On the Application of the Hamiltonian Swarm Design to Low Dimensional Minimalistic Swarms

Rohit Mittapalli and Ankit Agarwal under Dr. Sanza Kazadi

Abstract

We explore the applications and development of the Hamiltonian Method of swarm design on low dimensional swarm in order to tackle the swarm design problem. This problem includes questions such as the necessity of swarm capability, the various types of technologies needed for the swarm to function, and the minimal number of agents to use. In order to accomplish this task, we utilize the Hamiltonian Method of Swarm Design in order to describe the changing states of the swarm. We develop a partial derivative gradient matrix that is used to identify the technologies necessary to complete the task. We then utilize this technology matrix by applying it to a simple problem in swarm design by modeling a two agent and three agent gradient based swarm maximum search tasks. We then utilize the results of this method in order to better analyze the more conceptual questions of swarm design.

1 Introduction

Swarms are an innovative approach to coordinate multi-agent system to performe tasks with emergent behavior allowing the swarm to complete a task that the individual agents would not be able to do by themselves [1]. A simple natural example is grasshoppers, which incredulously all turn into locusts once a concentration is reached. By themselves, this is impossible, but with a group, this kind of transformation comes naturally. This swarm behavior illustrates one of the many types of complex behavior that swarms can exhibit from relatively simple agents.

The term swarm intelligence was invented to describe the study of multiple-agent complex swarms. As swarm intelligence started to grow as a field, studies gravitated towards swarm capability rather than intelligence of the performed behavior. For example, researchers Spears and Gordon [4, 6, 8] researched physicomimetics. Another study [5] exploited ant-colony algorithms to solved network-based problems such as the Traveling Salesman Problem or the Job-Scheduling problem.

These studies worked to obtain precise, measurable, and controllable results. These studies do not contribute to developing artificial intelligence through swarm agents. As a result, a new term Swarm Engineering has been coined to encompass such studies [10]. To a swarm engineer, the primary objective of his or her study is to design a swarm that accomplishes a well-defined task. Unfortunately, there does not to exist a well-defined theory on how to engineer a swarm design.

We need to develop a variety of questions that need to be answered in order to develop a working theory that will may assist any swarm engineer to solve his or her problems. One such theory presented is the Hamiltonian Method of Swarm Design (HMSD) [2]. This method is useful for calculation purposes, but it does not address the more fundamental questions: if a swarm is necessary or the minimal amount of agents. In this paper, we first discuss some important developments in swarm theory and introduce an

the Technology Matrix. We then apply our method of swarm design to a problem in swarm engineering known as the Maximum Search Problem. Finally, we confirm our results in the previous section by creating two different swarms that are applicable in the Maximum Search Problem and use those results to draw conclusions about swarm theory in general.

2 Defining a General Swarm Methodology

We will define these problems generally through phase space. First we will look at the Hamiltonian Method of Swarm Design as developed by [2]. We define three components of swarm design: global parameters, $P_1, P_2....P_i$, local measurables, $s_1...s_l$, and agents, $\vec{a_1}, \vec{a_2}...\vec{a_k}$. Global parameters accurately represent how close the swarm is to its final state. The number of global parameters determines the dimensionality of the swarm. For example, if the problem only requires one global measurable, then it is dubbed a *one-dimensional swarm*. Next, local measurables are variables that each agent of the swarm can take in and measure. Finally, we have the number of swarm agents that exist within the system. Each agent has a set of attributes that describe its state within the system. Using these components as guides, we may now define a multi-variable function that can describe the behavior of the system.

2.1 The Technology Matrix

We will first begin with the global parameters. Let $M(\vec{a})$ take in values of a vector $\vec{a} = (a_{1,1}, a_{1,2}...a_{1,n}, a_{2,1}...a_{k,n})$. Where $a_{i,j}$ represents the j-th attribute of the i-th agent in the swarm. Attributes may be any mathematical descriptor of the agent such as position, concentration, etc.. Additionally, let M output a set of global measurables $\vec{P} = (P_1, P_2...P_i)$. Therefore, M is a function that maps from the attributes of the agents to the global parameters. Since we wish to describe this function as it changes over time, we wish to take it derivative in respect to time. This can be represented as follows.

