Symbol Recognition using Particle Swarm Algorithms

Maha El Meseery, Mahmoud Fakhr El Din, Samyah Mashaly,
Magdah Fayek, Nevin Darwish

Abstract

Sketch recognition is defined as the process of identifying the symbols that the user draws using a single or multiple strokes. Users draw strokes using a pen and the system immediately interprets their strokes into objects that can be easily manipulated. This paper uses Particle Swarm Algorithm (PSO) to divide the strokes the user draws into meaningful geometric primitives. These geometric primitives are grouped to formulate symbols which will be further identified. The results showed that using PSO improved segmentation results which guide the symbol recognition phase. The system used SVM classifier which further improved the final recognition accuracy.

1. Introduction

Scientists generally and engineers specifically express thoughts and designs using sketches. Engineers use sketches to exchange designs as a natural method of communication rather than writing or speaking. Currently, engineers use paper and pencil design in early stages of design. Design engineers need a powerful computer based system with symbol recognition to help them design, manipulate and store sketches more effectively than with only papers. Increasing the interaction between the computer and the users in sketch and CAD systems has been recently the reason for the emerging of few advanced sketch recognition systems. Sketch recognition system converts the drawings and informal graphs the user draws into objects and symbols can be used in further interaction with the system.

Sketch recognition researchers divide the system into four main steps preprocessing, segmentation, symbol recognition and finally sketch understanding. The preprocessing process saves the location of points from the hardware and collects basic information about the stroke data then proceed to remove noise and primary data calculation. In segmentation step, the system divide the stroke into a set of simple geometrical primitives or segments. The third step of the system is clustering strokes and segments to formulate symbols that can be recognized by a classifier system. Finally, the sketch understanding step uses priori information about the sketch to support identification of symbols by excluding any possible invalid symbols.

Section 2 provides a brief literature review. The remaining of this papers will be as following: Section 3 will explain the general particle swarm algorithm. The system block diagram and system details will be presented in Section 4. The experiments and results achieved are explained in Section 5. Finally, section 6 will state the conclusion and future work.

2. Literature Review

The field of symbol recognition has gained interest in the last few years. A wide variety of techniques were used either on segmentation process or symbol recognition. A hybrid algorithm was introduced in [Sezgin *et al.*, 2001] where they generated different sets of segmentation based on both curvature and speed dominant points then followed by choosing a segmentation with the least error from the generated hybrid set. The system has a draw back as it only recognizes simple specific shapes with a set of low level recognizers. [Yu, 2003] introduced a feature area for each primitive and then computed the segmentation error for different types of primitives based on the computed feature area. The system achieved good accuracy in simple shapes (square, ellipse,...) but did not perform well in more complex shapes. Paulson and Hammond [Paulson and Hammond, 2008] introduced a set of low level recognizers that was reported to achieve 98% accuracy but there system like all low level recognizers only identify a small set of simple shapes.

Genetic algorithm was also used by [Chen *et al.*, 2003] to optimally divide digital curves into lines and curves. Chen *et al.* used digital curves scanned from paper as input to the system and did not take advantage of the curvature or local geometric properties of the digital curve. Yin used PSO to convert digital curves

into polygons[Yin, 2004], this paper adopted their system but improved it by adding curvature and other local information while segmenting strokes to achieve better segmentations.

Many different approaches have been investigated to achieve final recognition of symbols. Graph searching and template matching where used in [Lee *et al.*, 2007] where a graph of the drawn symbol is generated and matched with a stored set of graphs templates to reach a classification decision. These systems may generate better performance but they are very computationally expensive. Geometrical and spatial descriptive language was used in [Alvarado and Davis, 2007; Hammond and Davis, 2003] where an architecture similar to compiler was used to recognize symbols. SVM and HMM classification was implemented in[Peng *et al.*, 2004; Sezgin and Davis, 2005] to correctly classify symbols.

