DESIGN AND IMPLEMENTATION OF COMMUNICATION PLATFORM FOR AUTONOMOUS DECENTRALIZED SYSTEMS

Naga Sravani Gottipati

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

December 2010

APPROVED:

Kamesh Namuduri, Major Professor
Murali Varanasi, Committee Member and Chair of the Department of Electrical Engineering
Parthasarathy Guturu, Committee Member
Costas Tsatsoulis, Dean of the College of Engineering
James D. Meernik, Acting Dean of the Robert B. Toulouse School of Graduate Studies
Gottipati, Naga Sravani. Design and implementation of communication platform for autonomous decentralized systems. Master of Science (Electrical Engineering), December 2010, 29 pp., 23 illustrations, references, 6 titles.

This thesis deals with the decentralized autonomous system, in which individual nodes acting like peers, communicate and participate in collaborative tasks and decision making processes. An experimental test-bed is created using four Garcia robots. The robots act like peers and interact with each other using user datagram protocol (UDP) messages. Each robot continuously monitors for messages coming from other robots and respond accordingly. Each robot broadcasts its location to all the other robots within its vicinity. Robots do not have built-in global positioning system (GPS). So, an indoor localization method based on signal strength is developed to estimate robot’s position. The signal strength that the robot gets from the nearby wireless access points is used to calculate the robot’s position. Trilateration and fingerprint are some of the indoor localization methods used for this purpose.

The communication functionality of the decentralized system has been tested and verified in the autonomous systems laboratory.
Copyright 2010

by

Naga Sravani Gottipati
ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my major advisor Dr. Kamesh Namuduri for his guidance, constant mentoring and support. I would like to thank my advisory committee members Dr. Murali Varanasi, Dr. Parathasarathy Guturu for their motivation to finish my thesis successfully. I owe my gratitude to the faculty and staff of the electrical engineering department for their moral support. I would also like to thank Mr. Jim Byford for helping me to understand the UNT network. Lastly, I am very grateful to my family and friends, especially my parents and my roommates for their encouragement and support.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ACKNOWLEDGEMENTS</td>
<td>iii</td>
</tr>
<tr>
<td></td>
<td>LIST OF FIGURES</td>
<td>vi</td>
</tr>
<tr>
<td>Chapters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1.2 Problem Statement</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1.3 Objectives of the Thesis</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1.4 Contributions of the Thesis</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1.5 Organization of the Thesis</td>
<td>3</td>
</tr>
<tr>
<td>2.</td>
<td>EQUIPMENT USED</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2.1 Acroname Garcia Robots</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2.1.1 Processors and Memory</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2.1.2 Battery</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2.1.3 Serial Port</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2.1.4 Range Finders</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2.1.5 Primitives</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2.2 Remote Host</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2.3 UNT Network</td>
<td>6</td>
</tr>
<tr>
<td>3.</td>
<td>PROTOCOL USED</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>3.1 Socket Programming</td>
<td>11</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Garcia robot</td>
</tr>
<tr>
<td>3.1</td>
<td>Server received a message from one of the clients and sent a “hello” message as a response</td>
</tr>
<tr>
<td>3.2</td>
<td>Client transmitted a &quot;hi&quot; message to server and received a &quot;hello&quot; message as response</td>
</tr>
<tr>
<td>3.3</td>
<td>Server broadcasting “hi” message to all the clients</td>
</tr>
<tr>
<td>3.4</td>
<td>Client 1 received a ‘hi’ message from the server</td>
</tr>
<tr>
<td>3.5</td>
<td>Client 2 also received a “hi” message from the server at the same time</td>
</tr>
<tr>
<td>3.6</td>
<td>Each robot, acting like a peer, broadcasts messages periodically and also listens to the incoming messages from other robots</td>
</tr>
<tr>
<td>3.7</td>
<td>Sendto and recvfrom are the commands used as shown in the program</td>
</tr>
<tr>
<td>3.8</td>
<td>Telnet session of a robot using putty</td>
</tr>
<tr>
<td>3.9</td>
<td>\etc path in the Garcia robot</td>
</tr>
<tr>
<td>3.10</td>
<td>Network interface file in the Garcia robot</td>
</tr>
<tr>
<td>3.11</td>
<td>IP assignment to the robots</td>
</tr>
<tr>
<td>3.12</td>
<td>aBinary path in the Garcia robot</td>
</tr>
<tr>
<td>3.13</td>
<td>launching the aRelay application in all the robots</td>
</tr>
<tr>
<td>3.14</td>
<td>Communication among the robots using aRelay application</td>
</tr>
<tr>
<td>3.15</td>
<td>All robots move in equilateral triangle except the one which sent the “triangle” message</td>
</tr>
<tr>
<td>4.1</td>
<td>Garcia.config file</td>
</tr>
<tr>
<td>4.2</td>
<td>Relay.config file in the robot</td>
</tr>
</tbody>
</table>
5.1 Trilateration approach example 1 .................................................................24
5.2 Trilateration approach example 2 .................................................................24
5.3 Fingerprint database ....................................................................................25
5.4 Fingerprinting approach example 1 ..............................................................26
5.5 Fingerprinting approach example 2 ..............................................................26
1.1 Motivation