$$\frac{dM}{dt} = \mathbf{D}\vec{P} \times \Delta\vec{a} = \begin{bmatrix} \frac{\partial P_1}{\partial a_{1,1}} & \cdots & \frac{\partial P_1}{\partial a_{2,1}} & \cdots & \frac{\partial P_1}{\partial a_{k,n}} \\ \frac{\partial P_2}{\partial a_{1,1}} & \cdots & \frac{\partial P_2}{\partial a_{2,1}} & \cdots & \frac{\partial P_2}{\partial a_{k,n}} \\ \frac{\partial P_3}{\partial a_{1,1}} & \cdots & \frac{\partial P_3}{\partial a_{2,1}} & \cdots & \frac{\partial P_3}{\partial a_{k,n}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial P_i}{\partial a_{1,1}} & \cdots & \frac{\partial P_i}{\partial a_{2,1}} & \cdots & \frac{\partial P_i}{\partial a_{k,n}} \end{bmatrix} \begin{bmatrix} \frac{da_{1,1}}{dt} \\ \vdots \\ \frac{da_{2,1}}{dt} \\ \vdots \\ \frac{da_{k,n}}{dt} \end{bmatrix}$$

The matrix $\mathbf{D}$ above represents the Jacobian Matrix of function M. This matrix summarizes how the function M changes with respect to each swarm agent attribute. In other words, it describes how the swarm agents can change the local variables. Therefore, each term in this Jacobian represents a different technology that the swarm agent needs to apply in order to change the global parameters. Therefore, we will define this matrix to be known as the Technology Matrix.

2.2 Local Variables

We now wish to examine how these attributes change with respect to local variables. Local variables determine how the agent will behave in response to new information. This behavior can be represented as a change in the attributes of the agent through phase space. In other words, the change in each of the

attributes over time can be written as a function of local measurables measured by each agent. We will denote these as a vector $\vec{s_{a_i}} = (s_1 ... s_l)$. We will let $\vec{f_j}$ be the the function that maps the local variables to the change in the value of the attribute j over time. In other words, $\frac{da_i j}{dt} = \vec{f_j}(s_{a_i})$. We may substitute this into the equation above.

$$\frac{dM}{dt} = \mathbf{D}\vec{P} \times \Delta\vec{a} = \begin{bmatrix} \frac{\partial P_1}{\partial a_{1,1}} & \cdots & \frac{\partial P_1}{\partial a_{2,1}} & \cdots & \frac{\partial P_1}{\partial a_{k,n}} \\ \frac{\partial P_2}{\partial a_{1,1}} & \cdots & \frac{\partial P_2}{\partial a_{2,1}} & \cdots & \frac{\partial P_2}{\partial a_{k,n}} \\ \frac{\partial P_3}{\partial a_{1,1}} & \cdots & \frac{\partial P_3}{\partial a_{2,1}} & \cdots & \frac{\partial P_3}{\partial a_{k,n}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial P_i}{\partial a_{1,1}} & \cdots & \frac{\partial P_i}{\partial a_{2,1}} & \cdots & \frac{\partial P_i}{\partial a_{k,n}} \end{bmatrix} \begin{bmatrix} \vec{f_1}(s_{a_1}) \\ \vdots \\ \vec{f_1}(s_{a_2}) \\ \vdots \\ \vec{f_n}(s_{a_k}) \end{bmatrix}$$

We now have an equation that describes the global change of the swarm system as a function of the local variables for a certain time step.

2.3 Determining if a Swarm is Necessary

We wish to examine the necessity of swarms which require more than one agent. Since this equation works equally well with simply one agent, we wish to examine under what conditions a swarm is necessary. Generally this is done in a case by case basis, however, we can apply a general methodology.

1. Determine the set of all technologies needed for the task to be accomplished.
2. Determine whether or not all the technologies exist and can be practically implemented on a single agent.
3. If the above condition is satisfied, then a swarm is not necessary. Otherwise, a swarm is necessary to accomplish the task.

