

Model Independence in Swarm Robotics

S. Kazadi

*Jisan Research Institute
515 South Palm Avenue, Unit 3
Alhambra, California 91803, USA*

Abstract

Purpose

The development of a methodology for generating swarms is examined and illustrated using lossless flocking.

Design/methodology/approach

A general methodology for swarm design is described. Examples of this approach in the literature are examined. A general requirement for lossless flocking is developed. The requirement is used in developing two swarm behaviors.

Findings

It is possible to apply the approach to the lossless flocking and to use the swarm condition to develop two swarm behaviors which satisfy this condition in many situations.

Research limitations/implications

This research illustrates the general swarm engineering method and demonstrates how it can be properly applied.

Originality/value

The swarm engineering method is used to develop the “quark” model, a new physicomimetic model.

Keywords: swarm engineering; swarm robotics; physicomimetics; “quark” physicomimetic model.

Paper type: Research paper

1. Introduction

In the mid 1990s, swarms began to be a viable possible technology in the area of autonomous robots. Many researchers worked on a variety of different platforms that ranged from walking to rolling to flying robots with sensors that encompasses sound, light, and touch based technologies. Initially very expensive to work on, these platforms have become relatively inexpensive, able to work with a variety of computational platforms. Initially robots could only roll around and utilize limited computation and sensory platforms; they now have the ability to walk and fly; uti-

lize sophisticated computational and communication networks, deploy other devices including missiles, sensors, and communication apparatus; determine their position exactly via GPS or using the network of other agents; and carry out sophisticated tasks using ever improving tactile sensors and actuators.

The vast improvement in technology, computational capability, and software have yielded a stunning array of different types of agents designed. These have led many researchers to examine increasingly sophisticated swarm systems, generating often times brute force methods that accomplish a goal and other times elegant solutions to problems which accomplish the goal. However, despite the vast amount of work being done on swarms, very little in the way of a unifying theoretical approach has emerged. Several researchers (Martinoli, Spears, Winfield, Lerman, Mataric) have examined specific systems and taken a close look at theoretical models of laboratory swarm systems. However, despite very good agreement between the theoretical and actual measured quantities, this work has not yet yielded theoretical results that are global in the sense that they pertain to *any swarm that might be constructed for a problem using any technology, processing, or knowledge base*. Yet, it is precisely this kind of knowledge that is needed to move swarms forward, as such an understanding of the capabilities and limitations of swarms is required for their use to enter the mainstream.

There are many authors that have examined the stability and cohesion of swarms. These studies have often times used sophisticated techniques including Lyapunov stability analyses and various stability analysis methodologies. Examples of this are found in [Gazi and Passino, 2003] and in references found therein. However, in many of these cases, the agents were assumed to be points, have no delay in determining a new direction of motion, and have perfect knowledge of their own and all other agents' positions. These conditions are totally incongruous with real swarms which have noisy motion control, sensor limitations, imperfect knowledge of their own position, and a limited range on direct sensing and communication with other agents. As a result, these kinds of studies are of limited applicability in the area of swarm robotics.

One recent paper has characterized two methods of swarm design as "top-down" and "bottom-up" in the following way. "In the top-down approach, the design process starts with specifying the global system state and assuming that each component has global knowledge of the system, as in a centralized approach. The solution is then decentralized by replacing global knowledge with communication. In the bottom-up approach, on the other hand, the design starts with specifying requirements and capabilities of individual components, and the global behavior is said to emerge out of interactions among constituent components and between components and the environment." [Crespi et al, 2008]. The difficulty in the first one is that it is generally difficult to see how global information can be replaced by communication without a significant change in the agent model; the solution is rooted in the model, rather than the model being designed by a general solution. The difficulty in the

second one is that the emergence is generally the very thing one wants to design but cannot because of the nonlinear nature of the interaction. Moreover, there is little in this type of design and analysis that gives an intuition about the general solution.

We have elsewhere examined swarm systems that are designed using a *middle-meeting* methodology which is both top down and bottom up. It is top-down in the sense that the global goals are known and used to determine the behavior of the agents. However, it makes no assumptions about knowledge of global goals, and so is not dependent on having a global knowledge base or communication that provides that information. The method has been used elsewhere to make model-independent design requirements for economic systems and to determine, in a model-independent way, a simple economic system that cannot be built [Kazadi, 2009]. The advantage of this approach is that it determines general requirements for systems which must be satisfied in order for the global goal to be accomplished. It does not provide details of how to satisfy the requirement, but it has the advantage of creating a class of potential solutions, all of which are guaranteed to yield the global goal. When the requirement is impossible to satisfy, the desired swarm system cannot be built.

In this paper, we examine model independent solutions to swarm problems. We begin by reviewing the general swarm engineering approach and its application to several problems to generate model-independent results. We then examine its application to the generalized lossless flocking problem with limited agents. We develop a swarm requirement for lossless flocking, lossless directed flocking, and lossless flocking in the presence of obstacles. We develop two model swarms based on general considerations, examining their performance on the various flocking problems.

2. General swarm engineering considerations

2.1. Definitions

A good definition of a swarm is given in [Kazadi, 2005] as “a set of interacting agents within a system in which one agent’s change in state can be perceived by at least one other agent and effects the state of the agent perceiving the change. Moreover, the subset must have the property that every agent has at least one state whose change will initiate a continual set of state changes that affects every other agent in the swarm.” This definition communicates the idea that swarms are defined by their communication and interaction. The agents communicate directly or indirectly and interact with one another. Moreover, this communication and interaction is continual, and persists into the future. Using this definition, we can identify swarms as any group of autonomous things that basically control themselves and interact with one another. This includes swarms of robots that interact through explicit communication and/or through indirect communication as in stigmergic or sensed communication. The definition makes no implications about the nature of the agents, which might be stand alone computers, sentient individuals interacting

with but not controlling the other agents, and robots that behave independently and interact with, but are not controlled by, any other agents.

We can further delineate swarms to include those in which communication, information sharing, and processing are handled through one or more agents which might be stationary. These could be high throughput computation centers which generate strategy and combine information gathered from each of the agents. Communication sent out to the agents could assist in the agent achieving the task it is trying to accomplish. This is similar to a central communication and analysis hub of a decentralized activity such as a search or manhunt. The centralized agent could be coordinating the activities without directly controlling the activity. Its limitations could be limited to communication with all parts of the swarm, passing information on and creating a centralized database, and it could be as sophisticated as providing instructions about mission goals and steps one might take to achieve them to the agents. Such swarms are known as *centralized swarms*.

Swarms containing no central location with which each of the agents acting in the field communicate are known as *decentralized swarms*. These have two or more agents, each autonomously acting, and each communicating with another member of the swarm regularly, though no requirement is given as to whether or not the same subgroup of agents communicate with one another. There is also no requirement that decentralized swarms be homogeneous. Certainly, this simplifies the design requirement, but it does not always produce the most efficient swarm possible. For instance, at least one study [Kazadi et al., 2009b] describes a system for which the use of two different behavioral “castes” is more efficient than using only one caste, and separating them into two physical groups of robots improves performance. In this paper, we make no restriction as to the type of agents.

Each agent is assumed to be able to be able to sense and interact with one another and with elements of the environment. We denote the sensory modalities by $\{s_i\}_{i=1}^{N_s}$ where N_s represents the different kinds of sensors available on the robot. Each sensory modality s_i is assumed to produce one of a set of different possible sensor states. Each set of potential sensed states is given by $\{S_i\}$ and the i^{th} modality produces a state such that $s_i \in S_i$. We treat all processing required to generate these states as part of the sensor. That is, if a camera is used to take a shot of a field of vision, a “sensor” would consist of the state of the processed image which might contain a “target” or not along with the approximate location of the target. Therefore, we are abstracting a bit from raw data, and we argue that this does not pose a challenge to the control theory.

We also assume that the agent’s various actuators and emitters (of sound, of radio waves, of light, etc.) allow the agent to interact with the environment in a variety of ways. They might allow the agent to pick up, carry, and put down an object. They might allow the agent to push an object. They might allow the agent to illuminate an object or to generate a high powered electronic noise over a radio frequency. There is no restriction. However, we assume that the agent has the ability

to alter the numerical value of one or more variables $\{x_i\}_{i=1}^{N_a}$ where N_a represents the things the agent can change. As an example, an agent can change the position of a small object by pushing against it. The agent may also increase the general sound level at a particular frequency by emitting an appropriate sound.

