Symbol Recognition using Particle Swarm Algorithms

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

Abstract

Sketch recognition is defined by the process of identifying the symbols 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 strokes a 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 improves segmentation results which guide the symbol recognition phase. Final recognition have improved after the PSO algorithm using a SVM classifier.

1. Introduction

Scientists in general 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 symbol recognition and manipulation system to help them operate as freely as with papers. Increasing the interaction between the computer and the users in sketch and CAD systems was 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.
Most researchers in sketch recognition divided 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. After removing noise and calculating primary data the system proceeds to divide the stroke into a set of simple geometrical primitives. The second step of the system is clustering strokes and segments to formulate symbols that can be recognized by the system. Finally, priori information about the sketch is used to support identification of symbols by excluding any possible invalid symbols.
The next section will relate previous work in the field. The remaining of the paper will be as following; Section 2.1 will explain the general particle swarm algorithm. Section 3 will describe the system block diagram and then section 3.1 will explain the preprocessing step, section 3.2 will describe the segmentation algorithm in details. Finally, section 4 will describe the experiments that was preformed.

2. Related Work

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.
To segment strokes, [Yu, 2003] introduced a feature area for each primitive and then compute the segmentation error for different types of primitives based on this area. Complex and hybrid algorithm were also introduced as in [Sezgin *et al.*, 2001] where they generated different sets of segmentation based on both curvature and speed dominant points then they choose the segmentation with the least error from the hybrid set. A dynamic programming algorithm was also used to obtain optimum segmentation of the input strokes in [Gennari *et al.*, 2005]. Genetic algorithm was also used by [Chen *et al.*, 2003] to optimally divide digital curves into lines and curves. [Yin, 2004] used PSO to convert digital curves into polygons.

Many different approaches have been investigated to achieve final recognition of symbols. Graph searching and template matching where used in [Calhoun *et al.*, 2002; Lee *et al.*, 2007], where a graph of the drawn symbol is generated and matched to stored set of graphs templates to reach a classification. Geometrical and spatial descriptive language was used in [Alvarado and Davis, 2007; Hammond and Davis, 2003] where a parser like architecture was used to recognize symbols . Kara and Stahovich used trainable image template to classify different set of symbols [Kara and Stahovich, 2005] . The use of SVM and HMM classification was implemented in [Peng *et al.*, 2004; LiuWenyin1 *et al.*, 2001] and [Sezgin and Davis, 2005].

2.1. Particle Swarm Algorithm

In the Particle Swarm Algorithm (PSO) each agent represents a particle from the solution space. The agents move the particle with direction and velocity based on the general swarm equation [1]. Equation [1] shows how velocity and direction of each particle is computed where r_1 & r_2 are random variable 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. Equation 3 where t is a threshold and r_3 is a random variable, is used to change general swarm algorithm into binary particle which will handle particle values of either 0 or 1. .

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

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

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

3. System Description

As shown in fig. 1 the system is divided into four main blocks. Preprocessing, segmentation, clustering, feature extraction and finally classification.
The preprocessing block 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.
The segmentation process is divided into two steps ellipse fitting and curve segmentation. In the first step, the system tries to fit the stroke into an ellipse. If the system fails it passes the stroke into two PSO segmentation algorithms. The two algorithms will generate two segmentations, the system will choose the segmentation that has the minimum segmentation error.

After the user finish sketching the symbol by using any number of strokes, the set of unrecognized segments is passed to the feature extraction and classifier.

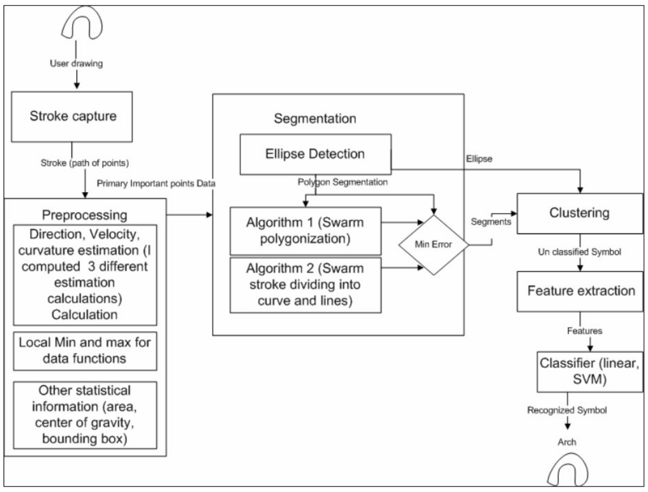
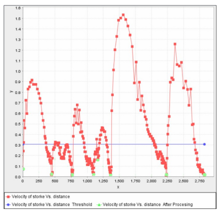