3. Particle Swarm Algorithm

The main idea of Particle Swarm Algorithm (PSO) is to represent each agent with a particle from the solution space. Each agent moves the particle with a direction and velocity based on equation [1].

$$v_{ij} = v_{ij} + c_1 r_1 (lbest_{ij} - p_{ij}) + c_2 r_2 (gbest_{ij} - p_{ij}) \quad (1)$$

$$p_{ij} = p_{ij} + v_{ij}, \quad (2)$$

Equation [1] shows how velocity and direction of each particle are computed where r_1 & r_2 are random variables and c_1 & c_2 are the swarm system variables. Each iteration the global best g_{best} particle and the agent local best l_{best} particle are evaluated based on the maximum fitness functions of all particles. The solution is found after a maximum number of iteration or after an error threshold is achieved.

$$P(i) \Leftarrow \{ \begin{array}{lll} 1 & if & r_3 > t \\ 0 & if & r_3 < t \end{array} \} \quad (3)$$

Equation 3 where t is a threshold and r_3 is a random variable, is used to change the general swarm algorithm into binary particle which handles particle values of either 0 or 1.

4. System Description

Figure. 1 shows that the main system blocks are 1) preprocessing, 2) segmentation, 3) clustering, 4) feature extraction, and finally 5) classification.

The preprocessing is responsible of capturing the input data and removing the noise from it. The system then proceeds to estimate a set of possible dominant points to help in the segmentation process.

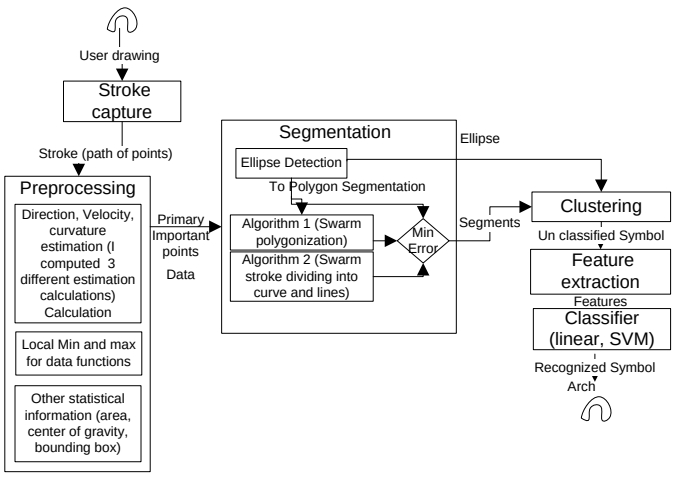


Figure 1. The system Block Diagram.
The figure shows the main system blocks. Preprocessing, segmentation, clustering, features extraction and classification.

The segmentation is divided into two steps ellipse fitting and curve segmentation. In the first step, the ellipse fitting process tries to fit the stroke into an ellipse. If the system fails it passes the stroke to the second step; curve segmentation which consist of two PSO segmentation algorithms. The two algorithms will generate two segmentations, the system will choose the segmentation that has the minimum segmentation error.

The clustering algorithms starts to group segments together after the segmentation step. The system let the user draws the symbol by using any number of strokes, a set of unrecognized segments is passed to the clustering algorithm to generate a symbol and compute a feature vector for it. The system uses a hybrid set of statistical, geometric and spatial features. The final step is a SVM classifier that will use the features computed to classify the segments into one of the previously trained classes.

4.1. Preprocessing

It is noted that as the user draws a shape the pen will slow down near corners and picks up speed when drawing a straight lines. Therefore, the speed and time difference data were widely used in sketch understanding systems [Sezgin *et al.*, 2001]. The direction and curvature data are used to determine the points with high angular changes along the path of points [Yu, 2003].

In the presented system, we computed time difference, direction, speed and curvature of each point along the stroke. The following equation $v = \Delta t / \Delta s$ is used

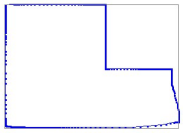


Figure 2. Stroke Sample
An example of the stroke drawn in the system.

for speed calculation where t is the time difference between two points and s is the length between them. The direction is calculated as the angle between two vectors and curvature is the change in direction with respect to length i.e. $c = \Delta d/\Delta s$.

