Static Fingerspelling Recognition Based on Boundary Tracing Algorithm and Chain Code

Ahmad Yahya Dawod
Pattern Recognition Research Group, Centre for Artificial Intelligence Technology (CAIT), Faculty of Information Science and Technology, The National University of Malaysia, 43600 Bangi, Malaysia.
+60147015877
ahmadyahyadawod@gmail.com

Md Jan Nordin
Pattern Recognition Research Group, Centre for Artificial Intelligence Technology (CAIT), Faculty of Information Science and Technology, The National University of Malaysia, 43600 Bangi, Malaysia.
+60193333705
jan@ukm.edu.my

Junaidi Abdullah
Faculty of Computing & Informatics, Centre for Assistive Technology, Multimedia University (MMU), 63000, Cyberjaya, Selangor, Malaysia.
+60133507899
junaidi@mmu.edu.my

ABSTRACT

This paper presents a novel method for the detection and extraction of shape feature for fingerspelling recognition using boundary tracing and chain code. The method includes several steps such as conversion of RGB to *YCbCr* color space of an image and segmentation of skin pixel regions using thresholding method in order to construct binary images. Edge detection is applied and the location of candidate fingertips is estimated based on boundary tracing process and local extrema. The modified 2D chain code algorithm is then applied to the edge image to extract the fingerspelling shape feature and Support Vector Machine (SVM) is used for the classification task. The experimental findings show that the accuracy of the proposed method is 97.75% and 96.48% for alphabets and numbers, respectively.

CCS Concepts

• Computing methodologies → Supervised learning by classification

Keywords

Fingertips detection, shape feature, hand tracing, fingerspelling, and chain code.

1. INTRODUCTION

Human-computer interaction plays a significant role in assisting a machine to understand any given input from human such as hand gestures. Hand gestures may include the fingertips that may represent rich information. Therefore, locating the fingertips in an image accurately is one of the essential tasks for human and computer to cooperatively accomplish a given task [1]. This process can be helpful for the assistance and integration of deaf people into society, through some particular task such as hand gesture recognition and interpretation in the domain of sign language.

ISMSI '18, March 24–25, 2018, Phuket, Thailand
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-6412-6/18/03….$15.00
https://doi.org/10.1145/3206185.3206195

The speech and hearing-impaired people normally use fingerspelling signs as basic communication tool. It typically consists of hand postures for numbers and alphabets of the sign language. The task of finger sign recognition using computer vision is complex and the challenges can be related to self-occlusion, one-handed posture, and out-of-plane rotation. In some cases, the task can become more complicated whenever one of the hands overlaps the other. This happens when the hands are articulating a combined posture. The implementation of a general hand posture recognizer becomes very challenging due to different degrees of freedom of each hand that perform different new shapes [2].

For hand gesture recognition, hand contour is normally used when real-time performance is needed. Related works for hand gesture recognition are categorized into software and hardware-based solution. Some works categorized under the software-based solution use hand contour to acquire characteristic points for computational time reduction [3]. This is similar to our proposed approach. However, their extraction of the interested points involve a lot of steps which include calculation of distance between all pixels and the contour of the object and other scanning operations which is done in full-frame. There are also other proposed methods for finger localization such as contour curvatures [4] and hand skeleton [5]. However, the performance of these methods is generally low.

If the standard sign for the desired word is non-existent, signers normally use fingerspelling gestures to spell out the word that represent the English alphabet letters. Unlike signs that represent words, fingerspelling is performed using one hand and most of them do not require motion. Different alphabet letters are normally distinguished by the finger position of the signer, which is also known as the hand shape. Fingerspelling recognition approach is basically about learning to recognize each letter's hand shape before dealing with letters in sequence.

In this paper, a method to represent contour based on chain code is proposed to classify the hand shape performing the static fingerspelling.

2. RELATED WORDS

There are many research that can be found in the literature in the area of hand gesture recognition. As a motivation to assist the people with hearing difficulty to communicate with normal people through sign language recognition, a system is proposed to recognize the static hand gestures representing 26 ASL alphabets from various background based on Euclidean distance measure [6, 7].

