Proceedings of the 24th IASTED International Multi-Conference
ARTIFICIAL INTELLIGENCE AND APPLICATIONS
February 13-16, 2006, Innsbruck, Austria

A SWARM INTELLIGENCE APPROACH TO COUNTING STACKED SYMMETRIC OBJECTS

Chad George
Indiana University
South Bend, Indiana
United States
cgeorge@cs.iusb.edu

James Wolfer
Indiana University
South Bend, Indiana
United States
jwolfer@iusb.edu

ABSTRACT
In a manufacturing environment it is often necessary to perform a manual inventory of finished goods and raw materials. These raw materials might be wood, plastic or metal and often represent a large investment of capital to procure and store. While the economic value of an accurate inventory is high, the process of obtaining a good count is tedious and fraught with human error. As a precursor to counting an inventory of tubular steel bar stock from digital images, we present a hybrid algorithm inspired by ant colony and particle swarm technology. This algorithm defines an environmental habitat for interacting particles to competitively cluster into segmented colonies. By forming colonies at bar stock end profile locations the algorithm provides potentially countable high-level information.

KEY WORDS
Computational intelligence, swarm intelligence, computer vision and manufacturing.

1. Introduction

In a manufacturing environment it is often necessary to perform a manual inventory of finished goods and raw materials. These raw materials might be wood, plastic or metal and often represent a large investment of capital to procure and store. While the economic value of an accurate inventory is high, the process of obtaining a good count is tedious and fraught with human error. It would be beneficial to have an automated system that could assist a human operator in determining inventory on hand.

One raw material used for this study is metal bar stock, specifically small diameter steel tubes as displayed in Figure 1. Like most raw materials this tubular stock is usually stored in homogenous stacks or bundles. Digital images taken of the tube stack end profiles usually contain edges defined at areas of large local grey scale changes. Additionally in the industrial environments studied, the homogeneous bar stock to be counted is almost always clustered in a single continuous region of fairly symmetrical objects that may be isolated from background and other foreground objects. By processing digital images of the tube ends we seek to identify individual tubes as a precursor to inventory assessment.

Figure 1 – Profile end of small diameter tube stock

As with most image processing problems, the task of accurately counting discrete bars from images acquired in an industrial environment is complex. A practical solution must handle a wide variety of bar stock shapes and sizes, inconsistent image quality and changes in camera angle. Given these challenges we feel that this task may be well suited to help assess the real world potential of swarm approaches to image processing.

Swarm processing is popular in computational intelligence, where a desired global behaviour is allowed to emerge from the local interaction of agents each utilizing much simpler individual behaviour. This processing model has been successfully applied to optimization problems. The two most common types of swarm processing models are the Ant Colony Optimization (ACO) and the Particle Swarm Optimization (PSO) algorithms. In recent literature, both of these algorithms have been adapted for image processing.

For example, Ramos and Almeida [1] use digital images as the habitat of an artificial ant colony. They used the simple ant model described by Chialvo and Millonas [2] to define the movement of ants within a digital habitat. A grey scale digital image was used to represent the initial pheromone concentrations for the ant's environment. As the ants moved throughout the digital habitat, they changed the pheromone concentrations in their environment. The resulting global patterns of movement that emerged correlated with edge locations in the source images, showing that even simple ant colony systems can be used to effectively extract edge features from digital images.

Zhuang and Mastorakis [3] present an alternative ant colony model for image processing. In their approach a perceptual graph of the connections between pixels was constructed. Specifically, as the ants explored their environment, they updated the connections on the perceptual graph based on the cost of the path they followed. By using appropriate path cost functions their model proved capable of extracting both image edge features and segmenting similar image regions.

Omran, Salman and Engelbrecht [4] apply a PSO algorithm on the problems of image segmentation and classification. This approach uses the PSO algorithm to search for an optimal clustering of pixels. Their results showed that the PSO algorithm can be applied effectively at the problem of image segmentation.

