A SWARM-BASED SYSTEM FOR OBJECT RECOGNITION

*Tanya Mirzayans, Nitin Parimi, Patrick Pilarski, Chris Backhouse, Loren Wyard-Scott, Petr Musilek**

Abstract: Swarm intelligence is an emerging field with wide-reaching application opportunities in problems of optimization, analysis and machine learning. While swarm systems have proved very effective when applied to a variety of problems, swarm-based methods for computer vision have received little attention. This paper proposes a swarm system capable of extracting and exploiting the geometric properties of objects in images for fast and accurate recognition. In this approach, computational agents move over an image and affix themselves to relevant features, such as edges and corners. The resulting feature profile is then processed by a classification subsystem to categorize the object. The system has been tested with images containing several simple geometric shapes at a variety of noise levels, and evaluated based upon the accuracy of the system's predictions. The swarm system is able to accurately classify shapes even with high image noise levels, proving this approach to object recognition to be robust and reliable.

Key words: *Swarm intelligence, agent, computer vision, object recognition, neural network*

Received: March 24, 2005
Revised and accepted: June 1, 2005

1. Introduction

The behavior of ants and other social insects has inspired the development of artificial distributed problem-solving systems. Each individual member of a society has its own agenda and follows very simple rules; more complex global-level patterns emerge solely through the agents' interactions with each other and their environment, without supervision or central control. There are primarily two types of interactions between insects in a colony: direct and indirect. Direct interactions involve tactile, visual, or chemical contact. Indirect interactions are initiated by individuals that exhibit some behavior that modifies the environment, which in turn stimulates a change in the behavior of other individuals. Although these interactions may be simple, together they can solve difficult problems, such as finding the

*Petr Musilek (Corresponding author)
Department of Electrical and Computer Engineering, University of Alberta, Edmonton, Canada T6G 2V4, Tel: (780) 492–5368, Fax: (780) 492–1811, E-mail: Petr.Musilek@ualberta.ca

shortest path to a food source [5]. This collective behavior of social insects by means of self-organization has been termed *swarm intelligence* [1]. Models of swarm intelligence used to solve various problems have recently been gaining attention in the research community. Swarm approaches offer many problem-solving benefits, including increased flexibility, robustness, decentralization, and self-organization [2].

This paper presents a system that uses artificial agents that model the behavior of social insects to perform feature detection and object recognition tasks. The basic objective is to detect and identify simple objects in images. The ultimate goal of the ongoing research is to develop systems capable of detecting complex objects, such as human facial features or pathological changes in medical images.

The paper is organized as follows. Section 2. provides background information on computer vision and swarm systems. Section 3. describes the proposed swarm-based algorithm for feature detection and object recognition. Results of experiments are summarized in Section 4. Finally, Section 5. brings major conclusions and gives insights on the importance and possible future extension of this work.

2. Background

2.1 Computer Vision

Computer vision is a well-established area with many efficient methods for image processing, including feature detection and object recognition. While these methods provide numerous solutions to the feature extraction problem, they all require computationally complex structures and operations that introduce performance and scalability problems. In addition, most of these methods are indifferent to the underlying meaning and boundaries of analyzed regions, and do little to reduce the complexity of the output data.

A common method for region extraction involves assigning each pixel a level of belonging to a region or group [3]. While this may accurately allow for the characterization of fixed regions, it does not provide insight into the underlying nature of the area such as its general geometry and characteristic features. This is a capability lacking in most current approaches. Through extraction of an object's key features, it is possible to create a simplified representation of the object that is compatible with a suitable classification system. By extracting and utilizing geometric properties of an image, it is possible to reduce the complexity of the image's representation as seen at the input of the classification system. This results in a significant decrease in the computational complexity required to analyze object data, without degrading the accuracy of the results.

The approach proposed in this paper uses a swarm system to simplify a greyscale image into a list of key geometric properties such as the vertex count, the number of aligned line segments, and their relation to the total size of the object. This form of object representation is suitable for analysis by a classification system, such as a rule based system or a trained neural network. It is the aim of this study to create an effective approach that may be further expanded to allow for rapid characterization of complex images based upon their principal spacial trends.

