Title of project placed here

Chongbin Ren

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

A dissertation presented in part fulfilment of the requirements of the Degree of Master of Science at The University of Glasgow

Date of submiss

Abstract

abstract goes here

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: _____ Signature: _____

Acknowledgements

acknowledgements go here

Contents

Chapter 1

Introduction

1.1 A section

1.1.1 A subsection

Please note your proposal need not follow the included section headings - this is only a suggested structure. Also add subsections etc as required

2

Chapter 2

Background

2.1 Robotics

Robotics is a branch of engineering that involves the conception, design, manufacture, and operation of robots[24]. Robots are machines that can be used to do jobs. Some robots can do work by themselves. Other robots must always have a person telling them what to do. Todays robotic systems generally consist of many small autonomous systems working together to form a coherent whole[25].

2.1.1 Multi-Robot Systems

Given the rate of technological advancements over the past decade or so, the adoption of robotics has become increasingly widespread. In most domains, the performance of multiple robots outperforms a single robot, both with respect to cost, efcacy and domain potential [14]. This has led to multirobot systems becoming a key area of research within the eld of robotics in general [15].

Multi-robot systems can consist of many intelligent agents (each of which may be comprised of many small autonomous systems) working to solve a task that any one system may not be able to solve alone. These multirobot are distinct from a multi-agent system in which individual nodes are generally stationary, as each agent in a multi-robot system is mobile[16]. Mobile robotics has been made more possible recently by advances in battery and wireless communication technologies[17].

2.1.2 ROS and ROS2

Robotic middleware is a software infrastructure that is intended to provide convenient abstraction and communication paradigms for facilitating this multi-subsystem approach. In general, a robotics middleware would provide a common interface design so that no matter what hardware is producing the data, the results are distributed in a consistent manner[18].

Robot Operating System (ROS or ros) is robotics middleware (i.e. collection of software frameworks for robot software development). ROS is not an operating system. It is a set of software

libraries and tools that help people build robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools and its all open source[12]. ROS is a software framework meant to allow people to write applications which operate robotic hardware. As its most fundamental level, it is an abstraction layer - offering hardware abstraction.

Robot Operating System is mainly composed of 2 things[13]: 1) A core (middleware) with communication tools. 2) A set of plug and play libraries.

ROS2, which is a new version of ROS that is under heavy development. Since ROS was started in 2007, a lot has changed in the robotics and ROS community. The goal of the ROS2 project is to adapt to these changes, leveraging what is great about ROS and improving what is not[26].

ROS1 uses a custom serialization format, a custom transport protocol as well as a custom central discovery mechanism. ROS2 has an abstract middleware interface, through which serialization, transport, and discovery is being provided. Currently all implementations of this interface are based on the DDS standard[27]. This enables ROS2 to provide various Quality of Service policies which improve communication over different networks. ROS1 is targeting Python 2. ROS2 requires at least Python version 3.5.

2.2 Pi Car Robots

The following sections introduce the hardware and operating system that used in this project. Pi Car S Robots is a car robot based on Raspberry Pi. It uses Raspbian based on Linux as operating system.

2.2.1 Raspberry Pi

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote teaching of basic computer science in schools and in developing countries.[1]

The Raspberry Pi is small sized like credit card, single board computer. Its capable of doing everything that a desktop computer can do, from browsing the internet and installing applications, even playing games. It runs Linux and other available operating systems from a micro SD card. With the addition of a keyboard, mouse and micro USB power supply it can be connected to a television or monitor and used as a fully functioning desktop computer. In short, the Raspberry Pi is a small computer with relatively limited memory and performance, but it contains all needed functions.

The first Raspberry Pi was released in 2012, since then it has been used in a vast range of projects due to its low cost, portability, programmability and both wired and wireless connectivity[8]. Several generations of Raspberry Pis have been released. The Raspberry Pi 3 and Pi Zero W (wireless) are equipped with 2.4 GHz WiFi 802.11n (150 Mbit/s) and Bluetooth 4.1 (24 Mbit/s) based on the Broadcom BCM43438 FullMAC chip with no official support for monitor mode. The Raspberry Pi 3B+ features dual-band IEEE 802.11b/g/n/ac WiFi, Bluetooth 4.2, and Gigabit Ethernet[2].

2.2.2 Ubuntu

2.2.3 Raspbian

Raspbian is an Linux based operating system which is free to download and use based on Debian optimized for the Raspberry Pi hardware. The Raspberry Pi can run different operating systems, they include: Windows 10 Arch Linux, Ubuntu and others. Raspbian is the most popular operating system for normal use on a Raspberry Pi and is also Sun Founders recommended OS for use with the Pi Car S robots used in this project.