Finally, Li [5] presents a particle swarm model where local neighbourhoods of particles are formed by allowing species to differentiate from the initial homogenous set of swarm particles. These individual species are able to independently search for separate local minima within the problems space. While not originally applied to image processing, this model seems useful for the problem of isolating symmetrical discrete regions within the image space.

The next section of this paper presents a swarm approach to image processing composed of a hybrid of these basic swarm models, followed by the results of applying this algorithm to the problem of counting small diameter bar stock.

2. Algorithm Description

As previously discussed, research using swarm techniques for digital image processing has shown that the distributed processing model utilized in the ACO and PSO algorithms can perform low level tasks such as edge detection or region segmentation. While extracting these features is fundamentally important to high level image processing, often they do not significantly simplify the problem of counting symmetric objects in an image. This encouraged the development of a new swarm based algorithm specifically designed to handle the problem of segmenting stacked symmetric objects. The final algorithm utilizes many of the principles found in the ACO and PSO algorithms but combines their strengths as needed by this specific problem domain.

Our approach to segmenting and counting the stacked objects uses the following steps:

1. Create an environment map to represent the source image of objects to be counted
2. Seed the image as represented by its environment map with a population of randomly placed particles.
3. Allow the particles to interact with each other and their environment in such a way that they form segmented colonies.
4. Utilize the survivability of particles in a colony to represent the fitness of the colony's size and position.
5. After a stabilization criteria is met, the number of colonies remaining approximates the number of target objects in the source image.

The four main components of this algorithm are a global habitat, an environment map, a distribution of particles and their associated colonies. The remainder of this section will discuss each of these components in greater detail.

2.1 Habitat

The habitat consists of an environment map, particle colonies and the particles themselves. The habitat performs the fitness evaluations of the colonies and handles house keeping issues such as collecting dead particles and controlling colony reproduction.

2.2 Environment Map

The environment map provides individual particles with a perception of their surroundings. Similar to the digital habitats described in [2], an environment map is produced from a source image by calculating a gradient vector and an intensity value for each pixel location.

During image pre-processing a region of the source image containing objects to be counted is cropped, scaled and converted to grey scale. Because of its well defined edges, tubular stock does not require additional image processing. After pre-processing the target image, the following procedure is used to create an environment map:

1. Gradient vectors are calculated for each pixel in the image by applying a 3x3 Sobel convolution over the image.

2. The intensity at each pixel location is calculated as the length of the gradient vector at that position.
3. Gradient vectors are then normalized to unit length and rotated such that closed regions in the source image produce an interior region where the gradient vectors point inward and an exterior region where the gradient vectors point outward.
4. The intensity values at every pixel location are normalized by dividing by the largest intensity found. This represents the energy level at that location in the environment map.
5. After normalizing intensity values, any gradient vector whose corresponding intensity is below a given threshold is set to be zero.

After the initial development of the environment map, the image pixel values are no longer processed directly. Only the local interaction of autonomous particles and their segmentation into colonies are used to determine the position and size of the target objects in the source image. Also, unlike the traditional Ant Colony Optimization where autonomous agents communicate with each other by changing their local environment, in this algorithm the agents (particles) have no mechanism to change the contents of the environment map and therefore cannot communicate with each other using stigmergy.

2.3 Particles

As with other swarm algorithms, the algorithm presented here is primarily based on the behavior of individual particles and their interaction with surrounding particles within a local context. While it is active, each particle exists at a position constrained by the limits of the environment map. In each generation, a particle performs three tasks.

1. Each particle moves based on its local environment, the location of its colony and its current state.
2. Particles continuously interact with surrounding particles to form colonies or change their own state.
3. A particle updates its own state based on its position, surrounding particles and the local environment

The performance of these basic behaviors determines the particles interaction with its environment and surrounding particles. Next we describe the individual particle's behavior rules in more detail.

2.3.1 Particle Movement

A particle's movement at any time is stochastically determined from four main components: a force component calculated from the environment map, an inertial component representing its momentum, a social component directing the particle to move toward or away from its colony and a random component to encourage local exploration.

