Variation as an Element in Multi-Agent Control for Target Tracking

Cortney Riggs[1] and Annie S. Wu[1]

***Abstract*— This work investigates the impact of inter-agent variation on the performance of a multi-agent system based control system. The control system is composed of a multi-agent team and aggregates the actions of the individual agents of the team to form a single output. In the process, this approach attempts to use error and variation that naturally occurs in physical systems to the system's advantage. We apply this multi-agent control system to a multiple task allocation problem and present an analytical model with which we can study the conditions under which such a system is expected to stabilize. We then compare the expected behavior of the analytical model with an empirical agent-based simulation applied to a tracking problem.**

I. Introduction

This work investigates the impact of inter-agent variation on the performance of a multi-agent system based control system. In this approach, the control system is composed of a multi-agent system consisting of a team of decentralized agents which do not communicate directly with each other. Variation is introduced into the system in the form of small errors or differences in individual agent perceptions of common goals. Each agent acts independently in response to outside stimuli and the actions of all agents are aggregated to form a single control system output. As a result, this agent-based control architecture is expected to inherit many of the robustness qualities that are typically expected of multi-agent systems. More interestingly, we believe that this approach provides a natural way of dealing with (and even taking advantage of) naturally occurring errors and variations that exist in physical systems.

Inter-agent variation is common in natural systems, such as biological [1], [2], [3] and social systems [4], [5], [6], [7], [8] and appears to be an essential element in how complex systems work [9], [10]. If we consider the number of ways that a team of agents may respond to stimuli, we find that variation in the behaviors or reactions of the individual agents lead to a larger response set for the team as a whole. Thus, having a variety of responses can potentially benefit the system as a whole by providing a wider range of system responses to external stimuli [11], [12]. If the system response is what we are interested in, variation in the agents that make up the system will lead to a more flexible system.

Engineered multi-agent systems such as robot teams can generate better team performance by using variation to improve agent coordination in the team [13], [14], [15], [16]. Inter-agent variation may be dynamic and change over time [3], [4], [15] or remain static throughout a run [13], [14].

Several classes of algorithms use the aggregation of multiple sources to deal with variation. Collaborative control algorithms combine inputs from multiple agents (including human agents) and attempt to exploit the strengths of each agent [17], [18], [19]. Consensus control algorithms or filters, such as those used in distributed sensor networks [20], [21], [22], [23] use the aggregate information from multiple "agents" to determine signal values in the presence of noise. Similar to consensus algorithms are a number of approaches that use multi-agent aggregation to filter out noise for robot control [24], [25], [26], [27].

Previous work finds that inter-agent variation in a multi-agent system can allow for more efficient responses by the system to changing demands in a single task problem [12]. We extend that analysis to multiple-task allocation problems and present an analytical model with which we can study the conditions under which such a system is expected to stabilize. We then compare the expected behavior of the analytical model with an empirical agent-based simulation applied to a tracking problem. The contributions of this work are to gain a better understanding of the effects of variation on multi-agent system coordination and to demonstrate potential use of such systems as a control system.

II. Formal Description

We define the tracking problem to be the attempt by a tracking unit to follow the movements of an object over time. This problem can be described as a discrete autonomous dynamic system. The position of the object is defined as a discrete function of time. The corresponding velocity function is the input to our dynamic system. The multi-agent system is the control unit to our dynamic system. The goal is for the multi-agent system as a whole to respond appropriately to the input, even while the agents that make up the multi-agent system act independently and without direct communication. The control system response updates the current belief in the position of the object.

A. Representation of the problem

An object's position function has components in each dimension. Thus, if there are $q \in \aleph$ dimensions, the overall position function is defined by the vector

$$\mathbf{d}(t) = (d_1(t), d_2(t), \ldots, d_q(t))^T. \quad (1)$$

We further break down the representation of the tracking problem by representing each dimension with two tasks. One task represents the motion in the positive direction and the

*This work is supported by ONR grant #N000140911043 and General Dynamics grant #100005MC.

[1]C. Riggs and A. S. Wu are with the Department of Electrical Engineering and Computer Science, University of Central Florida, Orlando, FL 32816 cortr@hotmail.com, aswu@cs.ucf.edu

other represents motion in the negative direction. Thus, for a q dimensional tracking problem there will be $n = 2q$ tasks. The state vector of all of the tasks that make up a q-dimensional tracking problem is defined as

$$\mathbf{x}(t) = (x_1(t), x_2(t), \dots, x_n(t))^T. \tag{2}$$

In our problem, $n = 4$, x_1 corresponds to the positive direction of the x-axis, x_2 corresponds to negative x-axis movement, x_3 corresponds to positive y-axis movement, and x_4 corresponds to negative y-axis movement.

The velocity of the object serves as the basis for the input signal, $\mathbf{u}$, to the dynamic system. For an object moving according to Equation 1, the velocity vector is

$$\mathbf{d}'(t) = \mathbf{v}(t) = (v_1(t), v_2(t), \dots, v_q(t))^T \tag{3}$$

Velocity in a dimension j at time $t-1$, $v_j(t-1)$, is assigned to the appropriate task input signal, $u_i(t)$, based on whether the velocity is moving in the positive or the negative direction in the dimension j. Our input signal mapping function is

$$u_i(t) = \begin{cases} K v_{(i+1)/2}(t-1) & \text{if } i \text{ is odd and } v_{(i+1)/2}(t-1) > 0 \\ K \left| v_{i/2}(t-1) \right| & \text{if } i \text{ is even and } v_{(i)/2}(t-1) < 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

where K scales the velocity measurement to the timestep size. We assume the timesteps are on the same scale with the velocity measurements. The input signal to a task is always positive because we represent the positive and negative directions in a dimension using separate tasks. Thus, the position change represented in the states is $\mathbf{x}(t-1) - \mathbf{u}(t)$.