In general, the global goal of the swarm is a complex function of the agents' measurables. The trick is to find a way to write the the global goal as a function of the measurables that the agents have access to. If the global goal can be written in terms of the measurables that the agents have access to, then the agents can sense all of the relevant quantities that yield the global goal. The function is depicted in equation (1).

$$P_G = f(s_1, \dots, s_{N_s}). \quad (1)$$

The swarm's goal is then to change the numerical value of P_G to a numerical value consistent with the desired state. Note also, that this implies that the numerical value(s) of P_G in the desired state can be determined. This also implies that the numerical value of P_G that corresponds to the desired final system state does not correspond to any other system states either. In this case, changing P_G to the desired numerical value is enough to create the desired change in the system state. The situation is pictorially represented in Figure 2.1.

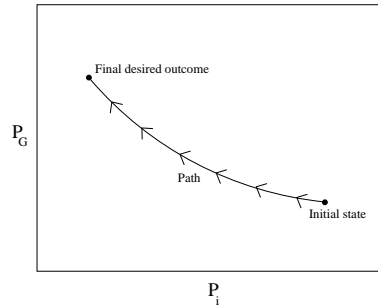


Figure 2.1: This figure illustrates, using a phase space diagram, the evolution of the global property P_G from an initial state to the final desired state as the property x_i is changed.

In some cases, the intermediate values between the initial state $(x_1^i, \dots, x_{N_s}^i)$ and the final state $(x_1^f, \dots, x_{N_s}^f)$ are infeasible. In this case, a more complex strategy must be employed. One option is to determine piecewise strategies with intermediate system designs, and evolve between these states. The situation is illustrated in Figure 2.2.

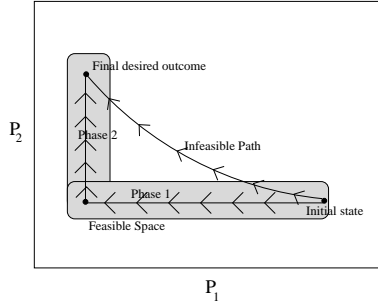


Figure 2.2: This figure illustrates the system evolving between the initial state and the final state through feasible areas, going around the feasible area that covers the most direct route between the initial and the final state.

Creating these way-points and generating ways of making this path occur defines the swarm engineering methodology. Taking the derivative of (1), we obtain

$$\frac{dP_G}{dt} = \sum_{i=1}^{N_s} \frac{\partial f}{\partial s_i} \frac{ds_i}{dt}. \quad (2)$$

Identifying each $\frac{ds_i}{dt}$ with the behavior of the agent, which corresponds to an action of one or more agents on a specific measurable in the system, we see that the goal is to have each of the constituent parts of the swarm change in a particular way. This is relatively straightforward to accomplish, once these things have been identified.

Many studies have centered around the development of systems containing *emergent* properties. These are properties that ostensibly are generated by the agents who themselves have no knowledge of it. Examples include the ability of puck clustering agents to generate a single cluster of pucks and of ant systems to exploit sources of food close to the nest prior to those further from the nest. Mathematically, this means that $\sim \exists s_i$ such that s_i observes property P_G . This is the mathematical definition of emergence.

2.2. Swarm engineered systems from the literature

Several studies have been performed which have generated model independent conditions to be satisfied in order for specific systems to emerge. These studies have determined specific conditions which must be achieved in order to have a system that generates the desired global property. In each case, the specific model used was not as important as the general conditions, and the validation of the condition consisted of constructing a system that satisfied the condition, and thereby generated the desired behavior.

2.2.1. Puck clustering

Perhaps one of the earliest engineered swarm systems utilizing a model-independent analysis comes from [Kazadi, 2002; Kazadi, 2000]. In these studies the general requirements for a puck clustering system were derived. Briefly, a puck clustering

system is any system in which pucks, or generalized materials which can be collected in separate piles, may be moved around by agents. These pucks may be data elements in computing systems, energy units in thermodynamic systems [Kazadi and Webb, 2006], or physical objects in physical systems^a. The two studies focused on how the agents could be parametrized so as to produce one or a specific number of equal sized clusters. Agents were parametrized by functions $h(N)$ and $f(N)$ which were functions of the number of pucks in a cluster and represented the probability that an agent would pick up a puck from a cluster of size N or drop off a puck at the same cluster of size N , respectively. The study determined that if the ratio of the two functions, denoted by the functional $g(N) = \frac{f(N)}{h(N)}$ is a monotonically decreasing function of N , then the system will form a single cluster. The system functions independently of any external control, and uses simple reactive agents with no processing capability and whose behaviors are controlled by simple state machines. This was validated in simulation, and is illustrated in Figure 2.2.

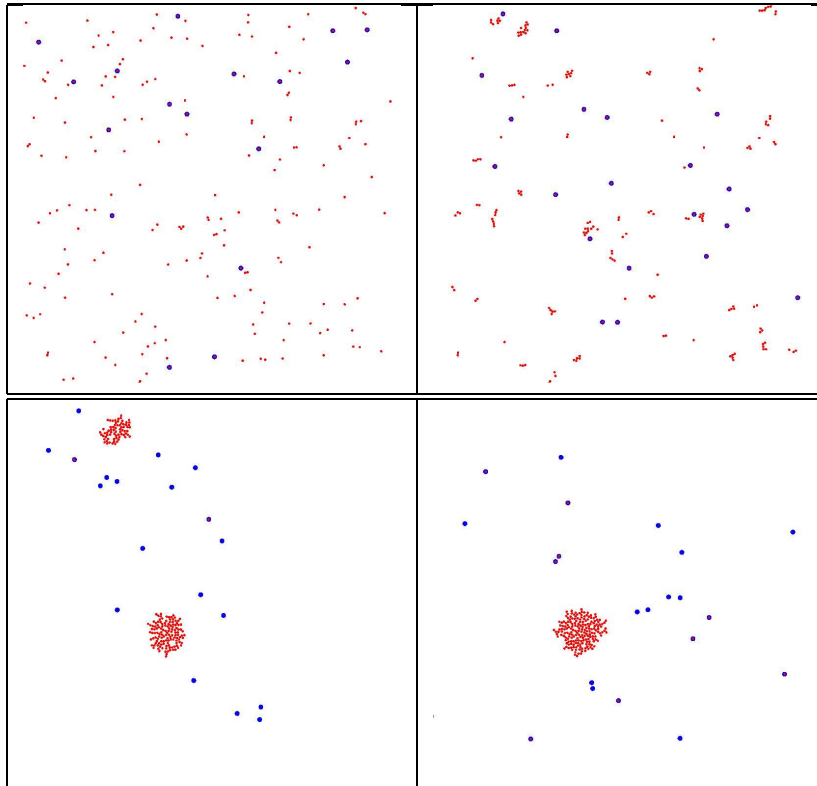


Figure 2.2: This figure illustrates the behavior of puck clusterers in the studies [Kazadi, 2002; Kazadi, 2000] and their ability to generate a single cluster despite having severely limited capabilities.

^aThe name derives from the initial studies done with real robots in which hockey pucks were pushed around a bounded arena by robots.

In this this figure, the agents (blue circles) are moving around the bounded arena picking up and dropping off the pucks (red). The agents are simple reactive agents with no memory, processing, or sophisticated sensing. The probabilities are denoted by the likelihood of an agent to encounter a cluster of a particular size at a given angle. Upon encountering a cluster, the agent denotes the cluster as “big” or “small” and then reacts accordingly, dropping off pucks at big clusters and picking up from small clusters. It can be shown that this behavior generates a monotonically decreasing g functional, thereby yielding the desired behavior.

It can also be shown that if the g functional has a single minimum, the system will generate a number of equal-sized clusters. The clusters will take a size roughly equal to the number of pucks at which g reaches a minimum. The situation is depicted in Figure 2.3.

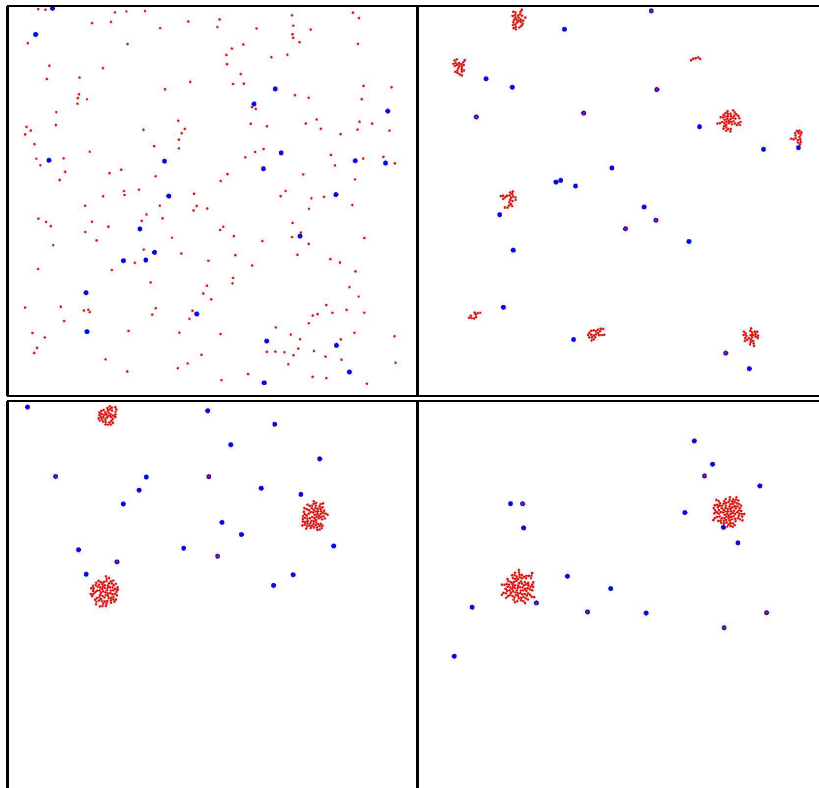


Figure 2.3: This figure illustrates the behavior of puck clusterers in the studies [Kazadi, 2002; Kazadi, 2000] and their ability to generate multiple equal-sized clusters.

