The Application of Swarm Engineering Technique to Robust Multi-chain Robot System

K. Chang, J. Hwang, E. Lee, S. Kazadi*
Jisan Research Institute; 28 North Oak Avenue; Pasadena, CA 91107; USA

Abstract

The swarm engineering technique construed in [1] attempts to develop a general methodology that can be applied in creating swarm-mediated systems. In this companion paper, we utilize the system established in [2] to further explore the swarm engineering technique and show how this methodology can be used to develop a robust multi-chain robot system in a rigorous manner. In addition, we show the benefits of building a multi-chain robot system using the swarm engineering technique.
keywords: swarm engineering

1 Introduction

In the past, there has been significant interest in the development of control algorithms for swarms of agents. Previous research efforts in the field of swarm engineering includes robot lattice formation using artificial physics [4]. The method based on artificial physics uses a dynamic set of equations that the given agents follow to produce a desired global behavior of the swarm. In [4], the artificial physics-based control algorithm involved a set of dynamic equation based on an attractive and repulsive force. These forces were utilized to create perfect hexagonal and square robot lattices, which was the desired global outcome. Other control methods in swarm systems include the use of social insect based control methods [8] and the application of stigmergy to control cluster building [6, 7]. In [12], individual agents are capable of estimating the group size, so that the agent can join or leave the group to leave the group size at a pre-determined target value. Though these efforts certainly have succeeded in their individual outcome, what is elusive is a more general method that can encompass swarm systems including systems based on methods mentioned above, as well as other systems.

*To whom correspondences should be addressed at sanza@jisan.org.

In the past, one of the many methods of creating swarms, called swarm engineering, is composed of two steps[3]. The first step involves generating a swarm condition, which is defined as a condition which, when satisfied, leads to the generation of a swarm of agents which is capable of carrying out a desired task. No specific method has been created for this step, though there have been numerous methods in past research efforts as mentioned above. The second step involves the creation of swarm behaviors that satisfy the given swarm condition. By understanding the swarm condition, swarm behaviors may be fabricated to fulfill that condition. The importance of these two steps in creating swarms is that they are a practical way to produce a general method to yield a desirable global behavior.

It is hoped that a more generalized and rigorous method would provide a standard set of techniques for designing swarms that can be assured dependability [15]. Among the recent efforts is the effort by [13], which proposes the two major models to identify swarms, macroscopically and microscopically. [1], the theoretical companion paper to ours, merges these two models together into a *middle-meeting method,* which is a general guideline for swarm generation. This method is a rigorous and general technique for building swarms with pre-specified properties provably. The method is designed to guarantee global properties of the swarm as long as local properties derived from the global properties are satisfied. This allows for the verification of the system design before construction.

In this paper, we adapt this new swarm engineering technique to the robust multi-chain robot system, which is an extension of the single-chain robot system in [2]. The single-chain robot system was created to search a field for resources without losing any robots, and bring the resources to the nest without the use of pheromones, GPS (Global Positioning System), positioned tracking, etc. This led to the robustness of the system, which allowed us to build upon the system, creating a robust multi-chain robot system. The multi-chain robot system improves on the searching mechanism set forth in [2], since it is described in [2] that a chain becomes unstable when it reaches a certain length. Thus, we seek to set a stable chain limit on these chains and attempt to engender a set number of chains that can search a field more efficiently than a single chain can. Although disorder and a certain degree of randomness is built into the multi-chain robot system, the properties of the swarm converges to a fixed point [14]. A method for generating a pre-determined number of chains in the system is developed. Moreover, we demonstrate how to build a swarm-mediated system with the new swarm engineering technique. Section 2 describes the system used for our study. Section 3 illustrates the development of our system by the application of the new swarm engineering technique. Section 4 describes our simulation and give our results. In section 5, we discuss the results and in section 6, we conclude.