User drawing
Stroke capture
Stroke (path of points)
Primary Important points Data
Preprocessing
Direction, Velocity, curvature estimation (I computed 3 different estimation calculations) Calculation
Local Min and max for data functions
Other statistical information (area, center of gravity, bounding box)
Segmentation
Ellipse Detection
Ellipse
Polygon Segmentation
Algorithm 1 (Swarm polygonization)
Algorithm 2 (Swarm stroke dividing into curve and lines)
Min Error
Segments
Clustering
Un classified Symbol
Feature extraction
Features
Classifier (linear, SVM)
Recognized Symbol
Arch

Figure 1. The system Block Diagram

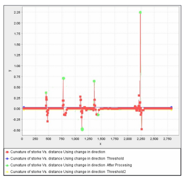
3.1. Preprocessing

It was noted that as the user draws a shape the pen will slow down near corners and will pick 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 decide the points with high angular changes along the path of points[Yu, 2003].

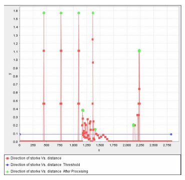
In the presented system, we computed time difference, direction, speed and curvature of each point along the stroke. For speed calculation $v = \Delta t / \Delta s$ where t is the time difference between two points and s is the length between them. The direction is calculated as the angle between two vectors and curvature is the change in direction with respect to length i.e. $c = \Delta d / \Delta s$.
All the calculations are performed in real time while the user draws his strokes. So the resultant computation complexity of computing of all these 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. So instead the following procedure was used, firstly the mean of the curve is used as threshold for lower values then the high values are further investigated to find the extreme points.



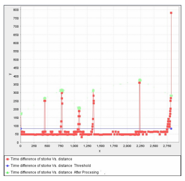
(a) The Speed of data



(c) Curvature



(b) Direction



(d) Time Difference

Figure 3.

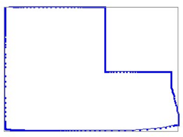


Figure 2. Stroke Sample

Hence, the mean of the direction 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. Only regions higher than the threshold value are processed when computing the maximum of each region. The point that corresponds to those maximum values are labeled 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 and guiding data for the next stage of segmentation.

3.2. Segmentation

After computing the primary data the system now 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.

3.2.1. 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 center of the ellipse. The axes of the ellipse are estimated as the w/2 and h/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 storke can be labled as ellipse or not.

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

3.2.2. 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.

3.2.3. 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 $P_i = \left\{p_{ij} \,|\, j = 1,2\cdots N\right\}$ where p_{ij} will have only two values 0 or 1 where $p_{ij} = 1$ will mean that point j is a dominant point.

3.2.4. The algorithm. The fitness function and error calculation are different in each algorithm. For the first algorithm, let's define the arc $\widehat{x_ix_j}$ as the consecutive set of points from point $x_i, x_{i+1}\cdots, x_j$. The line $\overline{x_ix_j}$ 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 $e(\widehat{x_ix_j}, \overline{x_ix_j})$ is computed as the sum of squared perpendicular distance from every point along the arc $\widehat{x_ix_j}$ to the line $\overline{x_ix_j}$.

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

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

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. As you can see when the error is larger than the threshold 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] where but 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.

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

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

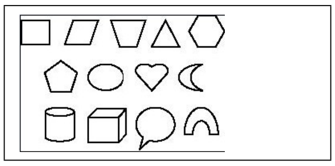


Figure 4. The Symbol Set

[Yin, 2004] used a merge and divide algorithm after each loop of the swarm system to refine the solution but we used another enhancement method. After each loop in the swarm algorithm, each particle loops on the set of selected dominant points to enhance the solution. Each dominant point is checked to find if it was labeled before as a possible dominant (computed as in section 3.1), if not the point is moved to the nearest labeled point.

3.3. Recognition

After the user draw 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 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].