Kazadi et al go on to determine that if two functionals g_1 and g_2 represent two different systems, the first system, represented by g_1 , will have a higher clustering efficiency than that of the second system iff $g_1 < g_2$ everywhere.

The significance of these results goes far beyond the simple clustering system described above. The result is model independent, which means that for swarm robotic systems, no matter what we equip the robots with, they must obey this requirement in order for the system to cluster. This gives us an idea of why other studies [Beckers et al, 2001] succeeded in producing a single cluster while others failed [Maris and Boekhorst, 1996]. In generalizing to other systems, such as thermodynamic systems or other data clustering systems, the result still holds.

2.2.2. Forming a hexagon-shaped hexagonal lattice

In [Kazadi et al, 2007] Kazadi et al describe a system of autonomous agents based on artificial physics or physicomimetics. This system was made up of a large number of identical agents, each of which behaved locally. Each agent is assumed to be able to see the other agents in the swarm, and to react to the other agents according to a summed force given as

$$\vec{F}_i = \sum_{j \neq i} \left(\frac{\vec{x}_i - \vec{x}_j}{|\vec{x}_i - \vec{x}_j|} \right) G(|\vec{x}_i - \vec{x}_j|) \quad (3)$$

where each \vec{x}_i represents the position of agent i and G is given as

$$G(x) = \begin{cases} -G_0 & x > D_0 \\ 300G_0 & x \leq D_0 \end{cases} \quad (4)$$

An energy function was defined as

$$E_{config} = \sum_{i,j} G(|\vec{x}_i - \vec{x}_j|) (D_0 - D_{ij}) \quad (5)$$

where D_{ij} represented the deviation from the perfect position.

The goal of the study was to find a way to arrange a specific number of agents (91 in this study) in a perfect hexagonal grid arranged as a hexagon. The difficulty of the task is more readily understood when considering how, without discussing with each other how, 91 people in a room with yardsticks are asked to arrange themselves in a hexagonal grid in the shape of a perfect hexagon. It is a very difficult problem.

It turns out that the physicomimetics system naturally anneals to a low energy state. Unfortunately, this energetic state is not necessarily the lowest energy of (5). As a result, a method of getting the agents to “step down” to the appropriate energy state was required. The paper offered two, developed independently and both of which generated improvements in the overall energy, eventually minimizing it. In both methods, the behaviors made regions of the hexagonal lattice to slide over one another, eventually bringing the extraneous agents into empty spaces. The procedure is illustrated in Figure 2.4.

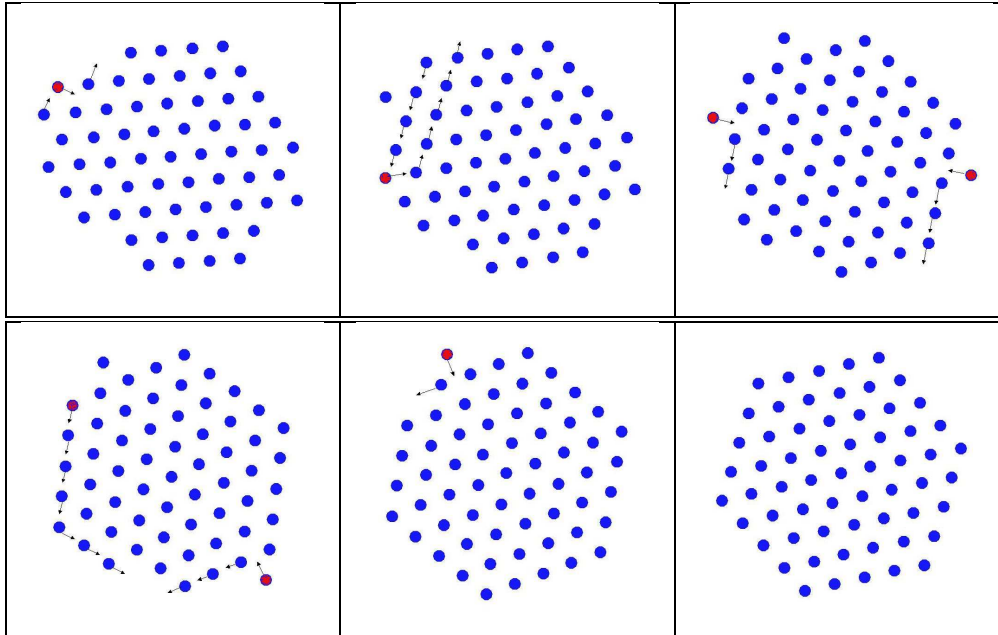


Figure 2.4: These images show how the physicomimetic swarm shuffles individuals along the edges until they fall into a minimal state.

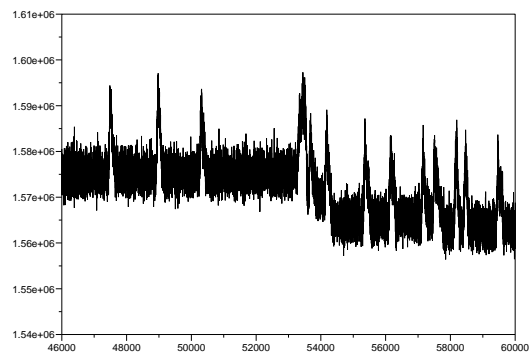


Figure 2.5: The energy of the system “steps down” to the minimal state during these state changes.

As in the previous example, this example demonstrates how the system’s design requirement can be indicated first, and then a swarm can be designed around that requirement. Without the swarm requirement, the problem description offers no indication of how the goal could be achieved. With the swarm requirement, the design question changed from “how can the swarm be made to create this behavior?” to “how can the swarm be made to lower its energy?”. The second one was answered in two different ways.

3. Engineering Flocking

Flocking has been very well studied in the swarm literature, beginning with the original Reynolds' boids [Reynolds, 1987] and continuing on through the years until today. In the natural world, animals flock together for a number of reasons including security, mating, and mutual predatory benefit. The ways in which flocks stay together are varied, though in many natural flocks, they are mediated by sight. Despite the ability of the flocking group to hold their form, to turn and move together, and to make seemingly seamless decisions, there are always cases in which the flock bifurcates, and must somehow come back together.

In most synthetic swarms, we desire a behavior that keeps the swarm together. In the literature, there have been many different approaches to this problem from chorusing [Holland et al, 1999] to methods validated by temporal logic [Winfield et al, 2005]. Yet, these are all based on models of agent interaction. The trick in these studies is to prove that the model chosen behaves as it is intended.

It is generally assumed that flocking is a function that is mediated entirely by the individuals using pairwise information that they can obtain by looking at their closest neighbors. Many authors have examined the problem and generated autonomous local-only information that can be had by using individual sensors. We examine whether the local-only information is insufficient to generate a control algorithm that prevents bifurcation of the swarm.

We seek a *model-independent* requirement for the flock to be maintained in the face of different swarm requirements. Therefore, in order to generate a model-independent condition for flocking, we begin by considering what it takes for the flock to stay together. Once this has been determined, we consider what it takes for the flock to move as a single unit. Finally, explore the requirements for the flock when navigating an obstacle and maintaining its cohesion.

3.1. *Simple lossless flocks*

In generating lossless flocks, the overall goal is to have the agents form a coherent or semi-coherent group that sticks together and loses no members. In order to guarantee that the flock stays unified, the agents must avoid making decisions based on the positions of subsets of the agents. Subsets of the agents have an average position that may differ significantly from the average position of the swarm. If so, decisions about how to move, depending only on the positions of the agents within sensor range of the given agent, may lead the agent continually further away from the center of the swarm. This kind of behavior can be seen in the natural world in flocks of birds which naturally break up into several parts, possibly coming back together later if the birds stay in the same general vicinity. Our strategy is to examine how to keep swarms of agents together once they have formed.