Debian has millions of users, excellent documentation and many online knowledge bases, as such it is easy to find solutions to a vast range of issues. Debian uses the apt and dpkg tools for software package and dependency management to install software from Debian's free online repository of over 37,000 pre-compiled software packages[4].

Raspbian was created by Mike Thompson and Peter Green as an independent project[3]. The operating system is still under active development. Raspbian is specifically optimized for the Raspberry Pi line's low-performance ARM CPUs due to the low-performance hardware[9].

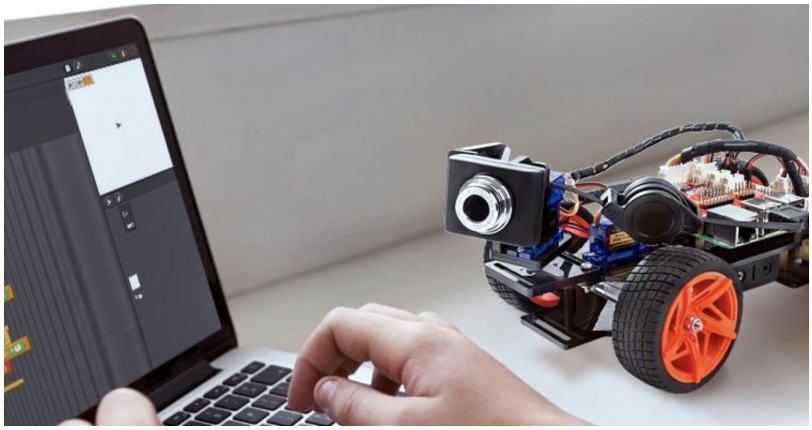
Being a linux computer, Raspbian supports basically many kinds of programming languages, mainly language like Python, Java, C/C++ etc. User just need to install the support if it's not installed. It is composed of a modified LXDE desktop environment and the Openbox stacking window manager with a new theme and few other changes[10]. Therefor, it is easy for user to interact.

2.2.4 SunFounder Raspberry Pi Smart Video Car

SunFounder is a technology company focused on Raspberry Pi and Arduino open source community development. The Pi Car-S Robot is a smart car which work based on a Raspberry Pi.

The raspberry pi car comes with three sensor modules including ultrasonic obstacle avoidance, light follower, and line follower. The car robot has durable and shatterproof plate and new style Servo, the MAX torque of the clutch gear digital servo is up to 1.6KG[11]. It rotates from 0-180 degrees.

Python code is provided for the car, and users can also program and debug it with Dragit, a Snap-based graphical interface, by just simple dragging and dropping the code blocks for more complex functions. A user can learn the programming conveniently on how to control the car.



2.3 Network Communication

2.3.1 Star Network Topology

A star topology is a network that has a central intermediate device. It is the most common WLAN topology in real life, as it is easy to install, scalable and easy to troubleshoot[21]. At the centre of the network sits the multifunctional wireless router providing the functionality of a wireless access point (WAP), a modem, a switch and a router.

Infrastructure wireless means having a device (access point) that gives structure to the network–in particular it will usually include a DHCP server, allocating IP address dynamically to every access devices. Using a wireless router is preferable because the networks are easier to set up and maintain, better security (WPA2 and WPA-PSK) is available, ethernet connected devices can be easily added, and internet access is much easier and better. However, a P2P wireless solution is necessary to allow free movement of the robots.

2.3.2 TCP vs UDP

Both TCP and UDP work on top of the IP (Internet Protocol). This is why there are terms such as TCP/IP or UDP/IP. In the OSI model, TCP and UDP are both ”Transport Layer” Protocols[19]. Where TCP is a connection oriented protocol and UDP is a connectionless protocol.

TCP and UDP are network protocols that are used to send data packets. These data packets are just bits of data that travel over the internet. When a user chat with his friends online, send an email, or send a page request through browser, he sends online data. This data is transferred in the form of tiny packets. While TCP and UDP are the most commonly used protocols, they arent the only ones used to transfer data packets. Another protocol that can be used is ICMP (Internet Control Message Protocol). However, most connections rely on either TCP or UDP.

There are three main differences between TCP and UDP[20]:

1. Reliability TCP is more reliable since it manages message acknowledgment and retransmissions in case of lost parts. Thus there is absolutely no missing data. UDP does not ensure that communi-

cation has reached receiver since concepts of acknowledgment, time out and retransmission are not present.

2. Ordering TCP transmissions are sent in a sequence and they are received in the same sequence. In the event of data segments arriving in wrong order, TCP reorders and delivers application. In the case of UDP, sent message sequence may not be maintained when it reaches receiving application. There is no way of predicting the order in which message will be received.

