in the other, the amount of calculations overhead required to attain precise gestures is not a significant factor [13], [14]. In addition, the improvements in *HGR* have attached importance to user experience. Consequently, loose hand gestures have to be as friendly to users as possible. Loose hand gestures allow the rotations in roll, yaw, and pitch; moreover, the fingers can have different degrees of bending, as shown in Fig. 1.

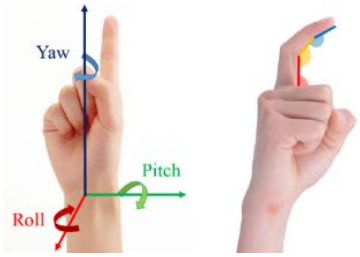


Fig. 1. A diagram of loose hand gestures.

In this article, a loose hand gesture recognition (*LHGR*) system based on relational features using a depth sensor is implemented via three stages, as shown in Fig. 2; we have realized improvements in all stages. In the hand detection stage, we can define an appropriate region of interest (*ROI*) size which completely covers the hand region depending on the depth of the hand skeleton point based on the human skeleton provided by Kinect *SDK*. Next, a wrist-cutting method that conforms to the characteristics of the human hand is proposed to remove the arm region. In the feature extraction stage, the required relational features are composed of three kinds of features, extracted in three steps. The first step is to use two signatures, namely the distance signature and the angle signature to find local features on a hand contour. The second step is implemented based on three rules of hand geometry and a double projection method to generate global features that describe the characteristics of a complete gesture. In the third step, we propose a simple method, called depth coding, to record the distribution of folded fingers within the palm region. The final stage is gesture classification by which we classify 16 kinds of gestures using three layers of classifiers which include finger counting, finger name matching, and coding comparison. Then the output can be adjusted more precisely through an adaptive decision in continuous time.

Manuscript received February 27, 2018; revised April 25, 2018.

Chen-Ming Chang and Din-Chang Tseng are with the Institute of Computer Science and Information Engineering, National Central University, Jhongli, Taiwan 32001 (e-mail: 985402003@cc.ncu.edu.tw, tsengdc@ip.csie.ncu.edu.tw).

doi: 10.18178/ijfcc.2018.7.2.515

In our study, we used an Intel Core i3-3220 as the CPU, NVIDIA GeForce GTX 650 Ti as the VGA, and a Kinect for Windows as the sensor. The average processing speed per frame was 38.6 ms; the average accuracy of standard gestures was about 98.29%, and the average accuracy of loose gestures was about 88.32%. The results show that the *LHGR* system we propose can be implemented on generic hardware devices and a reliable recognition of loose hand gestures can be achieved.

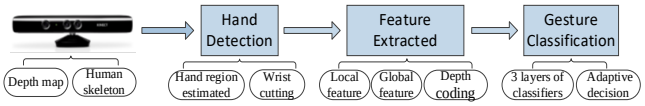


Fig. 2. The framework of our *LHGR* system.

II. Related Works

Generally, *HGR* can be divided into three stages: hand detection, feature extraction, and gesture classification. In this chapter, we will introduce a review of these stages.

A. Hand Detection

Hand detection is the first work in *HGR*. In this stage, the hand regions are estimated from images. In the past, appearance-based methods [15] were used to detect the hands, but were found to be unreliable. Wang [16] proposes a novel, real-time hand detection algorithm based on skin color. The detecting procedure is simple and fast. Nowadays, depth information acquired using depth sensors makes hand detection more easy and effective. One easy approach is to define a valid range through a depth threshold [17], [18] to detect objects. Some studies [19] use the body skeleton provided by Microsoft Kinect to assist hand detection. Ohn-Bar and Trivedi [20] developed a vision-based system that employs a combined RGB and depth descriptor to detect and classify hand gestures.

B. Feature Extraction

Feature extraction is used to obtain useful and sufficient features to describe a hand gesture; the three common methods to achieve this [11] are shape-based, 3D model-based, and skeleton-based. An alternative approach is to use the relationships among objects to define relational features [21].

Shaped-based. This approach uses hand contours which are easy to obtain with less computation; but the collected data is susceptible to noises. Ren *et al*. [22] propose the finger-earth mover's distance (*FEMD*) approach to measure the differences between hand shapes. Wong *et al*. [23] present a new superpixel-based *HGR* system based on a novel superpixel earth mover's distance (*SP-EMD*) metric, together with a Kinect depth camera.

