Diffusion-Based Path Planning in Mobile Actuator-Sensor Networks (MAS-Net): Some Preliminary Results

Kevin L. Moore[a], YangQuan Chen[a], and Zhen Song[a]

[a]CSOIS, Utah State University, Logan, UT, US

ABSTRACT

In this paper we present preliminary results related to path-planning problems when it is known that the quantities of interest in the system are generated via a diffusion process. The use of mobile sensor-actuator networks (MAS-Net) is proposed for such problems. A discussion of such networks is given, followed by a description of the general framework of the problem. Our strategy assumes that a network of mobile sensors can be commanded to collect samples of the distribution of interest. These samples are then used as constraints for a predictive model of the process. The predicted distribution from the model is then used to determine new sampling locations. A 2-D testbed for studying these ideas is described. The testbed includes a network of ten robots operating as a network using Intel Motes. We also present simulation results from our initial partial differential equation model of the diffusion process in the testbed.

1. INTRODUCTION

In this paper and its companion (in this same meeting[1]), we explore preliminary results and technical challenges, respectively, related to a concept initially proposed in.[2]

Our concern in both[1] and[2] is the need for homeland security, which has been highlighted by recent terrorism events. Along with the need to develop security systems to prevent terrorist acts comes the need to respond to such acts, should they occur. From this perspective, a particular concern is the security of hazardous materials storage facilities and capabilities to respond to chemical-biological-radiological (CBR) terrorism aimed at such facilities. Indeed, it has been suggested that terrorists view CBR storage facilities as possible targets. As such, there is an interest in a system that could assess and track the plume that would result from an attack on such a facility, and its associated concentration, as illustrated in Figure 1.

In[2] we proposed a unique, model-based approach aimed at this type of problem. The general system concept presented in that work is illustrated in Figure 2. Operationally,

1. A plume-generating event initiates deployment of the system. As shown, the idea is we have a small portable system of unmanned air vehicles (UAVs) that can be pre-staged or stored at fixed locations and which would be easily transported, deployed, operated, and maintained.
2. Using all possible data (such as weather information and information about the storage facility and the possible hazardous materials that have been released), the command and control system determines where sensor measurements should be made and directs the mobile sensors to appropriate locations.
3. The UAVs autonomously travel to the prescribed sampling locations within the plume, at which point sensors are used to assess the contamination hazards within the plume.
4. The sensor data is then transmitted to the ground station.
5. Using the up-to-date sensor data, the command and control system on the ground station updates its prediction of the plume location and then directs the UAVs to new sampling locations. Ground crew personnel would also relay this information to emergency management teams.

Center for Self-Organizing and Intelligent Systems (CSOIS), Utah State University, 4160 Old Main Hill, Logan, UT 84322-4160. Phone/Fax: (435)797-2924/2003. Email: http://www.csois.usu.edu

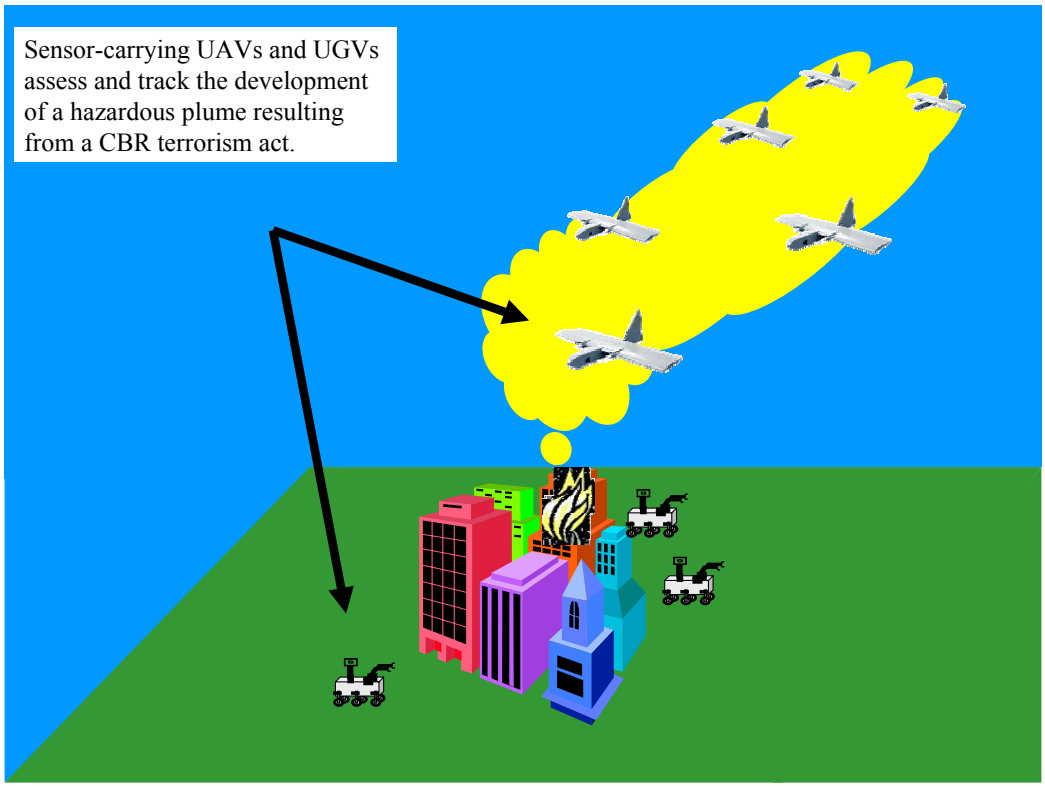

Figure 1. Overall system concept.