Robots are being used for a wide range of applications such as an underwater exploration, planetary surface exploration etc, which would not be possible with the humans [1]. Other than mechanics, electronics has become the driving force in robot design, develop and manufacturing [2]. Digital technology and programmable languages are used to build robots which can move and walk like humans [2]. These kinds of robots are used for the specific applications like space colonization.

In the National Aeronautics and Space Administration (NASA) space colonization project, a network of robots is expected to function as a decentralized autonomous system. A decentralized autonomous system is the one in which there is no centralized authority. Each robot acts independently as a peer, interacts with others, and participates in the decision-making process to solve a common problem. Any robot can join and leave the system whenever it needs. If any of the peers fail, it would be easy to reconstruct the system by replacing the peer without impacting its functionality. Each robot can send information about itself such as its position, continuously, to all other robots within its vicinity. Numerous applications such as terrain mapping can be implemented in such an environment.
1.2 Problem Statement

The goal is to create a test bed using Garcia robots. The robots should act like peers and should interact with each other as a decentralized autonomous system. Each robot should continuously monitor for messages coming from others and respond according to the messages. At the same time, each robot should be able to send its location to all the other robots periodically.

1.3 Objectives of the Thesis

The objectives of the thesis include the following:

- Exploring the features of Garcia robots
- Implementing a decentralized autonomous system using Garcia robots
- Creating a programming environment including compilation and execution
- Studying different indoor positioning algorithms, implementing them, and comparing their performance in MATLAB

1.4 Contributions of the Thesis

- A communication platform consisting of four Garcia robots is created
- A centralized test-bed is developed initially; later, it is reorganized into a decentralized autonomous system
- An interface named “aRelay” in Garcia robot is used to facilitate communication with the robots from the workstation
- Implemented two indoor positioning algorithms in MATLAB
1.5 Organization of the Thesis

A brief description of the features of Garcia robots is discussed in Chapter 2. Chapter 3 describes the protocol used for the thesis. The aRelay application, which is used to communicate with the robot, is discussed in Chapter 4. Study and implementation of indoor positioning algorithms are discussed in Chapter 5. Chapter 6 is about the conclusions and future work.
CHAPTER 2
EQUIPMENT USED

The following equipment is used to perform the required experiments:

1) Four Garcia robots purchased from Acroname
2) One Work Station
3) UNT's Network

2.1 Acroname Garcia Robots

Fig. 2.1 Garcia robot.
The Garcia robots are designed by the company Acroname [3]. Some of the specifications of Garcia robots are described below [4].

2.1.1 Processors and Memory

It has two 40MHZ processors and each processor has 32K read only memory (ROM). They are BrainStem Moto 1.0 and BrainStem GP 2.0. The former one handles the motion control and several sensor inputs, and the latter one provides a serial interface, infrared (IR) communication capability and additional IO.

2.1.2 Battery

The robot is powered by a standard 6-cell 7.2V 4200mAH nickel metal hydride (NiMH) battery pack. A charger is included with the robot. A smart charger is included with the robot. It can top off a completely drained battery pack in a couple of hours.

