A Mobile Sensor Network Testbed Using iRobots

Shadi Janansefat, Izzet Senturk, Kemal Akkaya and Michael Gloff
Department of Computer Science
Southern Illinois University
Carbondale, IL 62901 USA
Email: shadi.janan@siu.edu, isenturk@cs.siu.edu, kemal@cs.siu.edu, mgloff@siu.edu,

***Abstract*—Wireless sensor networks (WSNs) have been used in many applications by deploying tiny and stationary sensors. In recent years, a lot of studies proposed to introduce mobility capability to sensor nodes in order to exploit the advantages of mobility in WSN applications. In this proposal, we design and evaluate a mobile sensor network (MSN) testbed using iRobot-based mobile nodes. We integrate low-cost commercially available iRobot Create platform with IRIS sensor motes to design a new mobile sensor node called iRobotSense. Using these mobile nodes, we developed an MSN testbed at Southern Illinois University and implemented a connectivity restoration algorithm on this testbed that exploits node repositioning. Our demonstration will include the execution of this algorithm in a small testbed.**

I. Introduction

Wireless Sensor Networks (WSNs) have received significant attention from the wireless networking research community within the last decade [1] due to their potential to be deployed in many real-life applications where human accessibility is limited. While initial deployments of WSNs utilized stationary tiny sensors, later, due to wide range of application possibilities, deployment of mobile heterogeneous devices within WSNs were considered. Such devices include mobile robots [2][3], vehicles, and actuators [4]. Consequently, many variants of WSNs have been proposed including Wireless Sensor and Actor Networks (WSANs) [4] and Mobile Sensor Networks (MSNs) [5].

Although there exists a wide array of studies exploiting node mobility, very few of them considered the design and evaluation in a real MSN testbed [6]. Majority of the studies were simulation-based which did not take mobility issues in real-life contexts into consideration. These issues include the design of low-cost mobile sensors with sensing ability, evaluation of their energy consumption characteristics, consideration of their restrictions in moving in the region and evaluation of their performance in a testbed.

In this proposal, we present the design and implementation of a low-cost mobile sensing platform based on IRIS motes [7] and iRobot Create [8]. Both IRIS motes and iRobot Create are commercially available and used platforms with reasonable costs. We design a mobile sensing device by integrating IRIS mote with an iRobot Create which is named iRobotSense. Using this node, we will then demonstrate an MSN testbed. Under this testbed, we implemented a variant of the connectivity restoration scheme proposed in [9] under the same testbed. The testbed will be shown to detect network partitioning and restore the network connectivity via the cascaded movement of iRobotSense nodes towards the location of the failed node. The movement directions are controlled by using a compass module which is directly attached to the IRIS motes.

The rest of this demo proposal is organized as follows: Section II explains our mobile sensor node design based on IRIS and iRobot. In Section III, we explain the design and implementation of connectivity restoration scheme in a mobile MSN testbed. Section IV is dedicated to description of the demonstration and the requirements. Finally, Section V concludes the proposal.

II. iRobotSense: A Mobile Sensing Platform based on iRobot Create

In this section, we present the design of a new mobile platform which can sense the environment. The mobile platform is based on iRobot Create [8] which is cost effective ($130) and easily interfaceable with other devices such as sensors and sensor boards. As the sensor and board, we use IRIS motes from MEMSIC [7] ($99) and an interface board ($15) respectively. A compass module ($35) will also be used to determine the movement directions of the iRobotSense. The total cost of this platform will be roughly $280 which is cheaper than the existing robots and easily developeable. We first provide an overview of the hardware used in creating iRobotSense.

A. IRIS Motes

The IRIS Mote [7] is a wireless sensor module used in WSNs widely. This mote has a radio which implements the IEEE 802.15.4 wireless specification in the ISM 2.4 GHz band. The use of IEEE 802.15.4 is one of the major motivations to choose IRIS mote for our design since most of the simulation-based evaluations of WSNs assume a MAC layer protocol based on IEEE 802.15.4 and Zigbee.

Along with offering wireless communication, the module also has a 51-pin expansion connector to allow wired serial communication, ADC inputs and general purpose digital IO. The expansion connector can be used with a variety of sensor and data acquisition boards that can sense temperature, light and vibration.

B. iRobot

The iRobot Create (see Fig. 1) [8] is an educational development kit that can perform the common drive tasks of a robot without having to program a custom one. The drive

mechanism consists of two motors controlling each of the two drive wheels while a third wheel in the front provides support. These wheels can be powered and by specifying a turn radius, the robot can move in any direction. There are limited sensors onboard to detect a cliff, a wheel drop or a forward collision. iRobot is battery operated. Either 12 AA batteries or an optional rechargeable battery pack can power the robot.