A hand posture recognition in real-time is developed using a range camera that provides distance and visual data based on the closest pixels to the camera for hand area segmentation. For classification, a voxel representation on every incoming image is created and compared with a database [5]. The recognition rate of the system is 93.88%. However, the experiment involves only 12 hand gestures and some signs which is almost similar with the American Sign Language (ASL) signs such as “A”, “B”, “E” are not considered.

In [8], back propagation neural network is proposed for the hand gesture recognition. In [9], classification of hand gestures for alphabets is done using neural network after the hand segmentation steps from the input image. Peaks and valleys are extracted from the color space to be the features. Skin detection is known as the process selecting pixels that correspond to human skin in a given image. The process includes collecting the skin images of several people as image library or a sample. The collection of the sample should be as diverse as possible in order to achieve more accurate results. In addition, a large number of images is recommended to make sure the quality of the model remains at high level [10].

Fingerspelling in ASL consists of the same 26 letters as English alphabets to spell words out. A single hand is normally used for fingerspelling that comprises of 24 static and 2 dynamic hand gesture. Some of these gestures appear to be very similar. The two dynamic gestures are the letter J and Z; it requires the movement of the little finger and index finger for letter J and Z, respectively. Due to the similarity of some of the gestures, the task of fingerspelling classification is very challenging, even for a small subset (see Figure. 2 (b)) [11]. Common boundary tracking algorithm in the literature includes the worm following method. An ideal bug moving from white background pixel area to black background pixel area and the black pixel region is represented as a closed contour. As it enters the black pixels, it turns left and continues to the next pixel. If the next pixel is also black, the worm turns left. However, if the pixel is white, the worm turns right. This process continues until the worm reach the original point [12].

3. A PROPOSED ALGORITHM

An overview of the proposed method is described in this section. The method consists of five main phases: (i) image acquisition, (ii) morphological operation, (iii) edge detection, (iv) boundary tracing, and (v) classification. The steps of the proposed method is depicted in Figure 1.

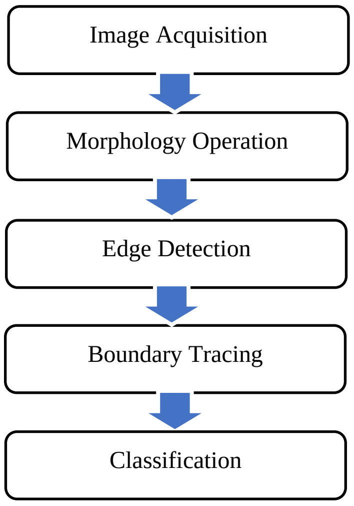
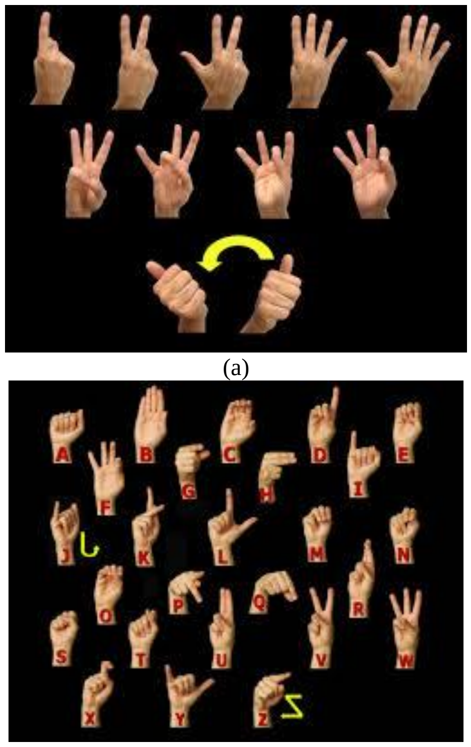


Figure1. Steps in proposed hand boundary tracing method

3.1 Image Acquisition

In this stage, hand gestures of the ASL alphabets are acquired. It is preferable to take images with simple background and having high contrast between the hand and the background to make the following stages to be less complex. In this study, a camera is used to capture the gestures and these images may possess cluttered background and varied illumination. The static ASL sign as well as normal static gesture images are captured using a webcam where the resolution is 320 x 240 pixels. Some of the images are collected from several open data sources [13] and the ASL fingerspelling images are obtained from an available online datasets (http://www.lifeprint.com/asl101/pages-ayout/handshapes.htm). In this study, only some image of gestures, i.e., numbers (1-20) and alphabets (A-Z) of the ASL fingerspelling are utilized (Figure 2 (a)).