2.1.3 Serial Port

Workstations can communicate with the Garcia through a TTL-level serial port.

2.1.4 Range Finders

Garcia has six IR range finders. They provide valid distance measurements in a range of 4 to 18 inches. These sensors enable the robot to wall-follow or detect obstacles while maneuvering. When not in use, pairs of sensors can be disabled to save battery power.

2.1.5 Primitives

A complex robotic task is a sequence of simpler tasks. These simpler tasks are called as “primitives.” These "primitives" may be quite small yet still provide a great deal of functionality.
Primitives are the building blocks of behaviors when programming a Garcia robot. Some examples of Garcia primitives are moving straight ahead a specified distance or turning a specified angle. To execute a primitive, Garcia runs a small program stored in its own memory. Garcia can store over a dozen primitives. They are stored in electrically erasable programmable read only memory (EEPROM) so they can be updated or changed when necessary. These primitives interact with the monitor to make sure that the robot acts appropriately when encountering special conditions.

The Garcia application programming interface (API) provides routines for creating behaviors, assigning behavior parameters, scheduling behaviors, and checking the results of each robot action.

2.2 Remote Host (Workstation)

In this configuration, the robot is connected to a workstation. This workstation could be a Windows PC, a Mac OS X, or Unix/Linux box. The robot can be connected with a serial cable to the host or through a telnet session. However, the ideal connection for real applications is a wireless connection. A Garcia robot already has space and connectors for installation of a spread-spectrum 2.4GHz radio frequency (RF) serial modem card. This is the recommended hardware for a Garcia-to-workstation connection.

2.3 UNT Network

In order to communicate with all the robots remotely, University of North Texas (UNT)’s wireless network is used. This network has various access points all over the Discovery Park (DP). The network administrator of UNT, Mr. Jim Byford, helped us by
assigning some IPs to the robots. He also opened some of the ports which are blocked by the university's network to help us with the project.
CHAPTER 3

PROTOCOL USED

The main objective of this research is to create a communication platform for the robots. Robots make use of user datagram protocol (UDP) messages for communication. Initially, the test bed was created as a centralized system following a client-server model. Whenever the server broadcasts a message, it gets either confirmation message or a response message from all the clients. This procedure was based on the simple socket program which was written in C language. Firstly, a one to one message model was created as shown in Fig. 3.1. The system is similar to a chat room. In this system, the server can chat with only one client at a time.

Fig. 3.1 Server received a message from one of the clients and sent a “hello” message as a response.
Fig. 3.2 Client transmitted a “hi” message to the server and received a “hello” message as a response.

Later on, server started broadcasting messages to all the clients as shown in the following Fig.

Fig. 3.3 Server broadcasting “hi” message to all the clients.
Fig. 3.4 Client1 received a “hi” message from the server.

Fig. 3.5 Client 2 also received a “hi” message from the server at the same time.

Later on, it turned into a decentralized system that means a single program runs in each robot. This program includes features such as broadcasting messages and listening to the messages coming from other robots simultaneously.
Fig. 3.6 Each robot, acting like a peer, broadcasts messages periodically and also listens to the incoming messages from other robots.

3.1 Socket Programming

User datagram protocol (UDP) is used for establishing communication between the robots. UDP is a connectionless network protocol in which the receiver does not send
an acknowledgment to the sender whenever it receives a message. UDP is compatible with the packet broadcasting.

Primarily, a socket was created and then bound to a port. A datagram channel was established for communication and is assigned with an IP address. Now, the server and client can communicate with each other by sending and receiving UDP messages. “sendto” and “recvfrom” are the commands used to send and receive UDP messages respectively as shown in the following Fig. 3.7.

```
if(!sendto(sockfd,buffer,strlen(buffer)+1,0,(struct sockaddr *)&serv_addr,sizeof(serv_len))==-1)
{
    perror("Error");
    //printf("ERROR TRANSMITTING DATA:%s\n",buffer);
    close(socket1);
}
else
{
    t=time(NULL);
    tm=localtime(&t);
    strftime(buf,sizeof(buf),"%Y-%m-%d %H:%M:%S",tm);
    printf("%s\n",buf);
    printf("DATA TRANSMITTED:%s\n",buffer);
}
else
{
    printf("Waiting for a message\n");
    bytes_recvd=recvfrom(socket1,buffer,sizeof(buffer),0,(struct sockaddr *)&serv_addr,&msg_len);
    if(bytes_recvd>0)
    {
    }
```

Fig. 3.7 Sendto and recvfrom are the commands used as shown in the above Fig.