2.4 Discrete Approximations of The Theory

Often agents work discretely rather than continuously, we wish to find a way to represent the state of the system as a discrete approximation. To do this, let the initial state or starting state of the system be equivalent to $\vec{P_s}$ and let the final state be equivalent to P_f. In general we will represent the state of the system at time t as $\vec{P_t}$. Additionally, we will describe the set of local variables available to the system at time t as $\{s\}_t$. Then, we can describe the change as time goes on as follows.

$$\vec{P}_{t+1} = \vec{P_t} + (\mathbf{D}\vec{P}|_{\vec{a}} \cdot \Delta\vec{a}|_{\{s\}_t})\Delta t.$$

This equation can be applied over an over again starting from $\vec{P_t} = \vec{P_s}$ until the value of $\vec{P_t}$ is sufficiently close to the $\vec{P_f}$.

3 Application to Maximum Search Problem

The Maximum Search Problem is a problem in swarm design where agents find the maximum of a field that increases to the maximum from all points. In the real world this would be analogous to a swarm finding a beacon of light or a plume source based on its vector field.

3.1 Applying the Hamiltonian Method

We will start by defining a global parameter P that accurately measures the state of the swarm system. Let us assume that the maximum point can be defined as a vector $\vec{m} = (m_1, m_2...m_n)$ in phase space. Then, let us define the k-th agent's position of l total agents as a vector $\vec{a_k} = (a_{k,1}, a_{k,2}, a_{k,3}...a_{k,n})$. We may develop the equation for the global parameter as follows:

$$P = \sum_{k=0}^{l} ||\vec{a_k} - \vec{m}||^2$$

Note that only when this function is 0 is the problem been satisfied. Splitting up into vector components we obtain:

$$P = \sum_{k=0}^{l} \sum_{i=0}^{n} (a_{k.i} - m_i)^2$$

This function describes the distance each agent is from the maximum point that it needs to reach. We wish this function to approach its minimal value at 0 in which case we will consider the swarm to have reached its goal. In order to do this, we want the change in the global parameter to be negative. We now wish to describe the change in P. Taking the time derivative of both sides, we obtain

$$\frac{dP}{dt} = \sum_{k=1}^{l} \sum_{i=1}^{n} 2 \cdot (a_{k,i} - m_i) \frac{da_{k,i}}{dt} \leq 0$$

In order for this equation to always be less than or equal to zero, we must ensure that all the terms inside the summation are less than or equal to zero. Therefore $\frac{da_{k,i}}{dt}$ must have the opposite sign to $(a_{k,i} - m_i)$. In order to calculate a possibility for this value, we will look at the difference quotient of vector field. Let the function be $I(\vec{x})$ be the function maximize at $\vec{x} = \vec{m}$. We will denote this as the intensity function. Then, since $I(\vec{m})$ is maximized, $I(\vec{m}) - I(\vec{a_k}) \geq 0$. Dividing by the value of $(m_i - a_{k,i})$ we obtain a difference quotient that is approximately equal to $\frac{\partial I}{\partial a_{k,i}}$. In fact, by the nature of our chosen function, this partial will always have the same sign as the difference quotient. This difference quotient is made up of a positive number divided by the negative of our desired quantity. Therefore, this difference quotient has the opposite sign to $(a_{k,i} - m_i)$. Therefore, we may set $\frac{da_{k_i}}{dt}$ equal to the $\frac{\partial I}{\partial a_{k,i}}$ to obtain a viable function.

$$\frac{dP}{dt} = \sum_{k=1}^{l} \sum_{i=1}^{n} 2 \cdot (a_{k,i} - m_i) \frac{\partial I}{\partial a_{k,i}} \leq 0$$

Therefore, the swarm needs to determine the partial derivatives with respect to each attribute for the intensity function. In other words, it needs to determine the gradient of the intensity function or ∇I as a function of its local measurables.