B. Multi-agent system

The multi-agent system is the control system for our dynamic system. Applied to the tracking problem, the goal of the control system is keep the task states of the dynamic system at a constant value. Constant values imply that the multi-agent system is properly responding to the input $\mathbf{u}$. Thus, the task states representing the problem are the cumulative difference between the input obtained from the object motion and the response by our multi-agent control system.

A set of m decentralized agents act independently with no direct communication with each other. At each time step, an agent decides whether or not it needs to act. If there are multiple tasks needing action, the agent must also decide on which task to act. Agents act by contributing a set amount, c, to the state of a task in response to the input signal, $\mathbf{u}$ (this could also be seen as a weighted vote where all the agents have the same weight on their votes). Each agent can only contribute to one task in a given time step; each agent has the ability to act on any task in a given time step. The aggregate of all agent contributions in a timestep forms the response of the multi-agent control system for that timestep.

Each agent, a_j, has a response threshold for task i that is defined as

$$a_j.target_i = \mu_i + \epsilon_{ij},$$

where μ_i is the mean of the response thresholds for all of the agents for task i and ϵ_{ij} is a small error value that represents an error in an agent's perception of when it needs to contribute to task i. The value ϵ_{ij} is drawn from a Gaussian distribution defined by $\mathcal{N}(0, \sigma_i)$. We use a Gaussian distribution to generate the error values ϵ_{ij} in order to model error distributions that potentially exist in the physical world. In our model, the error in perception is static, which means that ϵ_{ij} is constant throughout a run. Without loss of generality, we set the means of the response thresholds of all of the tasks to zero. We assume that all tasks have an identical standard deviation value. An agent will act on a task when the state of the task falls below the agent's response threshold for the task. If we have only one task whose state value is x and the input for a given timestep is u, then the proportion of agents that are expected to act in that timestep is defined by the cumulative distribution function,

$$Q(x-u) = \frac{1}{\sigma\sqrt{2\pi}} \int_{x-u}^{\infty} e^{-\frac{(\frac{z-\mu_i}{\sigma})^2}{2}} dz. \tag{5}$$

If the problem consists of more than one task, then it becomes possible to have more than one task fall below an agent's task response thresholds. As a result multiple tasks will be expecting a contribution from the same agent. Because an agent can only contribute to one task in each timestep, the actual expected contributions that a task receives will be less than the full amount that it expects due to loss of contributions to other tasks. Agents select randomly from among the tasks with states below the agent's response thresholds[1]. In order to model this as a dynamic system, we assume that the average response over multiple timesteps will be evenly divided among the tasks according to their need.

We are concerned with how the dynamic system is transformed in each timestep by a transformation array, E. The entries of the transformation array, E, are dependent on the states of the tasks relative to one another. The expected proportion of agents that will act on each task in timestep t, $\mathbf{z}$, is

$$\mathbf{z} = E\mathbf{x}^{-}, \tag{6}$$

is the vector of the expected proportion of contributions for each task when the other tasks are not considered, where

$$\mathbf{x}^{-} = \begin{bmatrix} Q(x_1 - u_1) \\ Q(x_2 - u_2) \\ \vdots \\ Q(x_n - u_n) \end{bmatrix}. \tag{7}$$

In order to find E, we begin by defining the expected proportion of agents that will contribute to task i as

$$z_i = B(\mathbf{x}^{-}, i) + C(\mathbf{x}^{-}, i), \tag{8}$$

where

$$B(\mathbf{x}^{-}, i) = \frac{\min_{x_l^{-} \in x^{-}} x_l^{-})}{n} \tag{9}$$

[1]Exploratory work has shown that random selection performs as would be desired for a tracking problem when compared to other selection mechanisms, such as choosing a task based on the largest or proportional difference between the task current and target values.

is the expected proportion of the contribution to the task with the highest state value which will be divided among all the tasks and

$$C(\bar{\mathbf{x}}, i) = \sum_{\bar{x}_j \leq \bar{x}_i} \left[\frac{D(\bar{\mathbf{x}}, j)}{\psi(\bar{\mathbf{x}}, j)} \right] \tag{10}$$

calculates the expected proportion of the contribution that task i will receive while taking into account contribution loss to the other tasks. In order to calculate $C(\bar{\mathbf{x}}, i)$, we look at the intervals defined by the task states, and calculate the expected proportion of contribution available within each interval.

The function $D(\bar{\mathbf{x}}, j)$ calculates the expected proportion of contribution in the interval from task j's state to the next highest task state value

$$D(\bar{\mathbf{x}}, j) = \frac{\bar{x}_j - \bar{x}_{next}}{\sum_{\bar{x}_l = \bar{x}_j} 1}, \tag{11}$$

where $\bar{x}_{next} = \max_{\bar{x}_g < \bar{x}_j}(\bar{x}_g)$ indicates the proportion $Q(x_g)$ for the task g with the next highest state value from task j. The summation in the denominator prevents repeat inclusion of the same contribution determined by $D(\bar{\mathbf{x}}, j)$ for all tasks with the same state value as task j in the summation of the terms defined in $C(\bar{\mathbf{x}}, i)$.