2 System

We utilize a two-dimensional simulation in our system. The simulation is a pseudophysical "world" containing circular mobile robots. Each robot moves linearly or in an arc along some direction, but is capable of changing its direction when faced with obstacles. Robots may not overlap or move through one another, and are constrained to a single maximal movement speed. Each robot is capable of sensing other robots in the immediate vicinity about a set radius. Also, each robot has an angular field of vision of 360 degrees[1]. We borrow our detailed base system from [2]. This system is capable of generating chains of robots which rotate around single stationary robots. This rotation allows the chains to grow more quickly than they would if they were static, and allows the robots to search the area around the central robot without the last robot in the chain losing connection to the central robot. The power of this method, as mentioned before, comes from the fact that pheromones, GPS, or positioned tracking is unnecessary in this field search mechanism.

Our perturbation to the simulation is designed to make a minimal or near minimal change which allows a specific number of chains to be formed and maintained. As with the original study, each robot is assigned a relation number which describes its state and behavior in the chain system. All robots start initially with a relation number of 0 or -1. Robots numbered 0 do not change state and always remain as the stationary robots. Likewise, robots starting off as -1 robots never transition into a 0 robot. Those with relation number 0 are less common, but a specific upper limit on these robots' states is not maintained. Robots in the -1 state spiral until they come in contact with an existing chain or a robot in state 0, which is the stationary base for the chain. Once the robots in state -1 come in contact with a robot in state 0 or a chain, the robot in state -1 finds the robot with the highest relation number in the chain by moving up toward end of the chain until it reaches a robot that cannot see a robot with a relation number one higher than itself. Then the robot that has found the last robot in the chain renumbers itself from -1 to a relation number one greater than the highest found, and joins itself to the chain. In some cases, we can set the robots in state -1 to stay in the -1 state for some pre-determined time, keeping clear of chains, in order to control the rate at which the chains are formed.

In order to rotate the chain about the 0 robot, each robot in the chain continually searches for and moves toward the midpoint of the line segment connecting the robots with relation numbers one above and one below it. This keeps the robots chain relatively linear. The last robot in this near linear chain sets the rotational speed, because the last robot moves perpendicularly to the line segment joining it ot the next robot at its highest speed; the other robots follow, using the method discussed above. So the other robots in the chain must travel at a slower rate in order to maintain the linear chain. This allows the system to rotate, compress, extend, reverse direction, etc. according to the actions of the last robot.

The system has the unforeseen property that when adjacent chains collide, they tend to collapse. This means that many of the agents in these swarms are released from the chains, and return to a -1 state. This makes the establishment of a stable chain system much harder.

One thing to note is that our system does not seem to be affected by a change in density. Because our system is robust in the sense that no robots are lost, the density of the robots in the field has little effect on the number of chains.

[1] This might be accomplished by facing a camera upward at a cone-shaped mirror with the tip facing down.

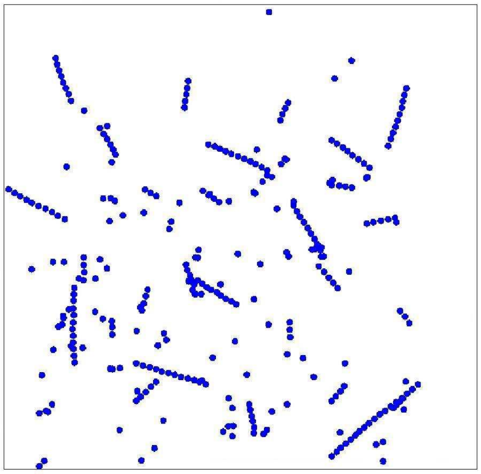

Figure 2.1: Above is a snapshot of our system. The size of the robots are exaggerated for viewing purposes. In this intermediate stage of our simulation, we can see the formation of chains that are spiraling around the stationary robots.