3.2 Using the Technology Matrix

We already know that each agent must be able to determine the gradient. But, let us look back at the equation above and utilize the technology matrix to determine other technologies. Since there is only one global measurable, P, the matrix will simply be a column vector.

$$\frac{dP}{dt} = \mathbf{D}\vec{P} \times \delta\vec{a} = \begin{bmatrix} \frac{\partial P_1}{\partial a_{1,1}} & \dots & \frac{\partial P_1}{\partial a_{2,1}} & \dots & \frac{\partial P_1}{\partial a_{k,i}} \end{bmatrix} \begin{bmatrix} \vec{f}_1(s_{a_1}) \\ \vdots \\ \vec{f}_1(s_{a_2}) \\ \vdots \\ \vec{f}_i(s_{a_k}) \end{bmatrix}$$

$$= \begin{bmatrix} 2 \cdot (a_{1,1} - m_1) & \dots & 2 \cdot (a_{2,1} - m_1) & \dots & 2 \cdot (a_{k,i} - m_i) \end{bmatrix} \begin{bmatrix} \frac{\partial I}{\partial a_{1,1}} \\ \vdots \\ \frac{\partial I}{\partial a_{2,1}} \\ \vdots \\ \frac{\partial I}{\partial a_{k,i}} \end{bmatrix}$$

We see that the expressions $2 \cdot (a_{k,i} - m_i)$ are equivalent to $\frac{\partial P_1}{\partial a_{k,n}}$. Therefore, the agent must have the ability to change these values over time. In other words, the agent must have the ability to move itself through phase space and directly change its attributes. The next thing we see is that the swarm must have the ability to accurately measure the gradient at its position in phase space as a function of some set of local variables. This is another technology that the swarm must be capable of using.

There are many ways in which to calculate the gradient in terms of its local variables. In the next sections, we discuss two types of swarms that may be used to solve the Maximum Search Problem and show how they both use the gradient to their advantage. For simplicity, rather than using the general maximum search problem, we simplify the problem down to the two dimensional case.

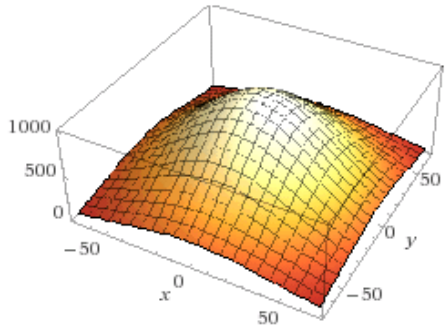
4 Modeling the Two Dimensional Maximum Search Problem

In the two dimensional maximum search problem, a swarm attempts to triangulate the global maximum. The problem is considered a two dimensional maximum search problem because each agent has two attributes x position and y position in the phase space. We model this phase space using a $x-y$ Cartesian plane where the x represents the x position in phase space and y represents the y position in phase space. Intensity functions report intensities given an x and y. In order to thoroughly model this problem, we use 3 different intensity functions.

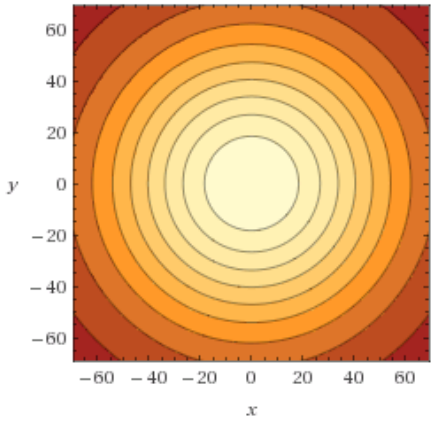
1. The Two Dimensional Normal or Gaussian Distribution

$$I(x, y) = Ae^{-(\frac{(x-x_0)^2}{h} + \frac{(y-y_0)^2}{k})} \tag{1}$$

Where A, h, k, x_0, y_0 are constants that do not affect the problem. Our simulation used $A = 1000$, $(x_0, y_0) = (0, 0)$, and $h = k = 3200$.



(a) 3D Graph of the Intensity Function (Eq.1)



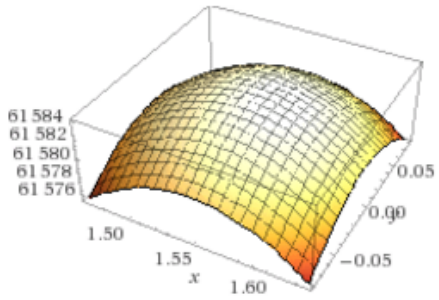
(b) Contour Map of the Intensity Function (Eq.1)

Figure 1: Representations of Intensity Function (Eq.1)