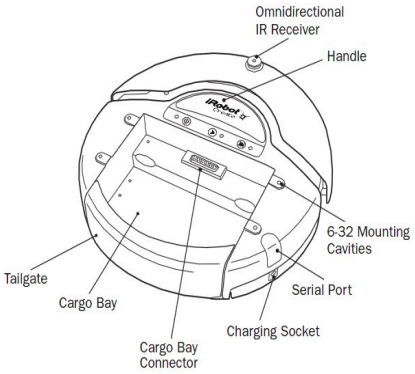

Fig. 1: iRobot Create for developing a mobile sensor

The robot is programmable via the iRobot Create Open Interface (OI). The OI provides a set of commands and responses for the robot to interact with external devices via the serial interface. Each command starts with a one-byte opcode. Some of the commands must be followed by data bytes. The commands that have been used in our design to control the robot are: *Start*, *Safe*, *Drive*, *Script*, *Play Script*, *Wait Distance*, *Wait Angle*, *Stream*.

C. iRobot-Mote Integration

To control the robot from an external source via the serial interface, two connection options are available: a DB25 cargo bay connector and a 7 pin mini-din connector as seen in Fig. 1. In our design, the DB25 cargo bay connector, shown in Fig. 2, is used to interface the iRobot with the mote.

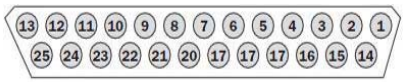

Fig. 2: DB25 cargo bay connector on iRobot Create

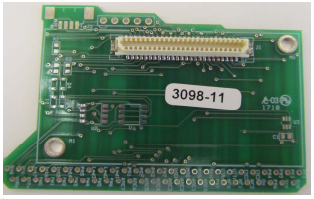

Fig. 3: Interface board to connect IRIS mote and iRobot

We used a separate interface board(IRIS interface board [10]) between the IRIS mote and the iRobot in order to interface them. This board is the simplest model of its kind, which has a breakout region where all 51 pins are exposed (see Fig. 3). This board allows easy interfacing of mote devices with peripherals and external sensors via the Inter-Integrated Circuit (I2C), Serial Peripheral Interface (SPI), UART and analog-to-digital converter (ADC) ports. Finally, this IRIS interface board was picked because of its low-cost.

In order to move an iRobotSense node to a certain destination with a known distance, iRobotSense needs to face its wheels towards this destination before it starts moving. To determine the direction of the iRobotSense, we also interfaced the IRIS mote with a compass. A Printed Circuit Board (PCB) produced by Sparkfun Electronics [11] provides us with a HMC6352 compass. Note that we assume that each IRobotSense node is assumed to know its location based on a local coordinate system. The final product along with compass wiring is shown in Fig. 4.

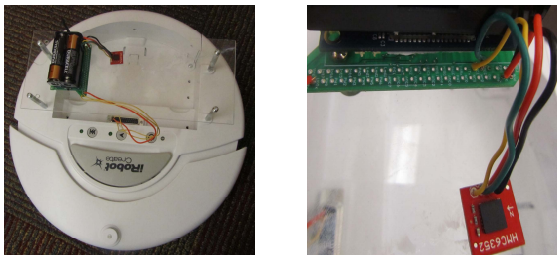
(a) (b)

Fig. 4: (a) A mobile sensor based on iRobot Create platform (b) Wiring with the compass

III. Demonstration

In this section, we describe our testbed consisting of iRobotSense nodes designed in the previous section and the connectivity restoration algorithm [9] that will be tested on this testbed.

A. Demonstration Testbed Topology

By deploying 7 iRobotSense nodes, we created a network topology as seen in Fig. 5. Each iRobotSense node is assigned a unique ID and the communication range for each node is set to the lowest value in order to force multi-hopping among the nodes. In this topology, each node is assumed to be broadcasting its readings every second to its neighbors. The neighbors further broadcast these readings until they are received by the sink node which is assumed to be node N_7 in Fig. 5. Since iRobotSense uses a IEEE 802.15.4 based MAC protocol, we follow the same protocol at the MAC layer. However, no specific routing protocol was implemented as we were relying on flooding.

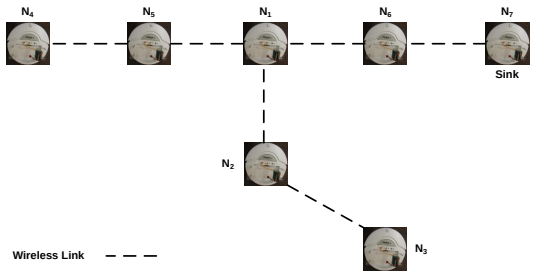

Fig. 5: Created MSN topology using 7 iRobotSense nodes

B. Demonstration Goal: Testing a Connectivity Restoration Algorithm

Before describing the algorithm, we define the problem as follows: Considering the topology in Fig. 5, if node N_1 fails

due to any reason, then the network would partition into three partitions as seen in Fig. 6. Then the problem is how to re-arrange the topology in order to restore the connectivity and data communication.