The function $\psi(\bar{\mathbf{x}}, j)$ counts the number of tasks whose state value is greater than or equal to that of task j, because these are the tasks to which task j will lose agent contributions.

$$\psi(\bar{\mathbf{x}}, j) = \sum_{\bar{x}_l \geq \bar{x}_j} 1. \tag{12}$$

For example, if all four tasks have unique state values, such that $x_1 - u_1 < x_2 - u_2 < x_3 - u_3 < x_4 - u_4$, then we have

$$\mathbf{z} = \begin{bmatrix} \bar{x}_1 - \bar{x}_2 + \frac{\bar{x}_2 - \bar{x}_3}{2} + \frac{\bar{x}_3 - \bar{x}_4}{3} + \frac{\bar{x}_4}{4} \\ \frac{\bar{x}_2 - \bar{x}_3}{2} + \frac{\bar{x}_3 - \bar{x}_4}{3} + \frac{\bar{x}_4}{4} \\ \frac{\bar{x}_3 - \bar{x}_4}{3} + \frac{\bar{x}_4}{4} \\ \frac{\bar{x}_4}{4} \end{bmatrix} \tag{13}$$

Separating the terms, we obtain

$$\mathbf{z} = \begin{bmatrix} \bar{x}_1 - \frac{\bar{x}_2}{2} - \frac{\bar{x}_3}{6} - \frac{\bar{x}_4}{12} \\ \frac{\bar{x}_2}{2} - \frac{\bar{x}_3}{6} - \frac{\bar{x}_4}{12} \\ \frac{\bar{x}_3}{3} - \frac{\bar{x}_4}{12} \\ \frac{\bar{x}_4}{4} \end{bmatrix}. \tag{14}$$

From Equation 14, we can determine that the transformation array for this example is

$$E = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{6} & -\frac{1}{12} \\ 0 & \frac{1}{2} & -\frac{1}{6} & -\frac{1}{12} \\ 0 & 0 & \frac{1}{3} & -\frac{1}{12} \\ 0 & 0 & 0 & \frac{1}{4} \end{bmatrix}. \tag{15}$$

Using the knowledge that this example is based on a vector $\mathbf{x}$ that is ordered in a decreasing fashion from x_1 to x_4, it is intuitive that the transformation array E will always be an upper triangular array if we order the vector $\mathbf{x}$. Even if some tasks have equal state values, the E acting on the decreasing ordered vector $\bar{\mathbf{x}}$ will still be defined as an upper triangular array. For example, if, in the ordered vector $\mathbf{x}$, the condition $x_1 - u_1 < x_2 - u_2 = x_3 - u_3 < x_4 - u_4$ is true, then we can describe the transformation array as

$$E = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{6} & -\frac{1}{12} \\ 0 & \frac{1}{3} & 0 & -\frac{1}{12} \\ 0 & 0 & \frac{1}{3} & -\frac{1}{12} \\ 0 & 0 & 0 & \frac{1}{4} \end{bmatrix}. \tag{16}$$

When all the tasks states are equal, E can be defined as diagonal array

$$E = \begin{bmatrix} \frac{1}{4} & 0 & 0 & 0 \\ 0 & \frac{1}{4} & 0 & 0 \\ 0 & 0 & \frac{1}{4} & 0 \\ 0 & 0 & 0 & \frac{1}{4} \end{bmatrix}. \tag{17}$$

Thus, even though the dynamic system has variable coefficients (changing entries in E), the transformation array can always be re-written as an upper triangular array by simply re-ordering the tasks' state vector by non-increasing values. This upper triangular array simplifies the stability analysis for fixed points, under the condition that the state vector $\mathbf{x}$ is ordered.

C. *Putting it all together*

Now that we have defined our problem representation and the decision making process of the multi-agent system, we can combine both into a discrete dynamic system representation of how the tasks states change over time. In this form, we can analyze the system to understand the behavior and determine conditions for stability. The dynamic system mapping of the states of the tasks is

$$\mathbf{x}(t) = \mathbf{x}(t-1) - \mathbf{u}(t) + \alpha E(\bar{\mathbf{x}}(t)), \tag{18}$$

where $\alpha = m \times c$ is the total contribution the multi-agent system can make at any given timestep. Since the update to the belief in position by the tracker is the sum of the agent contributions, the output mapping for the dynamic system is

$$\mathbf{y}(t) = \mathbf{y}(t-1) + \alpha A \mathbf{z}(\mathbf{t}), \tag{19}$$

where A is the aggregating $q \times n$ array combining the directional responses into the resulting dimensional position change. The goal is to minimize the difference between $\mathbf{y}(t)$ and $\mathbf{d}(t)$.

Because the multi-agent system has a finite number of agents and the total contribution, α, is dependent on the number of agents, the maximum response that the system can provide is bounded. The control system has the potential to track an object if the total contribution of agents is enough to counteract the sum of the reduction on the set of all tasks. A problem is considered feasible at a time step if the relationship

$$\alpha \geq \sum_i u_i \tag{20}$$

holds. For some period of discrete time steps, $[t_1, t_\tau]$, a problem is feasible (not necessarily stable) if the relationship in Equation 20 holds true during the span of that time period. In this work, we focus on feasible tracking problems.

III. Analysis

The components of our system (the tasks) are dependent on each other due to the resource constraints of a set number of agents. Because the tasks in our dynamic system must share the available agent resources, the multi-agent response to a single task can vary dynamically due to the states of the other tasks. Dependency between the components creates a variable coefficient dynamic system, which is generally difficult to analyze. Using the knowledge that the tranformation array E can be re-written as an upper triangular array, we can analyze the system using the partial derivatives of the dynamic system mapping.

To analyze the dynamic system, we need to calculate the fixed points of the system and determine the stability near the fixed points. Fixed points can be determined by solving for the state $\mathbf{x}^* = f(\mathbf{x}^*)$, where $f(\mathbf{x}^*)$ is the function describing the right hand side of Equation 18. Linearization of our non-linear dynamic system model can determine under what conditions the system is stable. The order of the states at the fixed point is important. If the system is stable at the fixed point, stability near the fixed point will only be true assuming the order of the states holds. Responses to arbitrary states may not result in orderings the same as the fixed point. Linearization cannot determine if the system is stable for arbitrary states near the fixed point, let alone for a broad range within the domain of possible state values. An understanding of the system behavior across arbitrary task state values allows for an effective application of the system to a given problem[2]. Thus, we analyze the system with two approaches to attempt understand how the dynamic system behaves. First, we will do a linearization near the fixed point to determine if there are any conditions that the system is stable. Second, we will attempt to show general behavior of the system for tasks starting from any arbitrary state values.