3.2 Testing on the Garcia Robots

After the code is tested successfully in PUTTY, it is implemented on Garcia robots. A login id and password were given for each of the Garcia robots. We can login into the robots either using a serial connection or by setting up a telnet session.
Fig. 3.8 Telnet session of a robot using putty.

After login is successful, the network settings of the robot such as IP address, host name etc., can be changed by choosing the "\etc" path as shown in the following Fig. [5].

Fig. 3.9 \etc path in the Garcia robot.
Fig. 3.10 Network interface file in the Garcia robot.

```
# if netduo attached uncomment lines below
#auto eth1
#iface eth1 inet dhcp
#
# Wireless interfaces
#
# Example of an unencrypted (no WEP or WPA) wireless connection
# that connects to any available access point:
#auto wlan0
iface wlan0 inet dhcp
    wireless_mode managed
    wireless_assid any
#address 129.120.3.14
#netmask 255.255.255.0
#/sbin/ifconfig $INTERFACE up
# Acroname configured Ad-hoc network
#auto wlan0
#iface wlan0 inet static
#    pre-up /sbin/ifconfig $INTERFACE mode Ad-hoc
#    pre-up /sbin/ifconfig $INTERFACE key st;jerry
    interfaces 37/80 46%
```

Fig. 3.11 IP assignment to the robots.
The IP addresses of all the robots are

**Blue robot:** 129.120.3.12

**Red robot:** 129.120.3.14

**White robot:** 129.120.3.13

**Orange robot:** 129.120.3.15

All these IP addresses belong to University of North Texas (UNT) network (eagle net). All the code or executable (.exe) files should be placed in the aBinary folder as shown in the Fig. 3.12.

![PuTTY Terminal](image)

**Fig. 3.12** aBinary path in the Garcia robot.

Placing the files in to the robots can be done by using file transfer protocol (FTP). Some of the examples for FTP are winscp, filezilla etc. It is not safe to run executable files directly on the Garcia robots. Hence, aRelay application is used for this purpose.
3.3 Task Implementation

The main task implementation involves moving the robot in an equilateral triangle by giving appropriate instructions to it. The robot moves in equilateral triangle by using its primitives. All these primitives are built in, given by Acroname. If any one of the robots broadcasts a message as *triangle*, then the remaining robots move in equilateral triangle. If the message broadcasted is *move*, then the remaining robots move straight. Other messages include *arc* or *circle* movement commands and they work in the same manner.

Before executing the code, aRelay application has to be launched in the Garcia robot. Once aRelay application has been launched, any code can be executed in the workstation as shown in the Fig. 3.13, 3.14 and 3.15.
Fig. 3.13 Launching the aRelay application in all the robots.
Fig. 3.14 Communication among the robots using aRelay application.
Fig. 3.15 All robots move in equilateral triangle except the one which sent the “triangle” message.
CHAPTER 4

aRelay APPLICATION

As Garcia robot has arm processor, executable files created on workstation would not work on the robot. Hence a cross compiler has to be used to execute a program. But we found that for some reason, the arm-linux-gcc cross compiler did not work for these robots. Therefore, we have to use an application called aRelay application, to execute the programs from the workstation itself. This application sends/receives all the information to/from the Garcia over a socket. We can run the Garcia API on our workstation and target the Garcia remotely. This allows our workstation tools to work on the robot without restrictions. This application is developed by Acroname. It is available in the “downloads” section of “Acroname” website. After downloading, place the file as a sibling to Acroname folder in the Garcia and extract it.

IP address and port number need to be assigned to the socket in order to give connection between workstation and Garcia. And at the same time baud rate and port name has to be assigned to the Garcia robot as shown in the following Fig.

