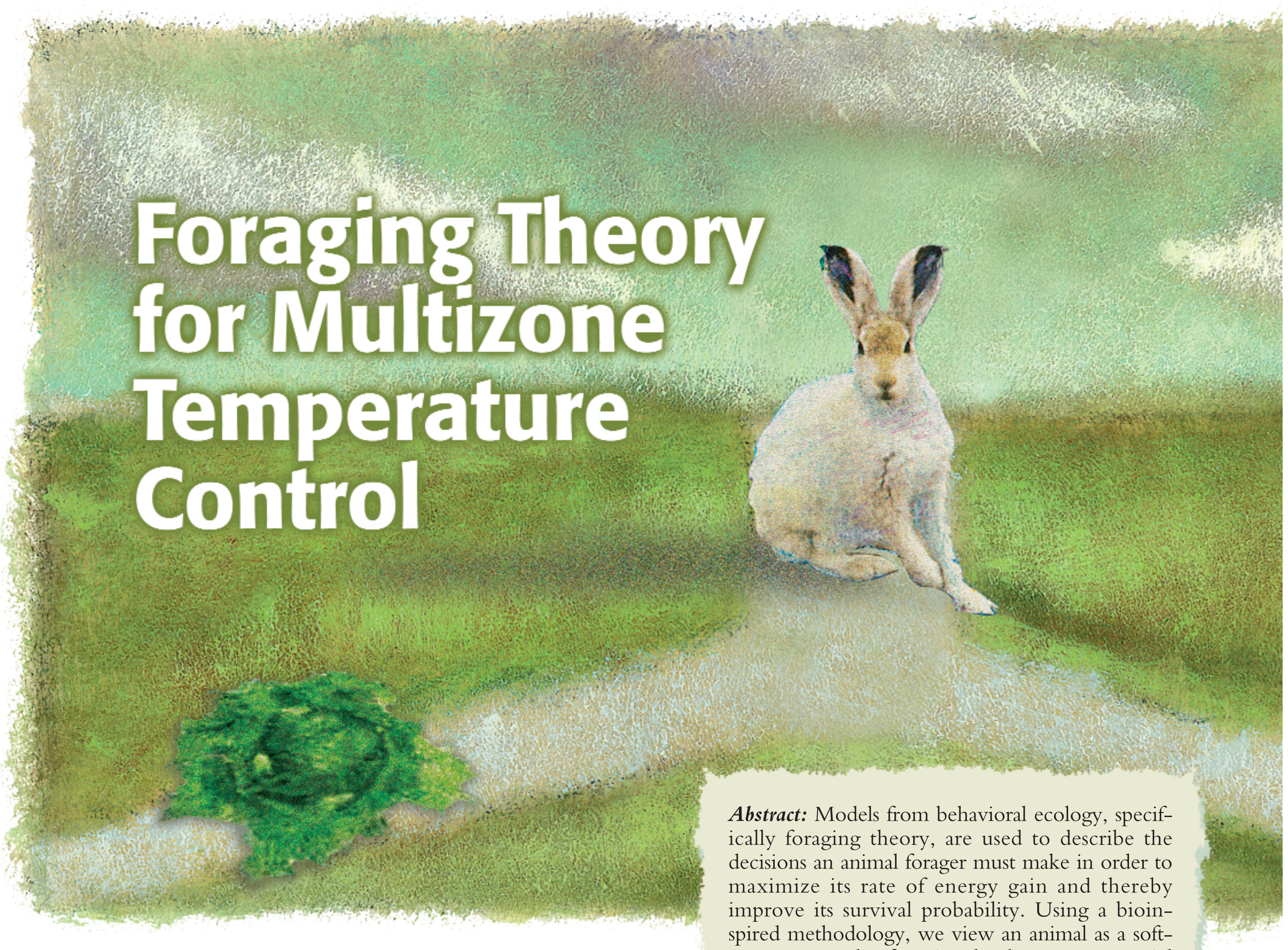
Nicanor Quijano and Kevin M. Passino
Ohio State University, USA
Burton W. Andrews
Johns Hopkins University, USA

Foraging Theory for Multizone Temperature Control



Abstract: Models from behavioral ecology, specifically foraging theory, are used to describe the decisions an animal forager must make in order to maximize its rate of energy gain and thereby improve its survival probability. Using a bioinspired methodology, we view an animal as a software agent, the foraging landscape as a spatial layout of temperature zones, and nutrients as errors between the desired and actual temperatures in the zones. Then, using foraging theory, we define a decision strategy for the agent that has an objective of reducing the temperature errors in order to track a desired temperature. We describe an implementation of a multizone temperature experiment, and show that the use of multiple agents defines a distributed controller that can equilibrate the temperatures in the zones in spite of interzone, ambient, and network effects. We discuss relations to ideas from theoretical ecology, and identify a number of promising research directions. It is our hope that the results of this paper will motivate other research on bioinspired methods based on behavioral ecology.

Index Terms: Foraging theory, intelligent control, temperature control.

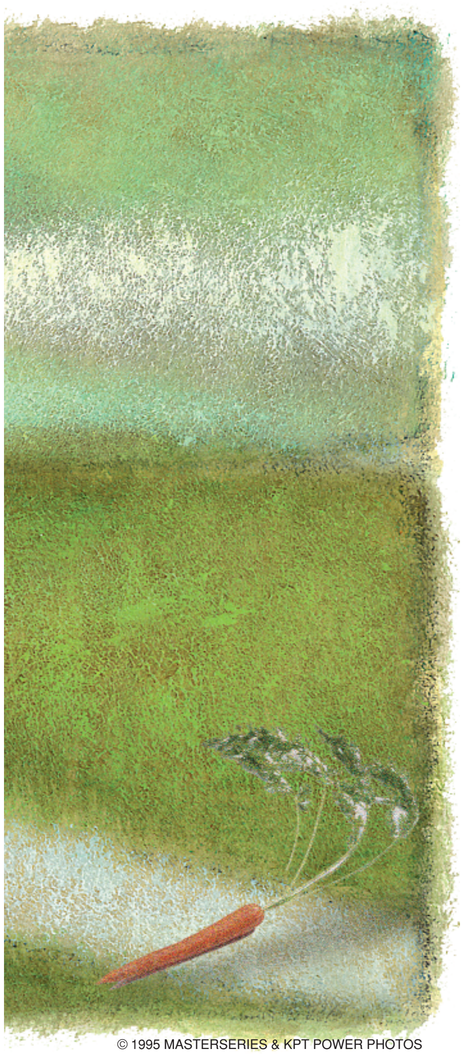
I. Introduction

Foraging theory is an area of behavioral ecology that mathematically describes a foraging animal searching for nutrients and choosing which ones to consume [1], [2]. One of the classical foraging models is the prey model. This model describes a forager searching for prey items individually dispersed throughout its environment and predicts which types of prey the forager should exploit in order to maximize its rate of energy gain. The analogy established in [3], [4] between a biological forager and an "agent" (autonomous vehicle or software module) allows for application of foraging models to engineering problems involving agents (i) searching for tasks dispersed throughout a domain, and (ii) deciding which task types to process and how long to process tasks or sets (patches) of tasks.