(a)

(b)

Figure 2. Images of ASL fingerspellingdataset. (a) Number, and (b) Alphabet.

3.2 Morphology Operation

In this step, the acquired image is converted into binary image and the hand is segmented using free-form skin color model [14]. Morphological operations is applied on the segmented binary image which include opening and dilation operation. In opening operation, tiny connected components are removed and contours are smoothened. Meanwhile, the dilation operation shapes the contour and increase the skin region size by employing a structuring element. This will aid the performance of the region labeling operation. The output of this process is a morphed image. In this work, the hand is separated from the noisy background using the dilation and erosion involving 3×3 structuring elements [15]. Let N ∈ X^2 , where the 2D space of (x, y), and let M ∈ X^2 be the SE, that controls the structure for the morphological operations. For a binary image I where N ⊆ I, we can define dilation and erosion as in Equation 1 and 2, respectively. Where

$\dot{\hat{M}}$ is the reflection of M and it is an alternative for the complement operation. The dilation of N by M is defined as

$$N \oplus M = \{X \mid \left(\dot{\hat{M}}\right)_x \cap N \neq \phi\} \qquad (1)$$

The dilation operator gives N and M sets in X^2. The erosion of N by structuring element M is defined as

$$N \ominus M = \{X \mid (M)_X \subseteq N\} \qquad (2)$$

Erosion operator gives N and M sets in X^2. The erosion of N by structuring element M is the set of all points x, where M translated by x, is contained in N. Figure 4 illustrates the comparison of hand image before and after morphological operations (dilation and erosion).

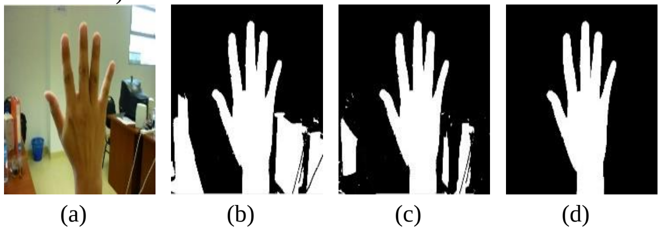


Figure 4. Morphological operations; (a) Original image, (b) Before dilation and erosion, (c) After dilation and erosion, and (d) After biggest blob detection

In this work, biggest blob detection is employed after the morphological operations. This is to remove unwanted objects that have similar color with the skin from the background. First, each blob is assigned a unique label to separate it from other blobs. All pixels that are spatially connected and exist within the blob of 1s are assigned with the same labels. Next, the blob with the biggest area will be chosen as the hand and all other blobs will be removed. Then, the geometric properties of the hand's posture is used to identify the fingers. If the tips of the fingers are closer to each other, then the identification of fingers on the image becomes more complicated. In order to find the candidate finger points, Euclidean distances are calculated between the pixels contour and the hand centroid the fingers can be identified through the extraction of the local maximas from the Euclidean distances.

3.3 Edge Detection

For edge detection, Canny edge detector is applied to the hand silhouette. It finds complex object boundaries in an image near the area where there is an occurrence of changes in brightness. Figure 5 shows the detected edge after applying the operator.

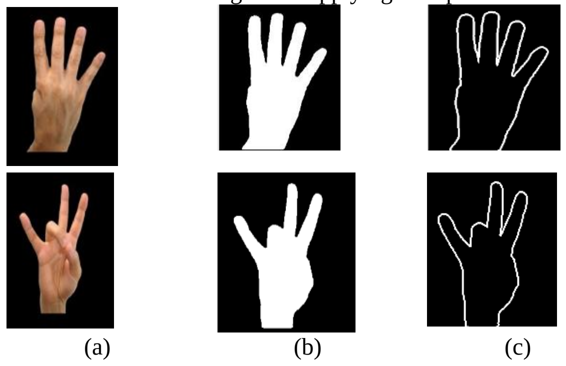


Figure 5. Edge detection (a) Original image, (b) Binarization images, and (c) edge detection.