6. The UAVs travel autonomously to the new locations, take new samples, and the process repeats.

The most important and unique aspect proposed in[2] is related to Step 5: the prediction of the plume and selection of the "best" new sampling locations. In our approach we promote the use of a partial differential equation diffusion model (Diff-PDE) to predict the plume dispersal. The key feature of our ideas is the use of the actual sampled data to be used as additional boundary conditions or constraints each time the model is run. This approach leads to a spatially and temporally sampled distributed feedback control system.

In this paper we present preliminary results related to the problem depicted in Figures 1 and 2. Specifically, we present a description of the design and modeling of a two-dimensional (2-D) testbed for studying the problem. We begin with more discussion of our ideas about mobile sensor-actuator networks. This is followed by a description of the general framework of the problem. We then describe a 2-D testbed we are developing and present simulation results from our initial Diff-PDE model of the testbed. A discussion of the control-theoretic aspects of the problem is given in the companion paper.[1]

2. MOBILE ACTUATOR-SENSOR NETWORKS (MAS-NET)

The more sensors we use to measure something, the better the chances of obtaining a good estimate of whatever we are measuring. However, when the thing we want to measure is spatially distributed, deploying many sensors for collecting data over a relative large area can be difficult. In the past, the most common approach was to distribute sensors spatially in a fixed array. Such strategies led to the so-called "array processing" theory, which considers the data in a batch fashion. Recently, however, advances in robotics and computer networks, especially wireless networks, have led to the ability to develop arrays, or networks, of mobile sensors. We use the phrase "sensor networks" to describe these new systems. Additionally, the mobile platforms (robots) that deploy such sensors can also be outfitted with actuators. Thus, we can consider more generally the idea of mobile actuator-sensor networks, which we denote MAS-Net, though in this paper we restrict our attention to mobile sensor networks only.

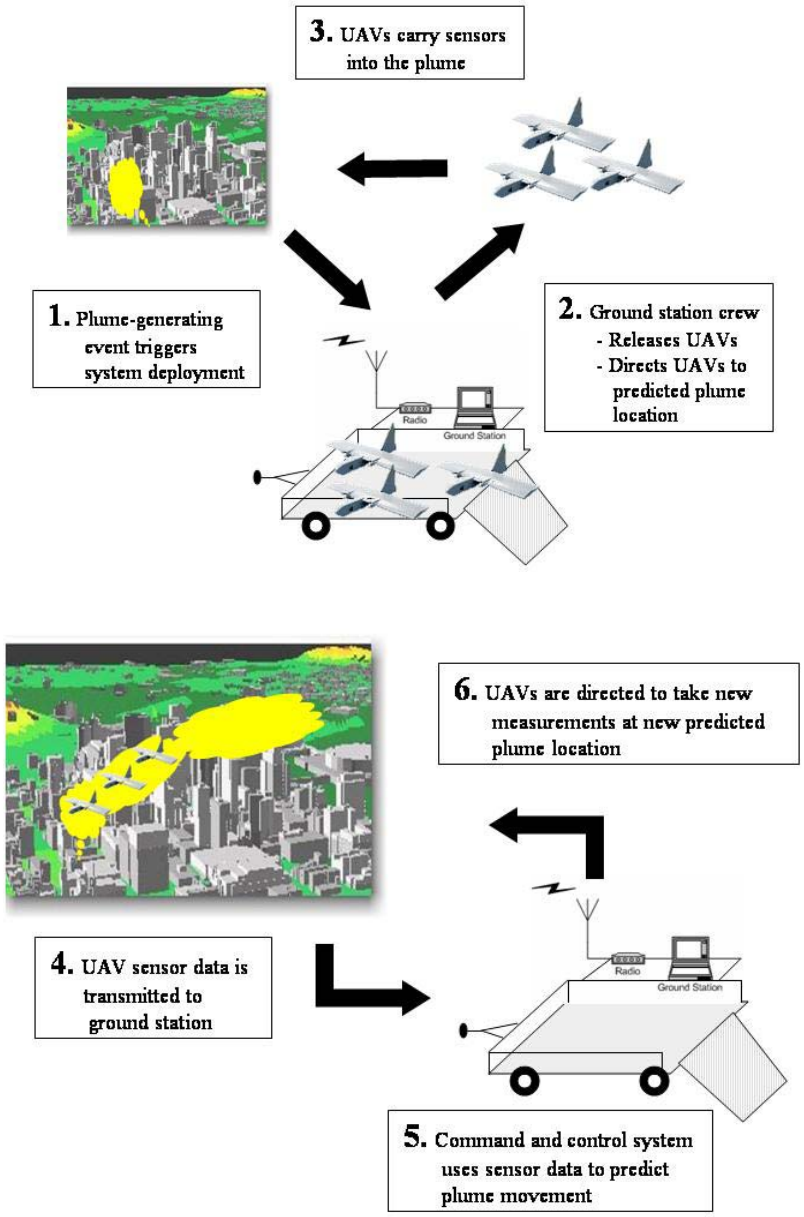

Figure 2. Detailed system concept.

From the perspective of the scenario of Figures 1 and 2, a key question is: "where should the next samples be taken?" In the context of mobile robotics, this is essentially a question of path planning. In our approach we will use a model-based (Diff-PDE) computation to answer this question. Thus, our ideas can be seen as a new model-based path planning methodology.

