S-NETS: Smart Sensor Networks

Yu Chen
University of Utah
Salt Lake City, UT 84112 USA
yuchen@cs.utah.edu

Thomas C. Henderson
University of Utah
Salt Lake City, UT 84112 USA
tch@cs.utah.edu

Abstract: The utilization of nonmobile, distributed sensor and communication devices by a team of mobile robots offers performance advantages in terms of speed, energy, robustness and communication requirements. Models of mobile robots with on-board sensors, a communication protocol and the *S-Net* system are established. Algorithms are defined for the *S-Net* which perform cooperative computation and provide information about the environment. Behaviors include robots going to or surrounding a temperature source. The simulation experiments show that the *S-Net* performs well, and is particularly robust with respect to noise in the environment. System cost versus performance is studied, and guidelines are formulated for which the *S-Net* system out-performs the non-*S-Net* system.

1. Introduction

At one extreme, mobile robots can be provided with a wealth of on-board sensing, communication and computational resources [1, 2]; at the other extreme, robots with fewer on-board resources can perform their tasks in the context of a large number of stationary devices distributed throughout the task environment [3]. We call the latter approach the *Smart Sensor Network*, or the *S-Net*. In this study, all the results are from simulation experiments using software (C and Matlab), and the performance of robot tasks with and without the presence of an *S-Net* (i.e., a set of distributed sensor devices) is evaluated in terms of various measures. See [4] for a more detailed account.

This approach can be exploited widely and across several scales of application; e.g., fire fighting robots. If mobile robots are used to fight forest fires, there may be several hot spots to extinguish or control. If sensor devices can be distributed in the environment, then their values and gradients can be used to direct the behavior of fire fighting robots and to transport fire extinguishing materials from a depot to the closest fire source. During this movement to and from the fire, collision avoidance algorithms can be employed. Sometimes coordinated activities are necessary and communication models are also important.

This study provides models for various components of study: (1) mobile robots with on-board sensors (2) communication, (3) the *S-Net* (includes computation, sensing and communication), and (4) the simulation environment. We have developed algorithms for the *S-Net* which perform cooperative computation and provide global information about the environment. Local and global frames are defined and created. A method for the production of global patterns using reaction-diffusion equations is described and its relation to multi-robot cooperation is demonstrated.

We provide the results of a set of simulation experiments designed to help us better understand the benefits and drawbacks of the *S-Net.* For behaviors of one mobile robot going to a temperature source, and multiple mobile robots surrounding a temperature source, in the ideal situation (which means no noise), the *S-Net* takes more time and distance. But when noise is added in, which is more realistic, the *S-Net* system is more robust than the non-*S-Net* system. For the task of multiple mobile robots going back and forth to a temperature source, there are thresholds above which the *S-Net* system out-performs the non-*S-Net* system.

2. Models

We have developed a mobile robot model, sensor models, an *S-Net* model, a communication model, and a model of the environment. The simulation provides a computational framework for the interaction of these models in terms of mobile robots performing useful tasks in the environment, and we define our simulation model as well. In order to act, a robot must receive current environmental information and calculate its movement based on the information received. On-board sensors (e.g., temperature, range, etc.) provide information about the environment and inform the robot's behaviors. In addition, the mobile robot may be able to communicate with other robots or the *S-Net.* The robot achieves movement by rotating or translating based on turning and motion primitives with given rotational and linear speeds.

The high-level behavior of the robot is specified by a program which maps the robot state and environmental information to primitive behavior sequences. For example, the behavior for the mobile robot to go to the closest temperature source includes: mobile robot sensing to get environmental temperature and gradients, turning as well as going forward to the source, and finally stopping when it reaches a certain distance from the temperature source.

Functions define source distribution of energy, material, etc. (e.g., heat, chemicals, etc.). The formula for distribution of temperature is:

$$T(x,y) = \frac{C}{\sqrt{(x-x_s)^2 + (y-y_s)^2} + 1} \tag{1}$$

where C is a constant related to the temperature source, (x_s, y_s) is the location of the temperature source, and (x, y) is the location at which we want to know the temperature.

