Demonstration of Robotic Repair for Wireless Networks

Thuy T. Truong, Rodolfo V. Bisol, James Giller, Hugh Whelan, Kenneth N. Brown and Cormac J. Sreenan
CTVR, Department of Computer Science
University College Cork, Ireland
Email: {t.truong, r.bisol, j.giller, h.whelan, k.brown, cjs}@cs.ucc.ie

***Abstract*—This paper describes our demonstration of a network repair problem where a robot bridges a gap between two disconnected wireless nodes by searching for a good position and moving there to forward data between the two nodes. It serves to show the potential for our published solutions for automated network repair. A simple Adhoc network consists of two Intel Galileo Gen 2 nodes exchanging messages and an NXT Mindstorm robot with another Galileo on board healing the network connection between the two Galileos in the case the two get disconnected. The demo showcases a solution that employs mobile agents to serve as relays to bridge the connectivity gaps in the wireless network.**

I. Introduction

Many applications for wireless networks will be in settings where network damage can be expected to occur, e.g. battlefield sensing, volcano monitoring, fire detection, etc. In addition, many remotely operated wireless sensor nodes are powered by batteries or renewable sources, and so nodes may fail as batteries deplete. The loss of nodes might cause network partitioning, thus leading to longer delivery delays and/or lost packets. To overcome node failure and to restore network connectivity, network repair should be initiated where we must place new nodes in the environment to restore connectivity to the network. In our prior published work ([1], [2] [3], [4]) we presented solutions for automated repair of damaged wireless sensor networks and in this demonstration we show the potential for healing a small network.

The goal of robotic network repair is to restore network connectivity for a partitioned wireless network. A robot agent will be deployed to discover network damage, RF environment characteristics, and physical access constraints, and run algorithms for optimisation of radio equipment use, network service quality, and physical deployment plans. These algorithms will guide the robot to go to specific locations to populate new nodes in order to restore the network connectivity. This paper demonstrates a scenario where a robot identifies connection problems and then moves into a strategic position to bridge the gap between two disconnected nodes. This robot will act as a relay to forward data between the two nodes.

II. System Description

The scenario is based on two Intel Galileos (Generation 2) [5] that are set to be in the same Adhoc network and communicate wirelessly. Incidents (e.g. physical obstructions) might occur during the transmission between the two Galileos, and this could block the wireless signals. When the signal is blocked, the robot will search for a location to reconnect them. Figure 1 shows the described scenario.

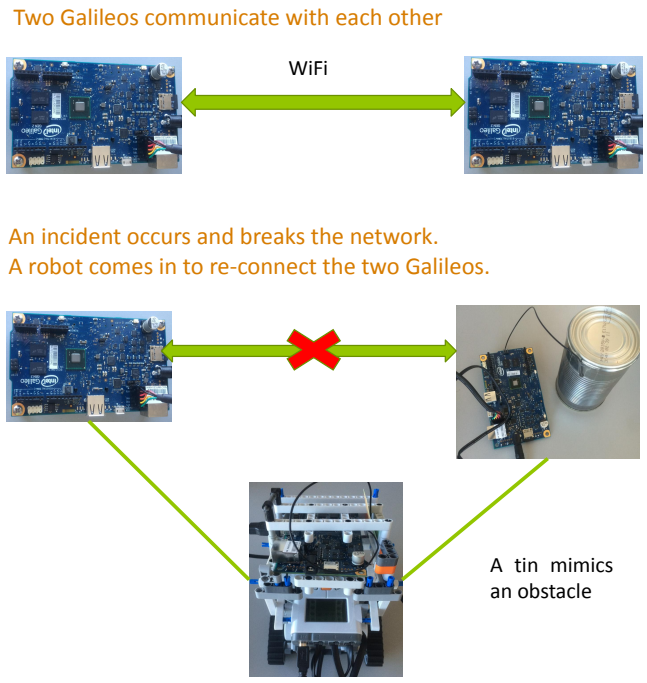


Fig. 1: A robot is healing the connectivity in an Adhoc network.

In order to test the communication, Galileo node 1 (G1) is attached with a display monitor (Figure 2) while Galileo node 2 (G2) is attached with a sound sensor (Figure 3). G2 will sample ambient sound and transmit the data (soundValue) at a fixed sampling rate to G1. Upon receiving the messages from G2, G1 processes the data and displays the results on the screen (Figure 2).

G1 is responsible for monitoring the condition of the radio link, and for determining whether or not the link is of the required quality. When the link is down, G1 sends a request message (REQ message) to a robot, asking it to come and heal

978-1-4673-7331-9/15/$31.00 © 2015 IEEE

the connection, and when the link is up again, G1 will send a retreat message (RET message) to the robot so that the robot knows to draw back to its starting location.