3.4 Boundary Tracing

The outline of a hand in a binary image is traced using boundary tracing algorithm, where the non-zero pixel area is separated from the zero pixel area in order to extract the hand contour. To find the first non-zero pixel, P0, the algorithm starts scanning the image in the order of left to right and top to bottom. The location of detected P0 is then used as the region border starting point. The tracing process then continues by searching for the next neighboring point in a clockwise direction by 8-pixel connectivity until it reaches the starting point again. The visited contour points from the described process are known as border-pixel-vector (BPV). Figure 6 shows an example of the image upon the application of the boundary tracing algorithm.

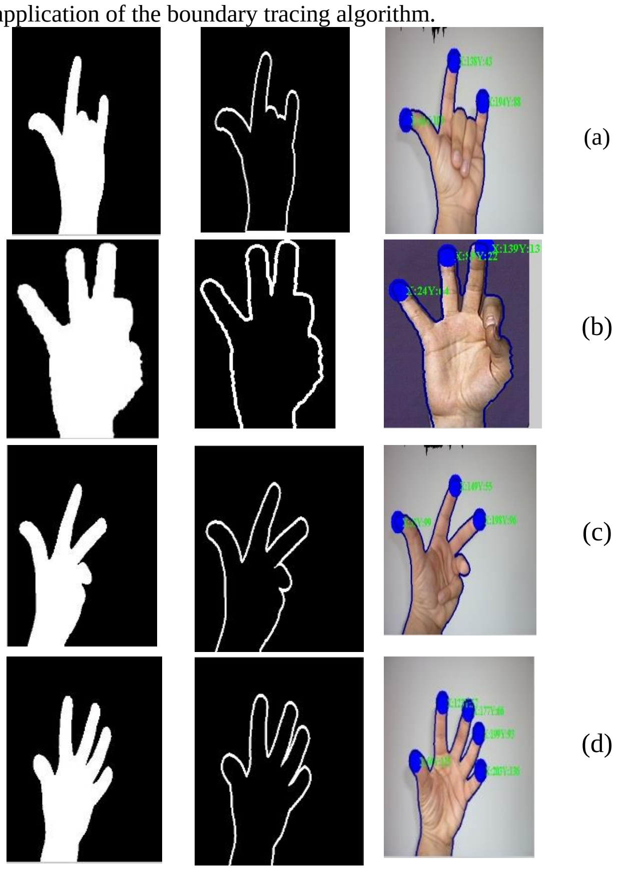


Figure 6. Blue line illustrates the hand boundary. (a) Love, (b) F, (c) 9, and (d) Claw.

3.4.1 Freeman Chain Code

Figure 7 illustrates two common methods used which define the neighborhood of a pixel namely the 4 and 8-connectedness. The 8-connectedness with diagonal connections is employed in this work. Since the boundary is the representation of the object shape, using chain code can provide an efficient representation of the boundary [16].

An object contour can be defined as a closed line representing all the pixels along the object border. Chain code can represent the boundary of an object through straight line segments which are connected in sequence. The direction and length of the line segments can be specified according to the nature of the boundary. A sequence of an integer is used to represent this straight line segment. Let sequence of integer $c = \{c_{0,}, c_1, c_{n-1}\}$, where c_0

and c_{n-1} represent the start and end point of the code respectively. The value of n is equal to 8 and 4 for 8- and 4-directional chain code, respectively, where $i = \{0,1,\ldots,7\}$, as shown in Figure 7. The sequence of the code is determined by 8-directional chain code. The object is scanned from top left to bottom right in order to find the chain code. After the determination of the object's starting point, the boundary is traversed until the end pixel. The directional code is identified and stored in an array.

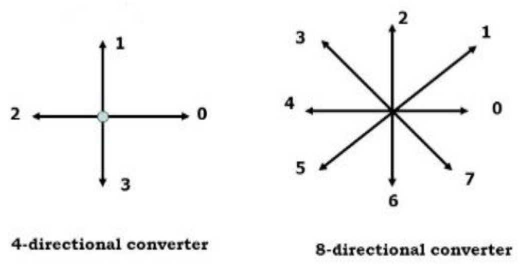


Figure 7. 4- and 8-directional chain code.