Garcia.config is the key file in which all the information has like IP address, port number etc. in the workstation. Based on the IP and port number, workstation decides to which Garcia it has to be connected.
Fig. 4.1 Garcia.config file.

```bash
### for ip linktype
# ip_port - Tells the program what ip port the stem is located on.
# This is typically a number above 6000 isn't limited to
# a range.
# ip_port = 8000

linktype=ip
ip_address=129.120.3.12
ip_port=9000
```

Fig. 4.2 Relay.config file in the robot.

```
# of the file name. If you are using Windows and have not already.
# you may wish to go to Control Panel > Folder Options > View
# and uncheck the box for hiding extensions for known file types

# THIS FILE IS NOT REQUIRED TO RUN THE aRelay PROGRAM

### for ip linktype
# port - Tells the program what ip port the relay is located on.
# This is typically a number above 6000 isn't limited to
# a range.
# port = 9004

# portname - Tells the program which device to relay with.
# On Unix, you may use 'ls /dev' to find the port.
# Values are case sensitive and must be typed correctly.
# For example, "com1" is not the same as "COM1".
# portname = ttyS1
baudrate=38400
relay.config 40/40 100%
```
We need to run aRelay application on the robot before we execute our code in the workstation. In order to connect all the robots to the workstation at one time, we need to maintain different Garcia.config files and the IP address, port numbers of all the robots should be different. Before executing the program, we need to compile it using an appropriate make file. Once all the robots are connected to the workstation, execute the .exe file which is in the path /acroname/aBinary.
CHAPTER 5
LOCALIZATION

After a detailed investigation of different indoor positioning techniques, the two techniques trilateration approach and fingerprinting approach are implemented [6]. The main aim of these approaches is to get the position of the robot at any given time. These approaches are based on the strength of the wireless signals received from wireless access points.

5.1 Trilateration Approach

Trilateration method calculates the distance ‘r’ between the access point (AP) and mobile user (MU). Based on the signal strength (SS) that MU gets from each AP, it calculates the distance ‘r’ from each AP. Circles are drawn with these radii and the location of AP’s as centers respectively. All these circles intersect at various points. The mean of all the points of the intersection gives the position of the MU as shown in the following Fig.
Fig. 5.1 Trilateration approach example 1.

Fig. 5.2 Trilateration approach example 2.
5.2 Fingerprinting Approach

In this approach, first some reference points (RP) are selected. SS from all the AP at each reference point is calculated and the finger print database is built as shown below in Fig. 5.3 [6].

![Diagram of Fingerprint Database](image)

Fig. 5.3 Fingerprint database.

Now, the signal strength from all the AP’s at mobile user is measured and stored in a vector. This vector is compared with the each row in the database using searching/matching algorithm. One method of comparison is to calculate the distance between the vectors. Then the weighted average of all the nearest neighbors (NN) is considered to get the position of the MU as shown below in Fig. 5.4.
Fig. 5.4 Fingerprinting approach example 1.

Fig. 5.5 Fingerprinting approach example 2.
CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

1. The design and development of a decentralized autonomous system using four robots is presented in this thesis. The following conclusions can be drawn from the analysis. First, the features of Garcia robots have been explored and the techniques for establishing communication among the robots are also investigated. Later, UNT network’s IP addresses are assigned to the robots. Once network has been established properly, a program is developed to establish communication among the robots and is implemented by using aRelay application that runs on the robot. Hence, all the robots started responding according to the message they received.

2. After rigorous literature survey, it is determined that making use of the wireless access points is the best way to find the position of the robots. Then trilateration and fingerprinting approaches which estimate robot's location based on wireless signal strength are implemented. Results show that the location of the robot is calculated very accurately using these approaches.

6.2 Future Work

There are some possible extensions to this current thesis work. The future work includes practical implementation of localization and video streaming. In order to implement localization techniques on robot, the signal strength of all the access points is necessary.
Then implementation of trilateration and fingerprint approaches on robot can be performed easily based on the MATLAB program that was developed in this thesis. Video streaming can be done by attaching a camera to the camera boom present on the Garcia robot. This video streaming is helpful for monitoring the public places without human’s presence and in many other ways.
REFERENCES


   http://www.acroname.com/robotics/info/gumstix/networking.html