# Reactive Behavior in Self-reconfiguring Sensor Networks

Qun Li, Ron Peterson, Michael DeRosa, and Daniela Rus
Department of Computer Science
Dartmouth College

**Algorithm 1** Sending messages in the presence of variables sleeping times.

Let $(s, r)$ be the sending/receiving pair of sensors
Let $t_s$ be the sleeping time of $r$
Let $R$ be the sampling rate of $r$
Add $\frac{t_s}{r}$ waste bits to the head of the message from $s$ to $r$

## I. Introduction

We wish to create more versatile information systems by using adaptive distributed sensor networks: hundreds of small sensors, equipped with limited memory and multiple sensing capabilities will autonomously organize and reorganize themselves as ad-hoc networks in response to task requirements and to triggers from the environment. Distributed adaptive sensor networks are reactive computing systems, well-suited for tasks in extreme environments, especially when the environmental model and the task specifications are uncertain and the system has to adapt to it. A collection of active sensor networks can follow the movement of a source to be tracked, for example a moving vehicle, it can guide the movement of an object on the ground, for example a surveillance robot, or it can focus attention over a specific area, for example a fire to localize its source and track its spread.

We build on important previous work by [1], [5], [2] and examine in more detail reactive sensors that can adapt to task and to the environment. We develop: (1) a protocol for adaptively selecting the active region in a sensor network for the tracking application and (2) a protocol for distributing the information in the sensor network for guiding the navigation of a user through the network area. We present algorithms for each of these problems, discuss an implementation using the Berkeley Mote sensors [3] and present some preliminary hardware experiments.

## II. Adaptive Activation of Sensors for Tracking

The main reason for putting nodes to sleep in a sensor network is to conserve power and extend the network lifetime. Deciding how much to keep a node asleep depends on many factors, including the nature of the application, the data flow across the network, and the environment. Sensors do not receive any packets while in the sleeping node. Thus, the node has to wake up occasionally in order to decide its future on/off status.

In the Mote implementation, when a node wakes up from its sleeping mode, it listens to its incoming channel to see if there are any incoming bits. It will keep alert (in waking mode) when it senses there are some incoming bits from the channel. Suppose that the sending Mote knows the sleeping time of the recipient node, but it has no knowledge of the sleeping period. When the sleeping time is unknown, the maximum sleeping time is assumed. In the current Mote implementation every node in the transmission range of the sender waits till the end of each packet sampling, in order to decide whether it is a recipient for the packet or not. Let the sleeping time be $t_s$. Since the sender does not know the when the recipient will wake up, it should assume that the sleeping period has just began. In order to ensure that the recipient hears the message, the sender should transmit waste bits for $t_s$ time. The number of waste bits should be $t_s/R$ where $R$ is the sample rate for incoming bits. Algorithm 1 summarizes these ideas.

The intuition behind the tracking protocol is as follows. Suppose there is some information about the entry point of the object to be tracked. This can be a small area, the boundary of the sensor network, or the entire area covered by sensors. All the nodes in this entry area periodically sense for the event to be tracked. Sensing may range from simply detecting the presence of the object, to predicting where the object is heading and how fast it is moving. As soon as a sensor triggers, it broadcasts an alert message to the right nodes in the network. In the simplest case, the broadcast goes to all the neighbors. If the sensor can make any predictions about how the node moves, it broadcasts to the nodes in the predicted area. This broadcast uses the communication strategy in Algorithm 1. The alerting node also wakes up the nodes that provide the communication route to the base station.

## III. Guiding a User

Sensors detect information about the area they cover. They can store this information locally or forward it to a base station for further analysis and use. Sensors can also use communication to integrate their sensed values with the rest of the sensor landscape. In this section we explore using sensor networks as distributed information repositories. We describe a method to distribute the information about the environment redundantly across the entire network. Users of the network (people, robots, unmanned planes, etc.) can use this information as they traverse the network. We illustrate this property of a reactive sensor network in the context of a guiding task, where a moving object is guided across the network along a safe path, away from the type of danger that can be detected by the sensors.