Put another way, each agent samples the flock using its sensors and makes decisions about what to do based on the sampling. In the case that the sampling is reflective of the characteristics of the swarm and the behavior is intended to have

a specific outcome relative to the characteristic, the agent can react appropriately. However, in the case that the sampling is not reflective of the swarm, the behavior may or may not be appropriate.

We start by examining what each of the members of the flock can sense about the other agents in the flock. Each agent has a position; let the set of all positions of the agents is denoted by $\{\vec{x}_i\}_{i=1}^{N_a}$ where N_a represents the number of agents. We assume that the agents have the ability to determine the relative positions of all other agents within a sensory range r_s .

Two groups of agents A and B are detached if \forall agents $x \in A$ and agents $y \in B$, $|\vec{x} - \vec{y}| > r_s$. Let $\vec{x}_{AB} = \vec{x}_m - \vec{y}_m$ where \vec{x}_m and \vec{y}_m are chosen so that $|\vec{x}_m - \vec{y}_m|$ is minimal. We can also let $\vec{x}_A = \frac{1}{N_{aA}} \sum_{i=1}^{N_{aA}} \vec{x}_i$ and $\vec{x}_B = \frac{1}{N_{aB}} \sum_{i=1}^{N_{aB}} \vec{x}_i$ be the average positions of the sets of robots from A and B , respectively. Then, in order for the flock to be truly lossless, it must be the case that any two groups of agents in the flock never become detached. That is, $|\vec{x}_{AB}| \leq r_s$. It can be shown that in order for a group to be connected, any group of N agents must be in sensory contact with at least one other agent outside of the group. If not, the group would be detached from the remainder of the group. As a result, the group of agents must be reactive to the agents not in the group in such a way that the group remains in contact with the agents outside of the group.

Another condition for the flock to be truly lossless is that if an agent or a group of agents becomes detached, the behaviors of the agent(s) re-attaches the group.

That is, whenever $|\vec{x}_{AB}| > r_s$, $\frac{d|\vec{x}_{AB}|}{dt} < 0$.

Let us suppose that A is a singleton group. Let us suppose also that $|\vec{x}_{AB}| > r_s$ for the moment. Let

$$\widehat{x_{A \rightarrow B}} = \frac{\vec{x}_B - \vec{x}_A}{|\vec{x}_B - \vec{x}_A|} \quad (6)$$

and

$$\widehat{x_{A \rightarrow AB}} = \frac{\vec{x}_{AB} - \vec{x}_A}{|\vec{x}_{AB} - \vec{x}_A|}. \quad (7)$$

Further let

$$\phi = \cos^{-1}(\widehat{x_{A \rightarrow B}} \cdot \widehat{x_{A \rightarrow AB}}).$$

Then it is straightforward to show that iff

$$\cos^{-1}\left(\widehat{x_{A \rightarrow B}} \cdot \frac{\dot{\vec{x}}_A}{|\dot{\vec{x}}_A|}\right) \leq \frac{\pi}{2} - \phi \quad (8)$$

and

$$\cos^{-1}\left(\widehat{x_{A \rightarrow AB}} \cdot \frac{\dot{\vec{x}}_A}{|\dot{\vec{x}}_A|}\right) \leq \frac{\pi}{2} - \phi \quad (9)$$

then $\frac{d|\vec{x}_{AB}|}{dt} < 0$.

For larger groups of agents than singleton groups, the condition remains the same. The only caveat is that the entire group must move within these limits in order to recover the remainder of the swarm.

If we combine these requirements, letting the second requirement apply when the agent is “near” or beyond the edge of the swarm, this makes the swarm lossless and capable of reforming after perturbations of the spatial organization. What this requirement states is a simple geometric requirement for the agents to accomplish in order for the swarm to recover from any separation. However, the requirement clearly uses information that is not possible to obtain from simple pairwise sensing; the agent must be able to determine the global position, however noisily, of the swarm in order to reliably implement this behavior. As a result, global information must be available to the agents. Note that this result is independent of the model of the agents. Thus, the result holds for all swarms; lossless swarms can be developed iff individuals have access to global information about relative position.

3.2. Directional coherent flocking

In [Kazadi and Chang, 2008] Kazadi and Chang examined the general requirements of hijacking swarms. They determined that the general requirement for a swarm of agents to be “hijacked” was that the swarm property in question was not actively under the control of the swarm. That is, if a property is affected by the behaviors of the swarm, but the controller does not use this property’s value in determining how the swarm will behave, then the property may be hijacked. In that case, the property can be affected with the smallest influence by any agent, internal or external.

In flocking systems, the swarm condition for lossless flocking centers around the use of the global information of the average *relative* position of the swarm and the agent(s) potentially leaving the swarm. The specific position of the swarm is not part of the control algorithm. This gives us the ability to control the exact position of the swarm *without interfering with the flocking of the swarm*. To see this, let

$$\bar{x} = \frac{1}{N_r} \sum_{i=1}^{N_r} \vec{x}_i. \quad (10)$$

In this case,

$$\frac{\partial \cos^{-1} \left(\widehat{x_{A \rightarrow AB}} \cdot \frac{\dot{\vec{x}}_A}{|\dot{\vec{x}}_A|} \right)}{\partial \bar{x}} = \frac{\partial \cos^{-1} \left(\widehat{x_{A \rightarrow B}} \cdot \frac{\dot{\vec{x}}_A}{|\dot{\vec{x}}_A|} \right)}{\partial \bar{x}} = 0. \quad (11)$$

This verifies that the average position of the swarm has nothing to do with the control algorithm, and we are free to move it. Now, the question is, how can it be moved?

[Kazadi and Chang, 2008; Celikkanat et al, 2008; Soysal and Sahin, 2007] all independently determined similar methods. In the first, the majority of the agents

were designed to simply maintain the swarm in formation. A single “rogue” agent was added to the system whose controller added a directional bias to its direction of movement. This meant that, while the main effect on the single agent was to keep it in the swarm, it tended to move in the bias direction, dragging the entire swarm along with it. Celikkanat et. al. made a similar system in which a bias was added to a group of agents. In this case, the idea was to have the robots not only move, but also to follow a path. Some robots were informed while others were not. The performance of the group improved with the number of “informed” robots.

In both systems, the additional behavior was to have a nudge in a particular direction, which handled the movement in that direction while maintaining the swarm cohesion. Another paper that examines this same problem in detail using different physicomimetics force laws and evolutionary computation-based optimization is [Hettiarachchi and Spears, 2005]. In this paper, by Hettiarachchi and Spears, the swarm moved through an obstacle field made up of relatively small obstacles while maintaining its form. The agents’ controllers added a repulsive term which allowed the robots to avoid a collision when close to the obstacle, making a minimal perturbation of the system.

Let us suppose that we have a coherent swarm in the absence of obstacles. Suppose that a single robot a_0 is informed, and therefore adds a bias to his direction of movement. In this case, the reaction to this bias will be for the swarm to follow the individual, and move to the right. However, we have said that this is true as long as the angles are as given above in equations (8) and (9). Let us understand when this is not so.

Suppose that the informed individual adds a directional bias to the robot’s behavior. That is, the controller calculates a direction and speed to move, and the agent adds \vec{v}_b to the vector. The resultant magnitude and direction are given by

$$\vec{v}_T = \vec{v}_c + \vec{v}_b. \quad (12)$$

This means that

$$\cos^{-1} \left(\widehat{x_{A \rightarrow AB}} \cdot \frac{\vec{v}_T}{|\vec{v}_T|} \right) \leq \frac{\pi}{2} - \phi \quad (13)$$

and

$$\cos^{-1} \left(\widehat{x_{A \rightarrow B}} \cdot \frac{\vec{v}_T}{|\vec{v}_T|} \right) \leq \frac{\pi}{2} - \phi. \quad (14)$$

Inverting these equations obtains

$$\widehat{x_{A \rightarrow B}} \cdot \vec{v}_c + \widehat{x_{A \rightarrow AB}} \cdot \vec{v}_b \leq |\vec{v}_c + \vec{v}_b| \sin(\phi) \leq (|\vec{v}_c| + |\vec{v}_b|) \sin(\phi) \quad (15)$$

and

$$\widehat{x_{A \rightarrow AB}} \cdot \vec{v}_c + \widehat{x_{A \rightarrow AB}} \cdot \vec{v}_b \leq |\vec{v}_c + \vec{v}_b| \sin(\phi) \leq (|\vec{v}_c| + |\vec{v}_b|) \sin(\phi) \quad (16)$$

If we subtract equation (16) from (15) we obtain

$$\left(\widehat{x_{A \rightarrow B}} - \widehat{x_{A \rightarrow AB}} \right) \cdot \vec{v}_c \leq \left(\widehat{x_{A \rightarrow AB}} - \widehat{x_{A \rightarrow B}} \right) \cdot \vec{v}_b \quad (17)$$

This requirement is also independent of the model of the agents. It gives limits on the magnitude and direction of \vec{v}_b that must be obeyed in order to move the flock in a lossless way. Too great an added movement, and the swarm will lose the agent rather than being dragged along by it.