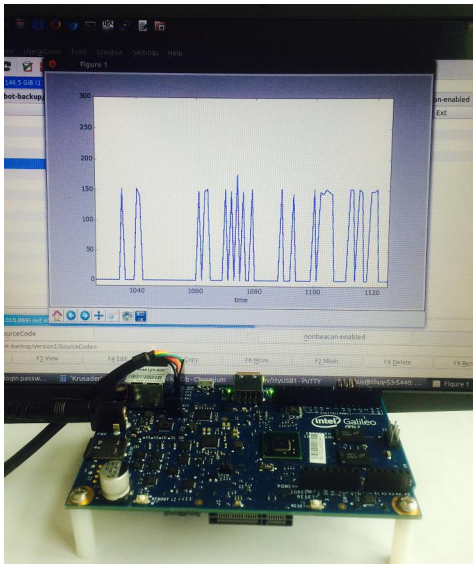


Fig. 2: Galileo G1 is attached with a display monitor. Upon reception of messages, G1 shows the received data on the screen.

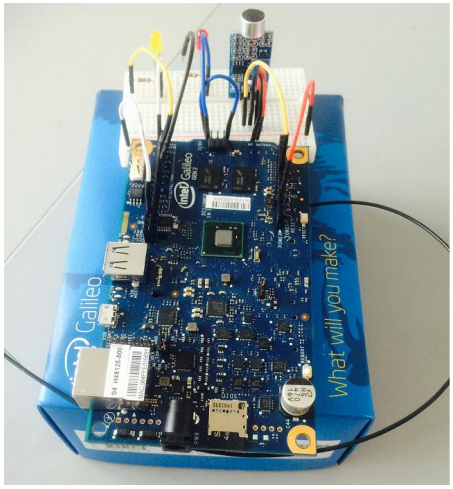
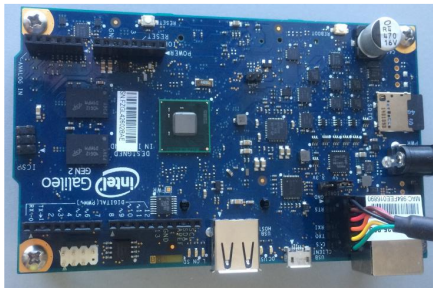
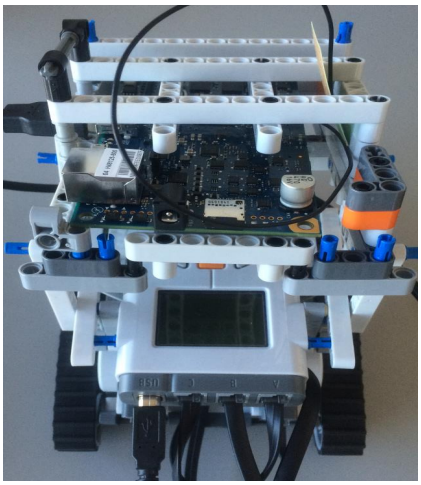


Fig. 3: Galileo G2 is attached with a sound sensor. G2 samples ambient sound and transmits the data at a fixed sampling rate to G1.

In this demo, we use an NXT Mindstorm robot for the actions. The robot will be equipped with a third Galileo Gen 2 (GR). This Galileo acts as the 'brain' of the robot. The Galileo is used to determine whether or not the robot is in a location which has connection with both G1 and G2, and if not, where the robot should go next to find such a location. It is also responsible for establishing a connection between two disconnected nodes. Figures 4 shows the Intel Galileo Gen 2 on the top and the NXT Mindstorm robot at the bottom. The robot



Intel Galileo (gen 2)



NXT Mindstorm robot
with Galileo on board

Fig. 4: Intel Galileo Gen 2 and an NXT Mindstorm Robot.

starts at a place (a starting point) where it can hear a message from G1 and waits for the request message. When it hears a REQ message from G1, it will turn on the proxy-arp and set up suitable parameters for the forwarding. GR calculates the shortest path to a target and controls the robot to move to the target, checking IP connectivity to both G1 and G2 on the way. The robot will stop at a location and start forwarding packets if it has connection to both Galileos; otherwise, it will search for different locations. When the robot hears a RET message from G1, it will turn off the proxy-arp and calculate the shortest path from current location to the starting point and then move there. At this point, the communication between the two Galileos (G1 and G2) is back to normal.

III. Implementation

We have four main programs running on three different Galileos (G1, G2 and GR) and the NXT Mindstorm robot. The three Galileos must be set in the same Adhoc network (see the script in Listing 1). The programs running on each device are as follows.