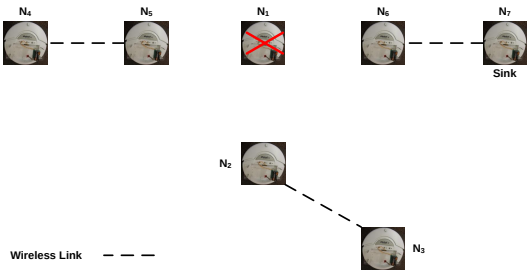

Fig. 6: Partitioned MSN topology after node N_1 fails

PArtition Detection and Recovery Algorithm (PADRA) [9] is designed to address this problem. PADRA first determines possible partitioning in advance and self-restores the connectivity in case of such failures with minimized node movement and message overhead. Since partitioning is caused by the failure of a node which is serving as a cut-vertex (i.e., node N_1), each node determines whether it is a cut-vertex or not in advance in a distributed manner. This is achieved by utilizing the Connected Dominating Set (CDS) of the whole network. Once such cut-vertex nodes are determined, each node designates the appropriate neighbor to handle its failure when such a contingency arises in the future. The designated neighbor picks a node from the CDS, called *dominatee* (i.e., whose absence does not lead to any partitioning of the network) to replace the failed node when it actually fails. Such a dominatee is found through a greedy algorithm by following the closest neighbors in terms of distance. The replacement is done through a cascaded movement where all the nodes from the dominatee to the cut-vertex are involved. The goal is to share the movement load so that the energy of the selected dominatees will not drain quickly as a result of a long mechanical motion.

In our testbed, node N_3 is selected as the dominatee node to replace node N_1. And node N_2 is the designated node to start the recovery process. We assume that these are computed in advance and hard-coded in the nodes. When the data transmissions start, we failed node N_1 at some point and waited for the network to restore the connectivity using PADRA. Node N_2 waits for a predetermined time until it decides to initiate the recovery process. During the recovery process, it determines node N_3 as the closest dominatee and thus sends a message to it to replace itself since it will be leaving to replace the dead node which is node N_1. The new topology as a result of running PADRA is shown in Fig. 7.

C. Demonstration Requirements

Due to logistics, we will bring only two iRobotSense nodes for the demo. The rest of the nodes will be replaced with fixed IRIS motes. We need the space layout of about 2mx3m to place the sensors and iRobots. No power or connection is needed. The failed node will be turned off during the demo to indicate that its not functioning and receiving any messages.

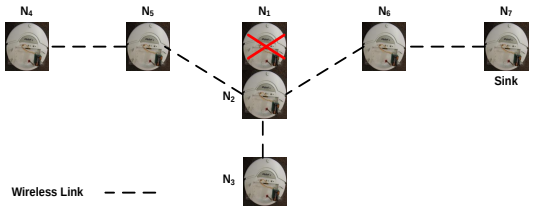

Fig. 7: New MSN topology after recovery using cascaded movement. N_2 replaced N_1 and N_3 replaced N_2

IV. Conclusion

In this demo, we demonstrate the design of a new mobile sensor called iRobotSense which uses iRobot Create as the mobile platform and integrates it with IRIS motes for sensing and communication capabilities. By using 2 iRobotSense nodes and 5 IRIS motes, we also created and demonstrated an MSN testbed which implements and tests PADRA. The testbed provided us several insights in designing and maintaining an MSN testbed.

Acknowledgment

This work is supported by US National Science Foundation under the grant number CNS 1018404.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Elsevier Computer Networks*, vol. 38, pp. 393–422, 2002.

[2] J. Friedman, D. Lee, I. Tsigkogiannis, P. Aghera, A. Dixit, D. Levine, D. Chao, A. Kansal, W. Kaiser, and M. Srivastava, "Ragobot: A new hardware platform for research in wireless mobile sensor networks," in *In Proceedings of International Conference on Distributed Computing in Sensor Systems*, 2005.

[3] G. T. Sibley and M. H. A. Rahimi, "Robomote: A tiny mobile robot platform for large-scale ad-hoc sensor networks," in *IEEE International Conference on Robotics and Automation*, 2002.

[4] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: Research challenges," in *Elsevier Ad hoc Network Journal*, vol. 2, 2004, pp. 351–367.

[5] G. Wang, G. Cao, T. L. Porta, and W. Zhang, "Sensor relocation in mobile sensor networks," in *Proceedings of the 24th International Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM05)*, Miami, FL, March 2005.

[6] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," in *Elsevier Ad-Hoc Network Journal*, vol. 6, no. 4, 2008, pp. 621–655.

[7] "Iris mote," in *http://www.memsic.com*, May 2012.

[8] "irobot," in *http://www.irobot.com*, May 2012.

[9] K. Akkaya, F. Senel, A. Thimmapuram, and S. Uludag, "Distributed recovery from network partitioning in movable sensoractor networks via controlled," in *IEEE Transactions on Computers*, vol. 59, no. 2, Feb. 2010, pp. 258–271.

[10] "Interface board," in *http://shop.samraksh.com/product.sc?productId=3*, May 2012.

[11] "compass," in *http://www.sparkfun.com/products/7915*, May 2012.