Here, we use the conceptual framework in [3], [4], along with the idea of "foraging for error" from [5], [6] to develop a

1556-603X/06/$20.00©2006IEEE

controller for a multizone temperature control problem. The goal is to achieve a uniform desired temperature across a grid of eight temperature zones, where a zone comprises a temperature sensor and a lamp. The controller (agent) moves around the grid searching for regions of temperature error (the task). The prey model approach to the problem relies on the definition of a "task type" as a zone with a particular error with respect to the desired temperature. Given this definition, an encounter with a task of type *i* occurs when the controller comes across a zone with a temperature error corresponding to task type *i*. The controller then uses the prey model algorithm [1] to decide whether to process the task, that is, whether to heat the zone associated with the error. We implement the experiment and controller in our laboratory [7] and provide data to illustrate the performance of the foraging algorithm.



The temperature control problem we are addressing is essentially that of a distributed feedback control problem. Recent relevant work in this area includes spatially distributed control [8], [9] modeling and estimation of distributed processes [10], distributed control of thermal processes [11]–[14], spatially interconnected systems [15], and semiconductor processing [16]–[18]. In [9], the authors implement various decentralized and hierarchical control ideas for the actuation allocation problem of an air-jet system. Distributed temperature control of thermal processes is addressed in [13], and the authors focus on multivariable distributed control in order to maintain a uniform temperature across a wafer during ramp-up. In [16], [18] the authors present methods where resources are allocated using geometric and adaptive techniques in order to utilize a heat source in designing a model-based control signal. In [19], the authors describe a lithographical system that is heated by 49 independently controlled zones. Distributed temperature control methods have been used to improve the performance of some devices in personal computers [20]. More recently, in [21] the authors studied dynamic resource allocation strategies for different processes (e.g., a multizone temperature control with 16 zones).

Our approach is novel with respect to what is found in the literature. Broadly speaking, we demonstrate for the first time the utility of merging the fields of engineering and behavioral ecology by using models from foraging theory to address an important class of distributed temperature control problems. Specifically, we show how a bioinspired distributed decision-making system (i.e., multiagent system) that communicates over a network can be used to control a complex dynamical system. The communication network consists of two "clients," which obtain data from each of the temperature zones, connected to a central "supervisor" that controls agent actions across the grids. Network delays from this topology as well as disturbances, such as interzone effects, ambient temperature changes, and wind, introduce additional challenges that, when overcome, highlight the robust nature of the agent-based controller. Restrictions on the number of agents and the amount of lamp voltage they can apply give our control strategy characteristics of dynamic resource allocation problems such as those mentioned above. Overall, while applications of foraging theory to autonomous vehicles are studied in [3], [4], [22], this is to our knowledge the only other existing control engineering application of foraging theory. This paper should, thus, be viewed as early work, but with results that show clear paths to further exploit concepts from mathematical behavioral ecology in engineering.

We begin by discussing the theory of the prey model, and how this theory can be used in a multizone temperature control experiment. Then, we show implementation results for tracking and regulation problems where we specify a desired temperature that must be reached by each zone in the temperature grid. In order to test the controller's performance, we add a disturbance and we limit the number of zones that a forager can search. Finally, to explain a key feature of the emergent behavior of the temperatures in our experiment, we discuss connections between our foraging algorithm approach and the "ideal free distribution" (IFD) [23], an idea from theoretical ecology that has recently been found to have potential uses in engineering applications [24]–[26].

II. Foraging Model

From a biological perspective, the environment of a foraging animal comprises prey or food items that are spatially dispersed. Each forager's primary "goal" is to obtain energy, and the only way in which to do so is by searching for, attacking, and consuming prey. The forager must make decisions about how to interact with its environment to maximize some correlate of Darwinian fitness. If there are different types of prey, which types should be attacked? Why not specialize on particular types to avoid wasting time on substandard prey? On the other hand, why not generalize and take advantage of all opportunities? This optimal diet problem is studied using the so-called prey model from foraging theory [1]. The work in [3], [4] discusses the applicability of this theory to many engineering problems by viewing a forager as an agent and sources of energy or prey

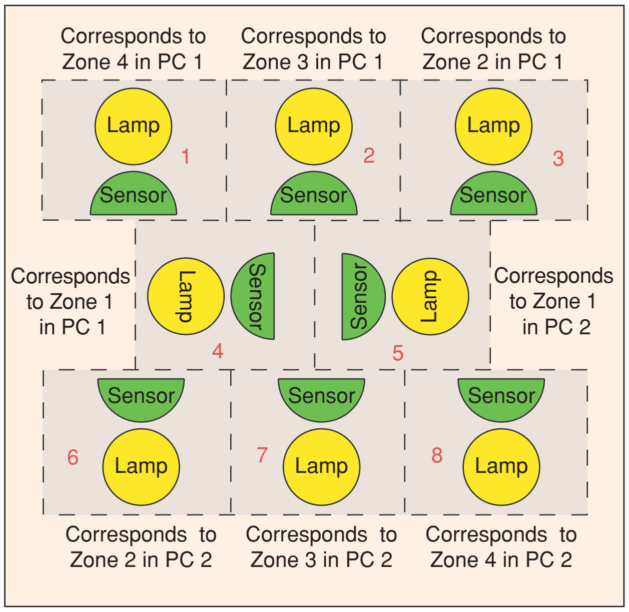

Corresponds to Zone 4 in PC 1
Corresponds to Zone 3 in PC 1
Corresponds to Zone 2 in PC 1
Lamp
Sensor
1
2
3
Corresponds to Zone 1 in PC 1
4
5
Corresponds to Zone 1 in PC 2
6
7
8
Corresponds to Zone 2 in PC 2
Corresponds to Zone 3 in PC 2
Corresponds to Zone 4 in PC 2

FIGURE 1 Zone layout on a single temperature grid. Each zone contains a lamp and a temperature sensor.

as tasks that must be processed. Here, using this general agent-based terminology, we provide an overview of the prey model following the treatment in [1].

The prey model describes an agent searching for tasks of different types in a particular environment. Each task holds a certain "point value" corresponding to the increment in the success level of an agent if it successfully processes the task. Processing is the equivalent of a biological forager handling prey. Point values quantify reward. The agent must search for tasks, recognize a task once it is encountered, and then decide whether to process the task based upon this recognition. The prey model, in its original form, assumes that a task is recognized correctly, that no time is required for recognition, and that the goal of the agent is to maximize its average rate of point gain.