Sensor models for temperature and range are both of the form:

$$\hat{T}(x,y) = T(x,y) + N(\mu, \sigma) \tag{2}$$

where $T(x, y)$ is the actual value at location (x, y) in the environment and N(μ, σ) is a normal distribution function with mean μ and variance σ.

The *communication model* consists of a protocol, a message layout, error model and performance characteristics. The protocol specifies the meaning of the bits in a message, as well as a set of commands for communication between robots and *S-Net* elements (*S-elements*). A group of *S-elements* sharing a common local frame is called an *S-clique*.

S-Net devices consist of three essential components: computation, sensing and communication. The *computation element* is described by the speed of the processor, its storage capacity, power requirements and cost. Sensors used by the *S-Net* devices are modeled as described above, but also include bandwidth, latency, power requirements and cost. The *communication model* is like that given for mobile robots, but includes power requirement and cost as well.

We use discrete event simulation with a fixed time step. In that we must model and simulate continuous events (e.g., during robot motion) as well as discrete events, we allow for an *every-time-step* event which can be put at the head of the event queue and must be handled every time step. Any number of these may be added to the event queue. The event list is a table recording all events that will happen in each time step. At the beginning of each time step, we copy it to a temporary list, and new events will be generated and added to the event list during the movement of the robots. During each time step, all scheduled events are handled and new events will be generated and added to the event list according to the different robot behaviors. At the end of each time step, the resulting state is evaluated to determine its feasibility. Once a possible state is achieved, the status of each robot such as position and local direction is updated. This procedure is repeated until the simulation terminates.

3. Algorithms

The *S-Net* is a collection of individual devices (*S-elements*) which have sensors, communication and computation abilities and are distributed either in a specified pattern or randomly to provide environmental information. The *S-Net* provides the framework for information gathering, analysis and presentation — it is an "information field" for the mobile robot. We propose three major algorithmic infrastructures for *S-Nets*:

- *Coordinate frames:* both local and global frames can be determined and exploited for robot tasks.

- *Pattern formation:* global patterns in the *S-Net* can be calculated and used to provide information for the robots [5, 6]; we use a stripe pattern to coordinate robot motion.

- *Level sets:* the ability to model and compute moving boundaries in the *S-Net* adds significant capabilities for robot tasks, including constrained shortest path information (see [7]).

4. Behaviors

A mobile robot's moving and turning behaviors are based on information provided by either its on-board sensors or the *S-Net*. A mobile robot may have four on-board temperature sensors located in different positions, thus providing four different spatial samples from which to compute the temperature gradient. The temperature gradient is used to control the heading of the robot. At the other extreme, with the *S-Net*, the robot will obtain the gradient information from the scattered *S-elements*.

The behaviors studied include:

- T_1: A single robot goes to a temperature source.
- T_2: Multiple mobile robots move to the temperature source and then cooperate and communicate to maintain a certain distance from the temperature source and to surround it.
- T_3: Multiple mobile robots going back and forth between the temperature source and a *Home* location (with the *S-Net*, *Home* is chosen as the origin of the *S-clique* that sensed the lowest temperature). Stripe patterns are formed along the gradient of the temperature source to *Home*. With no *S-Net*, an arbitrary location is selected.

5. Performance

We have compared the performance of mobile robots with and without the *S-Net* while solving the tasks: T_1, T_2, and T_3. For the robot behaviors that do not exploit the *S-Net*, the mobile robot obtains the information about the source (e.g., the temperature gradient) by itself. The mobile robot moves along the gradient towards the source, until the detected value (e.g., temperature) is a local maximum or is above some limit.

This set of tasks represents typical mobile robot tasks and can be configured to exploit many of the constraints described earlier. For example, a robot's path may be required to be the shortest, the gradient may be followed, or patterns in the *S-Net* may be used as road markers. Moreover, the last two tasks provide a setting to use multiple robots, ranging from few to many robots. In addition, robot interactions are necessary, at least as far as avoiding collisions. For each of these tasks, we propose a relevant set of performance measures, as well as a discussion of parameters and their possible values. Finally, we give the performance results and compare the two approaches.