In Figure 8, if the starting point is set at point A on the original object, the 8-directional chain code produces 0642 as the code. Initial chain code is defined by the starting point of the contour of an object. Normalization of the chain code is needed since the changes of the rotation and scale of the object affects the chain code. The basic chain code is limited to translation invariance, hence differential chain code is used for rotationally invariance. Meanwhile, scale invariance is obtained by modifying the sampling grid size.

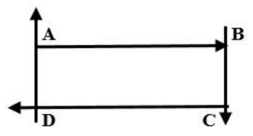


Figure 8. Original Object

3.4.2 Differential chain code

The first difference of chain code is used to obtain the differential chain code. In this case, we calculate the required number of transitions from the first number to the second number in counterclockwise direction as explained in [17]. The first difference of chain code is rotationally invariant. After the normalization of the differential code, the shape number is then obtained.

3.4.3 Normalization

The purpose of normalization is to get minimum number of integer in the sequence by redefining the starting point. The steps for normalization start with the finding of chain code object and redefining its starting point. The last number is set as the first position, the first difference of chain code is calculated and the shape number is determined.

Figure 9 illustrates the rotated image of Figure 7. The shape number of the rotated object is similar to the original object because the differential chain code is rotationally invariant. The chain code of the rotated object is 7531 (A-D), the redefine starting point is 17531 (DABCD), and the first difference are 1-7, 7-5, 5-3, 3-1

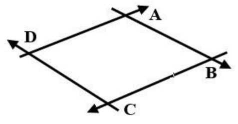
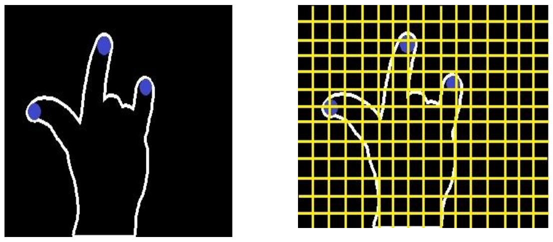


Figure 9. Rotation of original object.

3.4.4 Resampling Chain Code

One of the drawbacks of the chain code is its sensitivity to noise. This issue can be addressed by resampling the boundary by selecting larger grid spacing. Figure 10(a) shows the original image points and Figure 10(b) illustrates the resampling of the original image points. Less noise is observed in the resampled image compared to the original image. Apart from that, scale invariant can be achieved using resampling. Using Freeman chain code technique, the hand reconstruction can be made from the representation of its chain code.

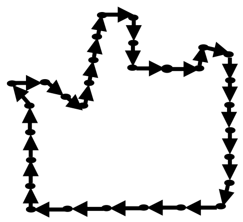


(a) (b)

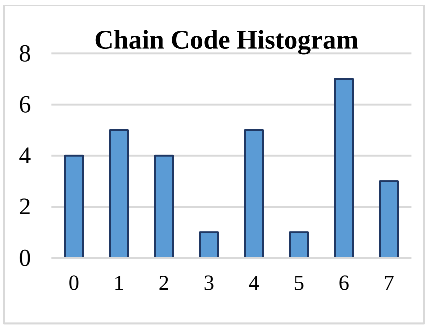
Figure 10. Original image, (a) Points of the original image, and (b) Resampling.

3.4.5 Chain Code Histogram

We employ the chain code histogram (CCH) to the group objects that represent the hands. The CCH is constructed based on the number of each directions in Freeman chain code representation of the object contour. This is shown in Figure 11.



(a)



(b)

Figure 11. (a) Freeman chain code: 066001766666544444222230771111, and (b) Chain code Histogram.

In this work, due to the simplicity and invariance in terms of translation, rotation and scale, the Freeman chain code descriptor is used.

3.5 Classification

For classification phase, the fingerspelling test data is compared with training dataset using machine learning algorithm. In this work, Support Vector Machine (SVM) is used for the classification task. SVM is known as a supervised learning model that contains machine learning algorithms that can acknowledge patterns and analyze information and generally used for multivariate analysis. Figure 12 shows the example of the classification of number and alphabets with the associate degree SVM model.