Let there be n different types of tasks in the environment described by: e_i, the expected time required to process a task of type i; v_i, the expected number of points obtained from processing a task of type i; λ_i, the average rate of encounter with tasks of type i while searching; and p_i, the probability of processing a task of type i if it is found and recognized. Encounters with type i are assumed to be sequential and to follow a Poisson process. While we have assumed no explicit cost for time spent searching, one may be accounted for by redefining v_i [1]. The average rate of point gain J for the agent is the expected number of points obtained divided by the expected total amount of time spent foraging, which includes both search time and time spent processing tasks. If an agent spends on average T_s time units searching, then we have

$$J = \frac{\sum_{i=1}^{n} p_i \lambda_i T_s v_i}{T_s + \sum_{i=1}^{n} p_i \lambda_i T_s e_i} = \frac{\sum_{i=1}^{n} p_i \lambda_i v_i}{1 + \sum_{i=1}^{n} p_i \lambda_i e_i}.$$

The probability of processing each task type is the decision variable for the agent. Thus, the goal of the agent is to choose the p_i that maximizes J. We first rewrite J as

$$J = \frac{p_i \lambda_i v_i + k_i}{c_i + p_i \lambda_i e_i},$$

where k_i is the summation of all terms in the numerator not involving task type i and c_i is a similar variable for the denominator. Differentiating with respect to p_i,

$$\begin{aligned}\frac{\partial J}{\partial p_i} &= \frac{\lambda_i v_i (c_i + p_i \lambda_i e_i) - \lambda_i e_i (p_i \lambda_i v_i + k_i)}{(c_i + p_i \lambda_i e_i)^2} \\ &= \frac{\lambda_i v_i c_i - \lambda_i e_i k_i}{(c_i + p_i \lambda_i e_i)^2}. \qquad (1)\end{aligned}$$

Note that if the numerator of Equation (1) is negative, then J is maximized by choosing the lowest possible p_i. Correspondingly, if the numerator is positive, then J is maximized by choosing the highest possible p_i. Therefore, because $0 \le p_i \le 1$, the p_i that maximizes J is either $p_i = 1$ or $p_i = 0$ for each i depending on the sign of $v_i c_i - e_i k_i$. This concept is known as the *zero-one rule*: to maximize its rate of point gain, an agent must either process a task of type i every time it encounters it or never process a task of type i. The question then is which tasks the agent should process and which tasks it should ignore. The answer must account for missed opportunity. If the rate of point gain that results from processing task type i is larger than that of searching for and processing tasks of other types, then the agent should process the task of type i. On the other hand, if the agent would gain more points by searching for other tasks and processing those, then the task of type i should not be processed. Summarizing, this results in the rule

$$\begin{aligned}&\text{set} \quad p_i = 0 \quad \text{if} \quad v_i/e_i < k_i/c_i \\ &\text{set} \quad p_i = 1 \quad \text{if} \quad v_i/e_i > k_i/c_i,\end{aligned}$$

where v_i/e_i is the rate of gain that results from processing task type i and k_i/c_i is the alternative rate of gain resulting from searching for and processing other task types.

We now describe the prey model algorithm in light of the above discussion. Denote the rate of point gain that results from processing type i (v_i/e_i) as the profitability of task type i, and rank the tasks in the environment according to their profitability such that $v_1/e_1 > v_2/e_2 > \cdots > v_n/e_n$. If type j is included in the agent's "task pool," those types that the agent will process once encountered, then all types with profitabilities greater than that of type j will be included in the task pool as well. Thus, the prey algorithm states that, after ranking the task types by profitability, include types in the task pool starting with the most profitable type until

$$\frac{\sum_{i=1}^{j} \lambda_i v_i}{1 + \sum_{i=1}^{j} \lambda_i e_i} > \frac{v_{j+1}}{e_{j+1}}. \qquad (2)$$

The highest j that satisfies this equation is the least profitable task type in the task pool. In other words, if task types in the environment are ranked according to profitability with $i = 1$ being the most profitable, and if type $j+1$ is the most profitable type such that the agent will benefit more from searching for and processing types with profitability higher than that of $j+1$, then tasks of types 1 through j should be processed when encountered and all other tasks should not. If the equation does not hold for any j, then all task types should be processed when encountered. A derivation of (2) is given in [1].

The exclusion of type $j+1$ does not depend on the rate of encounter with type $j+1$. This exclusion implies that if the expected missed opportunity gains exceed the immediate gains of processing a particular type, then it does not benefit the agent to process the type, no matter how often the agent encounters it. Equivalently, if a type's rate of encounter exceeds a critical threshold, then less profitable types should be ignored regardless of how common they are in the environment.

It is important to note that the prey model assumes the agent has knowledge of all parameters. In many engineering applications, it is reasonable to imagine the expected processing time e_i and the expected points obtained v_i to be parameters that are known or approximated. However, knowledge of the rate of encounter with tasks λ_i often may be an unrealistic assumption, and online estimation techniques may need to be used [3]. An additional assumption of the prey model is that the agent has infinite life; for example, infinite fuel for an autonomous vehicle. Also, since the rate of encounter is constant, an infinite number of tasks are assumed to exist. This idea might imply the ability of tasks to arrive within the environment, an infinite number of spatial task arrangements, or an infinite number of ways that the agent can move through the environment. More realistic, time-constrained situations are accounted for in a risk-sensitive version of the prey model [1]–[3].

III. Temperature System and Application of the Prey Model

The temperature system under consideration comprises a temperature grid divided into eight "zones" as shown in Figure 1. A zone contains a lamp and a National Semiconductors LM35CAZ temperature sensor. Two computers (clients) are each connected to four different zones via four analog inputs and four analog outputs on a DS1104 dSPACE card. The digital outputs transmit on/off signals from the client to the lamps, and the analog inputs transmit temperature data from the sensors on the grid to the client. The DS1104 card has eight analog inputs: four with 16 bit resolution and four with 12 bit resolution. Because of this, the use of two computers (and correspondingly two dSPACE cards) allows for all eight zones to have a 16 bit resolution input connection. Communication between each client and the supervisor takes place over a TCP/IP connection via Matlab. Data collected via dSPACE and the Real-Time Workshop is acquired with MLIB/MTRACE. In general, MLIB/MTRACE captures the data from the board and transfers the information to Matlab. This provides access to all variables from the application running on the dSPACE card, and allows the use of various Matlab commands. Using these capabilities, data is acquired and sent over the network connection to the supervisor. The supervisor gathers data from the clients, implements the appropriate control algorithm, and then sends information back to the clients as to which lamps should be turned on.

To apply the prey model to the temperature control experiment, we view a controller as an agent (animal) and a task (prey) as a zone with a specific error relative to the overall desired temperature. Hence, the number of task types n is free to be chosen as any reasonable number of discretized error quantities. In the specific application that follows, we choose $n = 100$ so that the agent may encounter any of 100 types of error. We choose these 100 types to span a range of error of 0°C (type 100) to 5°C (type 1) with the type distribution defined by the function

$$i = \text{ceil}\left(100e^{-1.2(T_d - T^k)}\right) \tag{3}$$

where $k = 1, 2, \ldots, 8$ is the zone number and "ceil" is the standard ceiling function for converting to an integer. Equation (3) is shown in Figure 2(a), where i is the task type and $(T_d - T^k) \in [0, 5]$ is the error of the kth zone with respect to the desired temperature T_d. This nonlinear function is chosen since it defines a larger number of types for errors with small magnitude, thus providing better accuracy near the desired temperature. The ceiling function discretizes the type, and a saturation function is used to assure that the error lies within the required domain. In other words, errors that correspond to T^k being above T_d are equivalent to 0°C errors and are considered to be of type 100.