3D model-based. This approach is an improvement from the traditional model-based method; it offers a breakthrough in the estimation of precision gestures. Oikonomidis *et al*. [15], [24], [25] propose a series of 3D model-based studies. A 3D hand model is represented as a vector of 27 parameters to encode the 26 degrees of freedom of a human hand. Particle Swarm Optimization (*PSO*) [26] is an efficient optimization algorithm used to minimize the difference between the hypothetical and observed gesture. Because the calculation of 27 parameters is computationally intensive, their studies also exploit GPU processing to speed up the *PSO*.

Skeleton-based. Here, the gesture based on the configuration of the hand skeleton is deduced. Skeleton generation is the key to skeleton-based methods. Keskin *et al*. [27] propose a real-time skeleton fitting algorithm based on random decision forests, which is used to perform per pixel classification and assign each pixel to a hand part. Fan *et al*. [28] present an algorithm for estimating a 3D hand skeleton model from a depth map based on the Active Shape Model framework.

Relational features. Using this approach, the original features are augmented with knowledge or guidelines such that the performance meets expectations. Relational features can be described in the form of graphs or rules using a specific syntax or language, which provide common relational information including adjacencies, geometrical relationships, behavior patterns, hierarchical structures, etc. Tian *et al*. [29] propose a set of comprehensive features, termed joints kinetic and relational features, for action recognition. Zweng *et al*. [30] evaluate a new algorithm for pedestrian detection using a relational feature model in combination with histogram similarity functions.

C. Gesture Classification

Gesture classification is used to classify the hand as a specifically defined gesture (static or dynamic) based on the features of the hand. Li [31] organizes three classifiers in a hierarchical manner which include the number of fingers, the finger names, the angles between each pair of fingers, to deduce the hand pose. Giulio *et al*. [32] introduce novel acquisition devices like Leap Motion and Kinect that can obtain a very informative description of the hand pose for accurate gesture recognition. This information is processed and fed into a multiclass *SVM* classifier to recognize gestures. Another new approach is Convolutional Neural Networks (*CNNs*)—a type of feed-forward artificial neural network. In recent years, *CNNs* have been used widely in computer vision [33]. So, some researchers have applied *CNNs* to hand gesture classification [34], [35] with good outcomes.

III. The Proposed Method

A. Hand Detection

We propose a *ROI* dynamic estimation method to find the most suitable *ROI* at different depths. First, the hand skeleton point provided by the Kinect skeleton is defined as the center point of ROI, and the largest size gesture is designated as the "maximally open hand". We record the length (in pixels) of the *ROI* needed for a "maximally open hand" at different depths; then the estimation formula

$$l = 0.00006d^2 - 0.279d + 360 \quad (1)$$

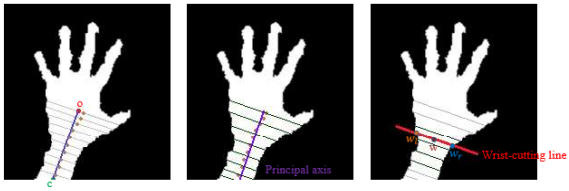
is applied based on the least squares estimation principle. In (1), d is the depth value (in millimeters) of the hand skeleton point and l is the length (in pixels) of the *ROI*.

In the general *HGR*, wrist cutting is not taken seriously. However, the quality of wrist cutting influences the result considerably. So, we present a stable wrist-cutting method

based on the length difference between the adjacent lines and the distribution of the probability histogram. An example of our wrist-cutting method is shown in Fig. 3. In Fig. 3(a), the red point o is a centroid of the whole gesture, the green point c is on the image boundary that belongs to the gesture, G' means all the gray lines which are orthogonal to $\overline{oc}$, and S is a set of the center points of G' shown in brown. In Fig. 3(b), L is the principal axis of the arm shown in purple, which is fitted by the least square estimation of S. W' is a set of candidates for the wrist-cutting line, which are orthogonal to L, shown in green. In Fig. 3(c), W'' is represented after removing the head and tail of W' to retain the middle half. Finally, the wrist-cutting line W_c shown in red can be selected from W'' by the following formula.

$$W_c = arg(\max\{ (|D_1 - D_{i-1}| + |D_i - D_{i+1}|) \times H_i \}) \quad (2)$$

In (2), $i \in W''$, D_i means the length of i presented by Euclidean distance, and H_i is the probability value of i in a histogram. If (2) is maximum when $i = t$, that t is the W_c. In addition, the left wrist point w_l, right wrist point w_r, and center wrist point w can be found on W_c. These three wrist points will play an important role in subsequent processing.