2.1. Additional Motivating Concepts

Before continuing, we digress a bit to note that the motivation of our research goes beyond the problem shown in Figures 1 and 2. Other scenarios include ground- and water-based systems:

- Determination of the safe ground boundary of the radiation field resulting from multiple nuclear radiation sources. In this case, an unmanned ground vehicle (mobile robot) is equipped with a radiation detector. The robots are actuated based on the spatial and temporal sensed information (radiation gradient, spatial position, etc.) integrated from all the robots.
- Determination of the nontoxic reservoir water surface boundary and zone control due to a toxic diffusion source in the water. In this case, the actuated or mobile sensors would be autonomous boats mounted with toxic chemical concentration sensors. Furthermore, we could assume that some (but not necessarily all) of the boats are equipped with the relevant neutralizing chemicals to make the water detoxified. By a proper design of distributed sensing and actuation/control strategies, it is possible to control the zone or shape of the toxic region to match the given desirable zone/shape. Now we have a complex distributed feedback control system that is more challenging than the networked actuators and sensors themselves.

We should note that to solve any of these problems, there are both theoretical and practical challenges, including:

1. Design of compact, low-cost, low-energy consuming, reliable robots as the mobile platforms that carry sensors and/or actuators.
2. Design of a reliable wireless communication system to link up all the robots as a network.
3. Collection and interpretation of the measurements from the networked sensors through models and data fusion algorithms.
4. Algorithms to direct the robot for the "best" possible active sensing of the diffusion.

Some of these problems, such as the design of a reliable wireless communication system and the selection of mobility platforms are essentially commercial-off-shelf (COTS) at this point. Our focus here will be directed to the more theoretical problems.

2.2. Sensor Networks

Before we discuss our path-planning approach, we give an overview and clarification of the concept of sensor networks. As noted above, the concept of a "sensor array" has been previously proposed, considering, as noted in,[3] "... the extraction of information from signals collected using an array of sensors." In the math formation of this reference, the difference among each sensor in the sensor array was described by phase shift only. In other words, this topic is in the range of traditional signal processing, studied by ordinary differential/difference equations (ODE) only, and thus assumes fixed sensor locations. A classical application of this theory is radar signal analysis. More recently, the concept of a "sensor network" has been defined as the "... use of multiple distributed sensors to collect information on entities of interest" in.[4] An explicit use of the word "distributed" implies the idea of of an underlying PDE to describe the phenomena of interest. A comparison of the notions of sensor networks and sensor arrays makes the following distinctions:

- Sensor location: The relative positions of the sensors in a sensor array are considered static, and most likely, uniformed placed. But, there is no such limitation on the sensor network.
- Sensing signal: This is not a hard rule, but normally when people discuss sensor arrays, they use ODEs to construct the signal model as shown in.[3] Meanwhile, it is common to see people formulate the sensor network problem using PDEs, such as,[2, 5–7] and other tools, such as probabilistic models.[8]

- Sensor type: The sensor arrays typically use the same type of sensor, such as radar or antenna, but the sensor network might mount different types of sensors on each mobile platform.[9]
- Communication: For an array of fixed sensors, communication can be achieved using wired channels, with fixed, reliable power sources. However, for a mobile sensor network we will need energy efficient, compact, low-cost, and reliable wireless communication channels. There are some papers dedicated to communication protocols,[10] or the system configuration.[11]

2.3. Related Work

The topic of sensor networks is quite comprehensive. It is common to find different people focusing on quite different aspects of the problem, with diverse methods and math tools, working from the perspective of distributed parameter systems, finite element analysis (FEA), diffusion modelling and parameter estimation, real-time system design, mechanical design, electronic design, reliable wireless communication, and software engineering, among others. Our particular focus is from a control engineering point of view, trying to answer questions such as: what is the requirement of the real applications? how do we drive the robot for boundary tracking? is it optimum? etc. We would point out the work of Masouds[12, 13] who presents papers on this topic from the control point of view, though instead of discussing the sensor network problem specifically, they talked about the path planning issue in general, using the classic approach of potential fields. Relative to our application, the potential field of interest is the chemical concentration of the plume. Other authors study sensor network from the aspects of computer science, communication engineering, computer networking, or detection and estimation theory. It is particularly useful to note the work of Arye Nehorai, et al., who published several papers[7, 14–16] on underwater contamination monitoring using a sensor array with moving sensors. These papers discuss this problem: an unknown source, e.g., a disposed container, releases toxic material under the ocean; how can we locate the source by deploying a group of chemical sensors? This problem is close to the problem we want to solve. The math formation provided in [16] is the following:

> When the contamination point is at $\vec{r}_0 = (x_0, y_0)$, we know that diffusion is subject to
>
> $$\frac{\partial c}{\partial t} = \kappa \nabla^2 c - \nabla \cdot (c\vec{v}),$$
>
> where c is the concentration, $\vec{v}$ is the velocity of the media, and t is the time. Then, the data from one sensor $y(\vec{r}, t)$, is defined by:
>
> $$y(\vec{r}, t) = c(\vec{r}, t) + b(\vec{r}, t) + e(\vec{r}, t)$$
>
> where
>
> - $c(\vec{r}, t)$ is concentration of the substance of interest at location $\vec{r}$.
> - $b(\vec{r}, t)$ stands for a "bias" term, representing the sensor's response to foreign substance that may be present.
> - $e(\vec{r}, t)$ is the noise, assumed to be Gaussian.
>
> Given an array of measurements $\mathbf{y}$, the problem is to find $\vec{r}_0$.

In the above papers, the authors used a maximum likelihood estimation approach to solve the problem. The concepts from array/signal processing, e.g. generalized likelihood ratio (GLR), Cramér-Rao bound (CRB), are frequently shown in the papers. They are different from our MAS-Net approach, since:

- The objective of MAS-Net is to identify the complete distribution instead of localizing the source of diffusion. We assume the source location is known.
- To simulate the real scenario, we need to consider that there are objects that interfere with the diffusion process. So, we used FEA theory during the simulation.
- We do not assume that the estimation noise e is Gaussian. Our math model does not involve statistical or probabilistic methods.