The controller "moves" around the temperature grid by being randomly placed on one of the eight zones and detecting

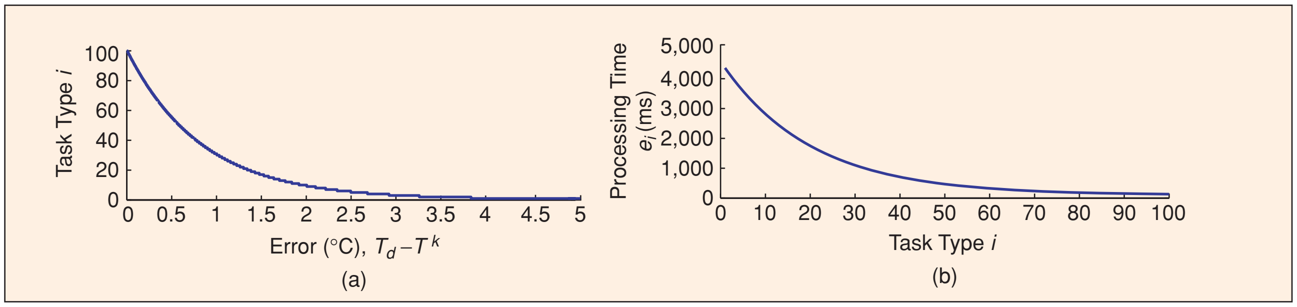


FIGURE 2 Parameter functions. Panel (a) depicts the determination of task type from encountered error, and panel (b) illustrates the processing times for each task type.

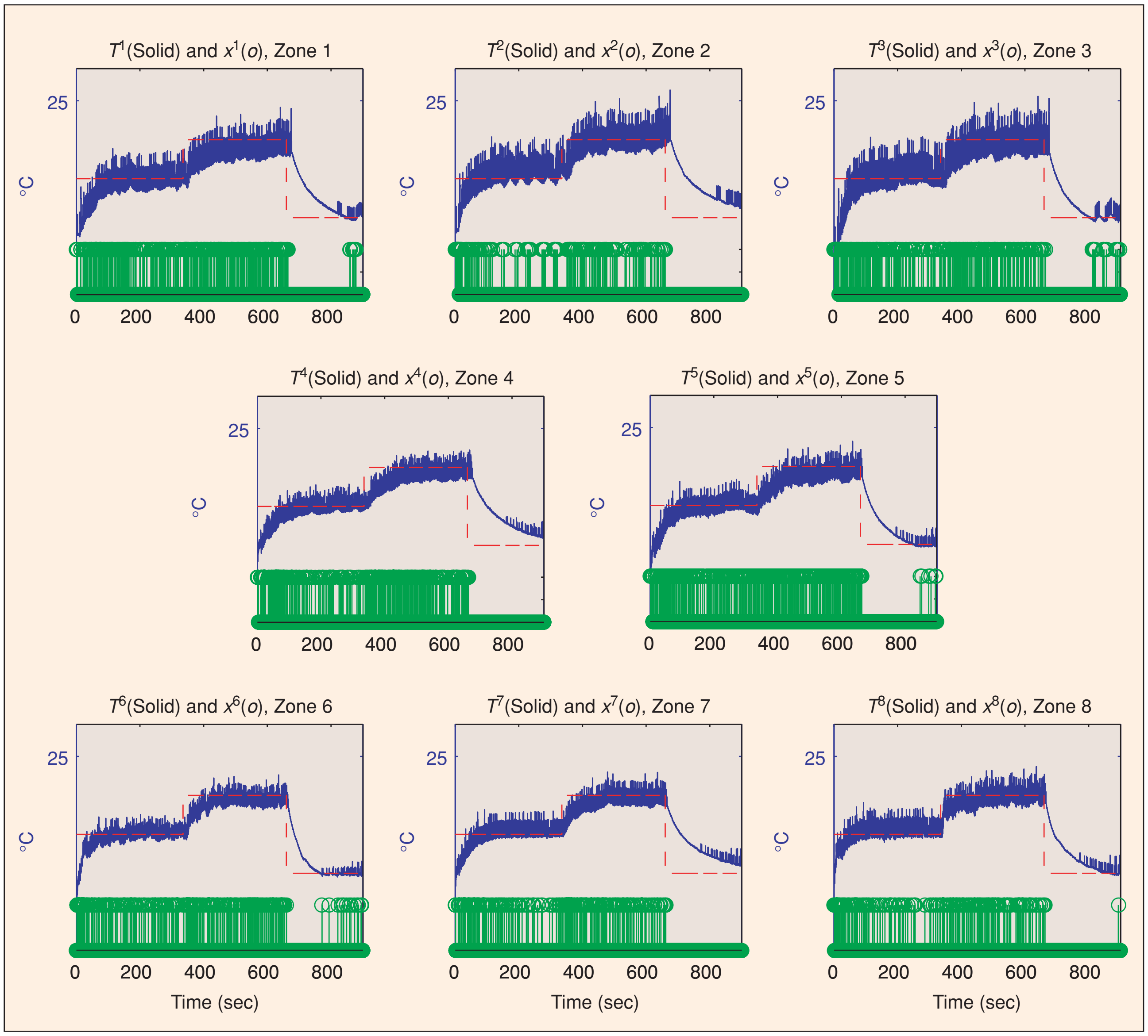


FIGURE 3 Multizone temperature control tracking performance, desired temperature (dashed), and actual temperature (solid) with plot layout corresponding to spatial zone positions in Figure 1. The stem plot at the bottom of each panel indicates the on/off state of the lamp x^i for zone i at a given point in time.

the temperature associated with that zone. Since the temperature of each zone constantly changes, placement of the controller on a specific zone does not imply an encounter with a particular task type. The type that the controller encounters depends on the temperature of the zone. The random zone selection adds to the stochastic nature of search and prevents oscillatory behavior that may result from systematic movement over the zones.

Processing times for each task type are determined by the function

$$e_i = 100 + 1000e^{-0.05(i-30)}$$

shown in Figure 2(b) where i is the task type. This function assigns longer processing times to task types that correspond to larger $T_d - T^k$ temperature errors since larger errors require a longer length of heating time by a lamp. The exponential characteristic of this function matches the distribution of processing times to the distribution of task types. Note that the processing times span a range of more than 4 seconds to 0.1 seconds. Although we examine one specific illustrative e_i, other processing-time functions may be chosen and will potentially result in different performance.

Task type point values are additional parameters that can be chosen freely and will affect the performance of the controller. Generally, task types corresponding to larger errors should have larger point values than task types corresponding to smaller errors. We choose point values according to the function

$$v_i = 101 - i$$

for $i \in \{1, \ldots, n-1\}$. This point gain function assigns point values ranging from 100 points for a type-1 task to two points for a type-99 task. For a type-100 task, we assign a negative point value since tasks corresponding to zero or negative error do not need to be actuated with the lamp.

Rates of encounter λ_i with different task types are estimated in real time as the experiment runs. The controller of an agent has a memory and is able to keep track of its number of encounters with a specific error. At any given time instant, an estimate $\hat{\lambda}_i$ of the rate of encounter with type i for that particular agent is calculated as the number of times type i has been encountered by the agent divided by the time that the agent has spent searching for tasks. Once the relationship between error and type, the processing time function e_i, and the point value function v_i are determined, the prey model algorithm described by (2) is implemented at each simulation time step using the rate of encounter estimates in order to determine which task types should be processed when encountered.

Summarizing, the controller agent is randomly placed on a zone (implying an encounter with, for example, task type i), $\hat{\lambda}_i$ is updated, and the prey model algorithm is calculated using the new rate of encounter estimate in order to determine whether the controller should stop searching for tasks momentarily and heat (with the lamp) the zone corresponding to the encountered task for the amount of time specified by e_i.