4. 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 three 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

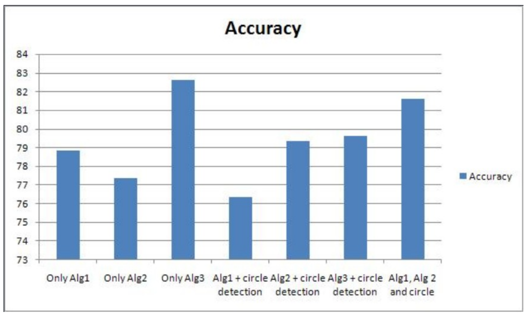


Figure 5. The results of first experiment

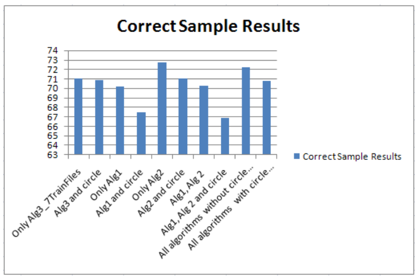


Figure 7. The results of third experiment

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. The combining of both algorithm improves the recognition rate of other symbols.
The final experiment we performed was to test the efficiency svm classifier. We implemented the linear discriminator classifier described in [Rubine, 1991](see figure 7).

5. Conclusion

This paper presented a new approach to sketch recognition using PSO. It was noted that the PSO in general improve the accuracy of the final symbol recognition in the system. The use of both speed and curvature data helps in improvement of the PSO algorithm over the original algorithms [Chen *et al.*, 2003; Yin, 2004]. The tradeoff between accuracy achieved and time complexity must be further investigated to achieve better results.

6. Future Work

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 can be the features extraction and classifier. Adding more spatial 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, New York, NY, USA, 2007. ACM Press.

[Calhoun *et al.*, 2002] Chris Calhoun, Levent Burak Kara, Thomas F. Stahovich, and Tolga Kurtoglu. Recognizing multi-stroke symbols, June 11 2002.

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

[Kara and Stahovich, 2005] Levent Burak Kara and Thomas F. Stahovich. An image-based, trainable symbol recognizer for hand-drawn sketches. *Computers & Graphics*, 29(4):501–517, 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.

[LiuWenyin1 *et al.*, 2001] LiuWenyin1, Wenjie Qian, Rong Xiao, and Xiangyu Jin2. Smart sketchpad - an on-line graphics recognition system. sep 2001.

[Peng *et al.*, 2004] Binbin Peng, Wenyin Liu, Yin Liu, Guanglin Huang, Zhengxing Sun, and Xiangyu Jin. An svm-

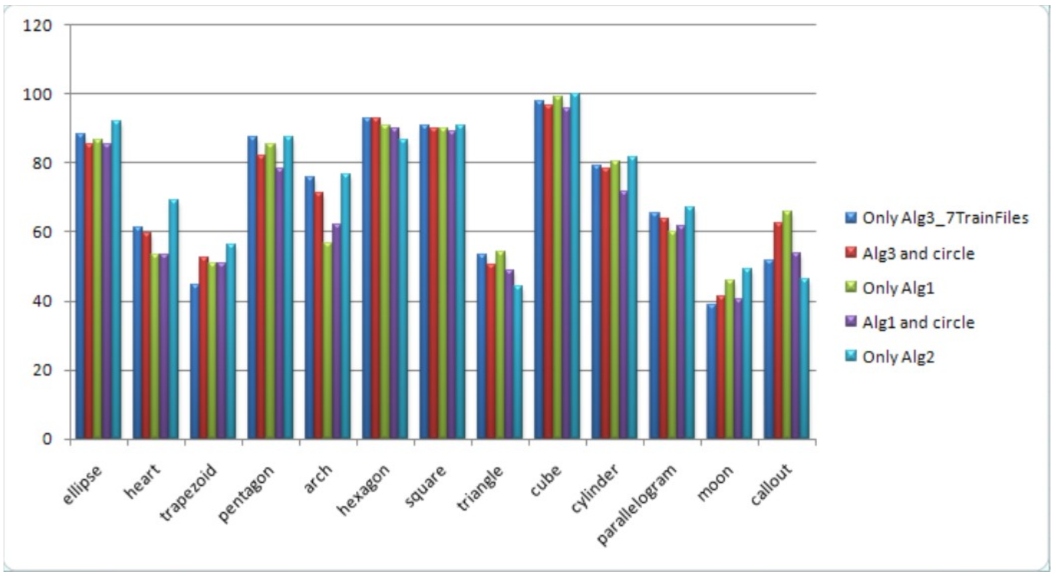


Figure 6. The results of second experiment

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.