A. Linearization near the fixed point

Intuitively, a fixed point, $\mathbf{x}^* = f(\mathbf{x}^*)$, exists when the order of the state vector corresponds to the magnitude of the disturbances on the states (the largest u_i must correspond with the largest response), where $f(\mathbf{x})$ is the function defining the right hand side of Equation 18. Since the linear model at any state, as well as at the fixed point, $\mathbf{x}^*$, is an upper triangular matrix when properly ordered, we can find the eigenvalues, λ, with the roots of

$$det(J_{f^r} - \lambda I) = \prod_{x_i} \left(1 + \frac{\partial Q_i'}{\partial x_i^{\text{-}}} - \lambda \right) \qquad (21)$$

where J_{f^r} is the Jacobian matrix of the function $f(\mathbf{x})$ when the system states are ordered in a fashion as described for the array in Equation 15 and I is the identity matrix of size n. The partial derivative, $\frac{\partial Q_i'}{\partial x_i} = \phi(x_i - u_i, \mu, \sigma^2)$, is the probability distribution function for state $x_i^{\text{-}}$ (this is the partial derivative of the terms in $\mathbf{z}$). For example, for the system with a fixed point resulting in a re-written $E(x_i^{\text{-}})$ as described in Equation 15, the eigenvalues are

$$\prod_{x_i} (1 - \frac{\alpha e^{-(x_i - u_i - \mu)^2/2}}{i\sigma\sqrt{2\pi}} - \lambda) = 0 \qquad (22)$$

A discrete dynamic system is stable when, for all eigenvalues, $\lambda_1, \ldots, \lambda_p$, we have, $|\lambda_j| < 1$. As we can see from Equations 21 and 22, the eigenvalues are always less than one. Thus, the driving restriction for determining stability is

$$1 - \frac{\alpha e^{-(x_i - u_i - \mu)^2/2}}{\sigma\sqrt{2\pi}} \geq -1. \qquad (23)$$

Noting that $e^{-(x_i - u_i - \mu)^2/2}$ is largest at $(x_i - u_i - \mu) = 0$ and solving for the parameters of the multi-agent system, the dynamic system is always asymptotically stable for the ordered states near the fixed point, x^*, when

$$\alpha < \sigma 2\sqrt{2\pi}. \qquad (24)$$

This is of most concern to avoid k-period[3] cycles and chaotic behavior in the system. Equation 24 shows how agents' perceptions (and therefore the distribution of the contribution) determined by α and σ determines the sensitivity of the system to changes. In fact, the asymptotic stability will have a monotonic convergence for all the tasks when

$$\alpha < \sigma\sqrt{2\pi}. \qquad (25)$$

B. General system behavior

In an attempt to analyze the stability for arbitrary task states near the fixed point, we will build up an analysis from a single task into a multiple task system. We begin with multiple single task systems each having their own populations of agents, then relax the assumption of mutually exclusive populations by allowing agents to cross over between populations, in effect merging the separate populations into one.

For a single task, the mapping simply becomes

$$x(t) = x(t\text{-}1) - u(t) + \frac{\alpha}{\sigma\sqrt{2\pi}} \int_{x_i(t\text{-}1)-u(t)}^{\infty} e^{-\frac{(\frac{z-\mu}{\sigma})^2}{2}} dz. \qquad (26)$$

This single task system was discussed in work by Campbell *et al.* [12], in which the authors empirically analyzed the stability. There, a loose bound on the fixed point, $x^* = f(x^*)$, for the system was derived using Chernoff approximation of the Q-function. Tighter bounds can be found using numerical methods for the equations

$$\frac{(x^*)^2}{2} + \ln(x^*) + \ln\left(\frac{u\sqrt{2\pi}}{\alpha}\right) < 0 \qquad (27)$$

and

$$0 < \frac{(x^*)^2}{2} + \ln\left(x^* - \frac{x^*}{(x^*)^2}\right) + \ln\left(\frac{u\sqrt{2\pi}}{\alpha}\right). \qquad (28)$$

[2]Given the assumption that the problem can be broken down into a task based representation.

[3]k is some integer describing the number of points in the cycle

The stability of the single task system around the fixed point has already been analyzed above via Equations 23, 24, and 25, because the derivative of the single task mapping function, $f'(x^*)$, is the left hand side of Equation 23, which was the partial derivative for the task with the largest contribution in the ordered system. The fixed point is stable when $|f'(x^*)| < 1$. When Equation 24 holds, all state values for the task have a derivative in the range $[-1, 1]$ ensuring that any feasible system is also stable.

Next, assume we have two independently stable single systems, task i and task j, each with their own multi-agent control system responding to disturbances. If we relax the disturbance constraints of each individual system to a broader constraint on both systems, such that $\sum_{l \in i,j} u_l \leq \sum_{l \in i,j} \alpha_l$ (similar to Equation 20), we have that one task could potentially be infeasible. However, if the multi-agent response systems are combined, the two individual tasks have enough resources to be feasible.

Combining the multi-agent responses of two tasks can be seen as an infeasible task "borrowing" contribution (agents) from the other task. The infeasible task will only take the contribution it needs to become feasible from the other task. It is assumed that the borrowed contribution will be distributed on the borrowing task according to its original distribution, $\mathcal{N}(\mu_i, \sigma_i)$. In order to ensure that the dynamic system for task i remains stable after it borrows from the other task, the relationship, $\alpha_i + (u_i - \alpha_i) < \sigma_i 2\sqrt{2\pi}$, must hold. The originally feasible task, task j, will remain stable since α_j will decrease (but not by so much that it becomes infeasible). Additional tasks can be added to the system in the same manner, as long as the total contribution available does not exceed $\sigma_i 2\sqrt{2\pi}$ for the any task i.