3.3. *Directional coherent flocking in the presence of obstacles*

Flocks of individual agents often times encounter obstacles, big and small. These obstacles can often times be overcome, with the entire flock swarming around a small obstacle. This can be achieved by small perturbations of the agent behaviors which allow agents to avoid obstacles in the way. This works fine when the obstacles are small, as the behaviors need only include an obstacle avoidance behavior (behavior based robots), a repulsion term in physicomimetics work, or other biases that allow the agents to avoid the obstacles. The difficulty occurs when the obstacle is large with relation to the swarm and the agents encountering the swarm.

In the case that the obstacles are physically large with respect to the swarm, it is quite possible for the swarm to encounter the obstacle and be split. For instance, consider the sequence of images in Figure 3.1. This swarm initially is moving to the right, and all agents are 'informed'. Upon encountering a large obstacle, the swarm splits into two separate swarms which then move away from one-another. Each agent is performing their behaviors flawlessly, but the overall action still causes the swarm to pull apart into two parts. This occurs because the obstacle avoidance behavior and movement behavior eventually overwhelm the flocking behavior, essentially pulling it apart a little at a time until the entire swarm has been split.

It can be shown, using a method similar to that in Section 2.2, that

$$\left(\widehat{x_{A \rightarrow B}} - \widehat{x_{A \rightarrow AB}}\right) \cdot \vec{v}_c \leq \left(\widehat{x_{A \rightarrow AB}} - \widehat{x_{A \rightarrow B}}\right) \cdot (\vec{v}_b + \vec{v}_a) \quad (18)$$

where \vec{v}_a represents the additional push due to the obstacle avoidance. This is the additional requirement for the swarm of agents undergoing obstacle avoidance while maintaining a coherent swarm.

3.4. *Swarm engineering*

In the swarm engineering methodology, we begin by examining the general requirements for a swarm to accomplish a specific goal. The next step is to create a model of the swarm and then verify that the model satisfies the general requirement. In this section, we introduce two models. We verify that the models satisfy the requirements in (8) and (9). We then examine whether they satisfy (17) and (18).

3.4.1. *Example of a lossless swarms*

Spears and others have developed and are currently extending the area known as *physicomimetics*. This areas involves the adaptation of physical laws to control laws in swarm robotics. Using a variety of different sensors, the agents can sense and

differentiate the positions of the different agents in the swarm, particularly those that are near by. Using these sensed robots' positions, the robots can determine their own actions. The actions are calculated by putting the relative positions through physical law analogs. As an example, one might imagine determining an update law for the direction an agent will move as

$$\hat{u} = \frac{\vec{f}}{|\vec{f}|}$$

where

$$\vec{f}_j = \frac{1}{N_r - 1} \sum_{i=1}^{N_r} \left[(1 - \delta_{i,j}) \left(A \frac{\hat{r}_{ij}}{|\vec{r}_{ij}|^a} - R \frac{\hat{r}_{ij}}{|\vec{r}_{ij}|^b} \right) \right]. \quad (19)$$

In (19) A is the magnitude of the attraction and R is the magnitude of the repulsion and $a < b$. This means that the two become equal when

$$r = \sqrt[b-a]{\frac{R}{A}}. \quad (20)$$

This is the point where the force changes from attraction to repulsion.

If an agent is outside the main swarm, the deviation of the direction from the average position of the swarm is given by

$$\vec{r}_{AB} \cdot \vec{f}_j = \frac{1}{N_r - 1} \sum_{i=1}^{N_r} \left[(1 - \delta_{i,j}) \left(\frac{A}{|\vec{r}_{ij}|^a} - \frac{R}{|\vec{r}_{ij}|^b} \right) \hat{r}_{ij} \cdot \vec{r}_{AB} \right] \quad (21)$$

As the agent becomes further and further from the main swarm, assuming it has no other agents near it, the sum above collapses to

$$\vec{r}_{AB} \cdot \vec{f}_j = \left(\frac{A}{|\vec{r}_{ij}|^a} - \frac{R}{|\vec{r}_{ij}|^b} \right) \hat{r}_{ij} \cdot \vec{r}_{AB} \quad (22)$$

for some i . In this case, $\vec{r}_{AB} \cdot \vec{f}_j$ is exactly equal to $\alpha \overrightarrow{x_{A \rightarrow B}}$ where α is a scalar multiplier. As a result, the direction of the agent becomes $\pi - \phi$, as required by equations (8) and (9). Therefore, the swarm will be lossless if the speed of the agent when moving away from the swarm does not exceed

$$|\vec{v}| = \frac{r_s}{t_u}, \quad (23)$$

where t_u represents the time between course corrections. This allows the agent to redirect itself toward the nearest agent in the swarm.

In the case that the agent is surrounded by other agents, these agents' behaviors will affect the agent's direction. It cannot be determined *a priori* whether or not these agents' behaviors will redirect the agent in question away from the swarm.

We implemented this swarm in an unbounded two dimensional simulated arena. Each of the 61 agents obeys equation (19) with $a = 1$, $b = 2$, $A = 1$, and $R = 40$. Each agent has a sensor range of 90 units. The agents are initially scattered about

a square area measuring 500 units per side. The agents react to other sensed agents and decide a direction of motion along with a magnitude according to (19). We implement three different scenarios. In the first, we allow the agents to sense one another and move accordingly. In the second, we have an “informed” agent who begins moving to the right, dragging the swarm to which it belongs with it. In the third, we have both an “informed” agent and an obstacle. The three scenarios are depicted below in Figures 3.1, 3.2, and 3.3.

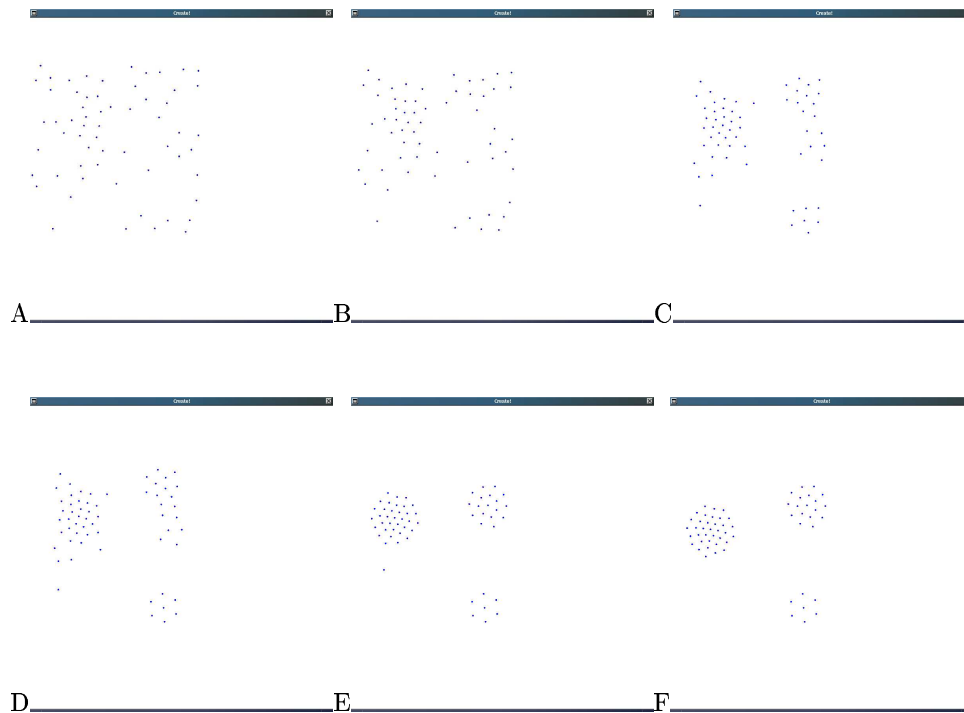


Figure 3.1: These images depict a typical swarm evolution of our unbounded and uninformed model. The agents form tight swarms, but fail to entirely merge.