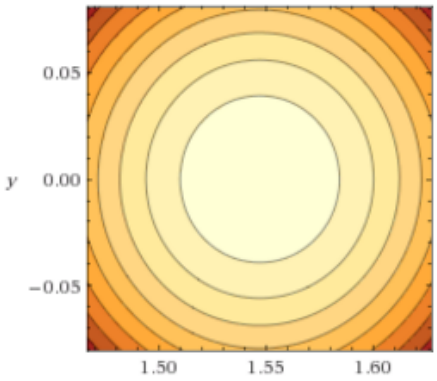
2.

$$I(x, y) = A \cdot \left(\frac{\tan^{-1}(k\sqrt{x^2 + y^2} + a)}{\pi} + b \right) \frac{(x^2 + y^2 - c)(x + d)}{e} \tag{2}$$

Where $A = 1000$, $k = 200000$, $a = 10$, $b = 0.5$, $c = 100$, $d = 30$, $e = 50$.



(a) 3D Graph of the Intensity Function (Eq.2)



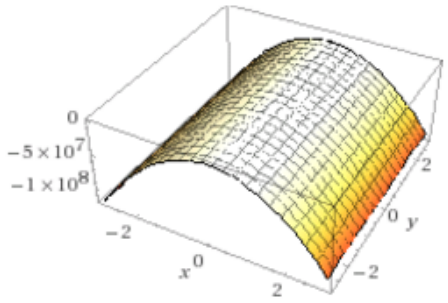
(b) Contour Map of the Intensity Function (Eq.2)

Figure 2: Representation of Intensity Function (Eq.2)

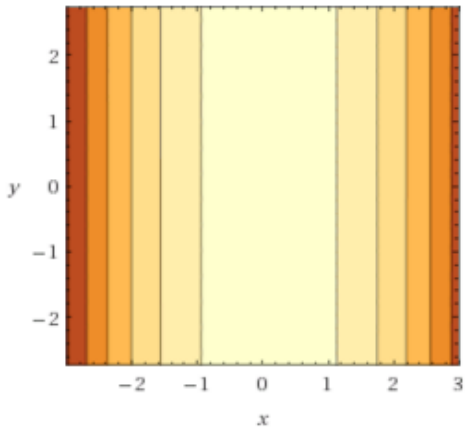
3. Polynomial Surface with an applied Gaussian Distribution

$$I(x, y) = A \cdot (ay - bx^2 + c) + e^{-(\frac{(x-x_0)^2}{h} + \frac{(y-y_0)^2}{k})} \tag{3}$$

Where $A = 750000$, $a = 3$, $b = 16$, $c = 10$, $x_0 = -30$, $y_0 = 50$, $h = k = 4000$.



(a) 3D Graph of the Intensity Function (Eq.3)



(b) Contour Map of the Intensity Function (Eq.3)

Figure 3: Representation of Intensity Function (Eq.3)

Each intensity function is differentiable and has a single local maximum. Each swarm uses a discrete approximation to complete the task. These swarms start in some initial state P_s and uses its method to estimate the gradient of the intensity function. Once it has done that and it has calculated its change for that time-step, the global parameter changes and it moves on until the swarm state is sufficiently close to its end or P_f. We use a the three agent swarm and a two agent swarm, each of which estimates the gradient in a unique way.

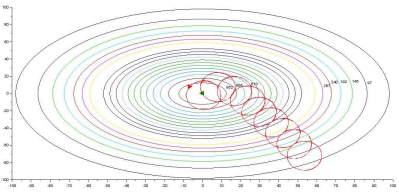
4.1 The Two Agent Swarm

In this swarm, the agents are configured to start a fixed distance r away from each other and are able to communicate each others intensities and track their own. The agent with the higher starting intensity stays stationary while the second agent orbits in a circle around the first, discretely sampling the values of the intensity during the circle. After making a full circle around the agent, the second agent moves to the maximum value it recorded on the circular path. Now the agents reverse roles and the cycle continues until the swarm reaches a maximum. By choosing the highest intensity in the circular orbit, the agents are moving in the direction of the greatest change and therefore approximating the gradient.