3 Swarm Engineering of Chain System

Our goal is to produce agents with behaviors such that there will be a set number of chains in a system each consisting of an equal number of agents. In this study, our system consists of 200 robots in the -1 state and the agent behaviors/properties we create influence the agents to form 10 chains of 20 robots each. In order to create these agent behaviors and properties, we utilize the method outlined in the companion paper [1]. First, we introduce important definitions. A property, P_i of a system is a characteristic of the system that can be measured using a process that is independent of the characteristics. A behavior, b_i, is defined as the way the properties of the system change in time. That is,

$$b_i = \frac{dP_i}{dt} \tag{1}$$

And generally we assume that the desired property,

$$P_s = f(P_{1,...,}P_{n,}b_{1,...,}b_n). \tag{2}$$

Then, the first step is to choose a global goal in terms of a set of properties of the individual members of the swarm, and organize them into corresponding initial and final characteristics. Once these initial and final characteristics are determined, we must specify the conditions under which the final characteristics become the result of the initial conditions and the system dynamics. This is known as the *swarm engineering condition.* In [1], the general swarm engineering condition is rigorously defined as follows: we want to find a set of conditions such that

$$\lim_{\tau\to\infty}\int_0^{\tau}\sum_{i\neq j}^{n_b}\left(\frac{\partial P_s}{\partial b_i}\frac{db_i}{dt}+\frac{\partial P_s}{\partial P_i}b_i\right)dt$$

$$+\lim_{\tau\to\infty}\int_0^{\tau}\frac{\partial P_s}{\partial b_s}\frac{db_j}{dt}dt+P_s^0=P_s^F. \tag{3}$$

Agents' behaviors depend on agents' sensors, memory state, behavioral strategy, and position. So the important thing is to create behaviors that are contingent on realistic sensor states and internal states which provably yield the desired global property.

3.1 Generation of Swarm Condition

First, we define our swarm condition that will lead to our global goal. Let the property of the swarm

$$P_s = \sum_{i=1}^{N_r} H_i, \tag{4}$$

where H_i is the chain length of the chain that robot i is part of and N_r is the number of robots. It is understood that if the robot is wandering, $H_i = 0$. This is one of many properties of the swarm that can be used to control the global goal. Then, if all robots are in chains of desired length l_d, then this is equal to $N_r l_d$. Plugging this into (3), we obtain

$$\lim_{\tau\to\infty}\int_0^{\tau}\sum_{i=1}^{N_r}\frac{dH_i}{dt} = l_d N_r. \tag{5}$$

This is the behavior we need to fulfill in order to generate our global property, where we assume that $\frac{dH_i}{dt}$ is controlled by the individual robot behavior that we have direct control over. So we attempt to use this behavior to make each chain grow toward l_d.

N ext, we describe the behaviors of each individual agent as a function of its different states. Since our system consists of $\{N_r\}$ agents, we assume that the lth agent's state may be completely described by its internal state, in_s^l, its sensor state, s_s^l, and its positional state, p_s^l. We may then represent the global behavior b_S as a function of these different states. First, we define the coupling between agent l and the global behavior by $C_S^l(p_s^l, in_s^l)$. Next, we describe the individual behavior of the agent by $AB^l(in_s^l, s_s^l)$, assuming that agents have no memory. Then, the overall behavior may be expressed as follows

$$b_S = \sum_{l=1}^{N_r} C_S^l(p_s^l, in_s^l)AB^l(in_s^l, s_s^l). \tag{6}$$

In our case the coupling is direct and so we may represent C_S^l as

$$C_S^l = 1 \tag{7}$$

because each robot can always determine the length of the chain it is part of. We then can rewrite (6) as

$$b_S = \sum_{l=1}^{N_r} AB^l(in_s^l, s_s^l). \tag{8}$$

This expresses the idea that we are only concerned with the robots in a chain. Since each robot in a chain has an internal state equal to the chain length of the chain it is in, equation (8) changes to