2.2 Swarm Intelligence

Swarm intelligence is a field concerned with creating algorithmic solutions based on the collective self-organizing behavior common to many species of social animals. In swarm systems, each individual, called an *agent*, follows a set of simple rules (encoded in the individual's nature) and reacts to environmental conditions. The combined behavior of these simple individuals gives rise to complex systemic behavior such as high-level problem solving and coordinated tactical action. A characteristic of these systems is that each individual has little or no knowledge of the overall layout of the environment or the high-level system goals. While some systems allow for direct communication between agents, most function solely on indirect interaction where agents take stimulus cues directly from the environment. In this scenario, past actions of a swarm on the environment affect future actions of individuals in the swarm [6]. By altering and responding to a common environment, agents can effectively communicate and complete high-level tasks. Initially proposed by Grassé in 1959 [9], this process is known as *stigmergy* and has been observed in the social functions of ants, termites, and some species of spider [3, 6]. It is possible to combine such stigmergic systems with another type of swarm behavior: *particle swarm optimization* (PSO). Modeled after the flocking behavior of birds and fish, PSO effectively flies a swarm of solutions through a multi-dimensional search space using state memory and inter-agent attraction. It has been shown that swarm systems employing stigmergic and PSO-like attractive behavior can effectively solve a large range of optimization problems, usually outperforming traditional evolutionary computing techniques in terms of both convergence speed and computational cost [7, 12, 16].

In addition to their use in solving traditional optimization problems, swarm systems have recently been applied to region detection and region mapping, including grey-scale feature extraction [3] and biomedical image transformation and registration [15]. Several groups have shown that PSOs can also be successfully applied to autonomous vehicle navigation [4, 8]. More closely related to the presented work, Ramos and Almeida introduce a swarm system involving the evolution of pheromone fields guiding artificial ant colonies to react and adapt to digital habitats [13]. Also, in [11], Liu and Tang propose an autonomous agent-based image segmentation approach. Their primary focus is on the computational aspects of behavior-based reactive agents as an efficient way to search and label specific homogeneous regions of known representations in a given image. Despite these initiatives, it appears that very little other work has been done to apply the power of swarm systems to traditional computer vision tasks. One of the common characteristics of current image processing systems is that they attempt to highlight features to make them more prominent. This has been done by isolating or matching a region of similar pixels based on shape and/or pixel intensity value [3, 15]. In addition to highlighting features, the system presented in this study allows for classification of image features. This is possible because the image being processed is represented as a cloud of data points, which is then analyzed by a classification subsystem.

3. Swarm-based System for Object Recognition

The primary contribution of this work lies in the unique combination of a swarm intelligence subsystem with a traditional classifier subsystem. The proposed swarm algorithm involves the fixing of agents to the edges of an object present in an image. The evolution of the agent population is driven by motion, feature detection, and fixation of the agents. The swarm produces statistics about the types of features present in the image, which are forwarded to the classification subsystem. This subsystem can be realized in many different ways. To demonstrate the feasibility of the swarm approach, two separate classification techniques are used: a fuzzy rule-based system (FRBS), and an artificial neural network (ANN). The overall structure of this evolutionary scheme is outlined in Algorithm 1.

3.1 Agent Motion

Initially, individuals in the swarm are randomly placed in the environment representing the image. In each iteration, every individual goes through two major steps: movement and possible fixing. Movement is governed by three components: attraction, momentum, and randomness. Attraction provides guidance to agent motion based on the assumption of object continuity; the agents are encouraged to explore areas of the environment that are known to contain features of an object. Momentum is necessary for consistent motion, and increases the likelihood that the swarm will sufficiently explore the environment. When momentum is insufficient,

Algorithm 1 Swarm-Based System for Image Recognition

```
1: Read picture
2: Initialize agent position randomly in environment (no overlap permitted)
3: Determine global threshold
4: Initialize momentum vector and iteration counter
5: while there exists a free agent do
6:     Increment iteration counter
7:     for all free agents do
8:         Check for features at current location
9:         if agent decides to fix then
10:            Re-categorize as a fixed agent
11:            Store the detected feature (e.g. horizontal or vertical edge, corner)
12:            Produce clones of the parent
13:        end if
14:    end for
15:    All free agents move (no overlap permitted)
16:    Decrement time-to-live for all free agents
17:    if time-to-live=0 for any free agent then
18:        The agent expires
19:    end if
20: end while
21: Statistics of detected features enter object classification system
```

individuals stay within the region in which they are initially placed, and their exploration capability is greatly reduced. Randomness provides diversity to an agent's motion pattern and also contributes to a more complete exploration of the environment.