IV. Experiments and Results

To illustrate the ideas described in the theory, we performed three different experiments. The first is a tracking problem where the desired temperature is altered over time. The second experiment illustrates how the foragers reallocate when a disturbance is applied after the desired temperature is reached in the temperature grid. Finally, we limit the number of zones that each forager can search to see whether a desired temperature is achieved. In all of these cases we use four foragers, each of which use the prey algorithm described in the previous section. Our results show that the desired temperature is achieved despite sensor inaccuracy, noise, and network delays. We also highlight an interesting connection with the "ideal free distribution" (IFD) concept from theoretical ecology [23], [27].

A. Tracking

To evaluate controller tracking abilities, we alter the desired temperature over time. Initially, the desired temperature is set to $T_d = 23°\text{C}$. We then change T_d at 340 and 680 seconds to $T_d = 24°\text{C}$ and $T_d = 22°\text{C}$, respectively. The ambient room temperature is $T_a = 21.3°\text{C}$. Typical results are shown in Figure 3. We ran the experiment many times and found similar performances for other T_d values and ambient conditions.

The prey model algorithm achieves the desired result: the temperature of each zone tracks the desired temperature changes. Note that data is acquired from the sensors at a sampling rate of 1 ms. This value was chosen to obtain accurate estimates of the rate of encounter with tasks for a particular agent. Although the sensors are very accurate ($\pm 0.2°\text{C}$ typical

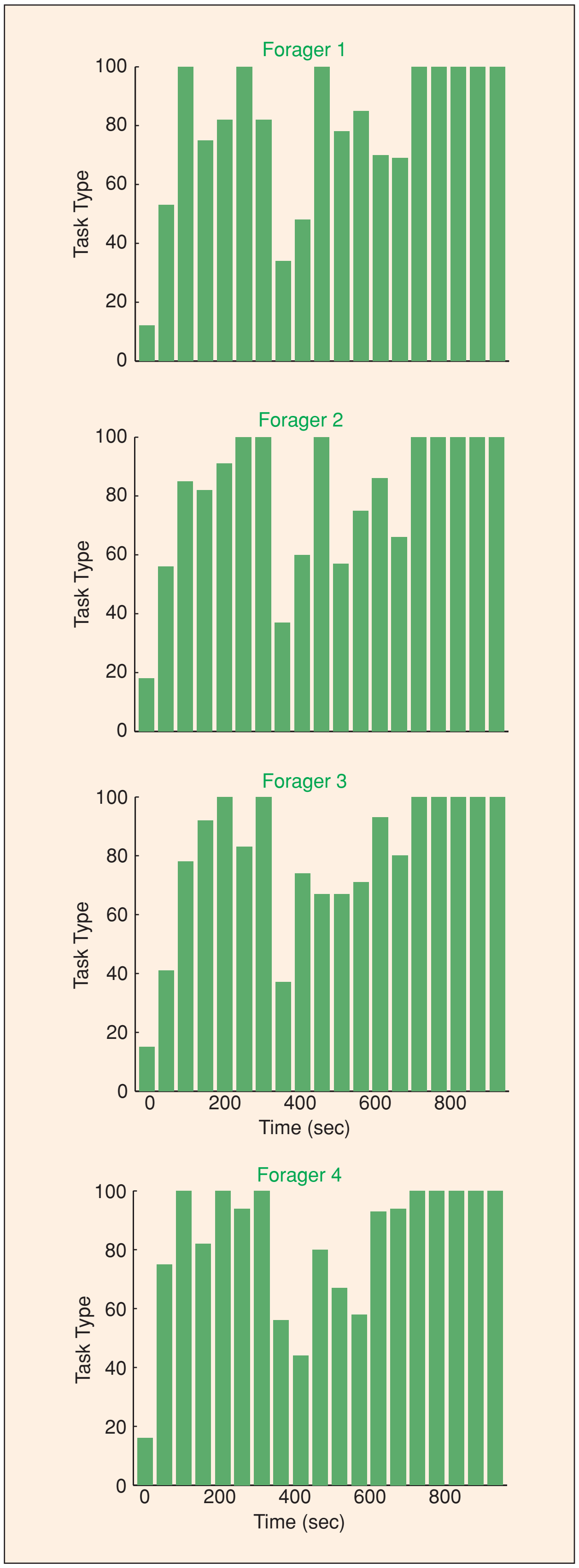


FIGURE 4 In this figure, we illustrate the task-type encounters of each forager with respect to time. Encounters are downsampled for visualization.

accuracy, and ±0.5°C guaranteed), their thermal constants are slow relative to the chosen sampling rate, yielding the noisy responses observed in Figure 3. Nevertheless, Figure 3 shows that on average good tracking performance is achieved.

The processing of tasks in each zone over time is indicated by the overlaying stem plot in Figure 3, which shows whether the lamp for a given zone is on or off at a particular point in time. It is clear from this plot that the decrease in desired temperature at 680 seconds results in no tasks being processed for a short period. This is because the temperature error of all zones during this time is negative (the actual temperature is above the desired temperature) implying encounters with only type 100 tasks, which have negative point value and are not worth processing. Also note that when the zones' temperatures are essentially at the desired temperature, some task types are still ignored. This is evident from the existence of gaps in the stem plot around steady state and is due to the exponential nature of the e_i curve. When all of the errors are small, the processing times of the encountered types do not differ much from one another (since they occur on the flatter part of the e_i curve). The controller then is not willing to waste time processing very tiny errors when it can search for and spend the same amount of time processing small (but not tiny) errors and receive a larger number of points.

The task types that each forager encounters over time are shown in Figure 4. Each forager encounters lower task types at the beginning due to the initial presence of large positive errors that correspond to low task types (Figure 2(a)). This characteristic is also seen when the desired temperature is increased at 340 seconds. As mentioned above, however, the second T_d change at 680 seconds causes encounters with only type-100 tasks because of a negative temperature error.