All these calculations are performed in real time while the user draws the strokes. The complexity of computation is $O(n)$ where n is number of points.
Figure 3 shows the computed data for the stroke drawn in fig. 2. If you look at the data curves you will see that dominant points are characterized by lower speed values and higher curvature and direction values.
After the system computes all the speed, time difference and curvature data it proceeds to detect the points with low velocity and high curvatures. Using simple differentiations to detect local extreme points resulted in false points due to the non smooth curves. Hence, the system adopted a process presented by [Sezgin *et al.*, 2001], where the mean of the curve is calculated and used as threshold. This threshold is used to separate the curve into regions; each region is defined as the part between two intersection points of the threshold line and the curve. Regions higher than the threshold value are processed to find the maximum of each region. The points that correspond to those maximum values are labeled as *possible dominant points*.
The system repeat this process to curvature, time difference and speed curves. All the points labeled as possible dominant points are saved into a single array. They are used as input to guide the next stage of segmentation.

4.2. Segmentation

After computing the primary data the system tries to segment the stroke into a set of primitives. The segmentation algorithm first tries to detect if the stroke is an ellipse. If the stroke proved to be an ellipse then the segmentation process ends and the system proceeds to the next step. Otherwise, the stroke is passed into two particle swarm algorithms that will divide the stroke to either lines or lines and curves. The algorithms takes the stroke points along with the possible dominant points computed then produce a set of dominant points which are connected with either lines or curves (see fig. 1). The next section will describe the ellipse detection algorithm and both the two particle swarm algorithms used to divide the stroke.

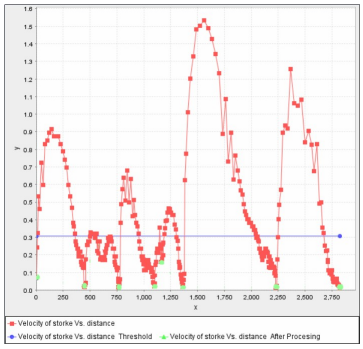
4.2.1. Ellipse Detection. The process starts with computing the center of the stroke bounding box. The bounding box center point is used as the first estimation of the ellipse center. The axes of the ellipse are estimated as $width/2$ and $height/2$ of the stroke bounding box. The least square fitting algorithm is used to minimize the fitting error of the ellipse equation (see 4 where N is number of points in the stroke, a,b are the length of ellipse axes, x_0 & y_0 are the coordinates of the center point, x_i & y_i are the coordinates of point i in the stroke). After only few loops the error and a confidence values are computed to check if the stroke can be labeled as ellipse or not.

$$E = \sum_{i=0}^{N} \frac{(x_i - x_0)^2}{a^2} + \frac{(y_i - y_0)^2}{b^2} - 1 \qquad (4)$$

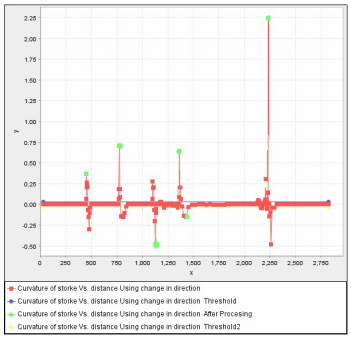
4.2.2. Discrete particle swarm algorithm. There are two PSO algorithm described in this section. The system generates segmentation from both algorithms and then chooses the segmentation with the minimum error value. The problem definition is the same in both algorithms but they differ in the method they compute fitness and error functions.

4.2.3. Problem definition. The input stroke with N points can be represented by set $S = \{x_1, x_2 \ldots x_N\}$ where x_i is the location of the point i . The swarm algorithms consist of M agents which are represented by the set $A = \{P_i \,|\, i = 1, 2 \cdots M\}$ where P_i is a single solution particle from the solution space. Each particle decodes the problem with binary array with the same length N as the input stroke. So the system will represent each particle P_i by $P_i = \{p_{ij} \,|\, j = 1, 2 \cdots N\}$ where p_{ij} will have only two values 0 or 1 where $p_{ij} = 1$ will mean that point j is a dominant point.

4.2.4. The algorithm. The fitness function and error calculation are different in each algorithm. For the first algorithm, let's define the arc $\widehat{x_i x_j}$ as the consecutive set of points from pointx_i to point x_j as in $x_i, x_{i+1} \cdots, x_j$. The line $\overline{x_i x_j}$ as the straight line connecting point x_i to point x_j. The approximation error is computed by the equation 5 where M is the number of dominant points in this solution. The error $e(\widehat{x_i x_j}, \overline{x_i x_j})$ is computed as the sum of squared perpendicular distance from every point along the arc $\widehat{x_i x_j}$ to the line $\overline{x_i x_j}$.