Attraction is closely related to the concept of stigmergy described in Section 2. A demonstration of this concept can be found in ant colonies, where ants release and detect pheromones as a method of conveying information. The algorithm has been simplified to shorten the time of each iteration by simulating stigmergy instead of using models of actual pheromones. These models are replaced by attraction to neighboring individuals, particularly those which are fixed at their locations in the environment, so that agents tend to follow each other and gather near interesting regions rather than wander through barren regions.

Motion is controlled by three system parameters: attraction, A, momentum, M, and a base constant, C, providing randomness. A fine balance between these parameters exists and suitable values have to be determined experimentally. When momentum is too high, it masks the other aspects of motion. If attraction between agents is too high, the swarm may not successfully locate all regions of interest in the environment, and the resulting population does not faithfully represent the object present in the image.

As mentioned above, motion of an agent is influenced by its momentum and by attraction to other agents in the neighborhood. Selection of a direction, i, in which to move is not deterministic, but rather stochastic, with the probability of a particular direction to be chosen defined as

$$p_i = \frac{C + a_i + m_i}{nC + \sum\limits_i a_i + M}, \tag{1}$$

where a_i is an attraction function, m_i is a momentum function, and n is the number of directions accessible from the current location. The probability is set to $p_i = 0$ for directions that are not accessible, i.e. where movement would lead outside the image boundary or to a location already occupied by another agent. The denominator performs normalization so that $\sum p_i = 1$. The base constant, C, provides each agent with equal probability to move in any direction before the influence of momentum and attraction is applied. The higher the value of C relative to the attraction and momentum function values, the more stochastic the agent movements become. This constant is kept relatively low in order to take advantage of the benefits provided by momentum and attraction.

The attraction to which the agent k is exposed is related to the distance between k and the neighboring agents, j, within a defined radius. Agents that lie beyond this radius are not considered neighbors to k. The attraction is increased according to the fixation status of the agent j by a constant value, F, such that $f_j = F$ if agent j is fixed, and f_j=0 otherwise.

The form of the attraction function a_i differs depending on the direction, i, being examined. The directions i are indexed starting from $i = 1$ corresponding to the direction up, and continuing clockwise to $i = 8$ corresponding to the direction `up-left`.

The following two equations show how the attraction functions are defined.

$$a_i = \begin{cases} \sum_l 7A - A \mid x_l - x_k \mid - A \mid y_l - y_k \mid + f_l & \text{for } i = 1,3,5,7 \\ \sum_m 4A - 2A \mid (\mid x_m - x_k \mid - \mid y_m - y_k \mid) \mid + & \\ +(2A - \mid x_m - x_k \mid)(2A - \mid y_m - y_k \mid) + f_m & \text{for } i = 2,4,6,8 \end{cases} \quad (2)$$

where set $\{l\}$ contains neighboring agents lying in the top, right, bottom, or left half-planes, set $\{m\}$ contains neighboring agents lying in top-right, bottom-right, bottom-left, or top-left quadrants, and A is a positive constant influencing strength of attraction relative to momentum and randomness. The relationships between the coordinates of an agent, k, and the coordinates of its neighbors, j, considered for each direction are defined in Tab. I. An example of attractions exerted on an agent by its neighbors is shown in Fig. 1.

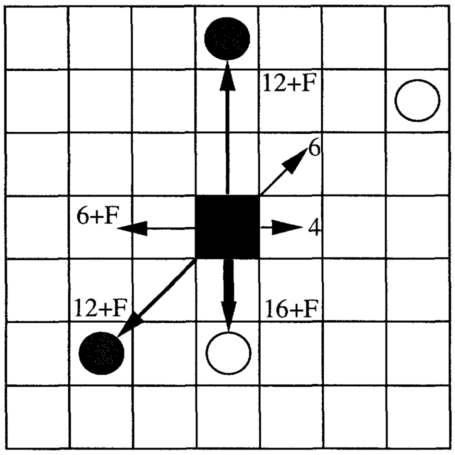

Fig. 1 *An example of attractions exerted on an agent (■). Legend: ○ unfixed neighboring agent, ● fixed neighboring agent, F – fixation constant*

The momentum function m_i evaluates to a positive constant, M, if the agent moved in direction i in the previous iteration, and to zero otherwise, as shown in the following equation

$$m_i = \begin{cases} M & \text{if } i_t = i_{t-1} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

