Proceedings of Engineering and Technology Innovation, vol. x, 201x, pp. xx - xx

Robotic Control by Teleoperation and Delay Time Issues Using the Internet

Anuntachai Machim*, Murray John Lawn, Ikuo Yamamoto

Department of Mechanical Science Graduate School of Engineering Nagasaki University, Nagasaki, Japan

Received 14 January 2019; received in revised form 13 February 2019; accepted 14 March 2019

Abstract

Internet usage has become an essential part of our daily lives, it has become universal. The aim of this research is to consider existing Internet-based networks with regard to their support for the remote operation of robotic technology, in particular, looking at transmission speed and delay time as it relates to teleoperation. Currently dedicated connections are typically used when a high Quality of Service (QoS) is required for mission critical services or safety-critical systems (SCS), however such connections are very expensive to set up and maintain. Therefore, this study focuses on modifying existing internet networks in a manner to provide a better QoS with little additional cost. Using a number of microcontrollers, computers, and routers, two different protocols were used to control a robotic device remotely. Use of the Point-to-Point Tunnelling Protocol (PPTP) used for implementing a Virtual Private Network (VPN) was found to provide higher average transmission speeds compared to the using a Uniform Resource Locator (URL) which is a regular internet connection.

Keywords: robotics, IoT, Internet, teleoperation, robot remote control, delay time, virtual private network, VPN

1. Introduction

From the infancy of the Internet on October 29, 1969, the first ever electronic message was sent between two computers that were literally side by side with their only physical connection being through the ARPANET, one of the first computer networks of the world [1-2]. The first message on the internet was “lo”, as in lo and behold! This technology has led to a technology explosion which continues to accelerate in many directions.

One application of the internet is as a medium through which one can control things for example, service mobile robots are controlled remotely over any TCP/IP enabled network [3], some research has shown the control of devices (servo motors) via internet, with accuracy between 97% to 99% [4], some researchers have used field-programmable gate arrays (FPGA) to interface robotic devices via the Internet [5], a VPN has been used to prevent robot control data from being accessible by other computers on the internet, creating secure connections between the Robot Operation System and authorized remote clients [6]. A Representational State Transfer (REST) architecture has been used to provide interoperability between computer systems via the Internet [7] in a similar manner to the use of PHP, SQL in this study. This type of control using the internet has come to be generally referred to as the Internet of Things (IoT). However, in regard to the sending and receiving of information, and the control of machines over the internet network in real time is problematic. In the case of controlling a robotic mechanism in real time, the issue of encountering loss of or the delay of data transmission is significant. This research focuses on correcting and/or improving the efficiency of existing internet connections to minimize this data loss and delay. In this research, we used

* Corresponding author. E-mail address: anuntachai.nut@gmail.com

hardware and software that are current, commonly used and relatively easy to use, specifically the authors used a Linux operating system (OS), python, and Arduinos. These are all open source ensuring good development support.

Transmission Control Protocol (TCP) Internet Protocol version 4 (IPv4) was used to transmit and receive data between the device's over the internet.

2. Methods

2.1. Hardware design

Regarding hardware design and the associated programming for such as a server, optimization of data handling to ensure efficient transfer of information or throughput is central.

2.1.1. Communication between computer, server and microprocessor - raspberry pi 3 model B (RPi3B)

The authors chose the SSH (Secure Shell) protocol for communication between the computer, server and microprocessor. In addition, the authors used PHP: Hypertext Preprocessor language and MySQL as an open source relational database management system (RDBMS) for checking the status of the devices, connected by LAN (Local Area Network) and WAN (Wide Area Network) networks through a router.

2.1.2. Communication between microprocessor and microcontroller - Arduino Pro Mini and Micropython

An ISP (In-circuit Serial Programmer) was used to communicate with Arduino Pro minis to upload programs and I²C (Inter-Integrated Circuit) for communicating with the raspberry pi and the Arduino - raspberry pi circuit schematics are shown in Fig. 1 and 2. Micropython was used to implement the UART bus to communicating with the raspberry pi.