Ture Lable	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	Classified
a	98	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	1	One
b	0	91	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	Two
c	0	0	92	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	Three
d	1	0	0	92	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	Four
e	0	0	1	1	96	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Five
f	0	0	0	0	0	95	0	0	1	0	0	0	0	0	0	0	0	0	0	0	Six
g	0	0	0	0	0	0	98	0	0	0	0	0	0	0	0	1	0	0	0	0	Seven
h	0	0	0	0	0	0	0	98	0	0	0	0	0	0	0	0	0	0	0	0	Eight
i	0	0	0	0	0	1	0	0	97	0	0	0	0	0	0	0	0	0	0	0	Nine
j	0	0	0	0	0	0	0	0	0	95	0	0	0	0	0	0	0	0	0	0	Ten
k	0	0	0	0	1	0	0	0	0	0	95	3	0	0	0	0	0	0	1	1	Eleven
l	0	0	0	0	0	0	0	0	0	0	0	97	0	0	0	0	0	0	0	0	Twelve
m	0	0	0	0	0	0	0	0	0	0	0	0	97	0	0	0	0	0	0	0	Thirteen
n	1	0	1	0	0	0	0	0	0	0	0	0	0	96	0	0	0	0	0	0	fourteen
o	0	0	1	0	1	0	0	0	0	0	0	0	0	0	97	0	0	0	0	1	fifteen
p	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	96	1	0	0	0	sixteen
q	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	95	1	0	0	seventeen
r	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	97	0	0	Eightteen
s	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	95	0	Nineteen
t	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	96	Twenty
	Predicted Label																				

(a)

True Label	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	classified
a	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A
b	0	97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	B
c	0	0	97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	C
d	0	0	0	93	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	D
e	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	E
f	0	0	0	0	0	97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	F
g	0	0	0	0	0	0	97	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	G
h	0	0	0	0	0	0	0	92	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	H
i	0	0	0	0	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	I
j	0	0	0	1	0	0	0	0	0	91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	J
k	0	0	0	0	0	0	0	0	1	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	K
l	0	0	0	0	0	0	0	0	0	0	0	96	0	2	0	0	0	0	0	0	0	0	0	0	0	0	L
m	0	0	0	0	0	0	0	0	0	0	0	0	92	0	0	0	0	0	0	0	0	0	0	0	0	0	M
n	0	0	0	0	0	0	0	0	1	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0	0	N
o	0	0	0	0	0	0	0	0	0	0	0	0	0	0	92	0	0	0	0	0	0	0	0	0	0	0	O
p	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	91	0	0	0	0	0	0	0	0	0	0	P
q	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	98	0	0	0	0	0	0	0	0	0	Q
r	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	91	0	0	0	0	0	0	0	0	R
s	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	91	0	0	0	2	0	0	0	S
t	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	91	0	0	0	0	0	0	T
u	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	94	0	0	0	0	0	U
v	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	97	0	0	0	0	V
w	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	99	0	0	1	W
x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	99	0	0	X
y	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	99	0	Y
z	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	95	Z
	Predicted Label																										

(b)

Figure 12. Classification SVM (a) Number, and (b) Alphabet.

4. EXPERIMENTAL RESULTS

Although there are several fingerspelling datasets available for training and evaluation of the proposed algorithm, the datasets are lack of quality and some are not suitable for our research purposes. SVM is applied to classify number and alphabets for fingerspelling and the result is shown in Table 1. The distance between the endpoints of the finger is calculated and the sign is recognized using a rule-based system. The rule-based system involves number of branches, number of fingertips, distance between the neighboring fingertips, chain code histogram and the shape number.

Other than that, the quantitative results are also examined in this study. The performance of the system is evaluated using standard metrics that are common in classification problems. The evaluation is done with two approaches: a) the statistical approach, which evaluate the fingertip detection in $YCbCr$ color space, and b) the precision, recall, and F- measure metrics, which evaluate the fingerspelling recognition accuracy as shown in Table 2, and also the comparison performances with two other methods (see in Table 3).