For a rectangular arrangement of pixels, there are $n \leq 8$ possible directions to move from any given position on the image. Probabilities corresponding to these directions obtained using (1) are arranged in a vector

$$D = [p_1\, p_2 \ldots p_8], \quad (4)$$

and a particular direction is then determined using a roulette wheel selection approach.

i	Direction	Range x	Range y
1	`up`	-	$y_j > y_k$
2	`up-right`	$x_j > x_k$	$y_j > y_k$
3	`right`	$x_j > x_k$	-
4	`down-right`	$x_j > x_k$	$y_j < y_k$
5	`down`	-	$y_j < y_k$
6	`down-left`	$x_j < x_k$	$y_j < y_k$
7	`left`	$x_j < x_k$	-
8	`up-left`	$x_j < x_k$	$y_j > y_k$

Tab. I *Relationship between the coordinates of an agent, k, and the coordinates of its neighbors, j, considered for each direction i.*

3.2 Agent Fixation and Characterization

Features in an image are identified using a feature-detection approach. For example, edges in an image can be found by considering amplitude discontinuities that are greater than a certain threshold. Conventional image processing involves techniques such as discrete differentiation, in which the original image is convolved with compass gradient masks in order to reveal image features. The method described in this paper uses similar masks, but in a different manner.

In the current system, the agents examine the 3×3 pixel area centered at their location in the image and apply compass gradient masks. A mask is applied by multiplying its elements with the corresponding pixel values in the image. The sum of products of these multiplications is compared to a pre-determined threshold value and indicates the orientation of an edge, if one exists. For initial experiments, the choice was made to use Sobel masks because they are symmetrical and simplified to detect gradients in two directions [14]. Many other existing sets of masks are more complex, requiring more steps, and therefore increase the time required to run the algorithm.

If a feature is found, the agent is fixed at the location and cloned. In order to speed up the object detection process, the clones are placed strategically according to the detected feature type. For example, when a horizontal edge is found, new agents (clones) are placed to the left and right of the original agent. Each agent is given a time-to-live, so that if it fails to find an object feature in its lifetime, it expires, thus saving computational resources.

3.3 Iteration Control

The status of all agents is evaluated during each iteration in random order so that the entire population is updated asynchronously. At any given time, an active agent must be in one of two possible states: free or fixed. Each free agent moves around the image and attempts to fix to the edges of the image, based on the motion heuristic and the agent fixing technique, detailed in Sections 3.1 and 3.2. Expired agents are removed from the population and no longer have any bearing on the remaining agents. If an agent fixes to a feature, it blocks that location in the image so that no other agent may occupy it. The entire population is continually updated until all active agents are fixed, at which point the swarm has reached a state of equilibrium.

3.4 Shape Classification

After the swarm has reached equilibrium, the statistics about the types of features present are retrieved and forwarded to the object classification subsystem, which has been implemented both as a FRBS and an ANN. The swarm outputs three raw parameters to the classification subsystem: the number of fixed corner agents, N_C, the number of fixed horizontal agents, N_H, and the number of fixed vertical agents, N_V. The inputs into the classification subsystem are different ratios of these parameters. Currently, the systems are able to identify and discern between squares, rectangles, crosses, triangles, and circles. Brief descriptions of the two implementations of the classification subsystem are provided in the following subsections.

3.4.1 Fuzzy Rule-Based System

The FRBS has been selected for the initial implementation of the classification subsystem as it is easy to design, using intuitive rules. The system is designed in the form of a Mamdani fuzzy inference system [10]. It classifies shapes based on three input parameters: the number of fixed corner agents, N_C, the ratio of fixed horizontal agents to fixed vertical agents, $R_{HV} = N_H \div N_V$, and the ratio of fixed corner agents to fixed horizontal agents, $R_{CH} = N_C \div N_H$. Instead of using exact parameter values, the FRBS allows their linguistic description in the form of fuzzy sets. This helps to account for incomplete objects, noise, and agents that may have erroneously fixed. The rules used to classify the objects have the following form:

(a)	IF $R_{HV} \approx 1$	and $N_C \approx 4$	THEN object is a *square*
(b)	IF $R_{HV} \not\approx 1$	and $N_C \approx 4$	THEN object is a *rectangle*
(c)	IF $R_{HV} \approx 1$	and $N_C \approx 12$	THEN object is a *cross*
(d)	IF $R_{HV} \gg 1$	and $R_{CH} > 1$	THEN object is a *triangle*
(e)	IF $R_{HV} \approx 1$	and $R_{CH} \approx 4$	THEN object is a *circle*