As alluded to above, the sub-swarms provide enough of a “pull” to bring in agents near their center, essentially cutting off the central agents from one another. While the swarms do not subsequently lose agents, it is clear that, as discussed above, the lack of global information yields sub-swarms, which themselves are capable of retaining agents. The simulation generates a single swarm an average of 19% of the time, due to sensor limitations and sparse distributions over the space. Also 19% of the time, agents are left out of the swarm entirely, randomly ending up initially where they have no contact with any other agents.

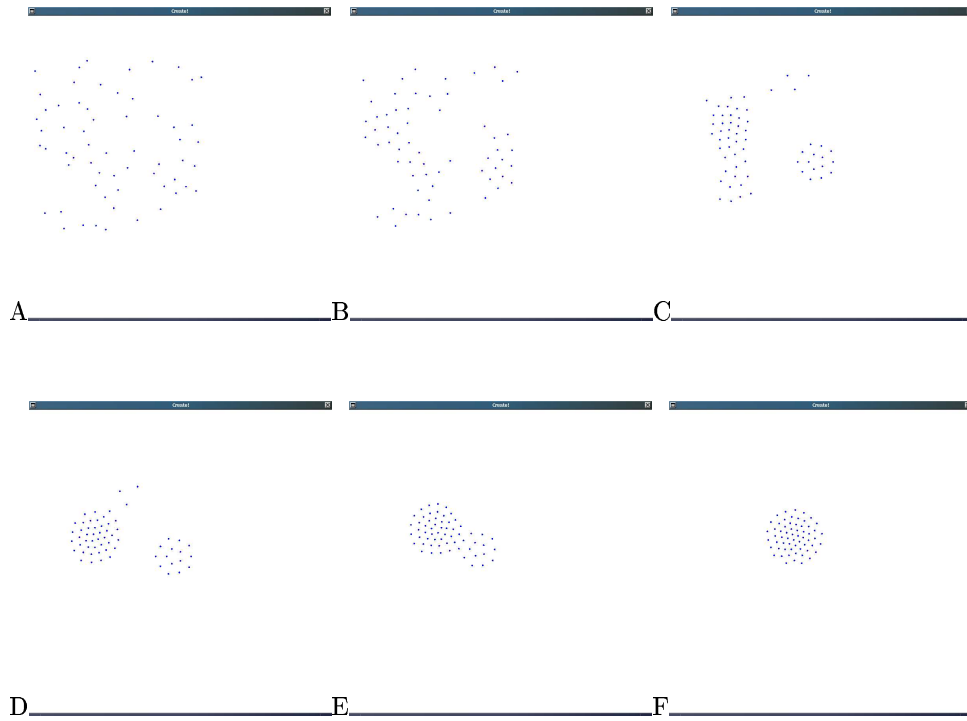
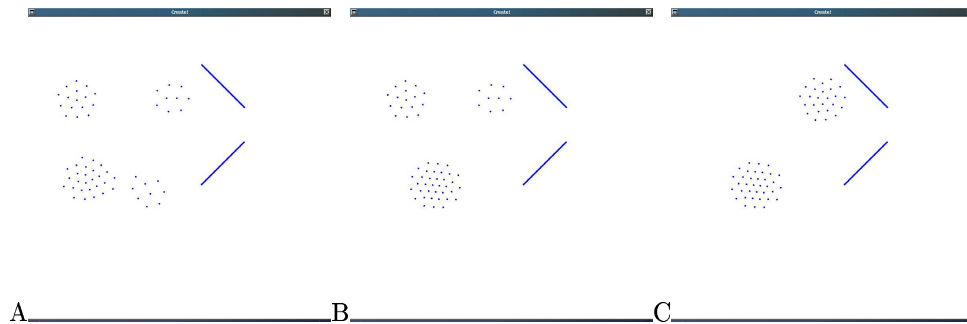


Figure 3.2: These images depict a typical swarm evolution of our unbounded model with a single informed agent. The agent pulls the swarm to which it belongs the right once it starts moving.

In the second scenario, the agents have an “informed individual within. As shown in Figure 3.2, once the “informed” agent becomes active, the entire swarm to which it belongs begins moving to the right. The agent essentially hijacks the swarm, moving it to a new global position since the swarm does not regulate its global position.



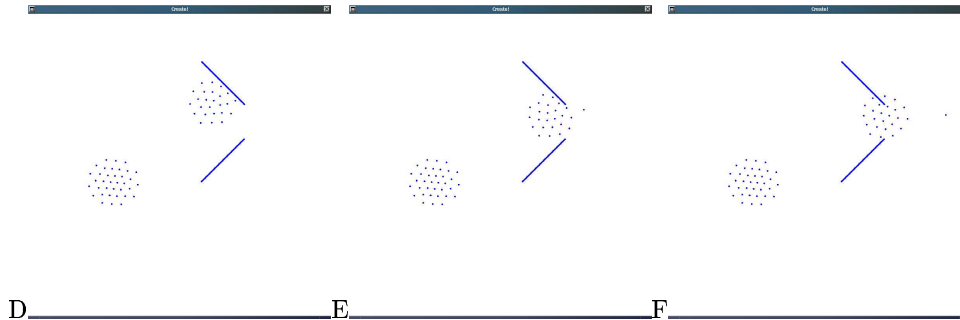


Figure 3.3: These images depict a typical swarm evolution of our unbounded model with a single informed agent as the swarm attempts to go through a large obstacle. The agent pulls the swarm to which it belongs the right once it starts moving, but then becomes separated after the swarm gets stuck at the obstacle.

In the third scenario, the “informed” agent becomes active and moves its swarm to the right. On the right, it encounters a large obstacle. The swarm is too large to get through the obstacle intact, and the “informed” agent is lost when the swarm gets stuck.

3.4.2. Second Example (“Quark Model”^b)

The previous example is somewhat unsatisfying because, while it satisfies the swarm requirements for a single agent separating from the group, it can fail to do so if the agent separates from the group along with other agents. As described above, this represents the case that the sampling of the group measurable is inconsistent with the actual measurable. In such a case, the decision made will not be the desired one.

A second example of a physicomimetic system satisfying the requirement of equations (8) and (9) is one in which the agents are most strongly affected in their decision-making process by the agents in their sensor range that are furthest away. In this case, the direction of motion is determined as follows

$$\vec{f}_j = \sum_{i=1}^{N_r} [(1 - \delta_{i,j}) ((a(|\vec{r}_{ij}|) - b(|\vec{r}_{ij}|)) \hat{r}_{ij})] \quad (24)$$

where a is a monotonically increasing real function with a vertical asymptote at $x = r_s$ and b is a monotonically decreasing real function with a vertical asymptote at $x = 0$. In this case, the two limits behave like an infinite potential well with curved walls. In the physical analog, the particle with finite energy remains indefinitely trapped between 0 and r_s .

^bThe name comes from the tendency of quarks to require more force to pull them further and further apart, eventually resulting in the spontaneous generation of new quarks. The new quarks allow for the existence of quarks in pairs and triplets.

In our system, the vector \vec{f}_j gives the *direction* of motion rather than the magnitude of motion. We assume that the agent moves at its top speed in all cases, and the direction of motion can be given by $\vec{u}_j = \frac{\vec{f}_j}{|\vec{f}_j|}$. It is straightforward to see that small groups of agents removed from the main group move directly toward the main group, and the main group responds by moving toward the small group. This method clearly satisfies equations (8) and (9) when approaching r_s . As a result, we expect it to generate coherent groups, despite the possibility of the group having a relatively changing structure.

In order to validate this design, we simulate a swarm of circular agents in a two-dimensional plane. As before, the agents are capable of determining the position of other agents within a sensor range r_s . Beyond this range, the agents cannot see others. In our system,

$$\vec{f}_j = \sum_{i=1}^{N_r} \left[\cot \left(\frac{\pi |\vec{r}_{ij}|}{r_s} \right) \hat{r}_{ij} \right] \quad (25)$$

which satisfies the definition given in (24). The sum of the different vectors produces a summed vector, which the agents then use to determine a direction of motion, which they head in at their maximum speed. As before, the sensor range is 90 units, and agents are randomly dispersed around a 500 by 500 unbounded arena. We begin by examining the case where there are no obstacles or “informed” individuals.