Fully combining the two task systems into a single combined system allows both sets of agents to act on either task according to the defined distribution of error for the task. The feasiblility of this combined system is defined by Equation 20. The stability of the individual tasks within this combined system is guaranteed if, for each task, Equation 24 holds true.

In the combined system, some of the contribution from the single multi-agent system that is expected by a single task is also potentially expected by other tasks. Thus, a *realized contribution* by a single task is the amount of the combined system's total contribution minus the contribution that will go to the other tasks. The contribution lost to other tasks will not destabilize the single task fixed point because lessening the realized contribution available to the task will not violate the condition in Equation 24. However, a lower realized contribution to a task does affect the fixed point of that task calculated in Equations 27 and 28 by lowering its value.

If the loss of expected contribution to other tasks is thought to remain constant, the state of a task will converge towards the fixed point defined by the amount of the realized contribution. Since the other tasks will change state values, the realized contribution for each task will likely change every time step. Thus, a task will be attracted to many changing "fixed points" during the span of a run. The range of these fixed points for a task denotes the region to which the task state will always be driven. This fixed point region is not necessarily a bounded stable area since the task could have oscillatory stability. If the stability is monotonic for all fixed points possible in the range (or when the available contribution is the largest) then the region is a bounded stable area (once in it the state will not leave the region unless a change in the input occurs).

As a result, the range of fixed points for a task i on a constant input is given by the interval

$$[x_i^* = f_i^{(4)}(x_i^*), x_i^* = f_i^{(1)}(x_i^*)] \tag{29}$$

where $f_i^{(j)}(x_i^*)$ is the mapping function for $x_i(t)$ when the contribution available to task i is α/j. This range of fixed points accounts for all possible combinations of state values for the system that include those near the task state corresponding to the combined system's fixed point. Since arbitrary states are attracted to a region that includes that fixed point, there is the potential the combined system will converge to that fixed point. However, this does not yet prove that arbitrary states will converge to the fixed point. This only shows that the behavior of the system will drive each of the task states to the region described in Equation 29. Further analysis needs to be done to show that the system is stable from arbitrary starting points[4], under the restriction that $\alpha < \sigma 2\sqrt{2\pi}$.

IV. Simulations

We compare the expected behavior defined by our analytical model with an empirical agent-based simulation. Our experiments investigate two aspects of system performance on a tracking problem. First, we examine the effectiveness of a multi-agent control system in tracking different trajectories. Second, we examine the robustness of the multi-agent control system to loss of agents.

Our experiments examine two tracking trajectories:

1) Square trajectory. The target starts at the center of the top of the square and moves clockwise. One complete square is tracked in 360 time-steps.The velocity input of the object is always 0.5 units per time step, $u_i = 0.5$, for a single task i.
2) Circular trajectory. The target starts at the twelve o'clock position and moves clockwise. One complete circle is tracked in 360 time-steps The velocity input will have a constant magnitude, $|u| \approx 0.39$, so sum of task inputs is never greater than 0.5 units per time step, $\sum u(t) \leq 0.5, \forall t$.

The multi-agent control system is made up of $n = 100$ agents. Each agent has a contribution of 0.01, so $\alpha = 1$. Since all problems have a maximum sum of velocities of 0.5, this maintains that the system is feasible across all time steps, $\alpha \geq \sum u$. Using Equation 24, σ is set near the limit to keep the system stable with a value of 0.2. The center of

[4]Intuitively it seems possible and experimental work seems to indicate so.

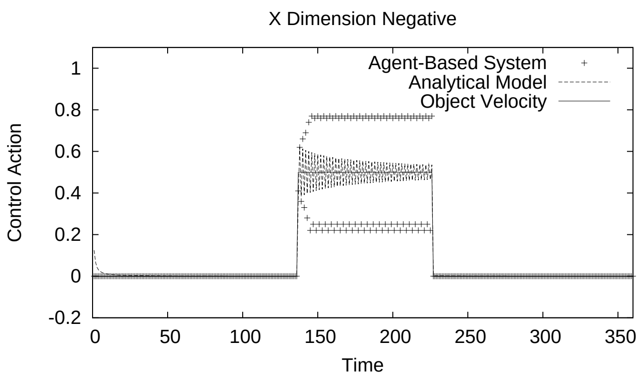

Fig. 1. This plot shows the object velocity in the negative direction of the X dimension (west) and compares that with the expected response of our analytical model and the behavior of our agent-based simulation. Because each task receives an input signal only when there is movement in the corresponding direction, this plot only has an active input signal when the object is moving west on the square trajectory. As a result, the region of interest is from timestep $t = 136$ to timestep $t = 225$.