Our goal is to find out under what conditions the *S-Net* system can outperform a non-*S-Net* system, and to study robustness and cost; cost is measured by time taken, distance traveled or total system cost. To summarize our results, we found that, for the first two behaviors (T_1 and T_2), the *S-Net* system does not perform better under ideal conditions (which means no noise at all). But when noise is added to the sensor data, we found that the *S-Net* system is more robust, especially in very noisy situations. For the third behavior (T_3), the *S-Net* system not only performs much better under realistic conditions, but also under ideal conditions. For certain round trip distance requirements, the

S-Net system can support more robots on the route while preventing collisions between robots. On the other hand, if there are too many robots on the route, in the system without the *S-Net*, some robots cannot move properly to prevent collision.

In these simulation experiments we test performance time and distance traveled with respect to sensor noise or variance (0 to 25), number of *S-elements* (100 to 300), and broadcast distance (1m to 2.5m) for the *S-elements.* According to [8, 9, 10], noise of a sensor includes inherent noise, transmitted noise, mechanical noise and so on. The temperature sensor model we choose here has a range of $[0, 1000]$, and we believe that 0.05% is a reasonable tolerance for the temperature sensors. That is why we choose σ^2 ranges from 0 to 25. Using standard statistical techniques, we compute 90% confidence intervals.

One Robot Goes to a Temperature Source and Multiple Robots Surround a Temperature Source: Our results in these two cases show that when the noise variance is above 10, for the mobile robot that utilizes the *S-Net*, the successful result does not change much. But for the mobile robot that uses on-board sensors, it so happens that the robot fails to locate the temperature source correctly – it takes the maximum time allowed for the task and does not locate the source. We believe that this is because the four on-board sensors are located too close to each other and cannot overcome the affect of noise. When noise is added to each sensor, the gradient computed from their values can have large error, which will further change the direction the mobile robot moves. One proposed solution to this problem is to have the mobile robot move to four widely spaced locations and get samples across a greater spatial scale to compute the correct gradient. This will certainly cost much more in time and energy. In fact, it also reduces the accuracy with which the robot can locate the source.

From all these measurement and comparison of the two systems, we can see that in the ideal situation, which means no noise, the *S-Net* takes more time and distance. Compared to the system without the *S-Net* (time used = 3.22 sec, distance traveled = 3.21m), our cost of time ranges from 3.6 sec to 5.5 sec, and distance traveled from 3.6m to 5.2m. But when noise is added, which is more realistic, the *S-Net* system basically does not change much, but the system without the *S-Net* gets much worse. We conclude that in real situations, especially a tough situation with lots of noise, the *S-Net* system will be more robust than the system without the *S-Net.*

Multiple Robots Go Back and Forth to the Temperature Source: This experiment is designed to explore the benefits of using the *S-Net* with regard to multiple cooperating mobile robots. The same behavior is used in each robot, so that by satisfying the same set of constraints, the robots can achieve the desired final result.

In the case that mobile robots use the *S-Net* (500 *S-elements*), *Home* is chosen as the origin of the *S-clique* that sensed the lowest temperature; this provides the longest path to the maximum temperature *S-element.* Then stripe patterns are formed along the gradient of the temperature source to *Home* (using the reaction-diffusion method). The straight line from *Home* to the temperature source (located at [10, 20]) is in the middle of the white stripe (pattern value is 1), the width of each stripe is a constant (5m in our case); black stripes (pattern value is 0) alternate spatially with white stripes. Different stripe patterns are formed for different random streams. The robots will move along the white stripe toward the temperature maximum and follow the black stripe *Home*. When a robot detects that a collision is about to happen, it will slow down to prevent the collision.

In the case that mobile robots do not use the *S-Net*, *Home* is arbitrarily located at the origin of environment (0, 0), and the temperature source is located according to the average distance from *Home* to the temperature source in the *S-Net* experiments. The purpose of this is to make sure that the distances of the round trips are basically the same for both setups. We believe that the gradient of temperature source to *Home* does not affect our experiment, so we choose the temperature source located in (40, 46). When robots detect an environment collision, they make a right turn and then try to get back on track again.