$$b_S = \sum_{l=1}^{N_r} AB^l(H_l, s_s^l) \tag{9}$$

where $AB^l(H_l, s_s^l)$ expresses the behavior when the internal state is H_l. Now, (9) mirrors (5) in the following way

$$\lim_{\tau\to\infty} \int_0^\tau b_s dt = \lim_{\tau\to\infty} \int_0^\tau \sum_{i=1}^{N_r} \frac{dH_i}{dt} = l_d n_r$$

$$= \lim_{\tau\to\infty} \int_0^\tau \sum_{l=1}^{N_r} AB^l(H_l, s_s^l) dt \tag{10}$$

Thus,

$$AB^l(H_l, s_s^l) = \frac{dH_l}{dt} \tag{11}$$

and we know (5) is a behavior, which is solely dependent on each agent's sensor states and its single-valued internal state. Now we may construct the local behaviors of individual agents to satisfy the swarm condition.

3.2 Design of Local Behaviors

The global top-down method provides for a requirement of local behaviors, but it does not explicitly define any specific local behavior. The bottom-up method is used to generate and theoretically test the local behaviors.

There are three cases we need to account for in regards to number of chains in a system. The first is the case that there are fewer chains than the desired number of chains in the system. This is controlled by incorporating a maximum chain length limit into the agent behavior. The agents are not able to join chains if they see that the last robot in the chain has a relation number that matches the maximum chain length. This prevents the creation of long chains that could cause the formation of fewer chains than our global goal. In the second case, there might be more chains than the desired number of chains. This may occur when relatively short chains are dispersed around the field. When this happens, there must be a restorative force that reduces the number of chains in the system. We accomplish this by implementing a chain degeneration probability that breaks apart some chains and add their constituents into the remaining chains. It is necessary, however, to build the system in such a way that the probability of the shorter chains breaking apart to join the longer chains is much greater than the probability of the longer chains breaking up to join the shorter chains. The third case is when we have exactly the same number of chains as the desired number of chains. In this case, each chain is of length l_d, and every chain has reached a static state.[2]

The behaviors outlined above will force $\frac{dH_i}{dt}$ to increase toward l_d. Thus, $\frac{dH_i}{dt}$ satisfies (3), which shows that our defined behavior will engender the global goal.

4 Simulation

Our simulation consists of 300 total robots in an arena 1000 units by 1000 units. We initially begin with 200 -1 robots and 100 stationary zero robots. Each of these robots has a diameter of 1 unit, but in the snapshots we use in this paper, their sizes are exaggerated for viewing purposes. Initially, the -1 robots randomly spiral as they search for stationary robots. We implement the chain construction as described in the last section. A chain will spontaneously disintegrate with a probability given by

$$p = -0.000425l + 0.0085 \tag{12}$$

where l is the chain length. This occurs via a mechanism mediated by a propagated signal from the robot at the end. This control mechanism based on the monotonically decreasing probability function tends to remove the smaller chains entirely, allowing their agents to add themselves to other chains. The result is lengthening chains whose number is controlled, as illustrated in Figure 4.1. The run illustrated is typical, as the number of chains is controlled so as to produce a size, in this case, of 10.14 ± 2.43 after 200,000 iterations (robots move a maximum distance of 2, each iteration).

[2] The random chain collisions can break even stable chains.