In each generation, the direction of movement for the i_{th} particle is calculated using the following equation:

$$v_i(t+1) = v_f(t)\, c_f\, r_f(t) + v_i(t)\, c_m\, r_m(t) + v_s(t)\, c_s\, r_s(t) + v_r(t)\, c_r\, r_r(t)$$

where r_f, r_m, r_s, and r_r are randomly generated from the interval [0,1],

c_f, c_m, c_s, and c_r are stochastic scaling constants, and

$v_f(t)$, $v_i(t)$, $v_s(t)$ and $v_r(t)$ are velocity terms from the previous time step.

The *force component*, $v_f(t)\, c_f\, r_f(t)$, represents the local environment's impact on the particles movement. The velocity term, $v_f(t)$, is the normalized gradient vector stored in the environment map at the current location. A scaling term, c_f, chosen heuristically is based on the scale and composition of the environment map so that this term dominates the movement equation as a particle approaches bounded regions. Because the gradient vectors always point inward for closed curves, a particle approaching a boundary will be repelled back toward the center of the bounded region, and likewise a particle outside the bounded region will be repelled away from the region.

The *inertial component*, $v_i(t)\, c_m\, r_m(t)$, represents the particle's momentum. Its velocity term, $v_i(t)$, is the normalized particle velocity from the previous time step. Momentum helps particles explore their environment and reinforces the effect of the force component after the particle has left a boundary region's field of influence.

The *social component*, $v_s(t)\, c_s\, r_s(t)$, represents a colony's tendency to explore and expand. The velocity term, $v_s(t)$, is a unit vector in the direction of the colony's center relative to the particle's current position. The scaling factor, c_s, is set based on the current state of the particle. A negative scaling factor will cause the particle to move toward the center of the colony, while a positive value causes the particle to move away from the colony.

The *random component*, $v_r(t)\, c_r\, r_r(t)$, represents the particle's tendency for local exploration. The velocity term, $v_r(t)$, is a randomly generated unit vector and the scaling factor, c_r, is small enough that it is not sufficient to overcome the force component in the bounded regions of the environment map.

The result of the movement equation is normalized and represents a stochastically weighted direction of movement for the particle. The magnitude of movement is determined by the local energy level of the environment which is the normalized intensity value from the

environment map at the particle's current location. The position of the i_{th} particle is then updated using

$$x_i\,(t+1) = x_i\,(t) + v_i\,(t+1)\,(c_{min} + \varepsilon_f(t)\,c_{max}) / |\,v_i\,(t+1)\,|$$

where $x_i\,(t)$ represents the current particle position, c_{min} represents a constant minimum speed and $\varepsilon_f(t)$ is the local energy level.

2.3.2 Particle Interaction

In swarm algorithms, local agent interaction is the mechanism by which global behavior emerges. In the ACO algorithm, agents interact by modifying their environment. In the PSO algorithm, agents interact by observing each other's position and velocity and by sharing information about the best solution found. This algorithm combines the spatial environment map of the ACO algorithm with the observation based particle interaction of the PSO algorithm. In this algorithm interaction between particles is based on two regions that determine when a particle can detect and influence surrounding particles.

Inside a *perception zone,* surrounding particles are considered as being nearby. These surrounding particles are counted as relatives if they belong to the same colony or enemies if they belong to a different colony. While this is similar to the communication mechanism in the PSO algorithm, our particles have no awareness of the position or velocity of surrounding particles. Only the presence of a nearby particle is noted and their colony status is recorded.

The *engagement zone* is a smaller region inside the perception zone where "enemy" particles are challenged. A system wide stochastic parameter determines whether a challenge occurs when a particle recognizes an enemy within its territory. When a challenge does occur between two particles, a contest is fought between the two particles involved. The aggressor attempts to force the enemy particle to join its colony, while the defender attempts to remain in its current colony. Both particles are given a score based on the number of relatives it detected within its own perception zone. These scores determine the probability of the attacker winning the challenge. After a defender loses a challenge it becomes a member of the winning colony.