For the *S-Net system*, when the number of robots and trips increases, the average time used and distance traveled by each robot increase linearly, and there are no major deviations from linear. When we take a close look at the data collected, we find that on occasion, due to the particular random number streams, the result is not ideal, which means the robots cannot exactly follow the stripe, but get lost looking for the correct stripe. Under detailed analysis, we found that it is caused by some particular distributions of the *S-elements*. Since we use the origin with lowest temperature as the *Home*, it is sometimes possible that it is on the border of the stripe. There may not be enough *S-elements* on the border for black stripes, and then when the robots try to follow the stripe to go *Home*, they may not get enough information to keep on the black stripe, and thus move away. This can be solved by making more *S-elements* on the border or making *Home* far away from borders.

For the non-*S-Net* system, when the number of robots and trips increases, the average time used and distance traveled by each robot increase linearly. After a detailed analysis of the data collected, we found that when there are more than eight robots on the same path, several robots may lose control. This is related to the robot behavior chosen. While there are lots of other behaviors, we believe that this is a rather standard collision avoidance algorithm and representative of many implementations in physical systems. Using methods described in [11], we find that the confidence interval for the mean difference in these two experimental setups is (0.2647, 9.5464). Since it does not include zero, we can say with 90% confidence that there is no evidence to suggest that there is not a statistically significant difference. Figures 1 and 2 show how noise affects the performance in both cases. From these figures, we found that the one using the *S-Net* can handle noise very well, when the noise variance is about 10, the performance and trace of robots generally stay the same. But for the case that does not use the *S-Net*, noise variance has a huge effect on the performance of the robots, where even a tiny variance as little as 0.5 can cause the robots to lose control (the robots wander erratically). We conclude

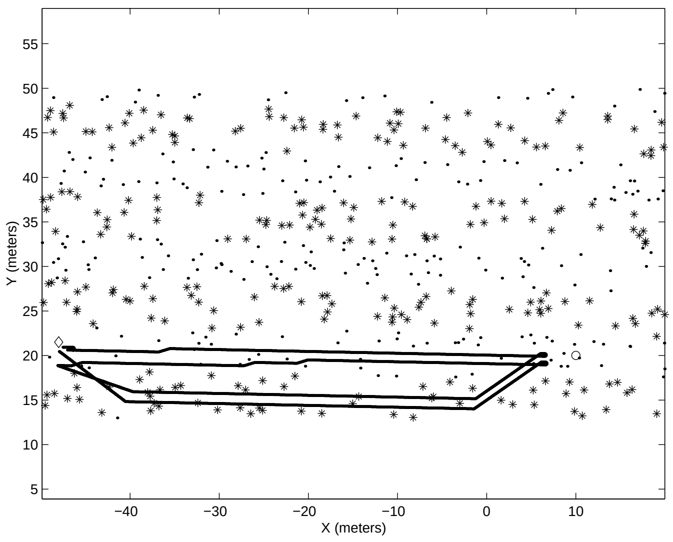

Figure 1. Trace of robots going back and forth with *S-Net* (2 robots, 2 round trips, noise = 10)

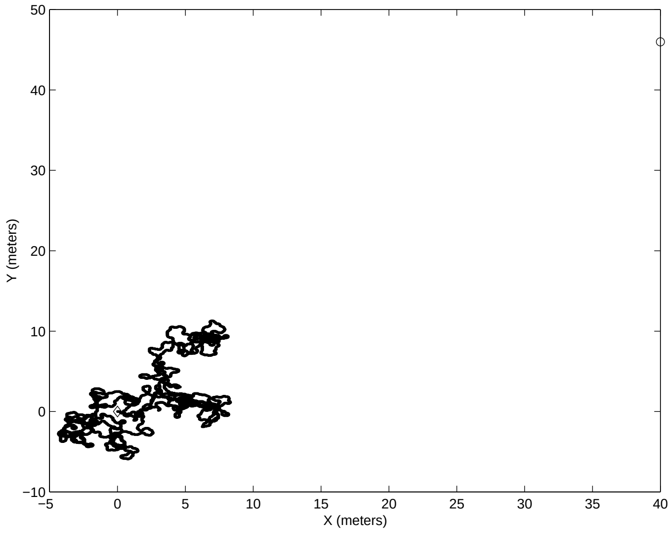

Figure 2. Trace of robots going back and forth without *S-Net* (2 robots, 2 round trips, noise = 0.5)

that, in terms of noise, the *S-Net* performs much better.