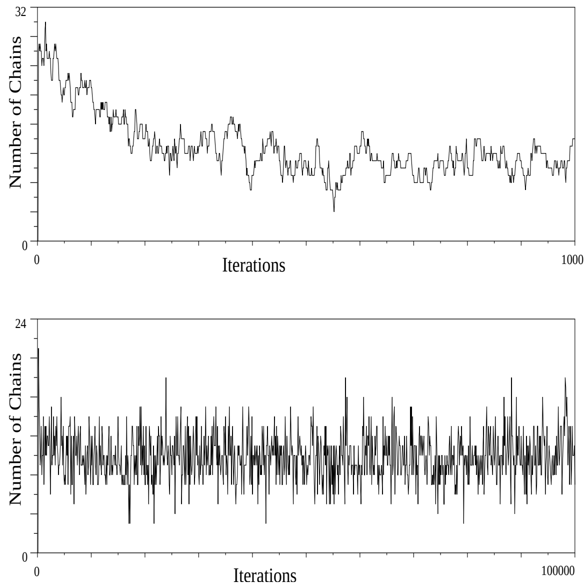

Figure 4.1: This graph illustrates the number of chains in the system under the behavior of the swarm. The top graph illustrates the first 1000 iterations of simulated time, while the lower graph expands this to the first 100000 iterations. Notably, a very quick decline in the number of chains is evident during the first 1000 iterations. The system remains relatively stable for one hundred thousand iterations, oscillating around the desired number of chains.

By running the simulation with different probabilities of chain destruction, we can produce different means of number of chains, as shown in table 4.1. This reflects the interplay between the chain dissolution mechanism and random collisions between chains. If the probability is high, many agents continue circulating. If it is too low, small chains tend to last long.

Probability	Mean	Standard Deviation
-0.0003l + 0.006	14.99	2.37
-0.00035l + 0.007	13.26	2.46
-0.0004l + 0.008	11.16	2.46
-0.00045l + 0.009	9.23	2.40
-0.0005l + 0.01	7.35	2.27

Table 4.1: This table shows the mean of number of chains under a specific chain destruction probability after 50 runs. As we can see, a smaller probability allows more chains to remain as stable chains, engendering more number of chains. As the probability of chain destruction increases, the number of chains decreases, as well. The standard deviation does not particularly vary with the probability of chain destruction, but the standard deviation we have is the result of the random interactions of chains that constantly collide, destruct, and then reform based on the probability.

5 Discussion

Here we introduce a formal proof on how the chain degeneration probability tends to increase the length of individual chains. The probability of an agent breaking away from an existing chain, P_b, is expressed as follows

$$P_b \propto (c - l\alpha), \tag{13}$$

where c and α are constants, and l is the length of chain. Next, the probability of an agent joining an existing chain is

$$P_j \propto l^2. \tag{14}$$

P_j is a square function of the length of a chain because the chains sweep around in a circle when searching for agents. Now considering two chains of lengths, l_1 and l_2, where $l_1 = l_2 - 1$, the ratio of the net movement from l_1 to l_2 over l_2 to l_1 is as follows

$$\frac{P_b(l_1)P_j(l_2)}{P_b(l_2)P_j(l_1)} = \frac{(c - l_1\alpha)(l_1+1)^2}{(c-(l_1+1)\alpha)(l_1)^2} > 1 \tag{15}$$

Since the ratio is bigger than 1, there will be net movement of robots from shorter chains to the longer chains, proving that $\frac{dH_i}{dt}$ satisfies (3).

One thing to note is that the variance results from the movement of the chains, which causes the chains to collapse in collisions between different rotating chains. However, we include the rotation of the chains to hasten the process of chain formation, as the sweeping chains can pick up more spiraling robots to join the chains. When chains do collapse, however, the system naturally builds more chains to reach the desired number of chains. This constant process of destruction and construction of chains is the cause of the variance.

6 Conclusion

One of the missing links in the development of a principled science of swarms is the development of techniques that allow the design and potential analysis of swarms. In this paper, we have applied a new method proposed in [1] to a decentralized linear swarm first described in [2]. By utilizing this method, we were able to develop local agent-level behaviors that were expected to generate a global property, in this case the number of chains. Application of the new agent-level behaviors demonstrated that the system is capable of exhibiting this property as desired. The importance of this new methodology is that we can verify, before building swarms, that the swarms will perform the specific global tasks that are desired.