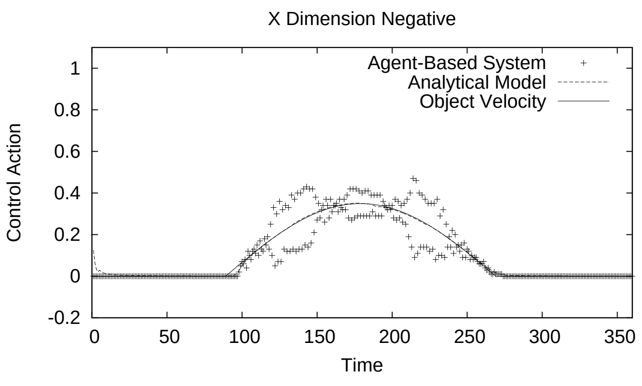

Fig. 2. This plot shows the object velocity in the negative direction of the X dimension (west) and compares that with the expected response of our analytical model and the behavior of our agent-based simulation. For the same reason as in Figure 1, this plot only has an active input signal when the object has a westward component in its motion on the circle trajectory. As a result, the region of interest is from timestep $t = 90$ to timestep $t = 180$.

the agent response thresholds for each task is defined to be $\mu = 0.0$. Thus, agents' error in perception is randomly drawn from the distribution defined by $\mathcal{N}(0, 0.2)$ for each task. The empirical agent-based simulations use initial task states of 0.6, (three standard deviations) to limit tracking error caused by agent contributions to tasks with no initial input. The analytical model uses initial states of 0.0 to demonstrate the the dynamic system's attraction towards the proper signal response.

A. Response effectiveness

We analyze the control actions for each trajectory by examining the directional control actions to determine how well the task needs are being met and analyze the displacement and heading control to see how well the system is performing on the tracking problem.

Figure 1 plots the object velocity, in the negative direction of the X dimension (west) for the square trajectory and compares that with the expected response of our analytical model and the behavior of our agent-based simulation. Because each task receives an input signal only when there is movement in the corresponding direction, this plot only has an active input signal when the object is moving west on the square trajectory. As a result, the region of interest is from timestep $t = 136$ to timestep $t = 225$. The analytical model shows asymptotic stability in response to the input. Oscillating convergence is caused by an $\alpha : \sigma$ ratio not fit for monotonic convergence. The oscillations are due to the higher sensitivity of the response near the center of the Gaussian distribution. In the agent-based simulations, the responses find “stable” cycles (in this case a 4-period cycle) centered around the object's velocity input for the task with the given input. The discrete nature of the multi-agent system does not gaurantee an accurate response for a timestep, but the system converges to a stable cycle. Plots for the other three directions are similar.

Figure 2 plots the object velocity, in the negative direction of the X dimension (west) on the circle trajectory and

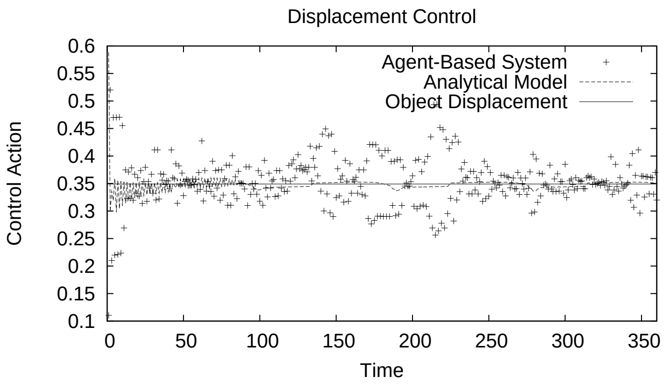

Fig. 3. Displacement control action is plotted with the actual displacement for a run of the circle trajectory. The analytical model displacement converges to a good approximation of the constant displacement of the circle trajectory. The agent-based simulation displacement fluctuates with an average difference from the actual displacement of 0.0075 units per timestep over 50 runs and a standard deviation of 0.007.

compares that with the expected response of our analytical model and the behavior of our agent-based simulation. For the same reason as in Figure 1, this plot only has an active input signal when the object has a westward component in its motion on the circle trajectory. As a result, the region of interest is from timestep $t = 90$ to timestep $t = 180$. The analytical model estimates the westward component of the object velocity well. Responses by the agent-based simulation fluctuate around the analytical model's estimations of the object velocity. Some ranges, such as around timestep 125, have large fluctuations in the multi-agent system responses. Near timestep 125, in our clockwise simulations, the southward direction also has a velocity component for input. Increased sensitivity at this time period is due to the input and spread of agent resources to other tasks causing the state of a single task to fall in to the more sensitive range near the center of the agent's response thresholds. The data for the other three directions is similar.

Figure 3 shows the displacement control action for an

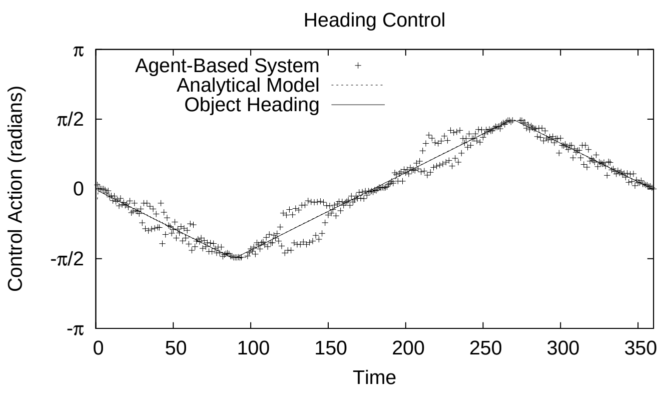

Fig. 4. Heading control action is plotted with the actual heading for a run of the circle trajectory. The analytical model's heading estimation is almost indistinguishable from the actual heading. The agent-based system heading has an average difference with the actual heading per time step over 50 runs of 0.05 radians with standard deviation of 0.013.