Since the size of the perception and engagement zones are system wide parameters, particles will play the role as aggressor and defender. By using the number of surrounding relatives as the score, the strength of the attack is proportional to the size of the colony, but it does not unfairly advantage a rouge particle that is far away from the rest of a large colony.

2.3.3 Particle State

Each particle has a number of parameters that collectively define the particle's state: a behavior parameter that encodes its current goals, an energy parameter that represents its fitness and survivability, and an affinity parameter that describes its proximity and relationship to surrounding particles.

The behavior parameter encodes the particle's affinity toward its colony's center or the degree to which it is repelled away from the colony. In effect this gives the particle one of two goals: to explore outward for boundaries or to return home to the colony. This is implemented through the social component of the movement equation. Changing the magnitude and sign of the stochastic scaling factor causes a general tendency to explore outward or return home.

The energy state parameter accumulates energy collected by the particle from its environment. At each generation, a particle's energy increases proportional to the intensity of the local energy in the environment. When a particle's energy level exceeds a global threshold its behavior parameter changes from exploration to seeking home. When the particle moves within some distance from the colony's position its energy is lowered back to a small starting value and its behavior parameter changes back to exploration.

For particles enclosed by an appropriately sized boundary, the behavior and energy parameters cause the particle to continuously cycle between gathering energy at the boundary edge and delivering that energy back to its colony. How well a particle can perform this task is based on its position, its colony's position and the local environment map. Over time, a particle's performance at finding and collecting energy from the environment determine its fitness and survivability.

Every particle also calculates an affinity parameter as the ratio of relative and enemy particles detected within its perception zone. This affinity parameter represents the proximity and composition of surrounding particles and is another measure of the fitness of the particle's position. When a particle is not surrounded by a properly sized closed boundary, it will expand far away from other particles in its colony. In this case, its calculated affinity parameter will be low and consequently its energy level is allowed to decay toward zero at a heuristically determined rate eventually causing the particle to be terminated.

Another adverse condition occurs when a particle is surrounded by a closed boundary that is too small. The geometry of stacked tubular objects along with the process of creating the environment map forms small bounded regions in the spaces between tubes. A particle trapped in this region will be constantly accumulating energy since it can not move outside of the field of

influence around the boundary. When this accumulated energy exceeds a maximum threshold the particle is killed.

In summary, a particle is considered to be dead when any one of the following conditions exists: the particle's energy deceases to zero, the particle's energy exceeds some maximum threshold or the particle moves outside the limits of the environment map.

2.4 Colonies

In this algorithm, a colony is a collection of particles which are similar to each other and different from other particles in the system. A colony represents a potential solution to the segmentation problem and the survivability of its member particles represents the fitness of that solution.

A colony's position is defined as the average position of all the particles that are a member of the colony. A colony's size is defined as a bounding region surrounding its member particles. In practice the bounding region is implemented as a circle that encloses most of the colony's member particles. It is determined by calculating the distance between each particle and the colony center, and then averaging the maximum particle distance over time. Each colony is then evaluated as being too small, too big or the proper size, based on its size compared with the average size of other colonies in the habitat. If a colony's size exceeds some threshold with respect to the average colony size all of its member particles are exterminated and it is considered a dead colony. If a colony is too small, its member particles are not allowed to release their accumulated energy and consequently they may exceed the maximum energy level and be terminated.

If a colony's radial size is within a heuristically determined tolerance of the average size, it accumulates the energy transferred from its member particles. When this stored energy grows above a threshold the colony is allowed to reproduce, creating a new particle at its current location. After the colony's population has reached a maximum limit any additional energy collected by its particles contributes to increasing the colony's fitness. When a colony becomes highly fit it is marked as a valid solution. The habitat then uses the colony's position and size to estimate where other valid solutions are likely to exist based on the symmetric nature of stacked tubular objects. As each valid colony is established, the habitat spawns new colonies in surrounding regions that are likely to also be valid.

3. Results

A simulation was written in Java using the MASON [6, 7] multi agent simulation toolkit to test the effectiveness of the algorithm and to investigate the effect of changing system parameters. Multiple simulations were performed on the sample image set to assess the effectiveness of the algorithm and to determine appropriate ranges for algorithm parameters under different conditions.