(a) (b) (c)

Fig. 3. The diagram of wrist cutting process.

B. Feature Extraction

Feature extraction is completed in three steps to find local features, global features, and depth coding. The above-mentioned features are used to compose the relational features of hand gestures.

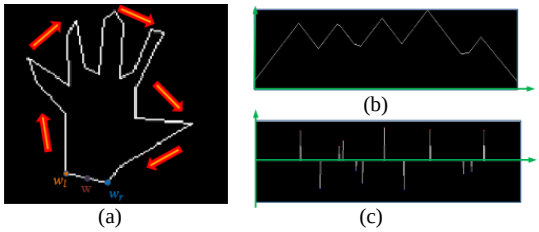
Local features include the peaks, valleys, and the known wrist points. Local features have the ability to describe the locations of the fingertips and gaps between fingers, which can be used to identify the number of protruding fingers in hand gestures. In this study, the process can be considered as a local optimum of an optimization problem.

First, we use the Douglas-Peucker algorithm (*DP*) to simplify the gesture contour, as shown in Fig. 4(a). Scanning the contour clockwise from w_l to w_r, the set of sequence vertices is recorded as $V = \{ v_1, v_2, \dots, v_n \}$, where $v_1 = w_l$ and $v_n = w_r$. Then, we make two signatures, namely the distance and the angle signature to assist local feature extraction. The distance signature is recorded as $D_s = \{ d(v_1,w), d(v_2,w), \dots, d(v_n,w) \}$, which represents the distance between each vertex and w; the angle signature is recorded as $A_s = \{ A(\angle w,v_1,v_2), A(\angle v_1,v_2,v_3), \dots, A(\angle v_{n-2},v_{n-1},v_n), A(\angle v_{n-1},v_n,w) \}$, which represents the values calculated by the following formula.

$$A(\text{angle}) = \text{Sign}(sin(\text{angle})) \times cos(\text{angle}) \quad (3)$$

In (3), the angle is that which is between each vertex and its two adjacent vertices, Sign(*sin*(*angle*)) is the direction of the cross product, and the value of *cos*(*angle*) can be used to determine the sharpness of the angle. An example for distance signature is shown in Fig. 4(b); the horizontal axis displays the sequence vertices, and the vertical axis displays their D_s. As regards the angle signature shown in Fig. 4(c), the horizontal axis displays the sequence vertices, and the vertical axis displays their A_s.

Next, we will explain how to extract the local features. First, A_s can be divided into many regions with each of the two vertices having the same direction of the cross product. Then, the local sharpest vertex in each region, the vertex with the maximum A(*angle*), called a peak is picked; otherwise the vertex with the minimum A(*angle*) is identified as a valley. Next, we use hill climbing to update the positions of peaks and valleys on D_s. Hence, the last peaks and valleys are the local optimal solutions.



(a) (b) (c)

Fig. 4. An example of the gesture contour and signatures: (a) A simplified gesture contour via *DP*. Scanning the contour clockwise from w_l to w_r; (b) A distance signature; (c) An angle signature.

Global features can be used to describe the overall gesture characteristics which include the palm region, the finger region, and the Metacarpophalangeal (*MCP*) joints. The most important of global features is *MCP* which is located at the junctions of the metacarpals and phalanges [36].

Global features are extracted by two processing works. The first work is to divide the gesture into the palm region and the finger region, according to three rules designed by the hand geometry which are:

1) *Generating the corresponding valleys*. Each peak and its two adjacent valleys can create a triangular block. But sometimes the peak nearby a wrist point may have only one adjacent valley; thus we need to generate a new valley with the same distances from the peak to the two adjacent valleys.
2) *Determining the finger region*. The angle between each peak and its adjacent valleys is examined. If the angle is less than 60, this triangle block belongs to the finger region.
3) *Correcting finger region based on trigonometry*. Ideally, the shape of each triangle block that belongs to the finger region should approximate an isosceles triangle. Here, we may generate a new valley on the longer side if needed, and the new tri-angle block forms an isosceles triangle.

Based on these three rules, we now have the local features and the newly generated valleys. Defining peaks as a set of fingertips $P=\{ p_1, p_2, \dots, p_i \}$, the remaining feature points are defined as $Q=\{ q_1, q_2, \dots, q_j \}$. The palm region is formed by Q.