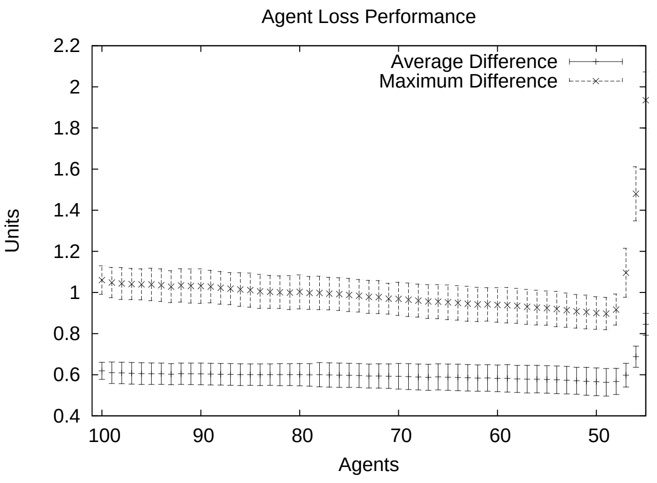

Fig. 5. Performance of the multi-agent system tracking a circular moving object as total number of agents decreases.

entire run of the circle trajectory. It shows the displacement of the object in each timestep and the response of the analytical model and the agent-based system to the object's movement in that timestep. The analytical model's estimation of the displacement converges to be close to the constant displacement of the object. The agent-based simulation response shows some fluctuation from timestep to timestep. The average difference in displacement per time step over 50 runs is 0.075 units with standard deviation of 0.007 and a max of 0.380 units.

Figure 4 shows the heading control action for an entire run of the circle trajectory. It shows the heading of the object in each timestep and the response of the analytical model and the agent-based system to the object's movement in that timestep. The plot implies that heading control performs well. The analytical model's estimations of heading are nearly indistinguishable from the object's heading. The agent-based simulation heading control again fluctuates from timestep to timestep. However, the performance over time is generally good. The difference in the multi-agent system response heading and the objects heading per time step over 50 runs averages 0.05 radians with a standard deviation of 0.013. The break down of the problem implicitly handles the angular change with the task relationships.

B. *System perturbations and tradeoff of small α*

An observant reader will have undoubtedly noticed the comfort of the input demand, u, never exceeding half of the available contribution, α. We next examine how the performance of our system adapts to smaller values of α; how it adapts as $(\sum \mathbf{u})/\alpha$ approaches 1. In effect, this also examines the multi-agent system's reaction to perturbations such as failure or loss of agents. We will focus on the circle trajectory problem. Each run consists of 100 completed circles. One agent is randomly selected to fail at the end of each complete circle. As agents fail, the α value of the system decreases.

Figure 5 shows the average and maximum difference in position of the object and the tracking system averaged per a completed circle over 50 runs. As agents are removed, the performance with respect to differences in tracking and actual position initially improves. Maximum distance shifts from about 1.07 at 100 agents to 0.89 at 49 agents. With less than 49 agents, the system performance starts to rapidly degrade. As the number of agents falls below 50, the system is infeasible for growing spans of time when the directional velocities of the object are near their highest sum, around tangent angles of $k\pi + \pi/2, k \in \mathbb{I}$.

The improvement in the performance of the system with agent loss is due to the thinning of the Gaussian distribution of agent response thresholds, causing the distribution of agent response thresholds becomes relatively more uniform. Uniform distributions of response thresholds have a linear relationship with the task state values. A linear gradient for the multi-agent system response results in a more reliable change in response with change in input.

V. Conclusion

In this paper, we present a formal model and demonstration of a multi-agent based control system. The agents in the system are decentralized, acting independently without direct communication. Agents differ in their perception of the system goals and the variation in agent perceptions has a Gaussian distribution. In reaching for the system goals, the agents contribute discrete amounts to tasks making up the goals. Summation of the discrete contributions of the agents creates a single control response. The variation in agent perceptions in the multi-agent system can create a robust and adaptable control unit, under sufficient parameters.

The performance of the system is dependent on two defining parameters of the system: the total contribution of the agents, α, and the amount of variation in the form of the standard deviation of the agent response thresholds, σ. Asymptotic stability in a single task dynamic system representation is always present when $\alpha \le \sigma 2\sqrt{2\pi}$. These conditions can be extended to a multiple task system, where the stability is evident in the drive of the system towards a region where the fixed point resides. Performance is better with lower total contribution than the upper limit $\sigma 2\sqrt{2\pi}$ due to a "thinner" Gaussian distribution of the agents' response thresholds. Less total contribution has a more uniform distribution of contribution which has a more linear gradient.

Applied to a tracking problem where the tasks are correlated with positive and negative direction-based components of movement, the multi-agent system response estimates the velocity, or the belief in position change of an object. In its estimation of the velocity, the control action by the multi-agent system appears noisy. The control actions of the multi-agent system produce fluctuating error in estimation of the position from timestep to timestep; however, the *average* error in position changes remains relatively small. If a rough estimate is all that is needed this is a sufficient system. However, one could shift the mean of the agents' thresholds each timestep in attempt to make the fluctuation less erratic and the controlling action more smooth.

The task based representation of the system that we present in this paper has both strengths and weaknesses. Two significant strengths are an inherent memory and robustness to system perturbations. The multi-agent system has a memory in the form of an accumulation of the difference in the system responses to the inputs. This memory is particular useful in the tracking problem example because it allows the multi-agent system to make up for past missed contributions to a task. The multi-agent system is robust in that loss of agents will not degrade performance as long as the total contribution of the multi-agent system is still enough to handle the input signal, in other words, as long as the system is feasible.

Noteable weaknesses include a restriction on the size of the multi-agent system for good behavior and the lack of agent loyalty to a task. If the multi-agent system is too large, the system responses are highly sensitive at levels near the center of the variation of the agents' response thresholds. By randomly choosing betweeen tasks with demand, there is potential for agents to switch between tasks frequently which is inefficient.

A key feature of multi-agent control systems such as the one presented here is that small inter-agent variations are essential to the effective performance of the system as a whole. This feature supports an recurrent and interesting possibility. Whereas most engineered systems perceive manufacturing variation, calibration errors, and other forms of variation that are unavoidable in physical systems to be negative features that must be actively dealt with, collaborative control systems may in fact be able to take advantage of some forms of variation to improve system performance.