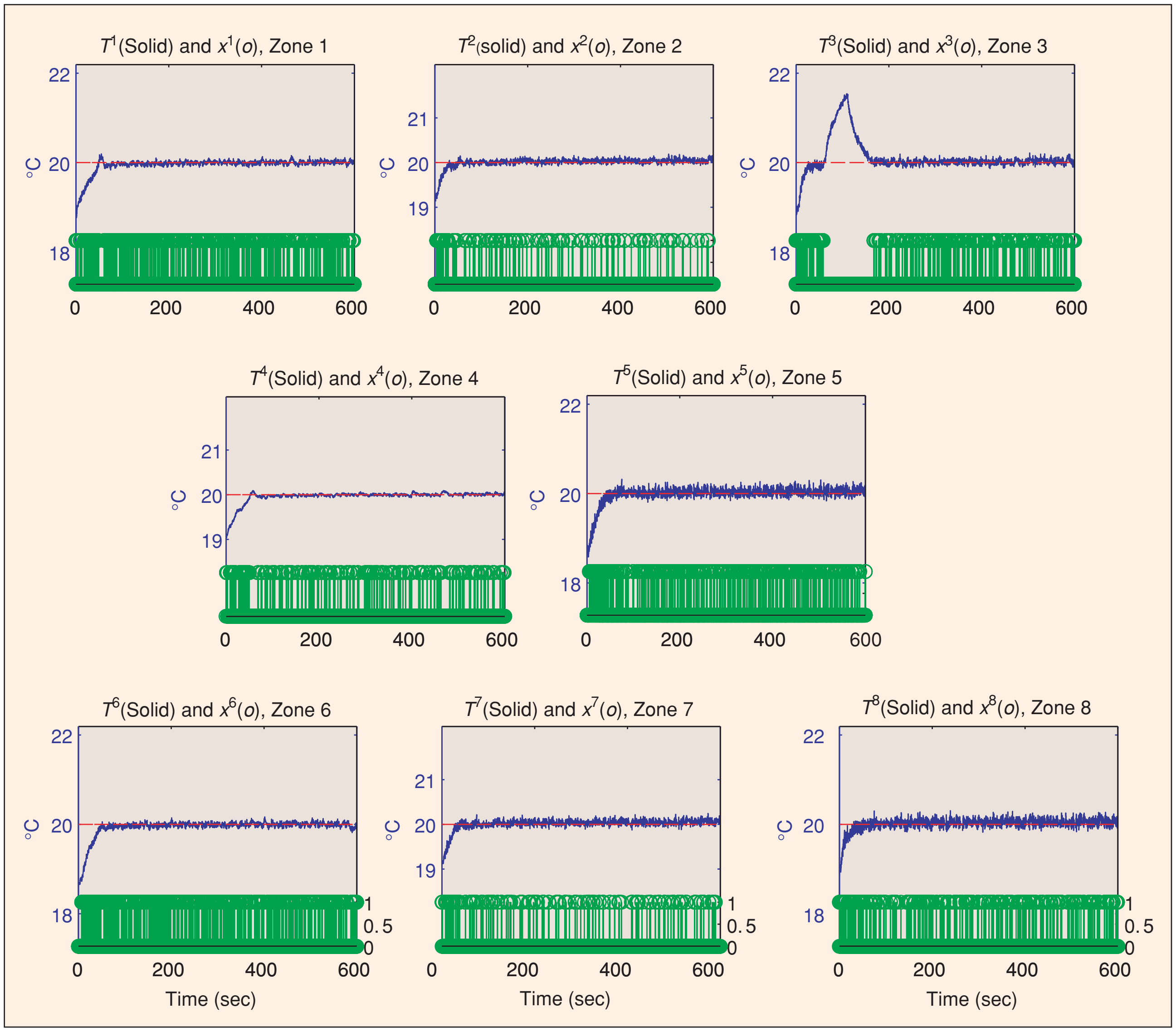


FIGURE 5 Multizone temperature control performance when a disturbance is applied. The desired temperature (dashed), the actual temperature (solid), and the lamps that are on (stem) are shown in the plot layout corresponding to spatial zone positions in Figure 1. The stem plot at the bottom of each panel indicates the on/off state of the lamp x^i for zone i at a given point in time.

B. Disturbance Effects

The tracking experiment showed that the prey model controller achieves good tracking performance. Here, we introduce a disturbance by means of an extra lamp located next to the sensor in zone 3. The idea is to regulate the temperature of the grid to a desired temperature $T_d = 20^\circ\text{C}$, and once a steady-state value is achieved, the new lamp in zone 3 is turned on.

The experiment was begun with an ambient room temperature of $T_a = 18.5^\circ\text{C}$. Figure 5 illustrates the temperatures and the lamps that were on for the first 10 minutes of the experiment. The controller was initially run in the absence of any disturbance, and the extra lamp in zone 3 was turned on after approximately 60 seconds. As seen in Figure 5, the temperature in zone 3 starts to increase, but the temperature in the other zones remains close to the desired temperature T_d. The disturbance is turned off around 110 seconds, and the temperature drops until it again reaches T_d. Figure 5 also shows the lamps that are on at a given time. We see that, during the time the disturbance was on, no foragers heated zone 3, even though the zone could have been selected by the algorithm. This behavior is expected considering the negative error in zone 3 after the disturbance. Foragers that randomly select zone 3 encounter a type-100 task and do not process it because it is not profitable enough to be included in the task pool. However, once the disturbance is turned off, and the temperature returns close to T_d, the foragers visit that zone in order to keep the grid at the desired temperature.

C. Search Limitations

The previous two experiments were based on the assumption that all foragers could sense the temperature in every zone of the multizone temperature grid. However, an interesting case

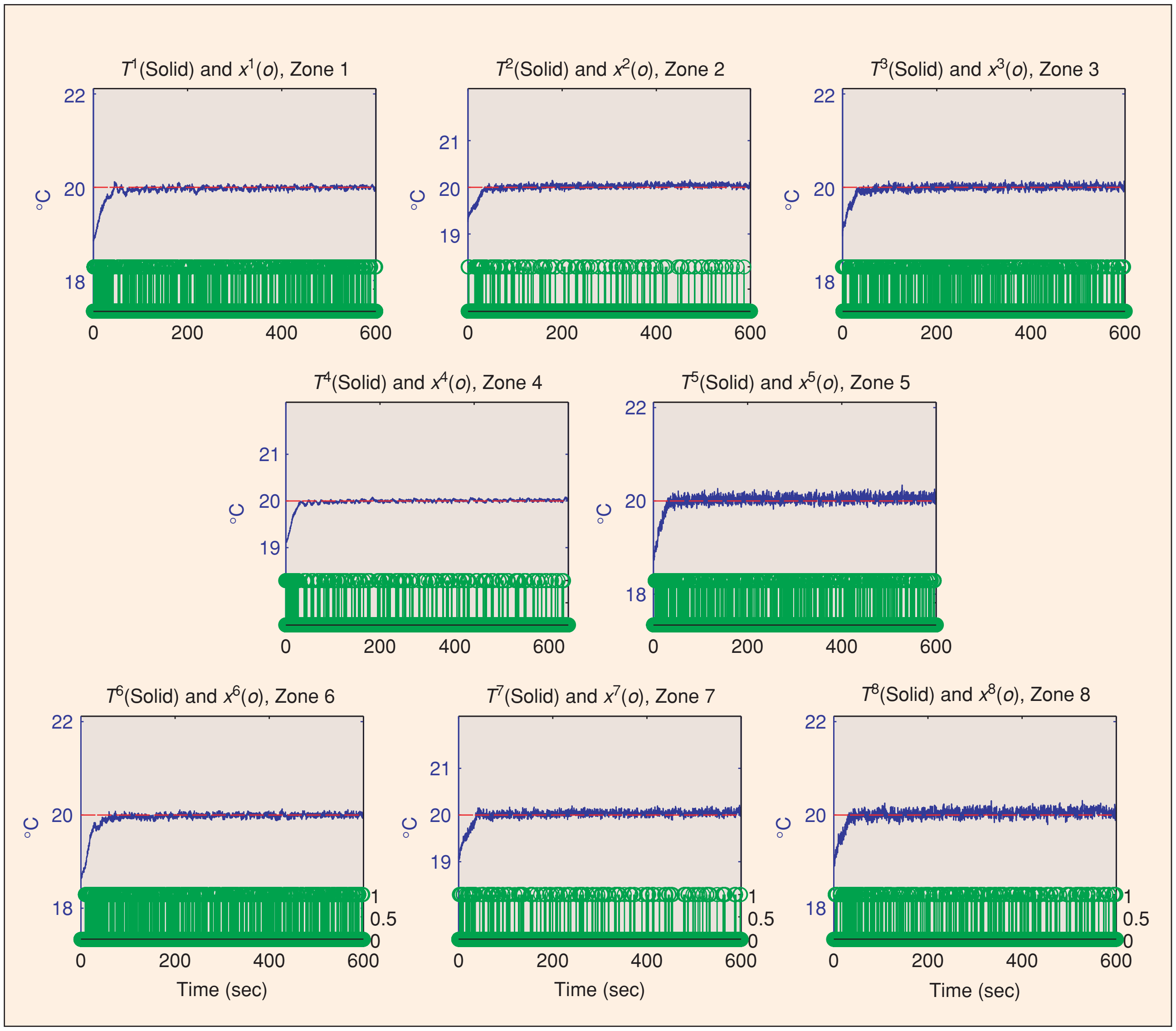


FIGURE 6 Multizone temperature control performance when there is not perfect information. The desired temperature (dashed), the actual temperature (solid), and the lamps that are on (stem) are shown with plot layout corresponding to spatial zone positions in Figure 1. The stem plot at the bottom of each panel indicates the on/off state of the lamp x^i for zone i at a given point in time.