In the second work, we project P onto a palm circle with a double projection method to calculate the locations of *MCP*. The center of the palm circle is the centroid of Q, called q_c;

the radius is equal to $(2\sum d(q_1,q_c)/3j)$. The first projection consists of projecting P onto their opposite sides according to the ratio of the hypotenuse side and the adjacent side, named p'. The second projection consists of projecting p' onto the palm circle, and the final projection points are the *MCP* joints. An example for estimating global features is shown in Fig. 5. In Fig. 5(a), the local feature points are displayed; the wrist points are represented in gray, the peaks in yellow, and the valleys in cyan. In Fig. 5(b), the new corresponding valleys are represented in green; the range within the red border is the palm region, and the range outside the red border is the finger region. In Fig. 5(c), the palm circle is represented in green; P is projected onto p' with the ratio $b/a = d/c$, and the *MCP* joints are represented in blue.

(a) (b) (c)

Fig. 5. An example for estimating global features: (a) Local features of hand gesture; (b) A division of a gesture into palm region and finger region; (c) The result of *MCP* joints.

Depth coding is proposed to record the distribution of folded fingers. First, we calculate an average depth value d_e of the palm region, and the points are a part of folded fingers when their depth is less than d_e. Next, the connections between two adjacent *MCP* joints are denoted right to left as $M = \{ m_1, m_2, \ldots, m_n \}$, where $0 \le n \le 4$. The proportion of the number of pixels belong to folded fingers on m_n is calculated as $rate^n$. Finally, we code M sequentially through a customized threshold T (T = 0.55 in this study). If $rate^n > T$, we code m_n to 1; otherwise, we code it to 0. An example of depth coding is shown as Fig. 6, in which the pixels of folded fingers are represented in gray, and M are represented in yellow. The codes are respectively 000 in Fig. 6(a) and 001 in Fig. 6(b).

(a) (b)

Fig. 6. Using depth coding to classify two gestures; (a) is coded as 000; (b) is coded as 001.

C. Gesture Classification

We propose three layers of classifiers to classify 16 kinds of gestures, which are “Num. 0,” “Num. 1,” “Num. 2,” “Num. 3,” “Num. 4,” “Num. 5,” “Num. 6,” “Num. 7,” “Num. 8,” “Num. 9,” “Let’s Go,” “Little,” “Rock,” “I Love You,” “OK,” and “No Ring”, is shown as Fig. 7.

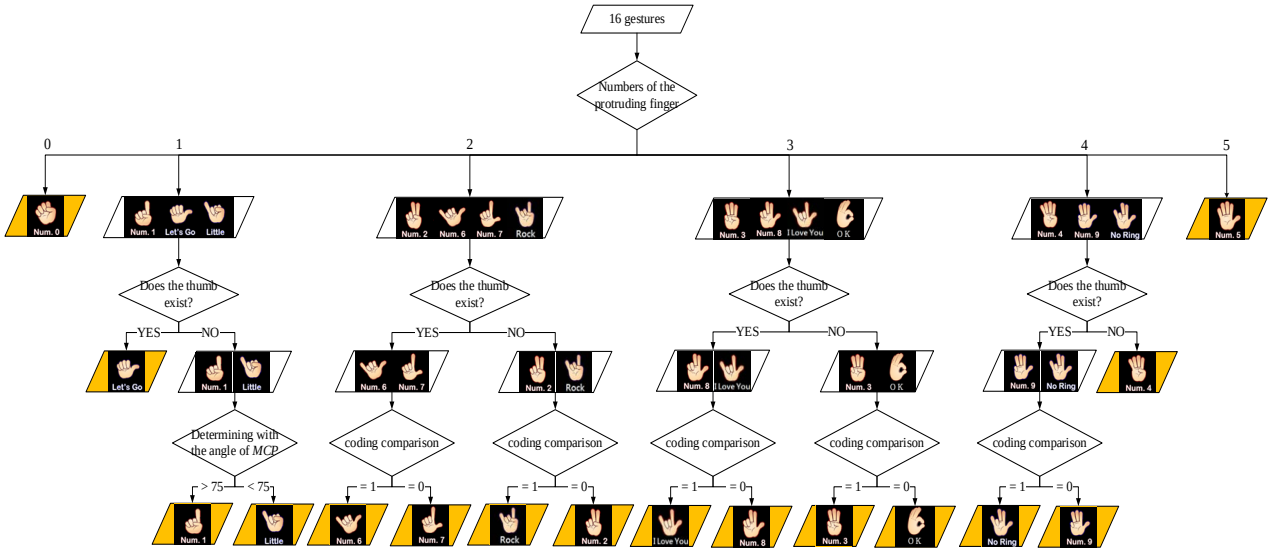


Fig. 7. Classification of 16 gestures using three layers of classifiers.