Source images were obtained in an industrial environment under normal lighting conditions with a digital camera at a resolution of 2048x1536 pixels. The source images where then tiled into 150x150 pixel regions and scaled to 500x500 pixels. From these tiled images, a test set was selected that represented the major features found in the source image. The algorithm's performance was evaluated based on how accurately it accomplished three specific tasks:

1. Finding the center of each bounded region completely contained inside the image perimeter.
2. Ignoring the small regions formed by adjacent tubes.
3. Identifying only bounded regions created by the tube profiles and ignoring the large empty spaces at the perimeter of the tube stack.

Figure 2 shows the environment map generated from a test image and the random distribution of 100 colonies. In this simulation each colony has an initial population of 15 particles and the maximum colony size is 25 particles.

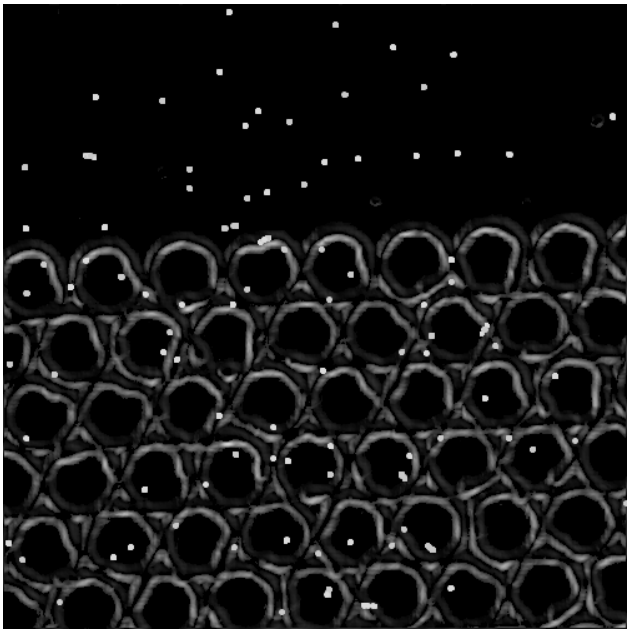

Figure 2 – Initial random colony distribution

Figure 3 shows the simulation after 50 generations. Many colonies have clustered into bounded regions. Colonies are represented as a circle with a small solid disk at the colony's current position. Colonies that are not bounded by a properly sized region have grown much larger than the average and will eventually be die out.

Figure 4 shows that after 500 generations all colonies that were bounded by a properly sized region have posted

themselves as being valid and their final locations correlate with the positions of tubes in the source image whose profiles were not cropped by the image boundaries.

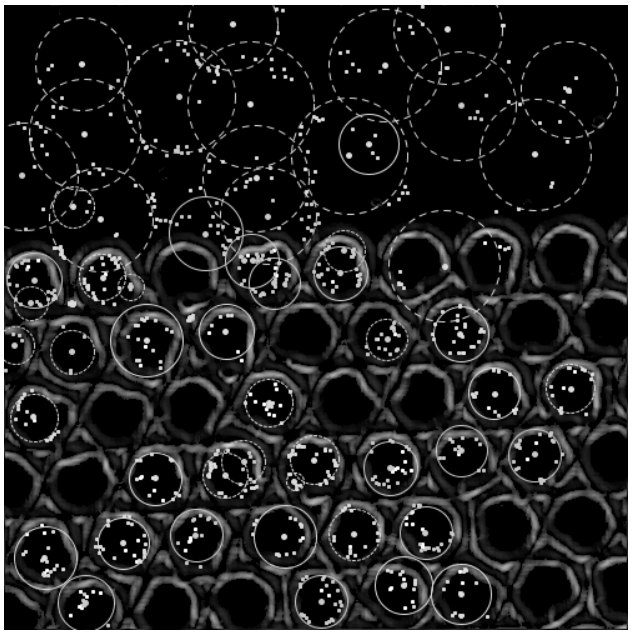

Figure 3 – Simulation results after 50 generations

Colonies occasionally formed in the regions between the target objects, but these colonies did not survive long enough to be counted. Also colonies that started in the empty regions of the image did not survive.