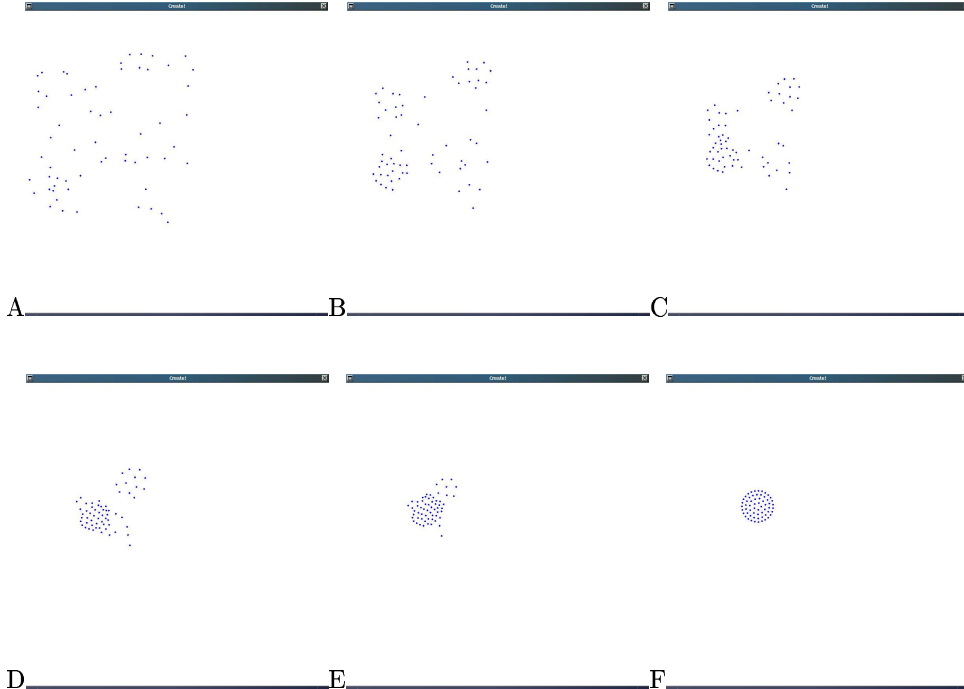
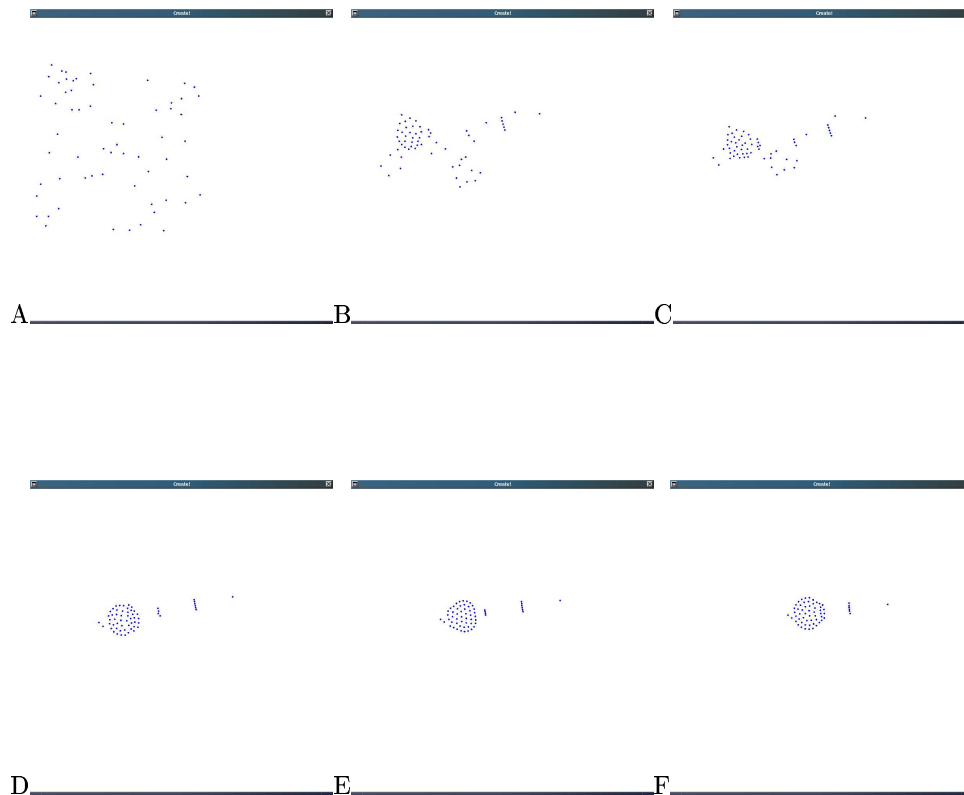


Figure 3.4: These images depict a typical swarm evolution of our unbounded uninformed quark model. The swarm contracts efficiently and merges to form a single symmetric swarm.

We depict an uninformed swarm contracting from random positions in Figure 3.4. The swarm quickly forms swarming centers, or areas of high density which might lead to the formation of a sub-swarm. This also occurred in the inverse force law model. However, the edges of each of the centers intermingles, and the centers get pulled together. When this is completed, the swarm forms a single symmetrically organized group. The swarm succeeds in forming a single group 91% of the time. 9 other agents when the simulation begins.

We continue with an examination of the behavior of the swarm under the action of a single informed individual moving to the right, almost immediately when the simulation starts. A typical run is depicted in Figure 3.5.



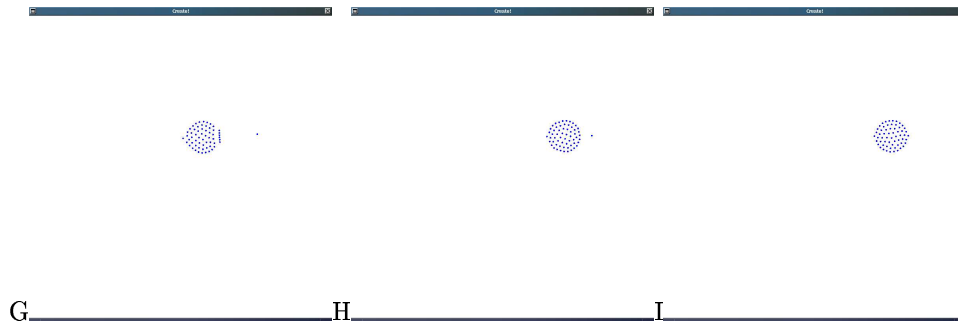
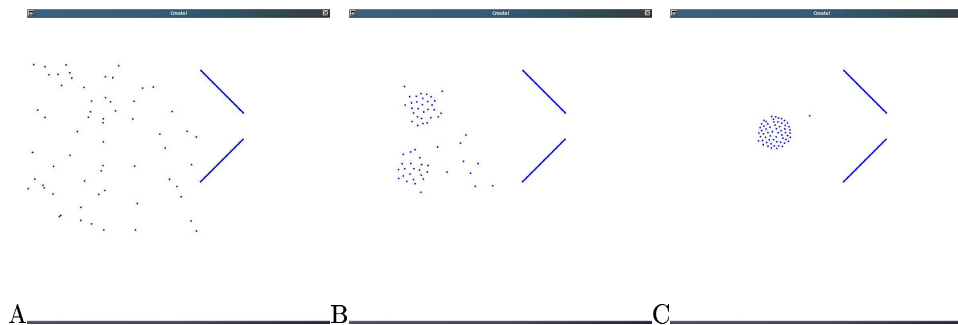


Figure 3.5: These images depict a typical swarm evolution of our unbounded informed quark model with a single informed individual. The swarm begins moving as it contracts. The informed individual moves in its preferred direction “pursued” by the bulk of the swarm, which eventually overtakes and absorbs the individual.

In the inverse force law model above, we saw that the informed individual began moving after the initial swarming event had occurred. This was needed because without the swarm already in existence, the agent almost invariably got lost. In this model the agent begins moving right away. However, the swarm congeals behind it, connected by a few intermediate agents. The bulk of the swarm becomes “reeled in”, catching up to the intermediate agents and absorbing them before catching the “informed” individual and absorbing it.

We also examine the behavior of the swarm in the presence of an obstacle. The inverse force law model above was unable to retain the informed agent because the pull on the swarm was insufficient to compress it enough to allow it to pass through the obstacle. We examine the same scenario in the quark model in Figure 3.6.