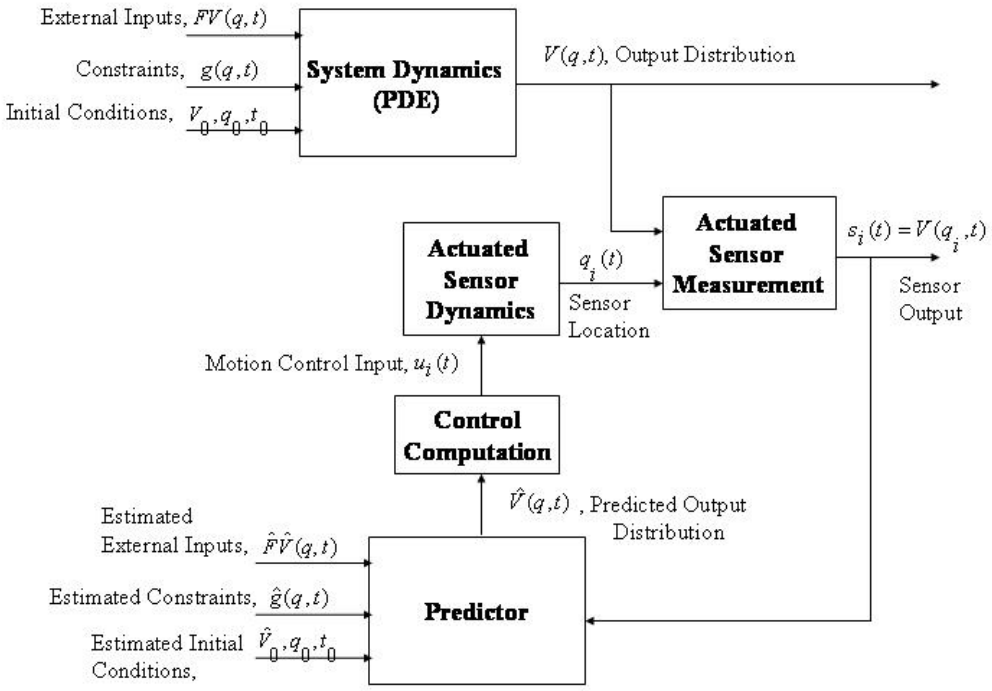

Figure 3. System block diagram (adapted from[2]).

3. GENERAL FRAMEWORK OF DIFFUSION-BASED PATH PLANNING

The general framework of the MAS-Net problem has previously been described in.[2] The technical concept is shown in Figure 3 (adapted from[2]).

As shown, the system to be characterized, which we think of as a "generating environment," is subject to the influence of media flow effects ($FV(q,t)$ in the figure), initial conditions (V_0, g_0, and t_0 in the figure) and constraints ($g(q,t)$ in the figure), where q denotes a location in space and t denotes time. The underlying PDE creates a spatial-temporal distribution of the concentration of interest (denoted $V(q,t)$ in the figure). Next, the i^{th} sensor, located in space at $q_i(t)$ takes samples, denoted by $s_i(t)$, which by definition is $s_i(t) = V(q_i,t)$. The measurements from all the sensors are then used by the predictor, along with estimates of the media flow, initial conditions, and constraints, to produce a predicted distribution, $\hat{V}(q,t)$. This is then used by a controller to decide on the control input $u_i(t)$ to the i^{th} mobile sensor. This decision is based on the solution of an optimal control problem with a suitable cost function. A more detailed formulation of this problem is found in.[2] Discussions about the control-theoretic aspects of this problem appear in the companion paper.[1]

4. USU MAS-NET 2-D TESTBED

4.1. System Configuration

As a means of exploring the algorithmic, computational, and practical issues of this problem, we have begun development of a 2-D MAS-Net testbed. The basic idea is shown in Figure 4. A sealed hollow box with obstacles that create multi-channel flows is covered with a transparent top. Air is circulated through the box and a contaminant is introduced into the flow. For our testbed we use a simple fog machine to generate the contaminant. To sense the fog's diffusion we have ten small COTS wheeled robots on the top of the box. The robots have the ability to move in both the x and y directions and are equipped with an optical sensor. The robots are configured into a wireless network as described below. The robots' "positions" are tracked using an overhead camera, which acts as a GPS system for the testbed. We also have encoders on the robots to help perform localization. In our working scenario, the sensor data from the robots is transmitted to the base station,