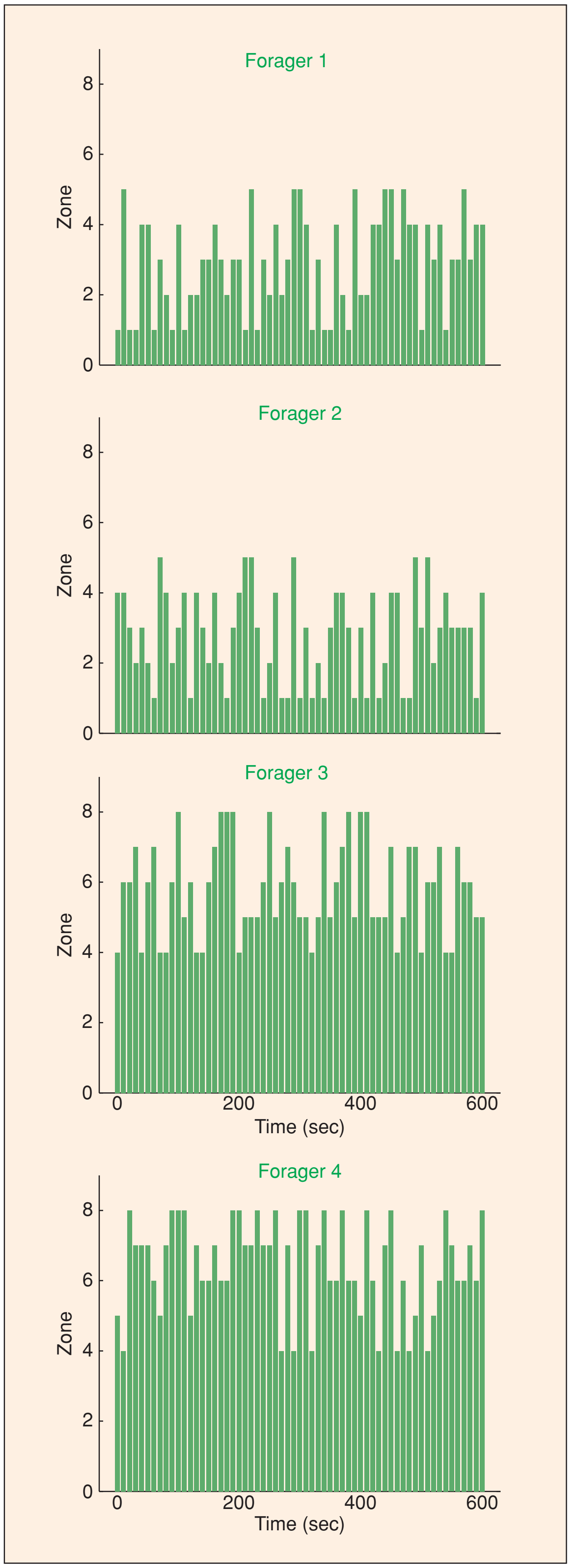


FIGURE 7 Corresponding zones for each of the four foragers.

arises when the foragers do not have access to all zones' temperatures. To investigate this issue, we divide the four foragers into two sets of two. The first set contains foragers 1 and 2, which are limited to search zones 1 through 5. The second set, comprising foragers 3 and 4, is limited to search only zones 4 through 8. Zones 4 and 5 are common to both sets. The prey model is still applied; however, the number of zones the forager can randomly encounter is limited.

The goal of this experiment is to regulate the temperature grid to a desired temperature $T_d = 20°$C when the ambient room temperature is $T_a = 18.9°$C. Figure 6 shows the results for 600 seconds. After 10 minutes, the temperature across the multizone temperature grid is essentially constant at the desired temperature. Some expected oscillations exist around the desired temperature due to the number of foragers, network delays, and sensing limitations. The search limitation is evident in Figure 7: only two foragers search zones 1 through 5, while the other two search zones 4 through 8. These limitations, however, do not affect tracking performance. The desired temperature is reached even in the absence of perfect information.

D. Discussion

Our results show that the temperature of each zone reaches each desired temperature that is applied despite several performance-limiting effects, namely resource allocation constraints, network delays, disturbances, and imperfect information. Resources are limited owing to the fact that only four foragers are allocated around the grid. Thus, the maximum number of lamps on at any given time is four. The result is at least four unattended zones that must experience a decrease in temperature and then later receive attention again due to the temperature error that has developed. This pattern yields slight oscillatory behavior in Figure 3 that is most evident in the zones where the desired temperature is reached the fastest. Network and communication delays also affect controller performance. The clients sample temperature data every 1 ms via the sensors and transmit this data back to the supervisor. Because these connections are implemented over a TCP/IP connection, internet delays exist. However, the controller performs quite well in the presence of such constraints. Additionally, good performance was achieved after the introduction of disturbances and information limitations.

It should be noted that the experiments were performed at different times of the day and year, leading to different ambient conditions. In addition to temperature fluctuations from experiment to experiment, wind current due to, for example, air conditioning systems and window drafts also exists. Although such adverse experimental conditions can significantly affect experiment results (e.g., making achieved steady-state behavior different, compare Figure 3 to Figures 5 and 6), the performance of the controller is quite good. Furthermore, parameter value and function tuning may lead to improved results. For example, the type function in (3), the processing-time function e_i, and the point-value function v_i were chosen in our experiments to simply demonstrate the utility of a foraging theory approach to control; however, different functions may be used and, with proper tuning, may result in improved performance.

An important characteristic of the experimental results given above is the connection to the "ideal free distribution" (IFD) concept from theoretical ecology [23], [27]. This concept has been used to analyze how animals (foragers) distribute themselves across different habitats. These habitats have different characteristics (e.g., one habitat might have a higher nutrient input rate than another), but animals tend to reach an equilibrium point where each has the same correlate of fitness (e.g., consumption rate). The term "ideal" means that the animals can sense the quality of all habitats and seek to maximize the suitability of the habitat they are in, and the term "free" means that the animals are free to move to any habitat. In the temperature control experiment, the group of foragers tends to reach an IFD. Why? If we think of temperature error in a given zone as a nutrient, the foragers will allocate themselves in the zones where the temperature is below the desired one, and they tend to choose those zones where the reward is higher (i.e., where they are getting a high number of points ν_i). In this way, they persistently move around the grid and maintain the same consumption rate for each of them that corresponds to the equilibration of the zone temperatures in Figures 3, 5, and 6.