The FRBS applies these rules to the input parameters and classifies the shape of the object based on how the parameters match the antecedent membership functions corresponding to each shape. For example, in rule (a), two separate triangular membership functions, centered around values 1 and 4, are used to represent the ranges of acceptable values within which R_{HV} and N_C may fall, respectively, for the shape to be classified as a square. Triangular membership functions are used because they are easily described and implemented. The consequents of the rules are represented by fuzzy singletons, with strongest membership achieved for input parameters exactly matching modal values of the input fuzzy sets.

3.4.2 Artificial Neural Network

Another instance of the classification subsystem has been implemented using a neural network. Compared to an FRBS, design of an ANN-based classifier is less intuitive and its operation less transparent. ANNs, however, provide greater flexibility and the ability to learn a variety of complex non-linear mappings. The latter property can be used to achieve an arbitrary classification performance without the need to explicate the classification rules. Rather, learning is performed based upon samples labeled with the desired output classes.

A three layer ANN has been implemented, consisting of two input units, six hidden units, and five output units corresponding to individual classes/shapes. Training occurs using a back-propagation algorithm applied to a labeled set of perfect image data over the course of 400 training epochs. The learning rate is set to 0.2 and the momentum term to 0.1. The activation function of the hidden and output neurons is of the sigmoidal type.

Based on the idealized feature profile observed in preliminary trials, it has been determined that all considered shapes could be represented by R_{HV} or R_{VH} and the ratio of the number of corners to the total number of features, $R_{CT} = N_C \div N_T$, where $N_T = N_C + N_H + N_V$. This set of input parameters is used because it provides higher resiliency to noise. Under noisy conditions, the swarm is prone to contain an abundance of corners, leading to a synthetically high N_C. However, if the corner statistics are represented using the R_{CT} ratio, the object classification is successful as long as this ratio falls within an acceptable range, partially compensation for the spurious corners.

The neural network is fed values $R_{CT} \in [0, 1]$ and the lower of R_{HV} or R_{VH}. Although not explicitly, the trained neural network considers the following relations between the input parameters and classes of objects. Circles and triangles have a very high R_{CT} ratio, but differ significantly in their R_{HV} ratios. Squares show similar R_{CT} ratios as rectangles, but can be distinguished by the close proximity of their R_{HV} ratio to unity. Finally, crosses can be identified as shapes similar to squares but with a R_{CT} ratio larger than that typically found in a square and less than that found in a circle.

4. Experimental Results

The complete system has been tested on images containing the five object shapes outlined in the last section. All tested shapes have been presented as 50×50 pixels grey-scale images, with white shape outlines set on to a black background, as shown in Fig. 2. The presented shapes occupy about five percent of the total image area, and are situated near the center of the image. To analyze these shapes, 20 swarm agents were evenly distributed on the image, each with a time-to-live of 15 iterations, and the potential to create 4 children upon fixation.

All five images, with the grey-scale values normalized to $[0, 1]$, have been analyzed for two global threshold values: 0.5 and 0.8. These values are defined as the intensity level in the range above which an agent bonds to a pixel. At both threshold levels, each shape is tested ten times and the results are averaged to obtain the classification accuracy. Each shape is tested at ten differing noise levels, ranging in 10% increments from 0% to 90%. The noise level is defined as the probability of each pixel in the image being modified by a uniform intensity value between 0% and 80% of the maximum grey-scale intensity level. Examples of the noisy images are shown in Fig. 3.

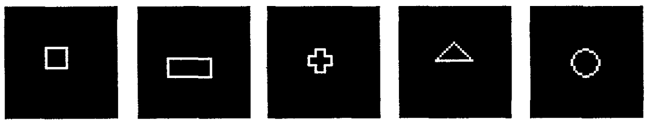

Fig. 2 *Shapes used for system testing.*

Analysis shows that the system accurately distinguishes between a range of sample shapes, even when the image is exposed to a moderate amount of noise or contains an incomplete shape. Tab. II summarizes the results of object classification under varying levels of noise. The system classifies rectangles and triangles quite well up to noise levels of 60%. Circles and squares show a more rapid drop in accuracy, with reliable classification up to 20% noise. These trends can be seen in Tab. II. The rapid decrease in accuracy is likely due to the fact that squares and circles must maintain rigid side-to-length ratios to be correctly classified. In support of this observation, test results show that the majority of misclassified squares are labeled as rectangles. More serious distortion only occurs at noise levels approaching 60%. Likewise, misclassification of circles yields 'triangle' and 'cross' results. Crosses show more robust classification than circles and squares, with good accuracy at noise levels up to 30%.