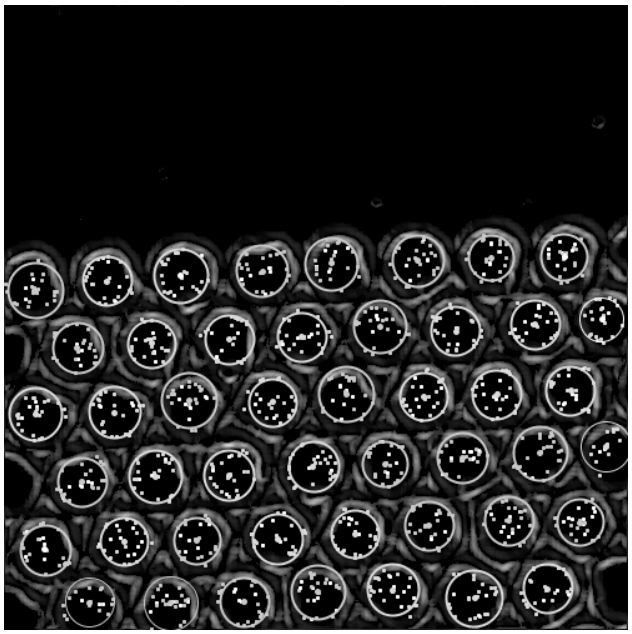

Figure 4 – Simulation results after 500 generations

4. Conclusion

One of the distinguishing features of this algorithm for image processing is that the algorithm extracts useful, simplified information. The final location and size of colonies directly correlates to bar locations in the source image. If missed regions or false positives exist they can be inferred from the available information and tested as appropriate. In the cases tested, the algorithm successfully segmented the target objects present that were not cropped by the image boundary.

In the future, we would like to apply the algorithm to larger images, using it to intelligently explore small regions of a total image. This would capitalize on the strengths of the algorithm, while providing an economic use of available resources. We would also like to compare the performance of this algorithm with other algorithms for detecting round objects such as the Hough transform and Cord Bisecting procedures described by Davies [8].

In conclusion, we believe this algorithm is effective at segmenting the target objects.

References

[1] V. Ramos, and F. Almeida, Artificial Ant Colonies in Digital Image Habitats - A Mass Behaviour Effect Study on Pattern Recognition, *Proceedings of ANTS'2000 - Int. Workshop on Ant Algorithms* (*From Ant Colonies to Artificial Ants*), Brussels, Belgium, 2000, 113-116.

[2] D. Chialvo, and M. Millonas, How Swarms Build Cognitive Maps, *The Biology and Technology of Intelligent Autonomous Agents*, NATO ASI (144), 1995, 439-450.

[3] X. Zhuang, N. E. Mastorakis. Image Processing with the Artificial Swarm Intelligence, *WSEAS Transactions on Computers, 4* (4), April 2005, 333-341.

[4] M. Omran, A. Salman, and A. P. Engelbrecht, Image classification using particle swarm optimization. *Proc. 4th Asia-Pacific Conference on Simulated Evolution and Learning 2002*, Singapore, 2002, 370-374.

[5] X. Li, Adaptively Choosing Neighbourhood Bests using Species in a Particle Swarm Optimizer for Multimodal Function Optimization, in *Proc. Genetic and Evolutionary Computation Conference 2004*, Seattle, WA, 2004, 105-116.

[6] S. Luke, G. C. Balan, L. A. Panait, C. Cioffi-Revilla, and S. Paus. MASON: A Multiagent Simulation Environment. *Simulation, 81*(7), July 2005, 517-527.

[7] MASON 10: A Java Multi-agent Simulation Library, http://cs.gmu.edu/~eclab/projects/mason, August, 2005.

[8] E. R. Davies, *Machine Vision: Theory Algorithms Practicalities 3rd Ed* (San Francisco, CA: Morgan Kaufmann, 2005)