V. Conclusions

The utility of foraging theory for decision-making system design was established in [3], [4] for autonomous vehicles via simulations. Here we examine an application of the theory to an actual physical experiment, namely temperature control of a grid of eight zones. A controller is thought of as an agent searching for error across the grid, and it uses the prey model algorithm to decide which types of error to actuate with a lamp to achieve a uniform desired temperature across the grid. The algorithm causes the controller to ignore certain types of error when the missed opportunity of more profitable types is too great to forgo. Results show that the controller does very well in tracking desired temperatures, even in the presence of disturbances and sensing limitations. The results also illustrate a connection between the prey model and the ideal free distribution. The desired temperature is reached in all three experiments by the foragers allocating themselves in the zones where the error is higher.

Future directions include applications of extended foraging theory concepts such as the patch model to determine how long to process certain error types and risk-sensitive foraging theory to decide which types to process when time is limited. Moreover, there is a need to mathematically analyze the stability of the controller; however, this is quite challenging due to the need to consider sensor noise, disturbances, lack of perfect information (i.e., decentralized control), asynchronous operation, and the fact that the control input is of the on-off type that is constrained so that only a limited number of zones can be heated at one time.

Acknowledgment

We would like to thank Jesús A. López for helping in the implementation of the TCP/IP connection using Matlab, and for giving us some insights regarding the MLIB Matlab-dSPACE interface. We would also like to thank Professor Thomas A. Waite of the OSU Department of Evolution, Ecology, and Organismal Biology for teaching us about behavioral ecology and for collaborating with us on other engineering applications. This work was supported in part by the OSU Office of Research. Kevin M. Passino is the corresponding author, and can be contacted at passino.1@osu.edu.

References

[1] D. Stephens and J. Krebs, *Foraging Theory*, Princeton, NJ: Princeton Univ. Press, 1986.
[2] A. Houston and J. McNamara, *Models of Adaptive Behaviour*, Cambridge: Cambridge University Press, 1999.
[3] B.W. Andrews, K.M. Passino, and T.A. Waite, "Foraging theory for decision-making system design: Task-type choice," *Proceedings, 43rd IEEE Conference on Decision and Control*, vol. 5, pp. 4740–4745, Dec. 2004.
[4] ——, "Social foraging theory for robust multiagent system design," *To Appear, IEEE Transactions on Automation Science and Engineering*, 2006.
[5] K. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.
[6] ——, *Biomimicry for Optimization, Control, and Automation*, London: Springer-Verlag, 2005.
[7] OSU Distributed Dynamical Systems Laboratory: http://www.ece.osu.edu/~passino/distdynamicsyslab.html.
[8] G.E. Stewart, D.M. Gorinevsky, and G.A. Dumont, "Feedback controller design for a spatially distributed system: The paper machine problem," *IEEE Transactions on Control Systems Technology*, vol. 11, no. 5, pp. 612–628, July 2003.
[9] M.P.J. Fromherz and W.B. Jackson, "Force allocation in a large-scale distributed active surface," *IEEE Transactions on Control Systems Technology*, vol. 11, no. 5, pp. 641–655, Sept. 2003.
[10] M. Boulvin, A.V. Wouwer, R. Lepore, C. Renotte, and M. Remy, "Modeling and control of cement grinding processes," *IEEE Transactions on Control Systems Technology*, vol. 11, no. 5, pp. 715–725, Sept. 2003.
[11] P.D. Jones, S.R. Duncan, T. Rayment, and P.S. Grant, "Control of temperature profile for a spray deposition process," *IEEE Transactions on Control Systems Technology*, vol. 11, no. 5, pp. 656–667, Sept. 2003.
[12] M. Zaheer-uddin, R.V. Patel, and S.A.K. Al-Assadi, "Design of decentralized robust controllers for multizone space heating systems," *IEEE Transactions on Control Systems Technology*, vol. 1, no. 4, pp. 246–261, Dec. 1993.
[13] A. Emami-Naeini, J. Ebert, D. de Roover, R. Kosut, M. Dettori, L.M.L. Porter, and S. Ghosal, "Modeling and control of distributed thermal systems," *IEEE Transactions on Control Systems Technology*, vol. 11, no. 5, pp. 668–683, Sept. 2003.
[14] M.A. Demetriou, A. Paskaleva, O. Vayena, and H. Doumanidis, "Scanning actuator guidance scheme in a 1-d thermal manufacturing process," *IEEE Transactions on Control Systems Technology*, vol. 11, no. 5, pp. 757–764, Sept. 2003.
[15] B. Bamieh, F. Paganini, and M.A. Dahleh, "Distributed control of spatially invariant systems," *IEEE Transactions on Automatic Control*, vol. 47, no. 7, pp. 1091–1107, July 2002.
[16] M. Alaeddine and C.C. Doumanidis, "Distributed parameter thermal controllability: A numerical method for solving the inverse heat conduction problem," *International Journal for Numerical Methods in Engineering*, vol. 59, no. 7, pp. 945–961, Feb. 2004.
[17] C.D. Schaper, T. Kailath, and Y.J. Lee, "Decentralized control of wafer temperature for multizone rapid thermal processing systems," *IEEE Transactions on Semiconductor Manufacturing*, vol. 12, no. 2, pp. 193–199, May 1999.
[18] M. Alaeddine and C.C. Doumanidis, "Distributed parameter thermal system control and observation by GreenGalerkin methods," *International Journal for Numerical Methods in Engineering*, vol. 61, no. 11, pp. 1921–1937, Nov. 2004.
[19] C.D. Schaper, K. El-Awady, and A. Tay, "Spatially-programmable temperature control and measurement for chemically amplified photoresist processing," *SPIE Conference on Process, Equipment, and Materials Control in Integrated Circuit Manufacturing V*, vol. 3882, pp. 74–79, Sept. 1999.
[20] P.E. Ross, "Beat the heat," *IEEE Spectrum*, vol. 41, no. 5, pp. 38–43, May 2004.
[21] N. Quijano, A.E. Gil, and K.M. Passino, "Experiments for dynamic resource allocation, scheduling, and control," *IEEE Control Systems Magazine*, vol. 25, pp. 63–79, Feb. 2005.
[22] P. Ulam and T. Balch, "Niche selection for foraging tasks in multi-robot teams using reinforcement learning," in *Proceedings of the 2nd International Workshop on the Mathematics and Algorithms of Social Insects*, Atlanta, GA, pp. 161–167, Dec. 15–17, 2003.
[23] S.D. Fretwell and H.L. Lucas, "On territorial behavior and other factors influencing habitat distribution in birds," *Acta Biotheoretica*, vol. 19, pp. 16–36, 1970.
[24] N. Quijano and K.M. Passino, "Optimality and stability of the ideal free distribution with application to temperature control," in *Proceedings of the 2006 American Control Conference*, pp. 4837–4842, 2006.
[25] J. Finke and K.M. Passino, "Stable cooperative multiagent spatial distributions," in *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference*, Sevilla, Spain, Dec. 2005.
[26] ——, "Stable emergent heterogeneous agent distributions in noisy environments," in *Proceedings of the 2006 American Control Conference*, pp. 2130–2135, 2006.
[27] T. Tregenza, "Building on the ideal free distribution," *Advances in Ecological Research*, vol. 26, pp. 253–307, 1995.