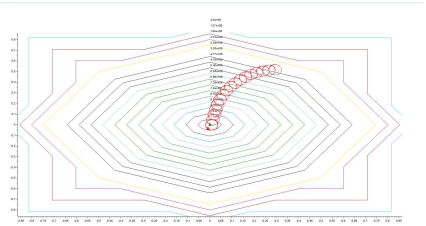
Additionally we wish to determine the local variables that the swarm is using to calculate its change in position. If the agent is stationary, it need not measure anything for that time-step. However, if the agent is moving, the local measurables available to the agent are its locally position in x and y, its measured intensity at discrete points around the circle, the other agent's position and their intensity.

For each of the intensity fields above we ran a simulation that demonstrates the ability for each of the swarms to find the maximum. The path taken by the agents are demonstrated below.

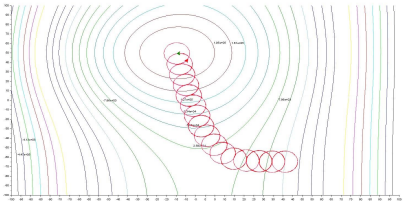
In each function, the swarm was able to quickly find the maximum of the function while also describing the path it took to get there.



(a) Path traveled by Two Agent Swarm through Eq.1



(b) Path traveled by Two Agent Swarm through Eq.2



(c) Path traveled by Two Agent Swarm through Eq.3

Figure 4: Path Traveled by Two Agents Swarm

4.2 The Three Agent Swarm

The three agent swarm is also capable of approximating gradient. Each agent is capable of taking the intensity and determining the location of the other agents. However, in order to estimate the magnitude of the gradient accurately, the swarm agents must be relatively close together. Therefore, the system utilizes a physicomimetic control algorithm, used multiple times in recent papers [4, **?**, 8]. The basic methodlogy is to implement an artificial gravitational force between agents far away and an artifical repulsive force for agents too close. This causes the agents to reach an equilibrium in the form of an equilateral triangle. For this swarm, we found it best to use a non-traditional gravity equation described below.

$$\vec{F_p} = 5\arctan(\frac{G}{r^2})\hat{r} \tag{4}$$

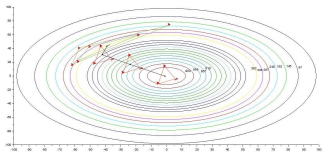
Where G is 200 and F_p is the force vector given by the physico-mimetic swarm. In the simulation, the three agents are initialized at random points in space. During each iteration, an agent receives the intensity from the other two agents and uses a first order linear approximation to approximate the gradient.

$$\vec{F_g} = \frac{E(\nabla I)}{||E(\nabla I)||} \tag{5}$$

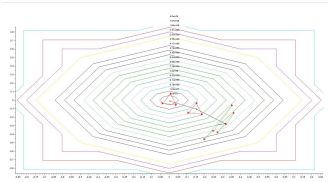
Where $E(\nabla I)$ is the estimated value of the gradient as determined by a first-order linear approximation. This produces a separate force vector. This force is added to the force of the physicomimemetic force vector. This vector is the estimated gradient normalized by dividing by the gradient's magnitude. Therefore our overall force vector is determined by the equation below.

$$\vec{F_{total}} = 5\arctan(\frac{G}{r^2})\hat{r} + \frac{E(\nabla I)}{||E(\nabla I)||} \tag{6}$$

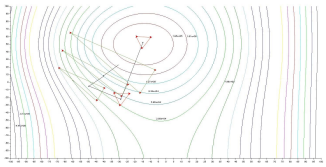
For each of the 3 intensity fields denoted, we ran a simulation demonstrating the ability of the three agent swarm to find the maximum. The path taken by the swarm is figured in the images below.



(a) Path of Three Agent Swarm through Eq.1



(b) Path of Three Agent Swarm through Eq.2



(c) Path of Three Agent Swarm through Eq.3

Figure 5: Path Traveled by Three Agents Swarm

As can be seen by the diagrams above, while the swarm agents moved erratically at first, due to a poor approximation of the gradient, the physicomimetic force was able congregate the swarm so that the gradient approximation would get better as time went on.

4.3 Utility of the Swarms

Both swarms effectively find the gradient of our test functions. These swarms are useful when functions are globally increasing to a singular maximum. However, each of the swarms has different utility when being applied to real-world functions. For instance, one of the major differences between the two swarms is that the two-agent swarms uses memory for each time-step, while the three-agent swarm is able to calculate the gradient in a single time step. However, the two agent swarm requires less material to build. The two agent swarm also requires a specific configuration to begin whereas the 3 agent swarm may take any initial position.