(a) The Speed of data



(b) The Curvature

Figure 3. Data Curves

The curve in a) shows the speed of the points in stroke in fig. 2. b) Shows the curvature curve for the same stroke.

$$E = \sum_{i=0}^{M} e(\widehat{x_i x_{i+1}}, \overline{x_i x_{i+1}}) \tag{5}$$

$$\max fitness(p_i) = \begin{cases} -E/\varepsilon N & if E > \varepsilon, \\ D/\sum_{j=1}^{N} p_{ij} & otherwise \end{cases} \tag{6}$$

The fitness is computed using equation 6 where N is the number of points in the stroke, D is the number of point in the solution that was previously labeled as a possible dominant point and E is the computed error and ε is the error threshold. Notice that when the error is larger than the threshold the fitness is given a -ve value to lower the value of solution otherwise the system will favor the lower number of vertices.

The second algorithm has the same problem formulation but different fitness and error functions. It was previously introduced in [Chen *et al.*, 2003] but a genetic programming was used as the optimizing algorithm. The error of both circle and line estimation are computed for each segment, the approximation with the lower error value will be the chosen approximation of this segment. The sum of the approximation error of all segments is the total error of the particle. The error is computed by equation (7) where M is the number of segments in the solution, D_i is the minimum approximation error of curve and line approximations $min(d_c, d_l)$ as computed by [Chen *et al.*, 2003]. The fitness is computed by the equation (8) where E is the error and M is number of segments and k is a parameter tweaked to get minimum number of segments.

$$E = \sum_{i=0}^{M} e(D_i) \tag{7}$$

$$\max fitness(P_i) = \frac{1}{E \times M^k} \tag{8}$$

Yin [Yin, 2004] used a merge and divide algorithm after each loop of the swarm system to refine the solution but this system is introducing another enhancement method. After each loop of the swarm algorithm, each particle is refined using the following procedure. For each particle each dominant point is checked to find if it was labeled before as a *possible dominant point* (computed as in section 4.1), if it is not labeled the point is moved to the nearest labeled point. After that the particles are tested to make sure that the distance between every two successive dominate point is larger than the constant min_D. If two points are found that they are nearer than min_D then based on the decrease in the error of the solution one of the points will either be removed or moved to the nearest possible dominate point.

4.3. Recognition

After the user draws all strokes of the symbol he has to wait 10 seconds or press finish button beside the drawing area. The set of unrecognized strokes is grouped together along with their segmentation as input to the feature extraction process. The system uses a composite set of features include Rubine feature set [Rubine, 1991], Zernike moments [Hse and Newton, 2005], ink density [Gennari *et al.*, 2005] and some structural and spatial information like number of perpendiculars lines , number of parallel lines and types of primitives in each symbol. After computing the features the symbol is introduced to the classifier. The system used a support vector machine (SVM) classifier with Gaussian kernel (RBF kernel) [Chang and Lin, 2001].

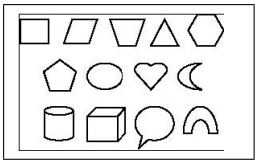


Figure 4. The Symbol Set
Ths figure shows the some of the symbols used in the dataset.

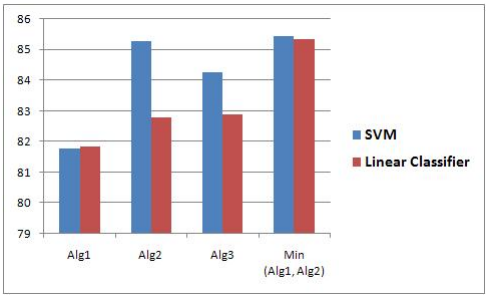


Figure 5. Algorithm comparison
The results of comparing the algorithms. The graphs displays the recognition rate of symbols.

5. Experiments

For testing the system used a data set collected by Hse and Newton. The data are drawn by 16 users each of them have drawn from 30 to 50 samples for each shape. Figure 4 shows the shapes used in the data set. We divided the dataset equally into training set and test set. The results displayed are the recognition accuracy resulted from the classifier after recognizing the symbols.