TABLE I: THE AVERAGE ANGLE OF INDIVIDUAL FINGERS IN DIFFERENT CATEGORIES

Finger name	Little	Ring	Middle	Index	Thumb
5 fingers	Don’t care	Don’t care	Don’t care	Don’t care	Don’t care
4 fingers	48	71	87	102	123
3 fingers	46	74	89	111	135
2 fingers	58	Null	103	117	143
1 fingers	52	Null	Null	106	128
0 fingers	Null	Null	Null	Null	Null

The three layers of classifiers are as below:

Finger counting. The number of protruding fingers of local features can be used to classify 16 kinds of gestures into 6 categories.

Finger name matching. Calculating the included angle between each *MCP*, w, and w_l from global features. Next, counting the average angle of each finger in different categories; the results are shown in Table I. Although we can match the protruding fingers using Table I to classify gestures, an ideal result for loose gestures is evasive. Eventually, we only determine whether the thumb exists using the relatively stable thumb angle. The categories can be further divided into two statuses—with thumb and without thumb. Notably, because there is no depth coding information for only one protruding finger, the classification is completed by matching only one finger name.

Coding comparison. For the last layer of classifiers, all cases of depth coding corresponding to the two statuses of gestures are shown in Table II(A). This form can be simplified into Table II(B); thus, we can just compare the last

number of depth coding to complete the classification.

TABLE II: DEPTH CODING CORRESPONDING TO THE GESTURES: (A) ALL CASES OF DEPTH CODING CORRESPONDING TO THE TWO STATUSES OF GESTURES; (B) A SIMPLIFIED FORM FROM (A)

(A)

Coding	With thumb	No thumb
0	"Num. 7"	"Num. 2"
1	"Num. 6"	"Rock"
00	"Num. 8"	"OK"
01	"I Love You"	"Num. 3"
10	X	"OK"
11	"I Love You"	"Num. 3"
000	"Num. 9"	X
001	"No Ring"	X
010	X	X
011	"No Ring"	X
100	X	X
101	X	X
110	X	X
111	X	X

(B)

Coding	With thumb	No thumb
0	"Num. 7"	"Num. 2"
1	"Num. 6"	"Rock"
—0	"Num. 8"	"OK"
—1	"I Love You"	"Num. 3"
——0	"Num. 9"	X
——1	"No Ring"	X

In the last work of our *LHGR*, we use an adaptive decision to record each finger state corresponding to gestures in the last five frames; a finger is considered reliable when it has a sufficiently big accumulated value. The final output will be determined by these reliable fingers; hence, the adaptive decision can effectively reduce the impact of misjudgments.

IV. RESULTS

A. Standard Gesture Recognition Results

TABLE III: THE STANDARD GESTURE RECOGNITION RESULTS OF THREE METHODS RESPECTIVELY

	(*i*)	(*ii*)	(*iii*)
Num. 0	1	1	1
Let's Go	0.99270073	0.99270073	0.97080292
Num. 1	0.980769231	0.980769231	0.974358974
Little	0.979310345	0.979310345	0.95862069
Num. 6	1	1	0.978571429
Num. 7	0.98013245	0.98013245	0.953642384
Rock	1	0.926380368	0.779141104
Num. 2	0.979591837	0.925170068	0.952380952
I Love You	0.984	0.848	0.776
Num. 8	1	0.873417722	0.772151899
Num. 3	0.920731707	0.841463415	0.756097561
OK	0.951048951	0.804195804	0.692307692
No Ring	1	0.776119403	0.656716418
Num. 9	0.965811966	0.794871795	0.735042735
Num. 4	0.993710692	0.993710692	0.962264151
Num. 5	1	1	1
Total	0.982942431	0.922814499	0.872921109

The standard gesture is defined as a normal hand with straight fingers and without intense rotation. In addition to our method (*i*), we designed two methods for verification; method (*ii*) with two layers of classifiers that does not use depth coding; and method (*iii*) with two layers of classifiers that does not use depth coding but uses peaks instead of *MCP* joints. The standard gesture recognition results of the above three methods are presented in Table III; the accuracy rate of standard gestures is about 98.29% when using our method (*i*), 92.28% using method (*ii*), and 87.29% when using method (*iii*). The experimental results have proved that *MCP* joints and depth coding are useful features.