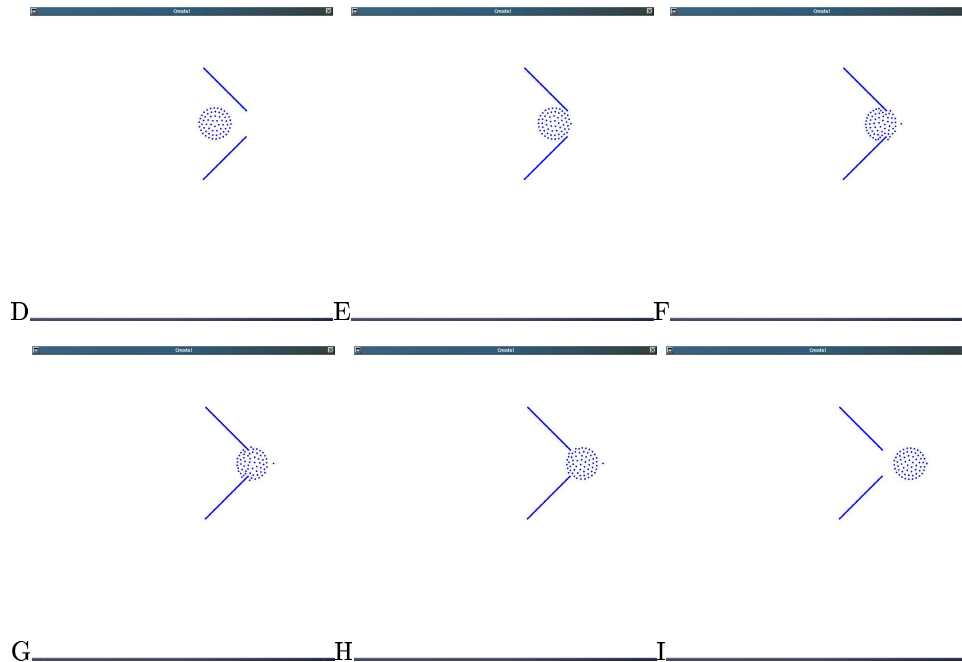


Figure 3.6: These images depict a typical swarm evolution of our unbounded informed quark model with a single informed individual and an obstacle. The swarm begins moving as it contracts. The informed individual moves in its preferred direction “pursued” by the bulk of the swarm, which eventually overtakes and absorbs the individual. When the swarm encounters the obstacle it temporarily loses the “informed” individual while it squeezes through, after which it recaptures the “informed” individual.

The behavior of the quark model in the presence of obstacles very clearly depicts the goal of the lossless swarm. Not only does it not lose an individual leaving the swarm, but it is capable of getting through obstacles whose apertures are significantly smaller than the swarm size. Note that the aperture in this obstacle is 25% smaller than that in used in the inverse square law model, discussed above. The swarm, which is made up of autonomous purely reactive agents, manages to get through intact, and resume its normal behavior on the other side of the obstacle.

4. Discussion and Conclusions

In general, the value of a swarm engineering approach lies in its ability to allow the development of robust models of swarm interaction that do precisely what is desired and expected. Not only might one want this to occur, but also to occur in such a way that quantitative predictions can be made as to the behavior of the swarm, and that things such as robustness and correctness can be measured and predicted.

Our method of swarm engineering centers around the development of model-independent requirements for the swarm system. These are requirements for the correct creation of a swarm system that do not depend on the specific model that

is being employed. In the puck clustering example given above, this translates to generating a requirement for the g functional that yields the desired clustering effect. This does not depend on how this requirement is implemented, and therefore represents a requirement that may be satisfied in many different ways. The physicomimetics swarm which generated a perfect hexagon was another example of the same principle. The lowering of the energy function was a requirement in order to reach the minimal state. How that was accomplished was the task for the designers of the system, and two different methods were put forth in that study.

We argue that creating a swarm system is significantly easier when the requirements for the correct system design are already known. It has been argued elsewhere that the lack of a complete solution using this methodology limits its applicability. We differ with this interpretation. We do not believe that the lack of a pre-formed solution allows maximal flexibility while the guidance of a set of requirements allows the examination of how a particular problem can be solved within a framework. This framework provides its own set of clues as to how one might approach the problem. For instance, with the clustering problem, one needs to create a behavior that decreases the g as the number of pucks increases. Knowing this provides a significantly easier problem to solve than simply needing to get the pucks clustered. In the case of the hexagonal swarm, the author is unaware of any other solution to this problem, and frankly has no idea how it might be approached otherwise. It is completely counter-intuitive to think that a solution (which was one of the two found) is to temporarily and randomly alter the equilibrium distance of agents at the corners of the swarm. Yet, being able to think of this within a framework allowed for the generation of the system.

In the present work, we have generated an entirely new physicomimetic system, which we've called the quark system. This system consists of dispensing with the traditional physicomimetic inverse square laws and utilizing a force law that is calculated with the cotangent function. The cotangent function naturally has two vertical asymptotes at zero and π . We used these to determine the force between individuals a given distance apart, scaled for the sensory range of the agents. The results were very unexpected, generating almost organic swarms whose first rule was to lose no agent. The robustness of this system far exceeded its inverse square law predecessor. Yet, it was the development of that model-independent requirement that allowed us to begin to think about what was really needed. We realized that the swarm had to keep track of the agents that were *most distant* first, rather than those that were closest. The vertical asymptotes were the result. The use of the cotangent function was a convenient way to implement it.

This paper has contributed to the growing work on how swarm problems may be approached. Equally important is the question of why swarms need to be used in the first place. We expect this to be an area of future research.

References

- R. Beckers, O. Holland, and J. Deneubourg. *From local actions to global tasks: stigmergy and collective robotics*. From R. Beckers, O. Holland, and J. Deneubourg, eds., **Proceedings of the 2001 IEEE Systems, Man, and Cybernetics Conference**, Tucson, Arizona, USA, October 2001.
- H. Celikkanat, A. Turgut, and E. Sahin. *Control of a Mobile Robot via Informed Robots*. **Proceedings of the 9th International Symposium on Distributed Autonomous Robotic Systems**, 2008.
- V. Crespi, A. Galstyan and K. Lerman. *Top-down vs bottom-up methodologies in multi-agent system design*. **Autonomous Robots**, 24(3), pp. 303-313, 2008.
- V. Gazi and K. M. Passino. *Stability Analysis of Swarms*. **IEEE Transactions on Automatic Control**, 48(4), pp. 692-697, 2003.
- S. Hettiarachchi. and W. Spears. *Moving swarm formations through obstacle fields*. **International Conference on Artificial Intelligence**, Volume 1, pp. 97-103, CSREA Press., 2005.
- O. Holland, C. Melhuish, and S. Hoddell. *Chorusing and controlled clustering for minimal mobile agents*. **Robotics and Autonomous Systems**, 28, pp. 207-216, 1999.
- S. Kazadi. **Swarm Engineering**. PhD thesis, California Institute of Technology, 2000.
- S. Kazadi, A. Abdul-Khaliq, and R. Goodman. *On the Convergence of Puck Clustering Systems*. **Robotics and Autonomous Systems**, 38 (2), 93-117, 2002.
- S. Kazadi. *On the Development of a Swarm Engineering Methodology*. **Proceedings of IEEE Conference on Systems, Man, and Cybernetics**, Waikoloa, Hawaii, USA, pp.1423-1428, October 2005.
- S. Kazadi and M. Webb. *Swarm Pumping*. Working Paper 3, Jisan Research Institute, 2006.
- S. Kazadi, J. R. Lee, J. Lee. *Artificial Physics, Swarm Engineering, and the Hamiltonian Method*. **Proceedings of the World Congress on Engineering and Computer Science 2007**, San Francisco, California, USA, October 24-26, p. 623-632, 2007.
- S. Kazadi and J. Chang. *Hijacking Swarms*. **Proceedings, 11th IASTED International Conference on Intelligent Systems and Control 2008**, Orlando, Florida, USA, November 16-18, p. 228-234, 2008.
- S. Kazadi. *Model independent economics based on swarm engineering*. Invited book chapter, in press, 2009.
- S. Kazadi, J. Yang, J. Park, and A. Park. *Orthogonality and optimality in non-pheromone mediated foraging*. Submitted, 2009.
- M. Maris and R. Boekhorst. *Exploiting physical constraints: heap formation through behavioral error in a group of robots*. **Proceedings of IROS '96**, Osaka, Japan, November 4-8, 1996.
- C. Reynolds. *Flocks, herds, and schools: a distributed behavioral model*. **Computer Graphics**, 21(4), pp. 25-34, 1987.
- O. Soysal and E. Sahin. *A Macroscopic Model for Self-organized Aggregation in Swarm Robotic Systems*. **Swarm Robotics: Second SAB 2006 International Workshop**. Sahin E., Spears W., and Winfield A.F.T (editors), 4433, Lecture Notes in Computer Science, pages 27-42, Berlin Heidelberg, Springer, 2007.
- A. Winfield, J. Sa, M. Gago, C. Dixon, and M. Fisher. *On formal specification of emergent behaviours in swarm robotic systems*. **International Journal of Advanced Robotic Systems**, 2(4), pp. 363-370, 2005.

Biography

Dr. Sanza Kazadi is the President and chief scientist at the Jisan Research Institute. Dr. Kazadi earned his PhD from the California Institute of Technology in 2000. Dr. Kazadi moved to the Jisan Research Institute in 2000 to serve as the chief scientist. Dr. Kazadi's research interests include swarm engineering and renewable and low carbon footprint technologies. Dr. Kazadi may be reached at skazadi@jisan.org.