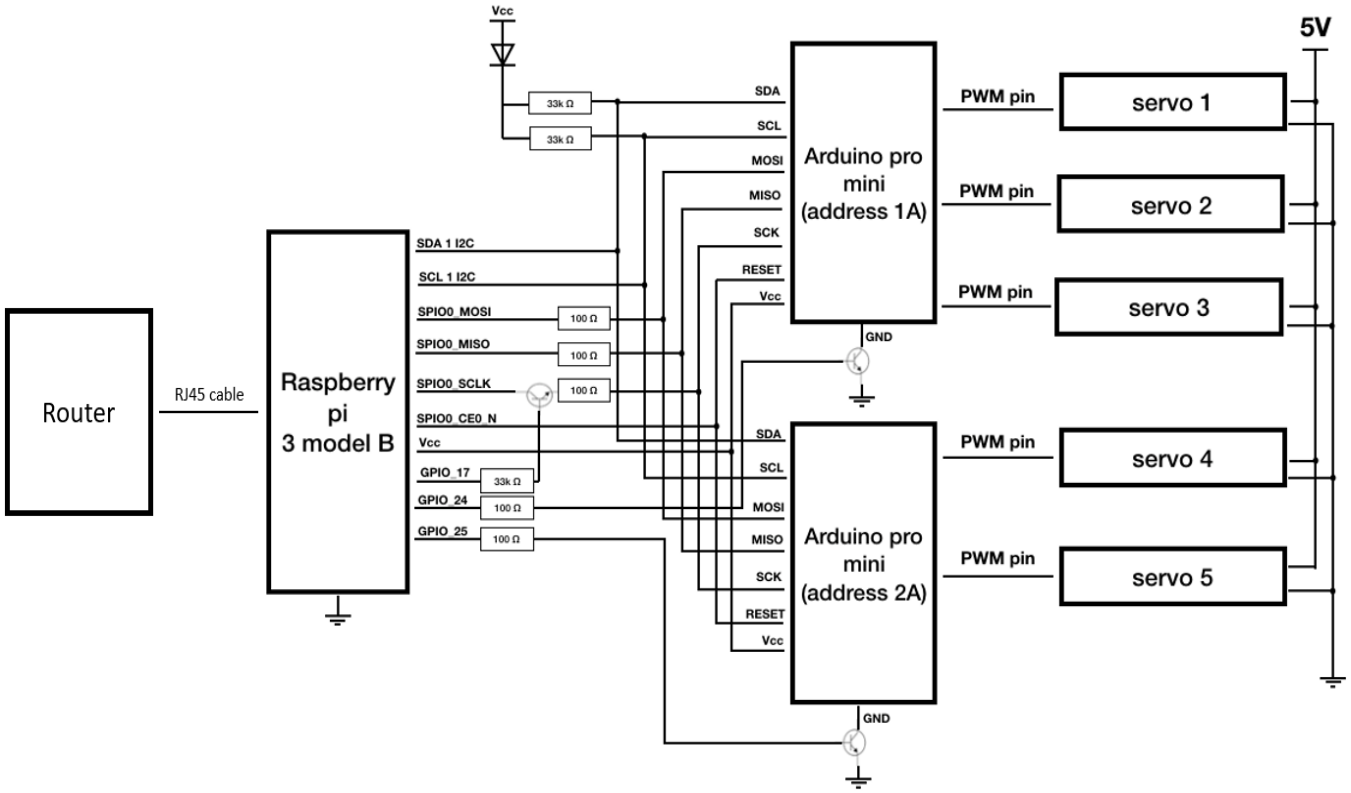


Fig. 1 Circuit for the data receiver

2.2. Programming

2.2.1. Server

A “Linus-Unix” based operating system using Debian 9 for programming was employed. Ubuntu 18.04 LTS (Long Term Support) was used for the server, supported by Linux Mint, Linux kali, and Raspbian functionality.

2.2.2 Control robot

The main program for the Arduino Pro Mini is written in Arduino, with other support programs such as Shell Script, PHP, Python3, and MySQL. Python2 allowed communication with Arduino using the I^2C and SPI bus for data transmission between the RPi3B and Arduino Pro Mini. The main Micropython program was written in Python3 to communicate with the Raspberry pi using Python2 for UART communication.