In essence, we must generate the local behaviors that are capable of being performed by the individual agents in the system. In the multi-chain robot system, the local behaviors consist of the robots' ability to change states and to either join or destroy existing chains. Next, we must generate a global property based on the local properties that have been previously determined. In the multi-chain robot system, the global property we want to generate is the desired number of chains. We are able to control the number of chains with a decentralized swarm system and without the individual agents having any knowledge of the number of chains in the system. By carefully controlling each individual agent's behaviors, what emerges is an emergent property that gave forth a pre-determined number of chains.

What we have laid forth is a general mechanism that can be generalized to many swarm-mediated systems. This will potentially allow us to analyze the steps in creating swarms in a more rigorous manner, thus ensuring provability before construction, as we have showed in the application of the new swarm engineering technique to the robust multi-chain robot system.

References

[1] Kazadi S. *The application of swarm engineering to puck clustering systems.* **Submitted**, 2005.

[2] Kazadi S., Kondo E., Cheng A. *A robust decentralized linear spatial search flock.* Proceedings of **IASTED International Conference Robotics and Applications**. Honolulu, Hawaii. USA. 52-59, 2004.

[3] Kazadi S. *The Genesis of Swarm Engineering.* **Proceedings of the SCI2003 Conference, Special Session on Swarm Engineering**, Orlando, Florida, USA, 2003.

[4] Spears W. & Gordon D. (1999). *Using Artificial Physics to control agents.* In the **Proceedings of IEEE International Conference on Information, Intelligence, and Systems** (ICIIS'99).

[5] Lerman K and Galstyan A: A General Methodology for Mathematical Analysis of Multi-Agent Systems, **USC Information Sciences Technical Report ISI-TR-529**, 2001

[6] R. Beckers, O.E. Holland, & J.-L. Deneubourg. From local actions to global tasks: Stigmergy and collective robotics. in H. Ritter, H. Cruse, & J. Dean (eds.), **Prerational intelligence: Adaptive behavior and intelligent systems without symbols and logic, Vol. 2. Kluwer Academic Publishers**, 2000 (pp. 549-563)

[7] Melhuish C. (1999) "Exploiting Domain Physics: Using Stigmergy to Control Cluster Building with Real robots" **Proceedings of 5th European Conference on Artificial Life**.

[8] O. Holland. The use of social insect based control mechanisms with multiple robot systems. **IEE CONTROL'96**, Exeter, 1996

[9] Eric Martinson and David Payton, "Lattice Formation in Mobile Autonomous Sensor Arrays", **Swarm Robotics Workshop, SAB '04**, Santa Monica, CA, July, 2004

[10] Lerman K and Galstyan A: A General Methodology for Mathematical Analysis of Multi-Agent Systems, **USC Information Sciences Technical Report ISI-TR-529**,(2001)

[11] Kelly, I.D. and Keating, D.A., Flocking by the fusion of sonar and active infrared sensors on physical autonomous mobile robots". **The Third Int. Conf. on Mechatronics and Machine Vision in Practice**. 1996, Guimaraes, Portugal. Vol. 1, pp 1/1 - 1/4.

[12] Holland O. & Melhuish C. (1997) An interactive method for controlling group size in multiple mobile robot systems. **International Conference on Advanced Robotics**, Monterey, USA.

[13] Lerman K, Martinoli A and Galystan A: A Review of Modeling Methods for Swarm Robotic Systems, **Proceedings of the SAD'04 Swarm Robotics Workshop**, Santa Monica (2004)

[14] Beni, G. (2004), "Order by Disordered Action in Swarms", **Proc. of Workshop on Swarm Robotics, 8th Int'l Conf. on Simulation of Adaptive Behavior (SAB '04)**, July 13-17, Los Angeles, USA. To be published in Lecture Notes in Computer Science, Springer-Verlag.

[15] Winfield, Alan FT. Harper C., Nembrini J. "Toward Dependable Swarms", **International Workshop (SAB '04)**, Santa Monica, CA, July 2004.