Listing 1: Setting up the Adhoc network.

```
// setting up the Adhoc network
#! /bin/bash
iwconfig wlan0 mode Ad-hoc
iwconfig wlan0 essid MyWiFi
ifup wlan0
ifconfig wlan0 192.168.2.x netmask
    255.255.255.0 up
```

Listing 2: Turning on/off the proxy-arp and forwarding feature.

```
// The proxy-arp and forwarding features.
#! /bin/bash
sysctl -w net.ipv4.conf.wlan0.proxy_arp=1|0
sysctl -w
    net.ipv4.conf.wlan0.proxy_arp_pvlan=1|0
sysctl -w net.ipv4.ip_forward=1|0
sysctl -w net.ipv4.ip_nonlocal_bind=1|0
```

- At node G1: a C program (g1_communication.c) listens to the communication, detecting link up/down from G2, and sends REQUEST/RETREAT messages upon the link's condition has changed. This node also runs a Python script (g1_display.py) to display the received sound values received from G2.
- At node G2: A Python script (g2_data.py) is run to receive the sound values from a sound sensor, and then send these values to G1.
- At node GR: a C program (gr_program.c) will listen to the REQUEST/RETREAT messages from G1. This program upon receives a REQUEST message, it will execute a script to turn on the proxy_arp and set up all necessary parameters for forwarding data (Listing 2), then it searches for a location which has connection to both G1 and G2. GR will send a target location to the robot which will move to the location. For simplicity, we input a number of target locations to GR. The path from the current location to the new target location is calculated by A* search [6]. When GR receives a RETREAT message indicating that the direct connection between G1 and G2 is now up again, GR also uses A* search to calculate the path from its location to the starting location. The path will be sent to the robot and the robot will move to the stating location.
- At the NXT robot: The robot will run nxt_controller program. This program is written in Java and uses the LeJOS NXJ API [7]. It uses FourWayGridMesh to create a map geometry (a grid of nodes where spacing between the grid nodes and clearance around map geometry can be specified). This map is used for all navigation. The robot is attached with an ultrasonic sensor and we use RangeFeatureDetector to detect objects. For navigation, we implement a DifferentialPilot object to control a robot to traverse a path (a sequence of waypoints) and OdometryPoseProvider to keep track of the robot's pose. The robot will follow the path (sent by GR). However, at each waypoint, it stops and looks for obstacles and updates the map if necessary. If no obstacle has been detected, the robot moves to the next waypoint. Otherwise, it sends the obstacle's location to GR which will decide what to do next (changing the current path or recalculating a new plan).

The communication between the Galileo (GR) and the NXT Mindstorm Controller is straightforward with the API developed for the project [8].

IV. Conclusion

This demo demonstrates a scenario in the *robotic network repair* for disaster response where a robot agent is deployed to heal the connectivity between the two disconnected nodes in the network. In this scenario, the agent runs an A* static path finding algorithm to search for a location where it can stay and forward data between the two nodes, acting like a relay, and moves back to its starting location where the direct connection between the two is up again.

Future work includes a larger scale of network and an agent which has an ability to pick up and drop nodes. This agent will run more sophisticated algorithms to find the positions for dropping new nodes in order to heal the network connectivity, as presented in our earlier papers. Future work also considers state-of-the-art path finding algorithms for dynamic environment.

Space and/or other equipment-specific requirements

Our demonstration will require a space of 2m x 1.5m area. We have all the equipment that we need.

Acknowledgment

This project is funded by the SFI Centre CTVR (10/CE/I1853).

We would like to acknowledge the support of Intel Ireland in providing the Intel Galileo boards.

References

[1] T. T. Truong, K. N. Brown, and C. J. Sreenan, "Integration of node deployment and path planning in restoring network connectivity," in *The 29th Workshop of the UK Planning and Scheduling Special Interest Group (PlanSig)*, 2011.

[2] ——, "Restoring wireless sensor network connectivity in damaged environments." in *International Workshop on Cooperative Robots and Sensor Networks (RoboSense)*, 2012.

[3] ——, "Repairing wireless sensor network connectivity with mobility and hop-count constraints," in *ADHOC-NOW, 2013.*

[4] ——, "Autonomous discovery and repair of damage in wireless sensor networks," in *38th Annual IEEE Conference on Local Computer Networks, Sydney, 2013*, 2013, pp. 450–458.

[5] "Intel galileo generation 2 datasheet," http://www.intel.com/content/www/us/en/embedded/products/galileo/galileo-g2-datasheet.html.

[6] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on In Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[7] "Nxj technology," http://www.lejos.org/nxj.php.

[8] "Robotic network repair," http://www.ucc.ie/en/misl/research/current/ctvr_intel/.