We now wish to describe under what conditions a swarm is necessary to solve this type of problem. Utilizing the method we developed earlier, we see that a swarm is necessary only if the technologies associated with swarm capability are not easily applicable to one single agent. For this problem a swarm in necessary in cases in which the gradient is impractical or impossible to predict or measure using a single technology.

5 Conclusions and Future Work

In this paper we described a general swarm theory and applied it to a low dimensional swarm problem. First, we defined a mathematical structure of a generic swarm problem using global parameters and their agents. We then described the change in this system over time which allowed us to set up requirements that a swarm must follow to achieve the task. Examining these requirements, we derived the technology matrix. Each of the term in the technology matrix represents a different way the agents must change their attributes, and therefore, represents a new technology required by the swarm.

We then discussed how the attributes themselves change over time and modeled those as a function of local variables. These partial differential equations describe how the swarm calculates its next position in time. From here we completed our mathematical framework for solving a general swarm problem.

We then modeled how a real swarm could accomplish the task. We used a discrete approximation, using

a function that would take in a state of the function, apply the agent's affect, and output a new state of the system on the next time-step. Additionally, we briefly discussed the utility of a swarm under certain conditions, and the reason for using a swarm based on the technologies necessary to accomplish the task.

In order to test the theory made, we applied the Hamiltonian method directly on the Maximum Search Problem in phase space. This problem essentially involves having agents methodically finding a global maximum of a given intensity function. Applying the Hamiltonian method, we were able to calculate the technology matrix for the general problem and prove that the if the swarms were able to accurately measure the gradient at their position, then they would be guaranteed to approach the maximum.

Finally, we modeled two swarms within the two-dimensional version of the Maximum Search Problem using a variety of intensity functions to test the Hamiltonian method. Both swarm computer models were able to find the maximum.

In the future, however, we may use our computer models to create functional robots that may find maximums in real life. Additionally, we wish to apply this methodology to multi-dimensional problems. This will allow us to accomplish a wider variety of swarm tasks that can perhaps later be used for important engineering developments.

References

[1] Bourgeous W, Rounds S, Chen Y. *A Swarm Engineering Approach to Mobile Sensor Network Design Towards Collaborative Phototaxis With A Slowly Moving Light Source.* **International Design Engineering Technical Conferences and Computers and Information in Engineering Conference**, 179-189

[2] Kazadi, S. and Lee, J. *Artificial Physics, Swarm Engineering, and Hamiltonian Method.* **Proceedings of World Congress on Engineering and Computer Science 2007**. San Francisco, CA, USA, Oct 24-26, 2007

[3] Kazadi, S. *Model Independence in Swarm Robotics.* **International Journal of Intelligent Computing and Cybernetics, Special Issue on Swarm Robotics**, 2(4), pp. 672-694, 2009.

[4] Spears, D., Spears, W., Sokolsky, O., and Lee, I. *Distributed Spatial Control, Global Monitoring and Steering of Mobile Physical Agents.* **IEEE International Conference on Information, Intelligence, and Systems**, November, 1999.

[5] Dorigo, E. and Gambardella, L. M. *Ant Colonies for The Traveling Salesman Problem.* **ScienceDirect**, 43(1), 73-81, 1997.

[6] Spears, W. and Gordon, D. *Using Artificial Physics to Control Agents.* **Proceedings of the IEEE Conference of Information Intelligence and Systems**, 1999

[7] S. Kazadi, D. Jin, and M. Li. *Utilizing Abstract Phase Space in Swarm Design and Validation.* **Advances in Swarm and Computational Intelligence.** Vol. 9140, pp. 14-29, 2015.

[8] Spears, W., D. Spears, J. Hamann, and R. Heil. *Distributed, Physics-Based Control of Swarms of Vehicles.* **Autonomous Robots**, Volume 17(2-3), August 2004.

[9] D. Spears, D. Thayer, an D. Zarzhitsky. *Foundations of swarm robotic chemical plume tracing from a fluid dynamics perspective.* **International Journal of Intelligent Computing and Cybernetics**, 2(4), pp. 745-785, 2009.

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