B. Loose Gesture Recognition Results

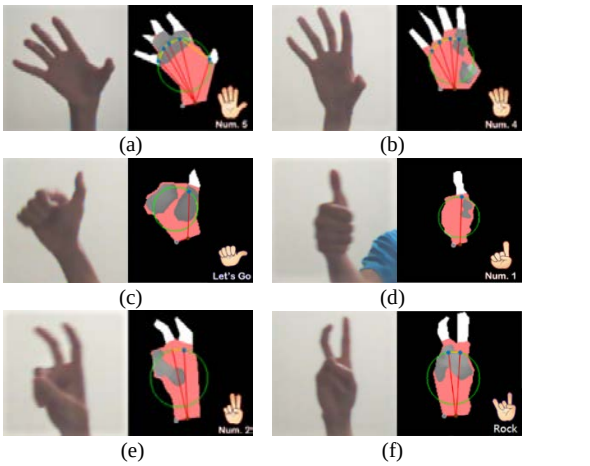
The loose gesture recognition results from our method are presented in Table IV. The accuracy rate of loose gestures is about 88.32%, and the worst gesture is "OK" at an accuracy rate of about 77.17%. Roughly, although the accuracy of loose gestures is much lower compared to that of standard gestures, we have achieved acceptable results for loose gestures.

There are six examples for loose gesture recognition, as shown in Fig. 8. From a comparison of Fig. 8(a) and Fig. 8(b), while both the actual gestures are "Num. 5," we can see that Fig. 8(b) is mistaken for "Num. 4" because of its incomplete shape. From a comparison of Fig. 8(c) and Fig. 8(d), while both the actual gestures are "Let's Go," we can see that the Fig. 8(d) is mistaken for "Num. 1" because of the violent gesture rotation. Comparing Fig. 8(e) and Fig. 8(f), while both the actual gestures are "Num. 2," it can be seen that Fig. 8(f) is mistaken for "Rock" due to the distribution of folded fingers (or say, the rotation in yaw is the major reason). Therefore, the yaw rotation is the most harmful to the system because it may simultaneously affect the gestural shape and distribution of folded fingers.

TABLE IV: THE LOOSE GESTURE RECOGNITION RESULTS ON OUR METHOD

	Num. 0	Let's Go	Num. 1	Little	Num. 6	Num. 7	Rock	Num. 2	I Love You	Num. 8	Num. 3	OK	No Ring	Num. 9	Num. 4	Num. 5
Num. 0	1															
Let's Go		0.896552	0.06977		0.0229											
Num. 1		0.103448	0.89922	0.074074		0.017391										
Little			0.03101	0.916667	0.01527											
Num. 6					0.9084	0.034783	0.11667	0.00735								
Num. 7						0.895652		0.02941								
Rock				0.009259	0.05344		0.85	0.16176			0.01408					
Num. 2						0.052174	0.03333	0.80147			0.01408	0.01575				
I Love You									0.9310345	0.11364			0.00794			
Num. 8									0.0258621	0.85606	0.00704	0.00787	0.02381			
Num. 3									0.0431034	0.00758	0.80282	0.20472			0.02459	
OK										0.02273	0.16197	0.77165				0.01361
No Ring													0.94444	0.1374		
Num. 9														0.82445		
Num. 4													0.02381	0.03817	0.91803	0.04762
Num. 5															0.05738	0.93878

Total accuracy of standard gestures = 0.88320326



(a) (b) (c) (d) (e) (f)

Fig. 8. Six examples for loose gesture recognition, the left side of which shows a color image, and right side shows the relational features and the recognition result.

V. CONCLUSION

We implemented a *LHGR* system based on relational features using a Kinect, whereby improvements in all the stages have been realized to maintain a good accuracy in real-time processing. The computer that we used for *LHGR* comprises commonly available parts in our study. The average processing speed of each frame was 38.6 ms, the average accuracy of standard gestures was about 98.29%, and the average accuracy of loose gestures was about 88.32%.

Some directions for future work are to increase the varieties of recognized gestures to improve the accuracy of recognition and to reduce the environmental constraints. Our study is very close to actual life conditions and presents a great potential for development. In future, the findings from our study can be easily applied to user interaction in a variety of control applications.

ACKNOWLEDGMENT

The authors would like to thank Enago (www.enago.tw) for the English language review.