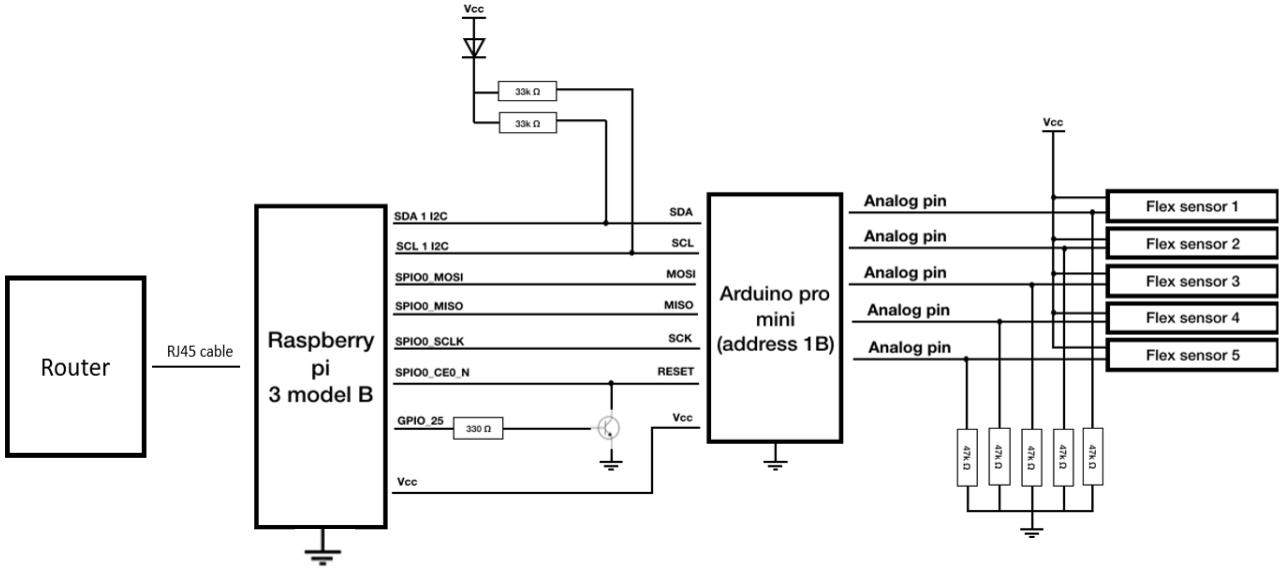


Fig. 2 Circuit for data transmission

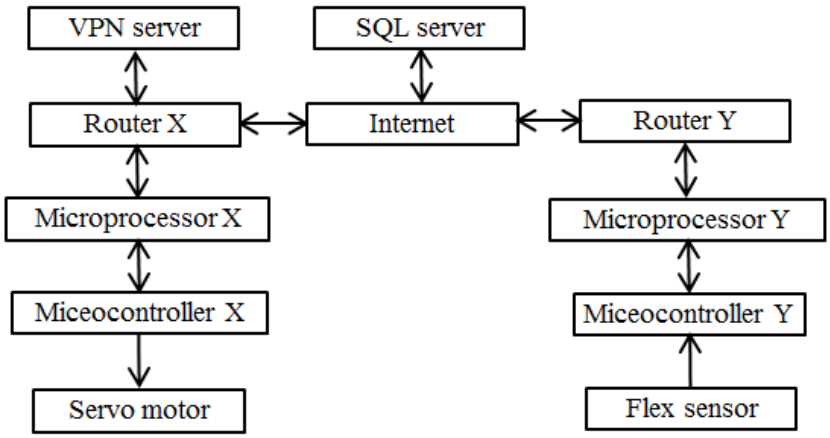


Fig. 3 Overall system communications schematic

2.3. Communication between microprocessor and microcontroller devices

Communication between the Raspberry pi and Arduino Pro mini enables the Raspberry pi to upload or edit Arduino programs using the SPI bus (Fig 1 and 2). It communicates via the I^2C bus, using Arduino for data transmission between the Arduino Pro Mini and the Raspberry pi, using an ethernet port (RJ45 port) to connect between the RPi3B and Router for data transmission over the internet. Using Python 2 on the Raspberry pi for sending data between the two devices using "import smbus" and "#include <Wire.h>" function libraries respectively. Two Micropython UART ports were used. One for writing and editing programs and another one for data transmission. The Raspberry pi used Python 2 "import serial" and Micropython used the "from machine import UART" command in Python 3.

2.4. Communication between the microprocessors

Using a raspberry pi 3 model B Microprocessor, the authors used Python to communicate between the two devices. By sending data via LAN cable using "import socket" from the Python2 library for transmission data. Both devices use the same

data port but are assigned as a receiver and messenger as required. The receiver identifies itself as the recipient, (the device as shown in Fig. 1 - schematic and Fig. 4 - raspberry pi) by specifying TCP IP as Localhost or specifying the IP Address of the device received from Router X, as shown in Fig. 3. and specifing PORT as 5005 (Fig. 5 - Receivers). The messenger will send the message to the target IP, or in this case the IP of the receiver (Fig. 5 - Messengers). In this case, the messenger has successfully connected to the VPN. The IP of the receiver was 192.168.12.198. The IP of the messenger was 192.168.12.x, where x was the number assigned by the PPTP VPN server [8] , so it is as if both are on the same LAN. From this point, both can communicate, despite being remotely located.