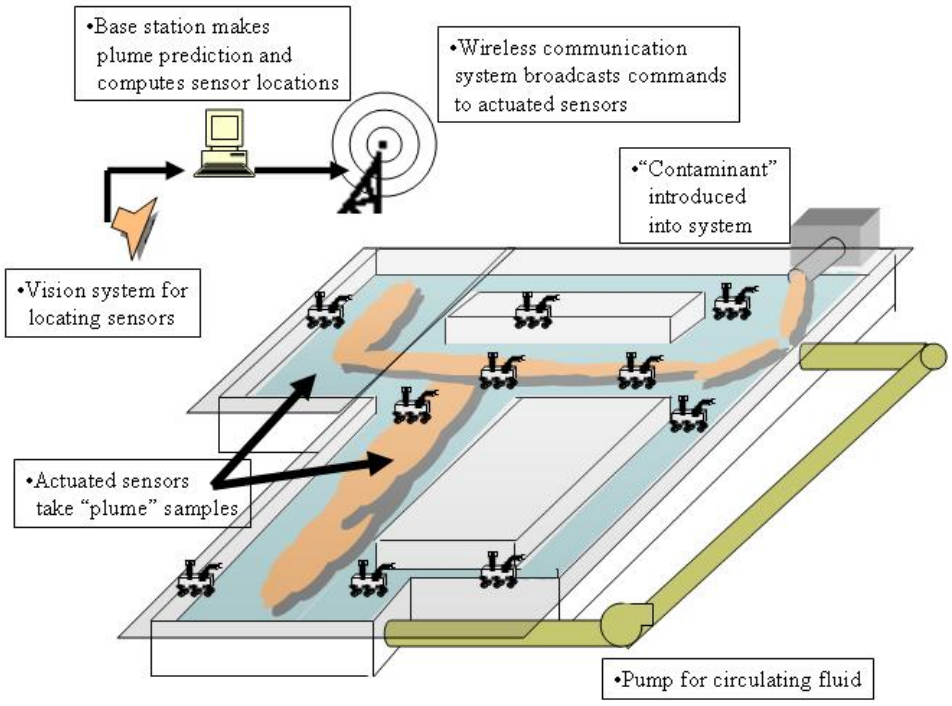

Figure 4. 2-D testbed.

which carries out a prediction of the spread of the fog, computes new sampling locations, and transmits these new locations to the robots.

We note that when considering the hardware and software platform, the MAS-Net system we have described is close to the system presented in,[17] which, as we have done, uses the MICA2[18] controller board with AVR Atmega128[19] microprocessor from Crossbow, the TinyOS[20] embedded operation system, and the nesC[21] programming language from Berkeley. However, the chassis for the robots in[17] is different than what we have used in our MAS-Net project (in[17] they used a control board called COTS-BOTS[22] to drive DC motors, whereas we used the MICA2 board to drive two modified servo motors). Other differences are:

- In,[17] the motivation is to solve a distributed pursuit-evasion game (DPEG). MAS-Net's aim is for distribution tracking for diffusion processes.
- Without a math model of the environment,[17] proposed to use a searching algorithm from the computer science point of view. In our application, we think the math model of the plant is important.
- In[17] they used different types of robots with different sensors. Our system is homogenous for now.
- Due to the heavy computation load of the PDE model, our system is more centralized, though the low-level control algorithm is executed on the local nodes. The system architecture presented in[17] is purely distributed. This feature implies differences in the software architecture, which is not the focus of this paper.

4.2. Hardware and Software

Here we give a brief overview of the hardware and software ideas for our MAS-Net testbed. At this time we have completed preliminary system design and component selection. Proof-of-concept testing of the Intel Mote-based communication system and the robot system has been completed (we refer to our robots as Motes since they use Intel's Mote technology[23]). Some of the pertinent details include:

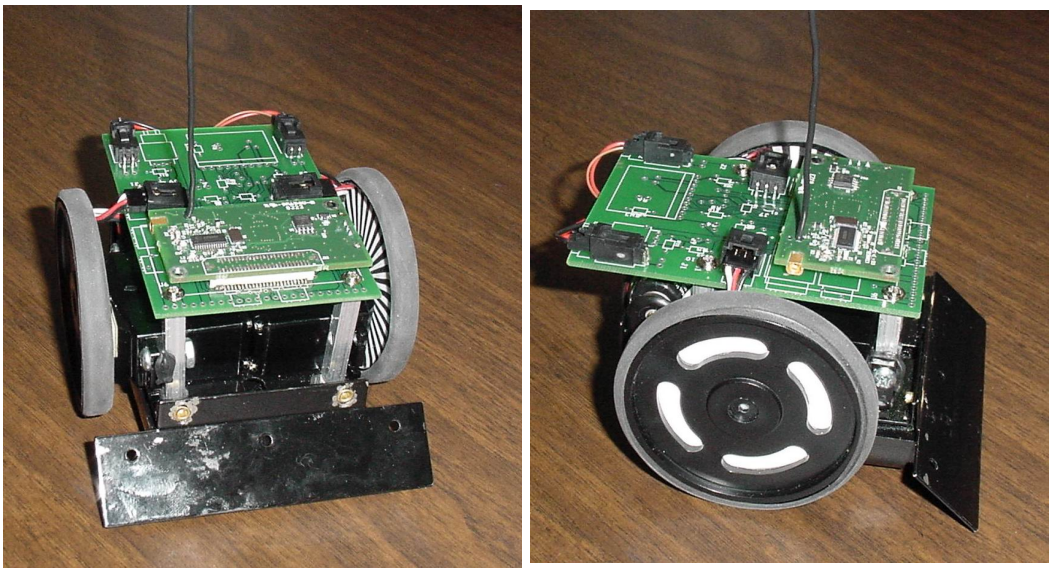

Figure 5. The Motes robot used in the MAS-Net system.

Robot Hardware. Figure 5 shows the robot system we are using. The chassis of the system is the Mark III.[24] As noted, we use a MICA2 as the controller board, which requires a 3V power supply. We mounted a 6V battery holder as the power supply for the two servo motors under the chassis. The servo motors are mounted symmetrically on left and right side for the differential driving. Typically, servo motors have a feedback loop inside, and rotate through a limited angle proportional to the input signal. Our servo motors are home-modified in order to control the rotation speed of the rotor rather than the rotation angle. On top of the standouts there is a printed circuit board (PCB) designed to integrate the peripheral circuit of motor control and sensors, and a socket to connect the MICA2 board.