REFERENCES

[1] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan, "Vision-based hand-gesture applications," *ACM Communications*, vol. 54, no. 2, pp. 60-71, 2011.
[2] J. M. Guo, Y. F. Liu, C. H. Chang, and H. S. Nguyen, "Improved hand tracking system," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 22, no. 5, pp. 693-701, 2012.
[3] A. Malima, E. Ozgur, and M. Cetin, "A fast algorithm for vision-based hand gesture recognition for robot control," in *Proc. IEEE 14th Signal Processing and Communications Applications*, Antalya, Turkey, Apr. 17-19, 2006, pp. 1-4.
[4] P. Pławiak, T. Sosnicki, M. Niedzwiecki, Z. Tabor, and K. Rzecki, "Hand body language gesture recognition based on signals from specialized glove and machine learning algorithms," *IEEE Trans. on Industrial Informatics*, vol. 12, pp. 1104-1113, 2016.
[5] N. Tubaiz, T. Shanableh, and K. Assaleh, "Glove-based continuous Arabic sign language recognition in user-dependent mode," *IEEE Trans. on Human-Machine Systems*, vol. 45, pp. 526-533, Aug. 2015.
[6] A. Saxena, S. H. Chung, and A. Y. Ng, "3-D depth reconstruction from a single still image," *International Journal of Computer Vision*, vol. 76, pp. 53-69, Jan. 2008.
[7] Y. Zhang, Z. Xiong, Z. Yang, and F. Wu, "Real-time scalable depth sensing with hybrid structured light illumination," *IEEE Trans. on Image Processing*, vol. 23, no. 1, pp. 97-109, Jan. 2014.
[8] S. Foix, G. Alenya, and C. Torras, "Lock-in Time-of-Flight (ToF) cameras: A survey," *IEEE Sensors Journal*, vol. 11, pp. 1917-1926, Sept. 2011.
[9] H. Sarbolandi, D. Lefloch, and A. Kolb, "Kinect range sensing: Structured-light versus time-of-flight Kinect," *Computer Vision and Image Understanding*, vol. 139, pp. 1-20, Oct. 2015.
[10] C. D. Mutto, P. Zanuttigh, and G. M. Cortelazzo, "TOF cameras and stereo systems: comparison and data fusion," in *TOF Range-Imaging Cameras*, F. Remondino and D. Stoppa, Eds. Germany: Springer-Verlag Berlin Heidelberg, 2013, pp. 177-202.
[11] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with Microsoft Kinect sensor: A review," *IEEE Trans. on Cybernetics*, vol. 43, no. 5, pp. 1318-1334, Oct. 2013.
[12] J. Hong, E. S. Kim, and H.-J. Lee, "Rotation-invariant hand posture classification with a convexity defect histogram," in *Proc. 2012 IEEE International Symposium on Circuits and Systems*, Seoul, South Korea, May 20-23, 2012, pp. 774-777.
[13] A. Erol, G. Bebis, M. Nicolescu, R.D. Boyle, and X. Twombly, "A review on vision-based full DOF hand motion estimation," in *Proc. 2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition - Workshops*, San Diego, CA, USA, June 20-26, 2005, pp. 75-82.
[14] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect," in *Proc. 22nd British Machine Vision Conf.*, Dundee, UK, Aug. 29-Sept. 2, 2011, pp. 1-11.
[15] P. M. Roth and M. Winter, "Survey of Appearance-based Methods for Object Recognition," Ph.D. dissertation, Institute for Computer Graphics and Vision, Graz Univ. of Tech., Graz, Austria, 2008.
[16] Y. R. Wang, W. H. Lin, and L. Yang, "A novel real time hand detection based on skin-color," in *Proc. 2013 IEEE 17th International Symposium on Consumer Electronics*, Zhubei City, Taiwan, June 3-5, 2013, pp. 141-142.
[17] K. K. Biswas and S. K. Basu, "Gesture recognition using Microsoft Kinect," in *Proc. 5th International Conf. on Automation, Robotics and Applications*, Wellington, New Zealand, Dec. 6-8, 2011, pp. 100-103.
[18] H. Du and T. To, "Hand Gesture Recognition using Kinect." Ph.D. dissertation, Dept. Elect. and Computer Eng., Boston Univ., MA, 2011.
[19] Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti, "American sign language recognition with the Kinect," in *Proc. 13th International Conf. on Multimodal Interfaces*, Alicante, Spain, Nov. 14-18, 2011, pp. 279-286.
[20] E. Ohn-Bar and M. M. Trivedi, "Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations," *IEEE Trans. on Intelligent Transportation Systems*, vol. 15, no.6, pp. 2368-2377, Dec. 2014.
[21] A. P. Dhawan, *Medical image analysis*; New Jersey: John Wiley & Sons, 2003.
[22] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using Kinect sensor," *IEEE Trans. on Multimedia*, vol. 15, no. 5, pp. 1110-1120, Aug. 2013.
