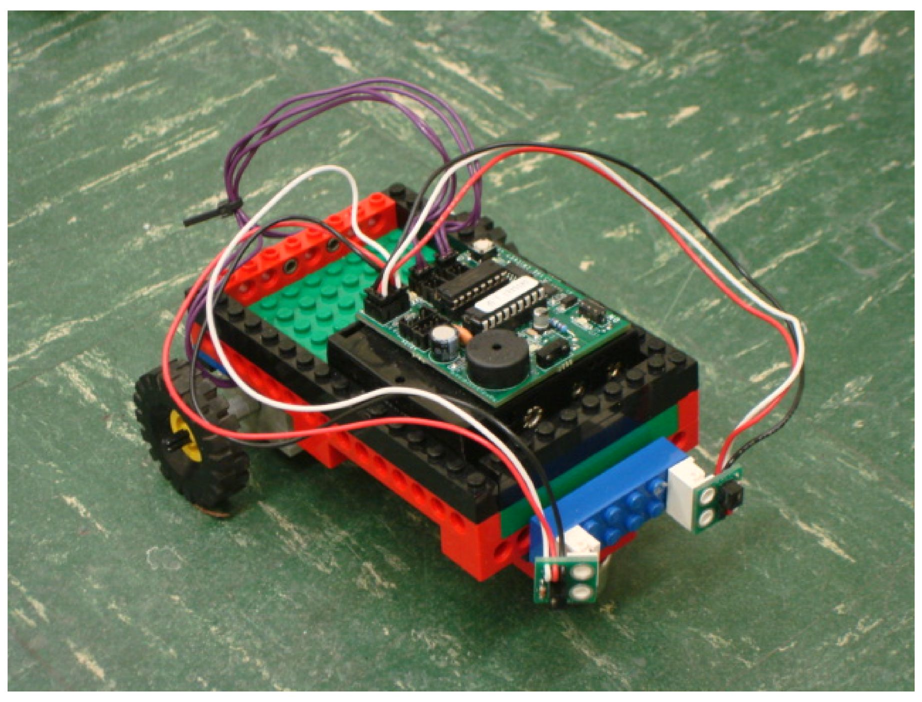
Implementing the Dispersion of Teams of Search/Rescue Robots

Binod K.C.
Augsburg College
Minneapolis, MN 55454
kcb@augsburg.edu

Introduction:

When unsafe areas such as collapsed buildings must be entered in rescue operations, the time it takes to find victims often determines whether or not those victims survive. Due to the fact that the area is unsafe, human rescuers are also in danger. One approach has been to send teams of small robots into the area. The rationale is that they will be able to move through small openings, spread out and find victims more quickly than a single robot would be able to do. The danger to human rescuers would also be decreased.

We have developed an algorithm using a tree structure for robot dispersion. This is in contrast to the two most common approaches of using a completely random dispersion or setting up a strong network of connections between robots.



One of our Cricket robots. All robots are configured identically with two light sensors for obstacle detection plus an infrared transceiver for communication.

Software:

The Handy Cricket provides 4096 bytes of non-volatile storage for user programs. It is programmed using a version of Logo called “Cricket Logo.” Code is downloaded through an external communications board which connects to the host computer’s serial port. Due to the small amount of memory, code must be carefully crafted. All Crickets contain the same code. There is no central control.

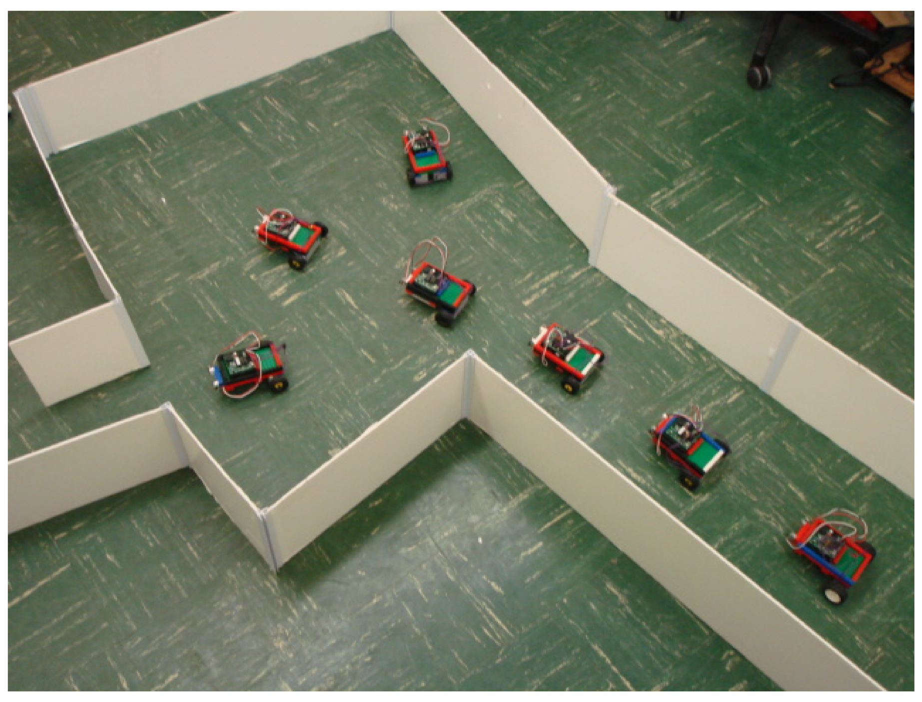
Datastructure:

Position	0	1	0
Children	1	0	0

The tree has a maximum branching factor of three. Thus, each robot can have up to three children. Two single arrays, as shown above, each of length three bits, hold local information for each robot. Its own relative position in the tree plus whether or not it has a LEFT, RIGHT or MIDDLE child is stored. The array in the table reflects a robot in the MIDDLE position with only a LEFT child.

Algorithm:

The algorithm we developed for growing our tree structure was tested in Player/Stage before we implemented it on the Crickets. Due to possible obstacles or narrow passages, the tree expands from root out, with nodes being “pushed” down from the root to fill the space. The tree grows in a depth-first/breadth-first manner, branching out into large spaces and snaking through small passages. The first robot to enter the area becomes the root of the tree. It checks ahead. If it sees space to be explored, it calls for a child. As soon as a robot answers that it is willing to become part of the tree, the root prepares to move forward, giving its root position to the new robot. Whenever an obstacle, such as a wall, is in the way of its progress in a given direction, it moves on to the next option. After reaching its designated destination, it sends back a confirming infrared signal to its parent.



Dispersion of seven robots. The robot in the right foreground is the current root and is located at the entrance to the area. The left leaf robot will be the next to request a child. It will be a middle child because expansion will occur down the passage straight ahead of it. The other two leaves will not request children because their sensors will report the walls ahead of them as obstacles. The left leaf robot’s request will propagate back to the root, an eighth robot will become the new root and all ancestor robots of the left leaf will move forward, with the left leaf moving into the passage ahead. If it senses space in that direction, it will request another child.

Conclusion:

Trials thus far have shown

- a decrease in dispersion time compared to using both a random dispersion and a strongly connected network.
- a need for fewer robots than in using a random dispersion.
- a workable system that depends on minimal communication, as opposed to using a system with a strong network of connections.

Future work includes

- running trials on the effectiveness of passing a bit string back to the root containing victim location.
- testing robot behavior in maze-like environments with numerous dead ends.
- running trials on removing the robot tree from the environment.

This work was supported by National Science Foundation grant #5038740.