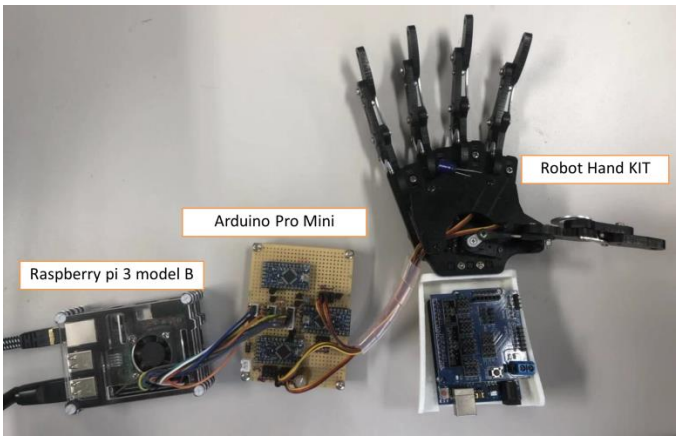


Fig. 4 Testing equipment

Messengers

```
import socket

TCP_IP = 192.168.12.198 #target ip
TCP_PORT = 5005 #target PORT
BUFFER_SIZE = 1024

...

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((TCP_IP, TCP_PORT))
s.send ( data ) #send data
s.close()

...
```

(a) Sending data

Receivers

```
import socket

TCP_IP = 192.168.12.198 #host ip
TCP_PORT = 5005 #open PORT
BUFFER_SIZE = 1024
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((TCP_IP, TCP_PORT))
s.listen(1)
...
conn, addr = s.accept()
...
        inputdata = conn.recv(BUFFER_SIZE) #Receivers
...
conn.close()
```

(b) Receiving data

Fig. 5 The main programs for sending and receiving data from both Microprocessors

Sending information on the URL using PHP and Python2 is illustrated in Fig. 6. Both microprocessors use Python2 “import urllib” and the “urllib.urlopen” to link the URL, then use PHP to connect the data to the database (SQL server). Here, it was necessary for Microprocessor X to read the data in a continuous loop.

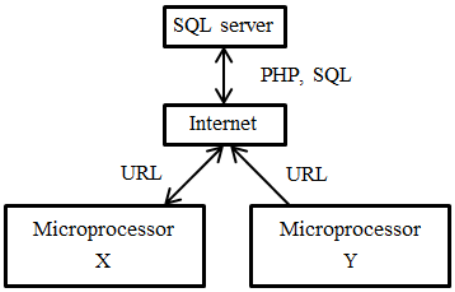


Fig. 6 Data transmission test on the URL and SQL server

2.5. System test

The system test configuration schematic is shown in Fig. 3. where Flex sensor data is generated by microcontroller Y, processed by Microprocessor Y and transmitted to the internet via Router Y, the circuit is shown in detail in Fig. 2. Router X has a public IPv4 or DDNS. Data reception is provided by Router X for processing by Microprocessor X for actuator control

via Microcontroller X, of which a detailed schematic is shown in Fig. 1. Data storage and monitoring are provided by the SQL server, Microprocessors X and Y send data to the database via the URL using PHP onto the MySQL database [9-11] and the VPN server functionality is provided.

First, both Microprocessors X and Y will send IPv4 values to the SQL server to check the device's operating status [12]. Secondly, Microprocessor Y will change the IPv4 status to emulate the same gateway as Microprocessor X at the same time Microprocessor X will check its status. Finally, both Microprocessors will connect as if they were on the same gateway and will be able to communicate using a shell script.

To test the system (remotely controlled robotic hand) the authors used a Mac OS terminal to simulate Microprocessors X and Y. The authors used a web browser on the Mac that is compatible with the “Raspberry pi zero w”. The Raspberry pi was equipped with a “Raspberry pi camera” and used the “Motion” program running on the native Raspbian OS. The authors used the “Terminal” on the Mac OS system to execute and check the data of both devices (Fig. 7). Here, Microprocessors X and Y functionality is simulated by a single physical device (Mac OS), however, Routers X and Y are separate routers connecting independently to different public IPv4 addresses (WAN).