3. Connection TCP is a heavy weight connection requiring three packets for a socket connection and handles congestion control and reliability. UDP is a lightweight transport layer designed atop an IP. There are no tracking connections or ordering of messages.

Chapter 3

Installation and configuration

3.1 Ubuntu and ROS system installation

3.1.1 Steps

1. Down load the latest version image of Ubuntu with ROS in the ROS official website.

2. Write the image to Raspberry Pi by using SD card.

3. Start the ubuntu system with installed ROS and envonriment automatically.

4. Type ‘roscore‘ to test whether ROS system exist.

3.1.2 Issues and Solution

Raspbian 10(buster) is not well supported by ROS system. There are some outdated dependencies which is needed in installing ROS. Therefore, there are many bugs and manually installation steps to solve when user want to install ROS in Raspbian operating system. Then the ubuntu is recommend operating system as official website said.

3.2 Router network configuration

1. Each Raspberry Pi should connect to the same LAN.

2. Assign static IP address to each Raspberry Pi for ssh remote control and management.

3. Using one computer to join this LAN for controlling the all Raspberry Pi.

4. Remote control other Raspberry Pi by ssh using ‘ssh hostname@ip address‘ with password.

3.3 Communication between different ROS hosts

3.3.1 Change the host file

1. each host configuration file list should be ‘sudo vim /etc/hosts‘

2. Add ip address and hostname in ‘/etc/hosts‘ file of your Raspberry Pi. Each ip address respond to one hostname which you give. Example hosts file:

```
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

192.168.1.11 com1
192.168.1.12 com2
```

3.3.2 Talker and Listener

Enter ROS system in every ubuntu system and ssh to com1. Then ping com1 and com2. If ping successfully meaning the network is fine, otherwise check the router and network problem.

The host name and ip address is respond to each Raspberry Pi, but the Master URI should be identical. Create a .sh file to export environment paths.

```
# /bin/sh
export ROS_HOSTNAME=com1
export ROS_IP=192.168.1.11
export ROS_MASTER_URI=http://com1:11311/
```

Configure the file and execute this file in every Raspberry Pi host. Then run the talker and listener python file separatlly in different hosts to check if listener can receive messages from talker.

3.3.3 Issues and Solution

Make sure setting the ROS hostname and IP address correctly and must explicitly set it. Otherwise the listener cannot receive talker because the host cannot find it in the network. The Talker will send messages continuously but listener can not receive these messages.

Check the hostname and ip address setting can use:

```
‘echo $ROS_HOSTNAME‘
‘echo $ROS_IP‘
```

Chapter 4

Experiment and Analysis

4.1 Experiment 1: Revising wifi router performance experiment

Objective
This experiment investigates the performance in different frequency of message passing by using Wi-Fi network connection in a router. This experiment was executed by Issac in the previous and my aim is execute this experiment to compare it with the previous one.

Rationale
In order to acquire a detailed understanding of the performance characteristics of ROS communication channels, this experiment use two ROS hosts to commnunicate with each other and get the message latency results in different message frequency. I reproduced Issac's experiment to see what is the difference between my experiment and the previous one then analyze the reason to cause these differences.

Procedure
1. Install ROS and ubuntu in Raspberry Pi 3B+
2. Sending message and receive message between two hosts to test communication.
3. Sending 10HZ message and test message latency.
4. Run the code which send timestamped messages from the sender host to the echoer host. The sender will receive the message and then record message id into a file. The code is run 3 times with a range of message frequencies from 1hz to 2000Hz to obtain averaged results for each message frequency.

Hardware Configuration
2 Raspberry Pi 3B+
TP-Link 150M router

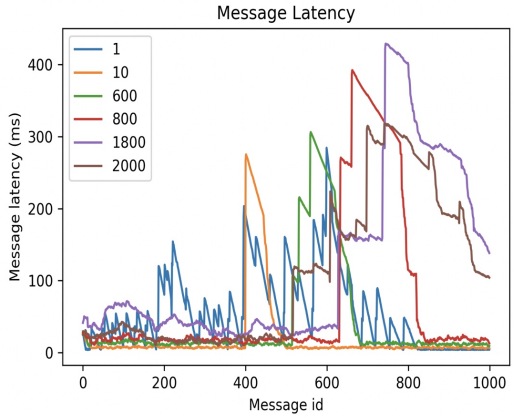
Software Configuration
Ubuntu 16.04
ROS Kinetic