Sensors. The sensors used by the system include:

- IR sensor: These are used for object detection and obstacle avoidance. We are using a Sharp GP2D12.
- Encoder: We mount two paper detection sensors for a simple encoder. On the inner side of each wheel, we paste a paper with black and white sectors. We can count the number of black and white sectors that pass these sensors, and then do an odometry computation. Hamamatsu photoreflectors were used as sensors.
- Optical sensor: We mount a photoresistor under the chassis for the fog concentration detection.

Communication System. The communication system follows the IEEE 802.11b protocol. On each MICA2 board, there is a CC1000 chip that responsible for the wireless communication. On the hardware side, we need to mount an antenna on the board. The MICA2 board plugged into each Intel Mote robot is the same as the one connected to the base station. On the software part, TinyOS provides a module called XNP, which is an implementation of the wireless communication protocol. For the Motes part, we just make sure that our application can talk with the XNP. As for the base station part, it is more complex. We are using the WinXP as the operating system (OS). On top of it, we run Cygwin to emulate the Linux OS. On the MICA2 connected to the base station, we installed a TinyOS which talks with the application running in Cygwin on the base station. Our graphical user interface (GUI) and camera-capturing application will be running under the Windows environment and talk with the application in Cygwin with TCP/IP protocol. This is a complex configuration, with three operating systems (WinXP, Cygwin and TinyOS) which need to cooperate properly. The reason for introducing this complexity is the constraints from different subsystems: First, our camera supports only the Windows environment. Second, TinyOS was developed under GNU/Linux. So, the easy way to port them into Windows is to use the GNU/Linux emulation. Since the communication system is not the focus of this paper, we will not give more details.

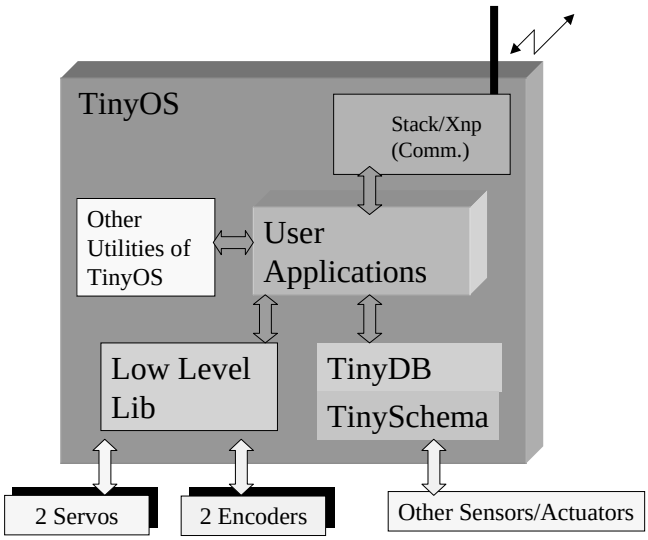

Figure 6. The software block diagram of each Mote robot.

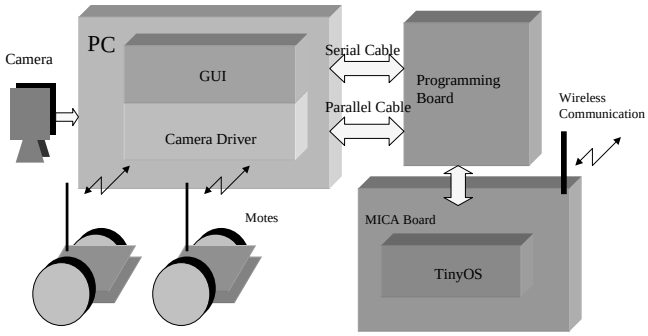

Figure 7. The system block diagram of the MAS-Net system.

Software The Mote system uses TinyOS as the embedded operation system and nesC as the programming language. The architecture we use on the robot, using TinyOS is shown in Figure 6. The overall system software block diagram is shown in Figure 7.

5. SIMULATION RESULTS

5.1. Math Model

We have conducted simple simulations to illustrate the prediction process depicted in Figure 3. To simplify the problem, we suppose that the air container is thin enough to be considered a 2-D problem. Thus, in the simulation, we consider a 2-D diffusion problem only. Figure 8 shows the simulation configuration we have assumed. The air flow enters at one corner (inlet) and exits at the diagonally-opposite corner (outlet). The "box" contains two objects that act as baffles to the air flow. We will assume the "fog" is injected at the air flow entry point. It is possible to model the concentration by the following equations:

$$\rho\frac{\partial \mathbf{V}}{\partial t} - \nabla \cdot \eta(\nabla \mathbf{V} + (\nabla \mathbf{V})^T) + \rho(\mathbf{V} \cdot \nabla)\mathbf{V} + \nabla P = \mathbf{F} \tag{1}$$

$$\frac{\partial c}{\partial t} + \nabla \cdot (\mathbf{V}c) - \nabla \cdot (D\nabla c) = 0 \tag{2}$$

where the parameters are

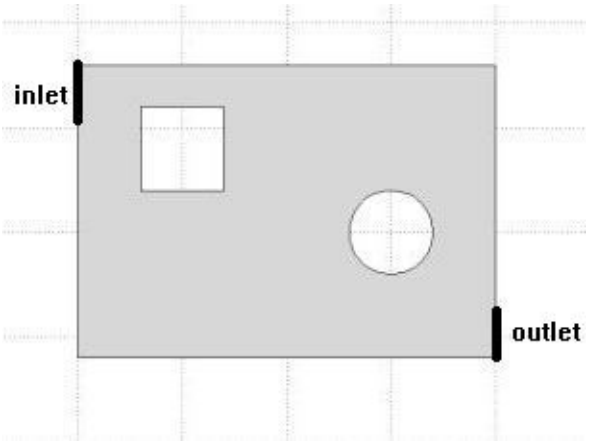

Figure 8. The objects (simulating obstacles) and the fog container for our simulation.