The performance measures used to this point have looked at success/failure, time to goal and distance traveled. Another crucial aspect is the more qualitative users' defined cost which is, in general, a function of the physical performance measures. For example, it may be that timeliness is extremely important, and the user may assign an exponential cost to time. Even if the cost is linearly related, it may have a steep slope.

To explore this aspect of performance cost, we have set up two models: (1) linear, and (2) quadratic. The three major terms included are:

- robot cost: we always assume this is linear in the number of robots.
- *S-Net* cost: we always assume this is linear in the number of *S-elements*.
- physical quantity (e.g., time and distance determined from simulation experiments): we apply a linear or quadratic form to this term.

In order to explore this issue, we examined linear and quadratic cost functions in terms of parameters in the equations in order to determine the existence of various cost relations to parameter values. We define the cost relation as:

$$C_l = C_s + C_p$$

where:

- C_l is the total cost
- C_s is the cost of the system infrastructure
- C_p is the cost of performance
- $C_s = N_r * C_r + N_{s-el} * C_{s-el}$
- $C_p = a_t * t^k + a_d * d^k$
- N_r is the number of robots
- N_{s-el} is the number of *S-elements*
- a_t and a_d are coefficients.

in which $k = 1$ in the linear case, and $k = 2$ in the quadratic case, t is the time taken to complete the task, and d is the distance traveled.

We compare the two systems (with and without the *S-Net*) by computing the percentage of cases for which the *S-Net* system outperforms the non-*S-Net* system (over the 100 cases of experiment – 1 to 10 robots making 1 to 10 round trips).

To establish C_s, we investigated mobile robot costs and a reasonable projection for *S-element* costs. For a given number of robots and *S-elements*, these costs are fixed and the cost variation comes from the C_p term. Rather than look at particular fixed a_t and a_d, we have assumed they are equal. Figures 3 and 4 show the percentages of times the *S-Net* outperforms the non-*S-Net* as a function of the coefficient value ($a_t = a_d$). As can be seen, for both the quadratic and linear cost function, there are thresholds below which the non-*S-Net* out-performs the *S-Net*. This indicates that for any particular implementation, a specific detailed analysis should be done to determine which is preferred.

In the quadratic distribution, we found that when a_t and a_d are chosen greater than 2, the percentage of times that the *S-Net* costs less is above 50%, which means the *S-Net* system is a better choice. In the linear distribution, when a_t and a_d are chosen greater than 2200, the percentage of times that the *S-net* costs less is above 50%. These graphs show that it is very likely that even in the ideal conditions, the *S-Net* is the better choice for a system with a