We performed two experiments to test the system firstly we tested recognition accuracy of shapes in the data set with both algorithms. We also implemented the segmentation algorithm described in [Sezgin *et al.*, 2001] to use as reference to our swarm algorithms. As you see in the fig. 5 shows the accuracy achieved by each algorithm. The results shows that both PSO algorithm achieve better result than other algorithms. The swarm algorithms were tested with and without the ellipse detection module. The ellipse detection module appear to be superior to results with only the PSO algorithms.

The second experiment we implemented was to test the effect of symbol complexity and type on the recognition rate. Figure 6 shows what the accuracy of each symbol, it clearly noted that symbols that have only line segments achieve higher accuracy rate than other symbols. Figure 6 also shows that algorithm 1 alone achieve better performance than algorithm 2 in the symbols that consist of lines only. This is natural as the algorithm divides strokes into line segments only but algorithm 2 can divide strokes into lines and curves based on the minimum error of the segment itself. Algorithms 3 can give good performance as long as the symbols consist of lines and curves, if the stroke consist of curve only the algorithm may lead to wrong segmentation result. This is because the system divides the stroke first to line segments then tries to decide if each segment can be represent better as a curve unlike algorithm 2 where the curve segments are tested while choosing the best segmentation. Combining both algorithm 1 and algorithm 2 improved the recognition rate of all symbols as the system now have the advantages of both algorithms. We could not test the result of the segmentation algorithm directly due to the fact that the correct segmentation is highly ambiguous. It is only based on what the user intended to draw while sketching the symbol. Figure 7 shows a sample of the output of the segmentation block.

6. Conclusion and Future Work

This paper presented a new approach to sketch recognition using PSO. The system used both speed and curvature data which helped in improving the PSO algorithms over the original algorithms [Chen *et al.*, 2003; Yin, 2004]. It was noted that the PSO in general generated an optimized stroke segmentation which improved the final recognition rate. The trade off between accuracy achieved and time complexity must be further investigated to achieve better results. The use of statistical moments and structural features improved the recognition rate. The system was tested on 13 different symbol and achieved satisfying results. The system dose not depend on low level recognizer but rather on a set of high level features which make the system easily expandable with aspect to symbols recognized.
The next step in this research is to complete the clustering algorithm for fully automated sketch recognition. The clustering must be performed without the user explicit involvement. Other area of modification will the features extraction methods. Introducing more spatial and geometrical features will improve classifications.

References

[Alvarado and Davis, 2007] Christine Alvarado and Randall Davis. Sketchread: a multi-domain sketch recognition engine. In *SIGGRAPH '07: ACM SIGGRAPH 2007 courses*, page 34. ACM Press, 2007.

[Chang and Lin, 2001] Chih-Chung Chang and Chih-Jen

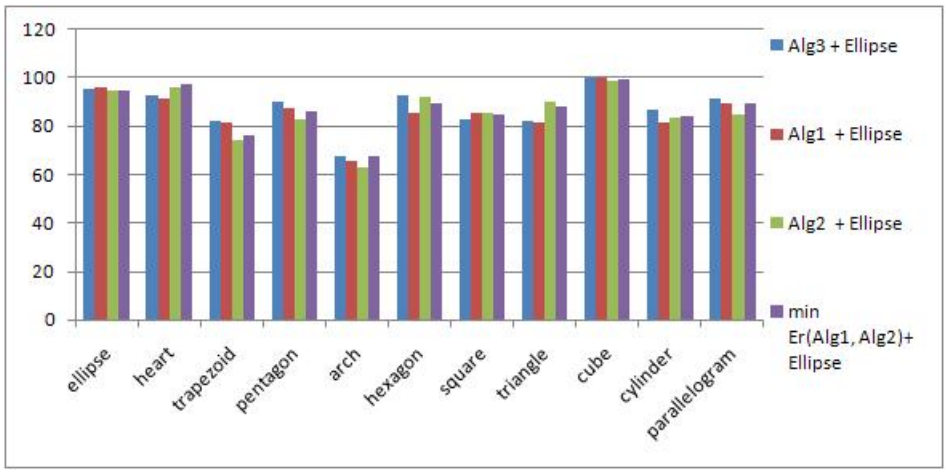


Figure 6. Symbols Comparison
The graph shows the recognition rate of each symbol using different algorithms.

Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[Chen *et al.*, 2003] Ke-Zhang Chen, Xi-Wen Zhang, Zong-Ying Ou, and Xin-An Feng. Recognition of digital curves scanned from paper drawings using genetic algorithms. *Pattern Recognition*, 36(1):123–130, 2003.

[Gennari *et al.*, 2005] Leslie Gennari, Levent Burak Kara, Thomas F. Stahovich, and Kenji Shimada. Combining geometry and domain knowledge to interpret hand-drawn diagrams. 29(4):547–562, 2005.

[Hammond and Davis, 2003] Tracy Hammond and Randall Davis. LADDER: A language to describe drawing, display, and editing in sketch recognition. *Proceedings of the 2003 Internaltional Joint Conference on Artificial Intelligence (IJCAI)*, 2003.

[Hse and Newton, 2005] Heloise Hwawen Hse and A. Richard Newton. Recognition and beautification of multi-stroke symbols in digital ink. 29(4):533–546, 2005.

[Lee *et al.*, 2007] WeeSan Lee, Levent Burak Kara, and Thomas F. Stahovich. An efficient graph-based recognizer for hand-drawn symbols. *Comput. Graph.*, 31(4):554–567, 2007.

[Paulson and Hammond, 2008] Brandon Paulson and Tracy Hammond. Paleosketch: accurate primitive sketch recognition and beautification. In *IUI '08: Proceedings of the 13th international conference on Intelligent user interfaces*, pages 1–10, 2008.

[Peng *et al.*, 2004] Binbin Peng, Wenyin Liu, Yin Liu, Guanglin Huang, Zhengxing Sun, and Xiangyu Jin. An svm-based incremental learning algorithm for user adaptation of sketch recognition. *IJPRAI*, 18(8):1529–1550, 2004.

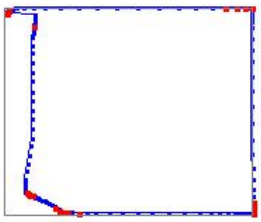
[Rubine, 1991] Dean Rubine. Specifying gestures by example. In *SIGGRAPH '91: Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, pages 329–337, New York, NY, USA, 1991. ACM Press.

[Sezgin and Davis, 2005] Tevfik Metin Sezgin and Randall Davis. Hmm-based efficient sketch recognition. In *IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces*, pages 281–283, New York, NY, USA, 2005. ACM Press.

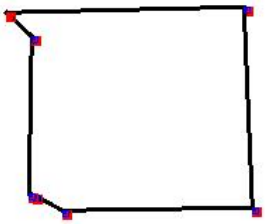
[Sezgin *et al.*, 2001] Tevfik Metin Sezgin, Thomas Stahovich, and Randall Davis. Sketch based interfaces: early processing for sketch understanding. In *PUI '01: Proceedings of the 2001 workshop on Perceptive user interfaces*, pages 1–8, New York, NY, USA, 2001. ACM Press.

[Yin, 2004] Peng-Yeng Yin. A discrete particle swarm algorithm for optimal polygonal approximation of digital curves. *J. Visual Communication and Image Representation*, 15(2):241–260, 2004.

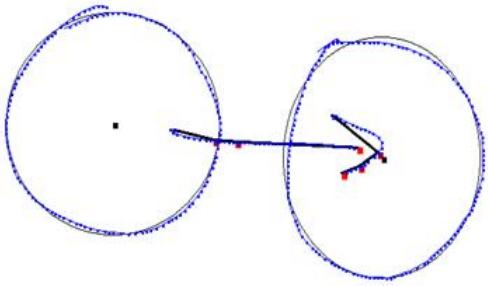
[Yu, 2003] Bo Yu. Recognition of freehand sketches using mean shift. In *IUI 03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 204–210, New York, NY, USA, 2003. ACM Press.



(a) User strokes



(b) Segmented strokes



(c) Multiple symbols

Figure 7. Segmentation outputs
The figure shows sample of the output of the segmentation phase. Figure a) shows the stroke after the user draws it. The red points are points labeled as *possible dominate points*. Figure b) shows the symbol after being segmented. Figure c) shows the segmentation of different and overlabed strokes. Notice the system mark the center of the ellipse that was identified.