Table 1. The accuracy of SVM for Number and Alphabet

Accuracy	Test Images	Number	Alphabet
SVM	100	97.75%	96.48

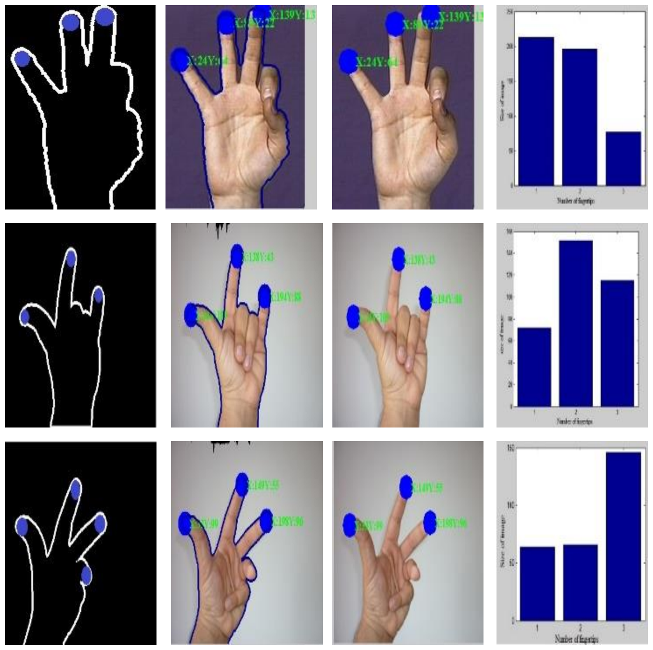
Table 2. Performance evaluation using standard metrics

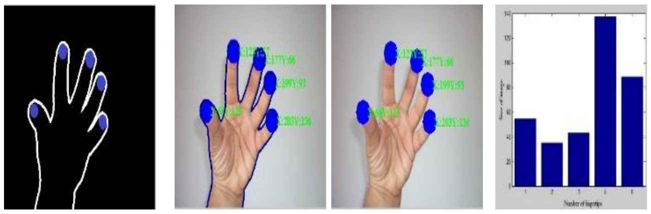
	Precision	Recall	F-Measure
Numbers	96.73	98.67	97.57
Alphabets	95.69	97.92	96.66

Table 3. Comparison of performance with other methods

	Precision	Recall	Accuracy
[18]	94.23	93.75	93.75
[19]	93.8	90.1	90
Our method	95.69	97.92	96.48

In this paper, chain code is modified to extract boundaries of the image. Chains can represent the contours of any shape. In this work, the length L is set to 1. The chains represent closed boundaries, thus, all chains are closed. In binary image, the hand contour is traced using boundary tracing algorithm. After the starting point is found, the next neighboring pixels with the same color and connected is searched by the algorithm. In this case, only pixels adjacent to at least one pixel of the other color are accepted as neighbors. This is to prevent the algorithm from deviating into the interior of the hand. In this work, the entire contour is traced in clockwise direction. Whenever the algorithm reaches its starting point, the contour tracing stops. Our proposed method is able to calculate the perimeter and dimension of a hand from the chain code efficiently. In addition, it also helps in reducing the noise along the contour as well as smoothing of the contour. Figure 13 illustrates the representation of the contour based on the chain code.





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

Figure 13. Hand boundary tracing in binary image; (a) Edge Detection. (b) Boundary tracing, (c) Fingertips detection, and (d) The corresponding histogram for different fingertips frames gestures.

5. CONCLUSION

This paper presents fingerspelling recognition method based on boundary tracing algorithm and modified chain code. A novel contour-tracing algorithm based on pixel is proposed for effective fingerspelling recognition. The related features obtained from the boundary tracing step, which include the chain code, shape number and chain histogram, combine with the number of detected fingertips and distance between fingertips provides better accuracy for recognition.

6. ACKNOWLEDGMENTS

The researchers wish to thank Universiti Kebangsaan Malaysia (UKM) for their support on this work via research grants DIP-2016-018.

7. REFERENCES

[1] Issa, T., and Isaias, P. 2015. Usability and Human Computer Interaction (HCI). In *Sustainable Design*. Springer London. 19-36.

[2] Kane, L., and Khanna, P. 2015. A framework for live and cross platform fingerspelling recognition using modified shape matrix variants on depth silhouettes. *Computer Vision and Image Understanding*, Vol.141, 138-151. DOI= https://doi.org/10.1016/j.cviu.2015.08.001.