References

[1] D. E. Jackson, S. J. Martin, F. L. W. Ratnieks, and M. Holcombe, "Spatial and temporal variation in pheromone composition of ant foraging trails," *Behavioral Ecology*, vol. 18, no. 2, pp. 444–450, 2007.

[2] J. C. Jones, M. R. Myerscough, S. Graham, and B. P. Oldroyd, "Honey bee nest thermoregulation: Diversity promotes stability," *Science*, vol. 305, pp. 402–404, 2004.

[3] F. Ravary, E. Lecoutey, G. Kaminski, N. Châline, and P. Jaisson, "Individual experience alone can generate lasting division of labor in ants," *Current Biology*, vol. 17, pp. 1308–1312, 2007.

[4] O. Kittithreerapronchai and C. Anderson, "Do ants paint trucks better than chickens? Markets versus response thresholds for distributed dynamic scheduling," *IEEE Congress on Evolutionary Computation*, vol. 2, pp. 1431–1439, December 2003.

[5] E. Bonabeau, G. Theraulaz, and J.-L. Deneubourg, "Phase diagram of a model of self-organizing hierarchies," *Physica A*, vol. 217, no. 3, pp. 373–392, 1995.

[6] D. A. Markiewicz and S. O'Donnell, "Social dominance, task performance and nutrition: implications for reproduction in eusocial wasps," *J. Comparative Physiology A*, vol. 187, no. 5, pp. 327–333, 2001.

[7] D. Maynes-Aminzade, R. Pausch, and S. Seitz, "Techniques for interactive audience participation," in *Proceedings of the IEEE International Conference on Multimodal Interfaces*, 2002.

[8] S. Picault and A. Collinot, "Designing social cognition models for multi-agent systems through simulating primate societies," in *Proceedings of the Third International Conference on Multiagent Systems*, 1998, pp. 238–245.

[9] W. R. Ashby, "Requisite variety and its implications for the control of complex systems," *Cybernetica*, vol. 1, no. 2, pp. 83–99, 1958.

[10] R. Axelrod and M. D. Cohen, *Harnessing Complexity*. New York: Free Press, 1999.

[11] W. R. Ashby, "Principles of the self-organizing dynamic system," *The Journal of General Psychology*, vol. 37, pp. 125–128, 1947.

[12] A. Campbell, C. Riggs, and A. S. Wu, "On the impact of variation on self-organizing systems," in *Proc. 5th IEEE Int'l Conf. Self-Adaptive and Self-Organizing Systems*, 2011.

[13] A. Campbell, A. S. Wu, K. Garfield, R. Shumaker, S. Luke, and K. A. D. Jong, "Empirical study on the effects of synthetic social structures on teams of autonomous vehicles," in *Proceedings of the IEEE International Conference on Networking, Sensing, and Control*, 2006, pp. 440–445.

[14] M. J. B. Krieger and J.-B. Billeter, "The call of duty: Self-organised task allocation in a population of up to twelve mobile robots," *Robotics and Autonomous Systems*, vol. 30, pp. 65–84, 2000.

[15] T. H. Labella, M. Dorigo, and J.-L. Deneubourg, "Division of labor in a group of robots inspired by ants' foraging behavior," *ACM Trans. Autonomous & Adaptive Systems*, vol. 1, no. 1, pp. 4–25, 2006.

[16] E. Bonabeau, M. Dorigo, and G. Theraulaz, "Inspiration for optimization from social insect behavior," *Nature*, vol. 406, pp. 39–42, 2000.

[17] R. Brooks, "A layered intelligence control system for a mobile robot," *IEEE Jour. Robotics & Automation*, vol. 2, no. 1, pp. 14–23, 1986.

[18] T. Carlson and Y. Demiris, "Collaborative control in human wheelchair interaction reduces the need for dexterity in precise manoeuvres," in *Proceedings of the ACM/IEEE HRI-08 Workshop on Robotics Helpers*, 2008, pp. 59–66.

[19] T. Fong, C. Thorpe, and C. Bauer, "Multi-robot remote driving with collaborative control," *IEEE Transactions on Industrial Electronics*, vol. 50, no. 4, pp. 699–704, 2003.

[20] Z. Li, Z. Duan, G. Chen, and L. Huang, "Consensus of multiagent systems and synchronization of complex networks: A unified viewpoint," *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 57, no. 1, pp. 213 –224, jan. 2010.

[21] R. Olfati-Saber and J. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, dec. 2005, pp. 6698 – 6703.

[22] W. Ren and R. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *Automatic Control, IEEE Transactions on*, vol. 50, no. 5, pp. 655 – 661, may 2005.

[23] A. Jadbabaie, J. Lin, and A. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *Automatic Control, IEEE Transactions on*, vol. 48, no. 6, pp. 988 – 1001, june 2003.

[24] K. Goldberg and B. Chen, "Collaborative control of robot motion: robustness to error," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 655–660.

[25] B. P. Gerkey, M. J. Mataric, and G. S. Sukhatme, "Exploiting physical dynamics for concurrent control of a mobile robot," in *Proc. IEEE Int'l Conf. Robotics and Automation*, 2002, pp. 3467–3472.

[26] M. D. Onsum and A. P. Arkin, "Autonomous mobile robot control based on white blood cell chemotaxis," in *Proceedings of the 2nd International Conference on Computational Methods in Systems Biology 2004 (CMSB 2004)*, 2005, pp. 9–19.

[27] D. Zarzhitsky, D. F. Spears, and W. M. Spears, "Distributed robotics approach to chemical plume tracing," in *Proc. IEEE/RSJ Int'l Conf. on Intelligent Robotics and Systems*, 2005, pp. 4034–4039.