The guiding application can be formulated as a robotics motion planning problem in the presence of obstacles. The dan-

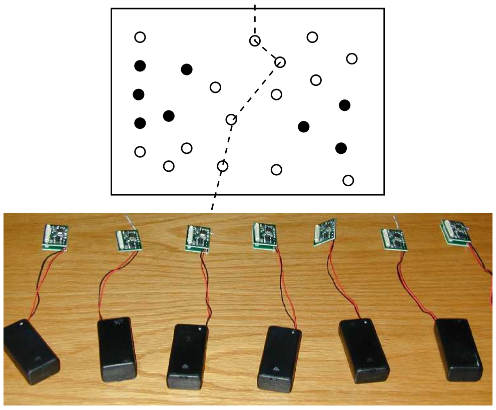

Fig. 1. The left figure shows a typical setup for the navigation guiding task. The solid black circles correspond to sensors whose sensed value is "danger". The white circles correspond to sensors that do not sense danger. The dashed line shows the guiding path across the area covered by the sensor network. Note that the path travels from sensor to sensor and preserves a maximal distance from the danger areas, while progressing to the exit area. The right picture shows some Mote sensors used for our experiments.

gerous areas of the sensor network are represented as obstacles. Danger may include excessive heat (volcanoes, fire, etc), people, etc. We assume that each sensor can sense for the presence or absence of such type of danger. A danger configuration protocol run across all the nodes of the network creates the danger map. We do not envision that the network will create an accurate geometric map, distributed across all the nodes. Instead, we wish for the nodes in the network to provide some information about how far from danger each node is. If the sensors are uniformly distributed, the smallest number of communication hops to a sensor that triggers "yes" to danger is a measure of the distance to danger. The goal is to find a path for the moving object that avoids the danger areas. We envision having the user ask the network regularly for where to go next. The nodes within broadcasting range from the user supply the next best step.

There are numerous solutions to motion planning in the presence of obstacles and uncertainty. We seek a solution that lends itself naturally to the discrete nature of sensor networks. In [4], Donald et al. describe an optimal solution for motion planning when the map of the world is given. The first step of the solution is to rasterize the configuration space obstacles into a series of bitmap slices. Dynamic programming is then used to calculate the optimal path in this space. Although this method can not be applied directly, it can be adapted for sensor networks. The motion planning algorithm fits a sensor network well in two ways. First, the sensors can be regarded as the bitmap pixels. Second, the dynamic programming component of the algorithm can be implemented by using the sensor communications.

In order to supply obstacle information to the planning algorithm we use artificial potential fields as follows. The "obstacles" (recall they correspond to the danger areas) will have repulsing values and the goal has an attracting value according to some metric (see Figure 1(Left)). The potential field is computed in the following way. Each node whose sensor triggers "danger" diffuses the information about the danger to its neighbors in a message that includes its source node id, the potential value, and the number of hops from the source of the message to the current node. When a node receives multiple messages from the same source node, it keeps only the message with the smallest number of hops. (The least hops message is kept because that message is likely to travel along the shortest path.) The current node computes the new potential value from this source node. The node then broadcasts a message with its potential value and number of hops to its neighbors.

After this configuration procedure, nodes may have several potentials from multiple sources. To compute its current danger level information, each node adds all the potentials.

Note that the potential field protocol provides a distributed repository of information about the area covered by the sensor network. It can be run as an initialization phase, continuously, or intermittently. The sensor network can self-reconfigure and update its distributed information content by running the potential field computation protocol regularly. In this way, the network can adapt to the insertion of sensor failure, to the addition of new nodes into the network and to dynamic danger sources, that can move across the network.

The potential field information stored at each node can be used to guide an object equipped with a sensor that can talk to the network in an on-line fashion. The safest path to the goal can be computed using a query protocol. The goal node initiates a dynamic programming computation of this path using broadcasting. The goal node broadcasts a message with the danger degree of the path, which is zero for the goal. When a sensor node receives a message, it adds its own potential value to the potential value provided in the message, and broadcasts an message updated with this new potential to its neighbors. If the node receives multiple messages, it selects the message with the smallest potential (corresponding to the least danger) and remembers the sender of the message.

## IV. Experiments

We have implemented the algorithms for variable sleeping time, tracking and guiding the navigation of a user across a sensor network using the Mote sensors [3].

## References


[1] Alberto Cerpa and Deborah Estrin. Ascent: Adaptive self-configuring sensor networks topologies. In *INFOCOM*, New York, NY, June 2002.

[2] M. Chu, H. Haussecker, and F. Zhao. Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks. *International Journal of High Performance Computing Applications*, 2002.

[3] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. System architecture directions for network sensors. In *ASPLOS*, 2000.

[4] J. Lengyel, M. Reichert, B. Donald, and D. Greenberg. Real-time robot motion planning using rasterizing computer graphics hardware. In *Proc. SIGGRAPH*, pages 327–336, Dallas, TX, 1990.

[5] Wei Ye, John Heidemann, and Deborah Estrin. An energy-efficient mac protocol for wireless sensor networks. In *INFOCOM*, New York, NY, June 2002.