[3] Yang, C., Feinen, C., Tiebe, O., Shirahama, K., and Grzegorzek, M. 2016. Shape-based object matching using interesting points and high-order graphs. *Pattern Recognition Letters*, Vol.83, 251-260. DOI= https://doi.org/10.1016/j.patrec.2016.03.013.

[4] Gong, D., and Liu, K. 2018. A multi-objective optimization model and its evolution-based solutions for the fingertip localization problem. *Pattern Recognition*, Vol.74, 385-405. DIO=https://doi.org/10.1016/j.patcog.2017.09.001.

[5] Pisharady, P. K., and Saerbeck, M. 2015. Recent methods and databases in vision-based hand gesture recognition: A review. *Computer Vision and Image Understanding*, Vol.141, 152-165. DIO= https://doi.org/10.1016/j.cviu.2015.08.004

[6] Nagarajan, S., and Subashini, T. S. 2013. Static hand gesture recognition for sign language alphabets using edge oriented histogram and multi class SVM. *International Journal of Computer Applications*, *82*(4). 28–35.

[7] Pansare, J. R., Gawande, S. H., and Ingle, M. 2012. Real-time static hand gesture recognition for American Sign Language (ASL) in complex background. *Journal of Signal and Information Processing*, *3*(03), 364- 367. DOI= http://dx.doi.org/10.4236/jsip.2012.33047.

[8] AtiqurRahman, M. D., Ahsan-Ul-Ambia, A., and Aktaruzzaman, M. D. 2011. Recognition of Static Hand Gestures of Alphabet in ASL. *IJCIT*, *2*(1), 1-4.

[9] Rajesh Mapari, and Dr.GovindKharat. 2012. Hand Gesture Recognition using Neural Network, *International Journal of Computer Science and Network*, 1(6), December.

[10] Shaik, K. B., Ganesan, P., Kalist, V., Sathish, B. S., and Jenitha, J. M. M. 2015. Comparative study of skin color detection and segmentation in HSV and YCbCr color space. *Procedia Computer Science*, Vol.57, 41-48. DOI= https://doi.org/10.1016/j.procs.2015.07.362.

[11] Rioux-Maldague, L., and Giguere, P. 2014. Sign language fingerspelling classification from depth and color images using a deep belief network. *Canadian Conference on Computer and Robot Vision (CRV)*, 92-97. IEEE. DIO= 10.1109/CRV.2014.20.

[12] Sun, Leiming, and Tianshun Huang. 2013. A new boundary tracing algorithm of the contour of objects in the binary image. Computer Modelling and New Technologies 17(5A), 63-67.

[13] Cheng, H., Dai, Z., Liu, Z. and Zhao, Y. 2016. An image-to-class dynamic time warping approach for both 3D static and trajectory hand gesture recognition. *Pattern Recognition*, 55, 137-147. DOI= https://doi.org/10.1016/j.patcog.2016.01.011.

[14] Dawod, A. Y., Abdullah, J. and Alam., M. J. 2010. A new method for hand segmentation using free-form skin color model, 3rd international conference on Advanced computer theory and engineering (ICACTE), V2-562-V2-566. DOI: 10.1109/ICACTE.2010.5579466

[15] Dawod, A. Y., Nordin, M. J. and Abdullah, J. 2015. Static hand gestures: Fingertips detection based on segmented images. *Journal of Computer Science*, 11(12), 1090-1098.

[16] Fating, K., and Ghotkar, A. 2014. Performance analysis of chain code descriptor for hand shape classification. *International Journal of Computer Graphics & Animation*, *4*(2), 9-19.

[17] Liu, Y. K., Žalik, B., Wang, P. J., and Podgorelec, D. 2012. Directional difference chain codes with quasi-lossless compression and run-length encoding. *Signal Processing: Image Communication*, *27*(9), 973-984. DOI= https://doi.org/10.1016/j.image.2012.07.008.

[18] Nagarajan, S., and Subashini, T. S. 2013. Static hand gesture recognition for sign language alphabets using edge oriented histogram and multi class SVM. *International Journal of Computer Applications*, 82(4). 28-35.

[19] Kwolek, B., and Sako, S. 2017. Learning Siamese Features for Finger Spelling Recognition. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, Vol.10617, 225-236.