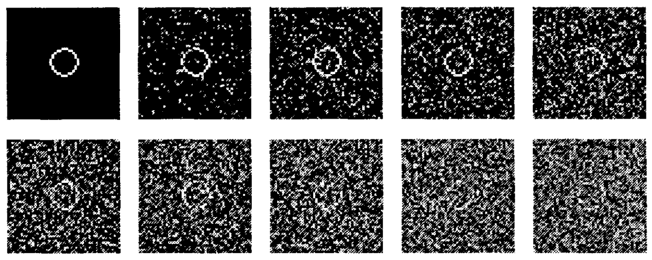

Fig. 3 *Noise applied to a test image – the individual images represent noise levels 0-90% in increments of 10%.*

As there are five object classes, classification by random chance is indicated by a classification rate of 0.2, while over 50% accuracy is indicated by classification rates greater than 0.5. The threshold level of 0.5 yields a better than random chance of classification at noise levels up to 50%, and over 50% accuracy of classification at noise levels up to 20%. Similarly, the threshold of 0.8 shows a better than chance classification up to noise levels of 60% and over 50% accuracy of classification for noise levels up to 20%. Interestingly, for rectangles and triangles, the 0.8 threshold tests show greater than 50% accuracy up to noise levels of 60-80% on rectangles and triangles, accurately classifying even highly distorted shapes.

A qualitative analysis showed that different types of noise have different effects on system performance. As described above, Gaussian noise was used to modify the test images. It was found that injected Gaussian noise does not prevent the system from identifying the major features of a shape. As such, the ratios of relevant features do not vary greatly from their noise-free values. The addition of Poisson noise had a more detrimental effect on the performance of the system, showing a greater divergence in feature ratios and a corresponding increase in misclassifications. Impulse and Laplacian noise lead to a plethora of erroneous corner features being detected in the area surrounding the shape. While the addition of new horizontal and vertical features did not greatly effect the deduced shape labels, the increase in the ratio of corners to total features caused a decrease in classification accuracy. While higher bonding threshold values allowed the rejection of some noise, a tradeoff was observed between erroneous corner deletion and the detection of continuous shape boundaries.

While this system was designed to identify single shapes, having multiple shapes in an image will not decrease the classification rate, as the average number of horizontal, vertical, and corner features should not change as long as the image is homogenous with respect to shape type. A system to deduce the nature of heterogeneous images will be presented in future work.

	Global threshold 0.5					*Global threshold 0.8*				
Noise	□	▭	+	△	○	□	▭	+	△	○
0%	1	1	1	1	0.7	0.9	1	1	1	0.9
10%	0.8	1	0.9	1	0.8	**0.9**	1	1	1	0.6
20%	**0.5**	1	0.8	1	0.7	0.3	1	1	1	**0.5**
30%	0.7	0.9	0.4	1	0.6	0.6	1	0.7	1	0.6
40%	0.3	0.9	0.7	0.9	**0.5**	0.3	0.9	**0.5**	0.9	0.5
50%	0.3	0.9	**0.5**	0.9	[0.2]	0.4	0.8	0.6	0.9	0.4
60%	[0.2]	**0.7**	[0.2]	0.9	0.4	0.3	0.5	[0.4]	0.9	[0.2]
70%	0.2	0.4	0.2	0.8	0.2	[0.1]	**0.5**	0	0.8	0.2
80%	0.3	0.3	0.3	**0.6**	0.1	0.3	0.5	0.3	**0.5**	0.1
90%	0.3	[0.2]	0	[0.2]	0.3	0.3	0.4	0.1	[0.2]	0

Tab. II *Classification rates for shape recognition in noisy conditions. Values set in bold font indicate the greatest noise levels at which accuracy is equal or exceeds 0.5. Values set in a pair of square brackets indicate noise levels where accuracy begins to fall below 0.2.*

5. Conclusions

In this paper, a swarm-based approach to the task of object recognition has been proposed. In an attempt to harness the many benefits of using such computational techniques as swarm intelligence, fuzzy sets, and neural networks, a robust system has been created which consistently and accurately classifies objects in images. The 'bottom-up' approach taken in this project allows the system to analyze the image indirectly by building an approximation of the image using a swarm and then analyzing the collected statistics of the converged swarm population. The system accomplishes this complex higher-level task while utilizing agents following simpler behavioral rules. The robustness of the proposed system is clearly demonstrated by its ability to correctly perform its task despite significant noise levels and occasional incomplete detection of the object's features by the swarm.