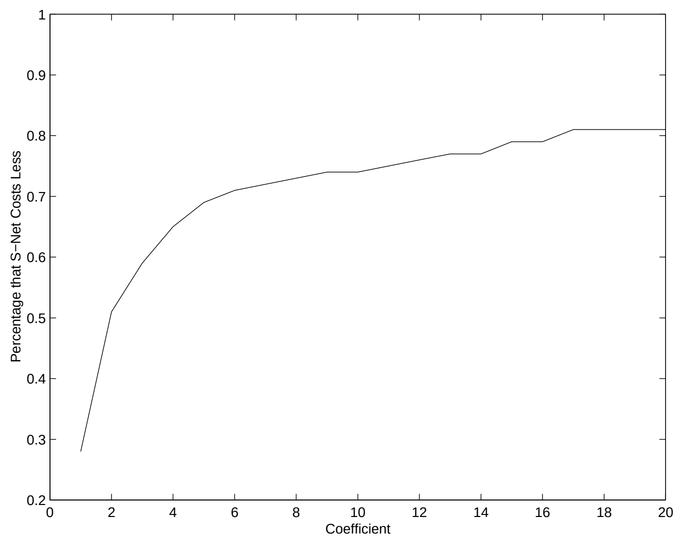

Figure 3. System cost comparison vs. coefficient in quadratic distribution

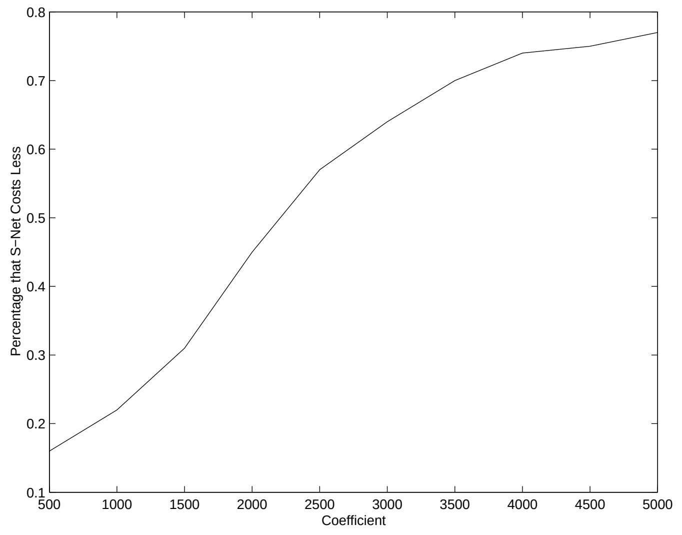

Figure 4. System cost comparison vs. coefficient in linear distribution

quadratic cost function. As an example, the measure of the circumference of a fire burning outward in a circular pattern grows quadratically with time, which could mean quadratic cost. In the linear cost case, it seems that the *S-Net* is a less performant option. However, it should not be overlooked that when noise is present, the *S-Net* dominates in performance.

6. Future Work

Future work includes:

- Physical implementation of the *S-Net* system, including the *S-elements* and mobile robots which exploit them.
- Optimization of the communication message code or layout, and study of communication errors, such as bad bits or lost message problems. In realistic situations, communication is never perfect, and communication error is an important aspect that needs to be handled in order to maintain the robustness of the *S-Net*.
- Development of a wider array of patterns to help multiple mobile robots cooperate. Computation of the shortest path with respect to realistic maps and topography would be useful.
- Exploration of gradient computation in the *S-Net*. The optimal computation of the gradient for a set of randomly sampled data has been solved for one dimension. It would be useful and interesting to expand the theory to two dimensions or above is a very interesting problem. By solving this problem, the efficiency of the *S-Net* can be further improved, and it's likely that the robustness of the system will be improved.

References

[1] Bares J E, Wettergreen D S 1999 Dante II: Technical description, results, and lessons learned. *Int J Rob Res.* 18(7):621-649 July

[2] Smith R, Frost A, Probert 1999 P A Sensor System for the navigation of an underwater vehicle. *Int J Rob Res.* 18(7):697-710 July

[3] Henderson, T C, Dekhil M, Morris S, Chen Y, Thompson W B 1998 Smart Sensor Snow. *IEEE Conf IROS.* Oct, pp 1377-1382

[4] Chen Y 2000 S-Nets: Smart Sensor Networks. MS Thesis, University of Utah

[5] Murray J 1993 *Mathematical Biology.* Springer-Verlag, New York

[6] Turing A 1952 The chemical basis of morphogenesis. *Phil Trans Roy Soc London.* B237:37-72

[7] Sethian J A 1999 *Level Set Methods and Fast Marching Methods.* Cambridge University Press, Cambridge UK

[8] Fraden J 1993 *AIP Handbook of Modern Sensors.* Americal Institute of Physics, New York

[9] Everett H R 1995 *Sensors for Mobile Robots: Theory and Applications.* A K Peters Ltd, MA

[10] Prasad L, Iyengar S S, Min H 1995 *Advances in Distributed Sensor Integration.* Prentice-Hall Inc, New Jersey

[11] Lilja D J 2000 *Measuring Computer Performance, A Practitioner's Guide.* Cambridge University Press, Cambridge UK