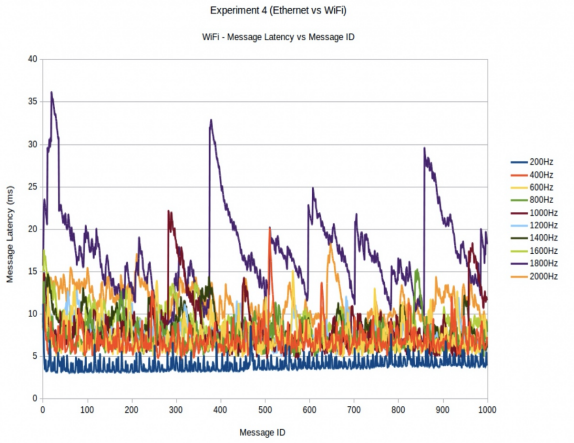
Hypothesis
1. The latency should be not regular(explain) due to the wifi connection is not stable.

2. Different message frequency may perform different latency due to the wifi environment.

Results

The data which I collect and plot is as follows, the left one is mine and the right one is Issac's experiment graph.





Conclusion

The previous experiment by Issac shows lower latency and I think that is due to the different hardware and some other environment factors.

The message latency is not regular and different frequency can perform relatively different latency. Wifi connection can be different in different time so that is why the latency is obvious in specific time period.

3

4

5

6

Chapter 5

Conclusion

1

2

Appendix A

First appendix

A.1 Section of first appendix

Appendix B

Second appendix

Bibliography

[1] Cellan-Jones, Rory (5 May 2011). ”A15 computer to inspire young programmers”. BBC News.

[2] ”seemoo-lab/nexmon”. GitHub.

[3] ”RaspbianAbout - Raspbian”. www.raspbian.org. Retrieved 2016-06-05.

[4] ”Debian”. https://www.computerhope.com/jargon/d/debian.htm

[5] ”batman-adv The Linux Kernel documentation”. www.kernel.org. Retrieved 2019-04-14.

[6] ”Linux 2 6 38”. Linux Kernel Newbies.

[7] ”P2P netwokr”. Dye et all 2008.

[8] ”Zero WH: Pre-soldered headers and what to do with them”. Raspberry Pi Foundation. Retrieved 12 January 2018.

[9] ”FrontPage - Raspbian”. www.raspbian.org. Retrieved 2016-04-04.

[10] ”Introducing PIXEL - Raspberry Pi”. Raspberry Pi. 2016-09-28. Retrieved 2017-01-07.

[11] ”SunFounder PiCar-S Kit V2.0”. https://www.sunfounder.com/picar-s-kit.html.

[12] ”ROS Melodic Morenia”. wiki.ros.org. Retrieved 10 June 2018.

[13] ”The Robotics Back-End”. https://roboticsbackend.com/what-is-ros.

[14] Avinash Gautam and Sudeept Mohan. A review of research in multi-robot systems. In 2012 IEEE 7th International Conference on Industrial and Information Systems (ICIIS), pages 15. IEEE, Aug 2012.

[15] Pedro U. Lima and Luis M. Custodio. Multi-Robot Systems. In Innovations in Robot Mobility and Control, pages 164. Springer, Berlin, Heidelberg, Aug 2005.

[16] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. International Journal of Advanced Robotic Systems, 10(12):399, 2013.

[17] Deborah Estrin, David Culler, Kris Pister, and Gaurav Sukhatme. Connecting the physical world with pervasive networks. IEEE pervasive computing, 1(1):6263, 2002.

[18] ”Robotic Middleware” Isaac Jordan. Experiment 1 - No Disk Writing Code. https://github.com/Sheepzez/ros-multirobot-evaluation/tree/master/experiments/message_latency/no_echo_delay_no_disk_write. Accessed: 2017-03-18.

[19] ”Communication Networks/TCP and UDP Protocols”. https://en.wikibooks.org/wiki/Communication_Networks/TCP_and_UDP_Protocols.

[20] ”Whats the Difference Between TCP and UDP?”. https://www.howtogeek.com/190014/htg-explains-what-is-the-difference-between-tcp-and-udp/.

[21] ”Star topology”. Cisco Networking Academy 2014.

[22] ”Mesh topology”. https://www.computerhope.com/jargon/m/mesh.htm.

[23] ”Open Mesh”. https://www.openmesh.com/datto-networking.

[24]”What Is Robotics?” https://www.nasa.gov/audience/forstudents/k-4/stories/nasa-knows/what_is_robotics_k4.html. Nov. 9, 2009

[25] B. Bauml and G. Hirzinger. Agile Robot Development (aRD): A Pragmatic Approach to Robotic Software. In 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 37413748, Oct 2006.

[26] ”Why ROS 2?”. https://index.ros.org/doc/ros2/

[27] ”Changes between ROS 1 and ROS 2”. http://design.ros2.org/articles/changes.html.