The primary focus of this paper is the successful combination of the swarm and classifier subsystems. It is not difficult to imagine the numerous potential applications to which the proposed swarm system may be applied in the future. The problem of distinguishing healthy from pathological conditions in medical images is of particular interest to the authors, and the work presented in this paper establishes a solid groundwork for future investigation in this field.

Acknowledgements

The work described in this paper has been supported by an Undergraduate Student Research Award provided by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

[1] Bonabeau E., Dorigo M., Theraulaz G.: Swarm Intelligence: From Natural To Artificial Systems, Oxford University Press, New York, NY, 1999.

[2] Bonabeau E., Theraulaz G.: Swarm Smarts, Scientific American, March 2000, pp. 72-79.

[3] Bourjot C., Chevrier V., Thomas V.: How Social Spiders Inspired an Approach to Region Detection. In: Proc. International Conference on Autonomous Agents and MultiAgent Systems, Bologne, Italy, **1**, 2002, pp. 426-433.

[4] Broggi A., Cellario M., Lombardi P., Porta M.: An Evolutionary Approach to Visual Sensing for Vehicle Navigation, IEEE Transactions On Industrial Electronics, **50**, 1, pp. 18-29, 2003.

[5] Dorigo M., Maniezzo V., Colorni A.: Ant System: Optimization by a Colony of Cooperating Agents, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics, **26**, 1, 1996.

[6] Dury A., Vakanas G., Bourjot C., Chevrier V., Krafft B.: Multi-agent Simulation to Test a Coordination Model of the Prey Capture in Social Spiders. In: Proc. 13th European Simulation Symposium, pp. 831-833, Erlangen, 2001.

[7] Engelbrecht A. P.: Computational Intelligence: An Introduction, Wiley, 2002.

[8] Gaing Z. L.: A Particle Swarm Optimization Approach for Optimum Design of PID Controller in AVR System, IEEE Transactions On Energy Conversion, **19**, 2, 2004, pp. 384-391.

[9] Grassé P.: La reconstruction du nid et les coordinations inter-individuelles chez Bellicositermes natalensis et Cubitermes sp. La théorie de la stigmergie: essai d'interprétation du comportement des termites constructeurs, Insectes Sociaux, 6, 1959, pp. 41-81.

[10] Jang J. R., Sun C., Mizutani E.: Neuro-Fuzzy and Soft Computing, Prentice Hall, Upper Saddle River, NJ, 1997.

[11] Liu J., Tang Y.: Adaptive Image Segmentation With Distributed Behavior-Based Agents, IEEE Transactions on Pattern Analysis and Machine Intelligence, **21**, 6, June 1999, pp. 544-551.

[12] Parsopoulos K. E., Vrahatis M. N.: On the Computation of All Global Minimizers Through Particle Swarm Optimization, IEEE Transactions On Evolutionary Computation, **8**, 2004, pp. 211-224.

[13] Ramos V., Almeida F.: Artificial Ant Colonies in Digital Image Habitats – A Mass Behaviour Effect Study on Pattern Recognition. In: Proc. ANTS 2000 – 2nd Int. Works. on Ant Algorithms (From Ant Colonies to Artificial Ants), 2000, pp. 113-124.

[14] Robinson G. S.: Edge Detection by Compass Gradient Masks, Computer Graphics and Image Processing, **6**, 5, 1977, pp. 492-501.

[15] Wachowiak M. P., Smolikova R., Zheng Y. F., Zurada J. M., Elmaghraby A.S.: An Approach to Multimodal Biomedical Image Registration Utilizing Particle Swarm Optimization, IEEE Transactions On Evolutionary Computation, **8**, 3, 2004, pp. 289-301.

[16] Yin P. Y.: A Discrete Particle Swarm Algorithm for Optimal Polygonal Approximation of Digital Curves, Journal of Visual Communication and Image Representation, **15**, 2, 2004, pp. 241-260.