- **V**: velocity of the media (i.e., the air; calculated everywhere except at the boundary; velocity is time-invariant at the inlet and is initially zero everywhere else).
- $\rho = 1$: density of the media.
- $\eta = 1$: viscous of the media.
- P: pressure (calculated everywhere except at the boundary).
- **F**: the force applied to the diffusion chemical (fog; calculated everywhere except at the boundary).
- c: concentration (calculated everywhere except at the inlet and outlet; the initial value of c is zero everywhere except at the inlet, where a steady fog input concentration is applied).
- $D = 0.1$: diffusion constant.

At this point we do not have actual constants for our testbed, but for the simulation we have chosen approximate values. Equ. 2 is referred to as a subdomain condition, where the subdomain is denoted by Ω. Thus, Equ. 2 is applicable to Ω. For the border of the diffusion, i.e., the surface of the fog container and the objects inside, we need to apply other math relations called boundary conditions. The exact form of the boundary conditions we have applied will not be discussed here, other than to note that the simulation has been spatially separated into segments. The segments identified with the inlet and outlet of the flow are constrained using the Dirichlet condition:

$$hc = r, \tag{3}$$

where h and r are constants. This condition is applied to define the initial velocity of the media and concentration of the fog at the inlets and outlets. Pressure is initialized to be zero at the outlet and is calculated everywhere else from the solution of Equ. 1. For all other segments except the inlet and outlet, the boundary conditions are computed using the Neumann boundary condition for the diffusion, which is given as:

$$\mathbf{n} \cdot (c\nabla c) + qc = g, \tag{4}$$

where $\mathbf{n}$ is the normal vector of $\partial\Omega$, and q and g indicate the properties of the current segment of $\partial\Omega$ (the boundary of Ω is denoted as $\partial\Omega$).

5.2. Simulation Tool

We used Femlab[25] to solve the finite element problem and perform the simulation. Based on Matlab,[26] Femlab is a general purposes FEA software package. Due to limited space, we will not give a tutorial on Femlab or describe all the details of our simulation. Rather, we will outline our simulation procedures in brief.

One of the benefits of the Femlab is that it has embedded many physical problem-oriented math models, together with their solutions. For example, for our case, we need to consider the fluid dynamics of the air together

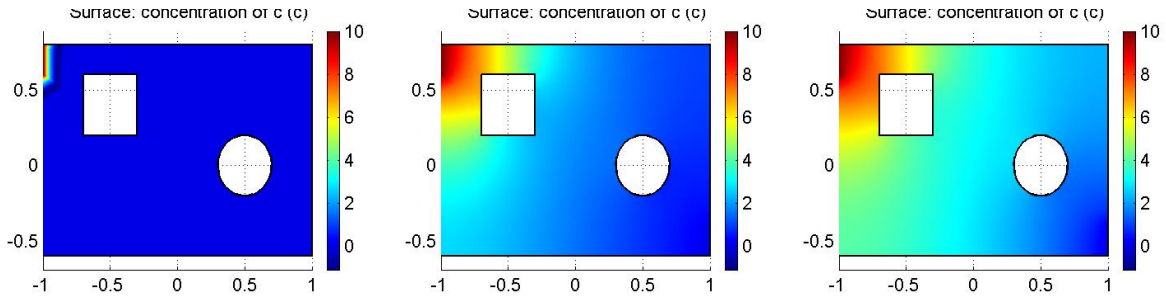

Figure 9. The simulation of the convection and diffusion of the fog.

with the diffusion of the fog. For the fluid dynamics computation, we can use the "incompressible Navier-Stokes" model in Femlab. This model includes Equ. 1 and its solution. All the coefficients of this model are defined before execution. Thus, without the information from other equations, Femlab can compute the velocity field for the air flow first. The second model to be simulated, "convection and diffusion," is Equ. 2, whose output is the concentration of the fog, c. The inputs to this model actually come from the "incompressible Navier-Stokes" model. That is, the output of the "incompressible Navier-Stokes" model is the velocity of the air flow, which is exactly the input of the "convection and diffusion" model. For our application, the quantity of interest is actually the concentration c only. In future work, the robots will measure the real concentration c, compare it with our predicted concentration $\hat{c}$ and adjust their behavior accordingly.

In Femlab, we selected the "Multiphysics" mode, and choose the two models as described above. In summary, we:

- Construct 2-D objects by the "create composite object" command. First, we draw a container and two obstacles on the screen. Second, we compose a new object via Boolean computation of those objects.
- Type in the numerical values of the coefficients for each model. We did this under the "subdomain" model, Ω. Thus, these rules apply to the inner part of the composite object.
- Setup boundary constraints, also under the subdomain model. We need to define boundary constraints with respect to each model. Also, for each model, the constraints for each segment of the boundary could be different. For the "incompressible Navier-Stokes" model, we choose one corner of the container as the inlet, and assigned the velocity of the incoming flow. Then, we choose another corner as the outlet, where the pressure is 0. All the other segments are "slippery" in the sense that they follow the constraint of Neumann equation. For the "convection and diffusion" model, we assign the inlet with a constant as the inflow volume, and assign the concentration of the outlet as 0.
- In the solver configuration dialog box, we assign the variable to be solved as the concentration c.