[23] C. Wang, Z. Liu, and S.-C. Chan, "Superpixel-based hand gesture recognition with Kinect depth camera," *IEEE Trans. on Multimedia*, vol. 17, no. 1, pp. 29-39, Jan. 2015.
[24] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Markerless and Efficient 26-DOF Hand Pose Recovery," in *Proc. 10th Asian Conf. on Computer Vision*, Queenstown, New Zealand, Nov. 8-12, 2010, pp. 744-757.
[25] I. Oikonomidis, N. Kyriazis, A. A. Argyros, "Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints," in *Proc. 2011 IEEE International Conf. on Computer Vision*, Barcelona, Spain, Nov. 6-13, 2011, pp. 2088-2095.
[26] J. Kennedy and R. Eberhart, "Particle swam optimization," in *Proc. 1995 IEEE* International *Conf. on Neural Networks*, Piscataway, New Jersey, USA, Nov. 27-Dec. 1, 1995, pp. 1942-1948.
[27] C. Keskin, F. Kirac, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *Proc. 2011 IEEE International Conf. on* Computer *Vision Workshops*, Barcelona, Spain, Nov. 6-13, 2013, pp. 1228-1234.
[28] C.-Y. Fan, M.-H. Lin, T.-F. Su, S.-H. Lai, and C.-H. Yu, "3D hand skeleton model estimation from a depth image," in *Proc. 14th IAPR International Conf. on Machine Vision Applications*, Tokyo, Japan, May 18-22, 2015, pp. 489-492.
[29] X. Tian and J. Fan, "Joints kinetic and relational features for action recognition," *Signal Processing*, vol. 142, pp. 412-422, Jan. 2018.
[30] A. Zweng and M. Kampel, "Performance evaluation of an improved relational feature model for pedestrian detection," in *Proc. 2013 IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, Clearwater, FL, USA, Jan. 15-17, 2013, pp. 53-60.
[31] Y. Li, "Hand gesture recognition using Kinect," in *Proc. 2012 IEEE 3rd International Conf. on Software Engineering and Service Science*, Beijing, China, June 22-24, 2012, pp. 196-199.
[32] M. Giulio, F. Dominio, and P. Zanuttigh, "Hand gesture recognition with leap motion and kinect devices," in *Proc. 2014 IEEE International Conf. on Image Processing*, Paris, France, Oct. 27-30, 2014, pp. 1565-1569.
[33] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information* Processing *Systems*, pp. 1097-1105, 2012.
[34] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand gesture recognition with 3D convolutional neural networks," in *Proc. 2015 IEEE Conf. on Computer Vision and Pattern Recognition Workshops*, Boston, MA, USA, June 7-12, 2015, pp. 1-7.
[35] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "Robust 3D hand pose estimation in single depth images: From single-view CNN to multi-view CNNs," in *Proc. 2016 IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas, Nevada, USA, June 26-July 1, 2016, pp. 3593-3601.
[36] I. M. Bullock, J. Borras, and A. M. Dollar, "Assessing assumptions in kinematic hand models: A review," in *Proc. 2012 4th IEEE RAS & EMBS International Conf. on Biomedical Robotics and Biomechatronics*, Roma, Italy, June 24-27, 2012, pp. 139-146.

Chen-Ming Chang received his B.S. degree in information engineering and computer science from Feng Chia University, Taichung, Taiwan, in 2007; and M.S. degree in information and computer engineering from Chung Yuan Christian University, Jhongli, Taiwan, in 2009. He is pursuing the Ph.D. degree in the Department of Computer Science and Information Engineering at National Central University, Jhongli, Taiwan. His research interests include computer vision, image processing, and deep learning, especially in the topic: computer vision techniques for human computer interaction.

Din-Chang Tseng received his Ph.D. degree in information engineering from National Chiao Tung University, Hsinchu, Taiwan, in June 1988. He has been a professor in the Department of Computer Science and Information Engineering at National Central University, Jhongli, Taiwan since 1996. He is a member of the IEEE. His current research interests include computer vision, image processing, and virtual reality; especially in the topics: computer vision system for advanced safety vehicle, computer vision techniques for human computer interaction, and view-dependent multi-resolution terrain modeling.