The results of these preliminary simulations are plotted in Fig. 9. The plots show the evolution of the concentration of the fog as time progresses. The leftmost plot shows the initial condition of the system. The middle plot shows how the fog has diffused after 10 seconds. Higher values of concentration are shown as the color progress from blue to red. The right-hand figure shows the concentration after 20 seconds. Using this data, in our future work we will then derive paths for the robots. For instance, based on the distribution of the concentration, we can calculate gradients, and drive the robots along these gradients until they arrive at the boundary of the fog, following the framework described in[12, 13] for the robot team path planning.

6. CONCLUSIONS

In this paper we have presented some preliminary results related to path-planning problems for diffusion processes. The use of mobile sensor-actuator networks was proposed in the context of a general framework of the problem. A 2-D testbed for studying these ideas was described, based on a network of ten mobile robots operating as a network using Intel Motes. Preliminary simulation results from our initial partial differential equation model of the diffusion process in the testbed were also presented. Future efforts will be aimed at a complete end-to-end demonstration of the ideas given in the paper.

ACKNOWLEDGEMENTS

The authors would like to thank the Space Dynamics Laboratory and the Center for Self-Organizing Systems at Utah State University for financial support of this project. We would also like to thank Zhongmin Wang, Anisha Aurora, Dan Stormont, and Bruce Christensen for their help with the project.

REFERENCES

1. Y. Chen, K. L. Moore, and Z. Song, "Diffusion boundary determination and zone control via mobile actuator-sensor networks (mas-net): Challenges and opportunities," in *Intelligent Computing: Theory and Applications II, part of SPIE's Defense and Security*, (Orlando, FL), Apr. 2004.
2. K. Moore and Y. Chen, "Model-based approach to characterization of diffusion processes via distributed control of actuated sensor networks," in *IFAC Symposium of Telematics Applications in Automation and Robotics*, (Helsinki University of Technology Espoo, Finland), June 2004.
3. D. G. Manolakis, V. K. Ingle, and S. M. Kogon, *Statistical and Adaptive Signal Processing*, Mc Graw Hill, 2000.
4. C.-Y. Chong and S. Pumar, "Sensor networks: Evolution, opportunities, and challenges," *Proceedings of the IEEE* **91**, pp. 1247–1256, Aug. 2003.
5. A. Bernoussi and A. E. Jaï, "Distributed parameter systems spreadability and evolving interfaces," in *14th Int. Symposium on Mathematical Theory of Networks and Systems*, (Perpignan, France), June 2000.
6. B. Porat and A. Nehorai, "Localizing vapor-emitting sources by moving sensors," *IEEE Trans. on Signal Processing* **44**, pp. 1018–1021, Apr. 1996.
7. A. Nehorai, B. Porat, and E. Paldi, "Detection and estimation of vapor-emition source," *IEEE Trans. on Signal Processing* **43**, pp. 243–253, Jan. 1995.
8. C. Sestok, M. Said, and A. Oppenheim, "Randomized data selection in detection with applications to distributed signal processing," *Proceedings of the IEEE* **91**, pp. 1184–1198, Aug. 2003.
9. L. E. Navarro-Serment, R. Grabowski, C. J. Paredis, and P. K. Khosla, "Millibots," *IEEE Robotics and Automation Magzine* , pp. 31–40, Dec. 2002.
10. H. Gharavi and K. Ban, "Multihop sensor network design for wide-band communications," *Proceedings of the IEEE* **91**, pp. 1235–1249, Aug. 2003.
11. H. Qi, Y. Xu, and X. Wang, "Mobile-agent-based collaborative signal and information processing in sensor networks," *Proceedings of the IEEE* **91**, pp. 1172–1183, Aug. 2003.
12. S. A. Masoud and A. A. Masoud, "Constrained motion control using vector potential fields," **30**, pp. 251–272, May 2000.
13. S. A. Masoud and A. A. Masoud, "Motion planning in the presence of directional and regional avoidance constraints using nonlinear, anisotropic, harmonic potential fields: A physical metaphor," **32**, pp. 705–723, Nov. 2002.
14. A. Jeremic and A. Nehorai, "Design of chemical sensor arrays for monitoring disposal sites on the ocean floor," *IEEE J. Oceanic Eng.* **23**, pp. 334–343, 1998.
15. A. Jeremic and A. Nehorai, "Landmine detection and localization using chemical sensor array processing," *IEEE Trans. on Signal Processing* **sp**, pp. 1295–1305, May 2000.
16. B. Porat and A. Nehorai, "Localizing vapor-emitting sources by moving sensors," *IEEE Trans. on Signal Processing* **sp**, pp. 1018–1021, Apr. 1996.
17. B. Sinopoli, C. Sharp, L. Schenato, S. Schaffert, and S. Sastry, "Distributed control applications within sensor networks," *Proceedings of the IEEE* **91**, pp. 1235–1249, Aug. 2003.
18. Crossbow, "MICA2." `http://www.xbow.com/Products/productsdetails.aspx?sid=72`.
19. Atmel, "Atmega128." `http://www.atmel.com/dyn/products/product_card.asp?part_id=2018`.
20. "TinyOS." `http://webs.cs.berkeley.edu/tos/`.
21. "nesC." `http://nescc.sourceforge.net/`.
22. S. Bergbreiter and K. Pister, "COTS-BOTS." `http://www-bsac.eecs.berkeley.edu/~sbergbre/CotsBots/cotsbots.html`.
23. Intel, "Intel's mote concept." `http://www.intel.com/research/exploratory/motes.htm`, 2002.
24. "Mark III Robot." `http://www.junun.org/MarkIII/Store.jsp`.
25. "Femlab." `http://www.comsol.com/`.
26. "Matlab." `http://www.mathworks.com/`.