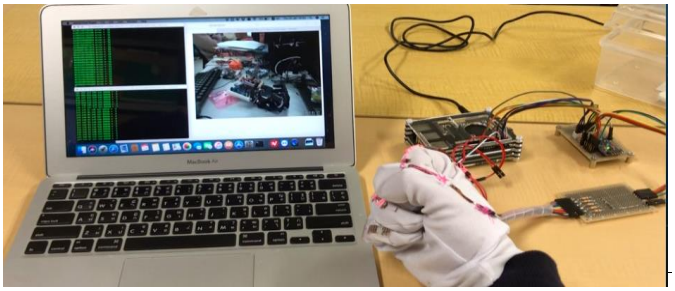
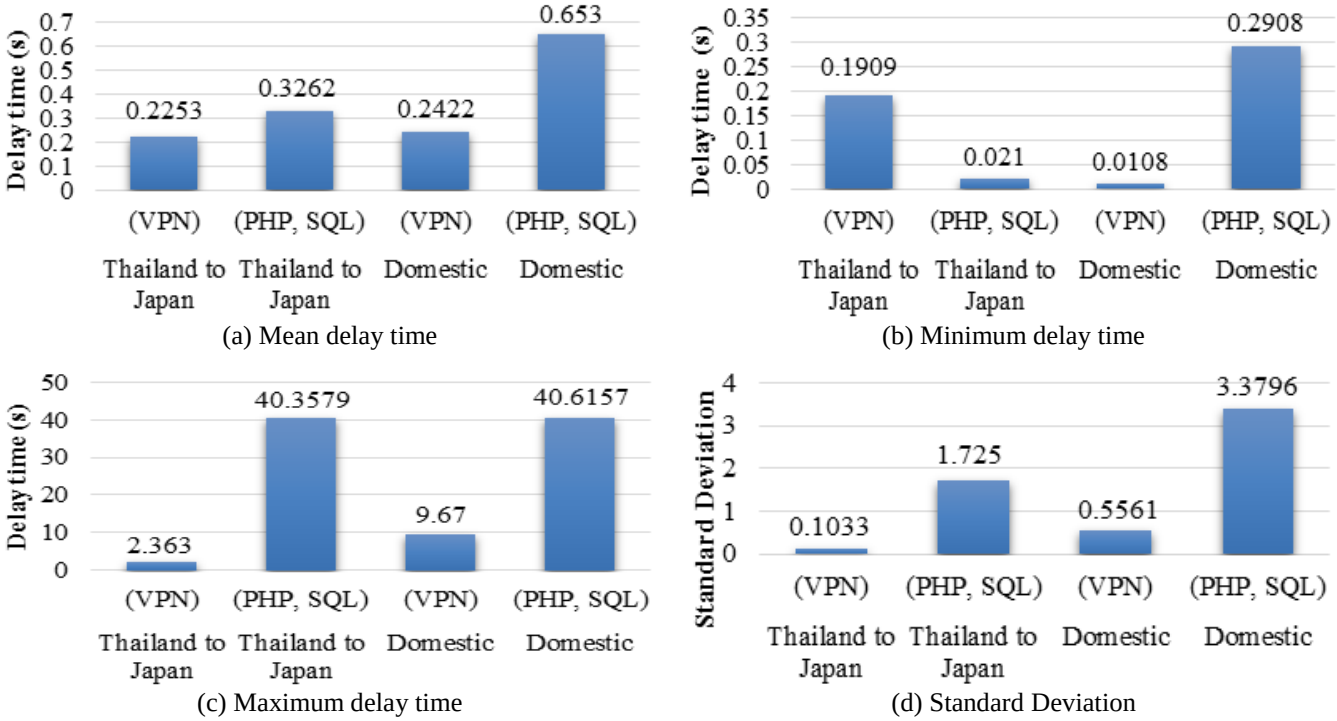


Fig. 7 Remotely controlled robotic hand test

3. Results



(a) Mean delay time

(b) Minimum delay time

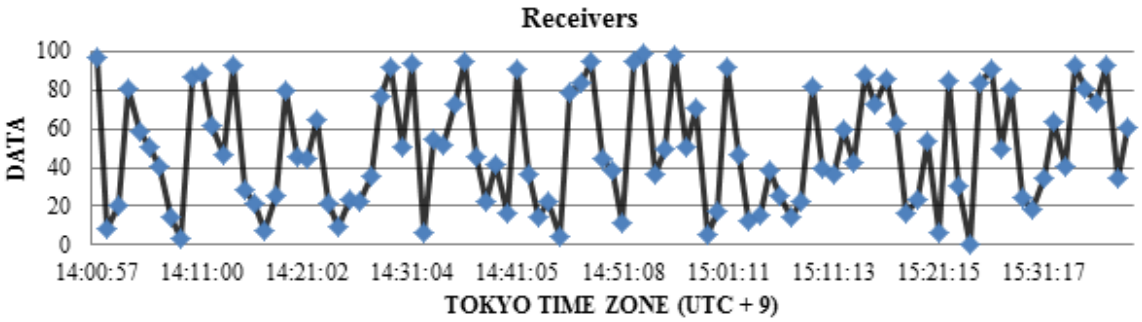
(c) Maximum delay time

(d) Standard Deviation

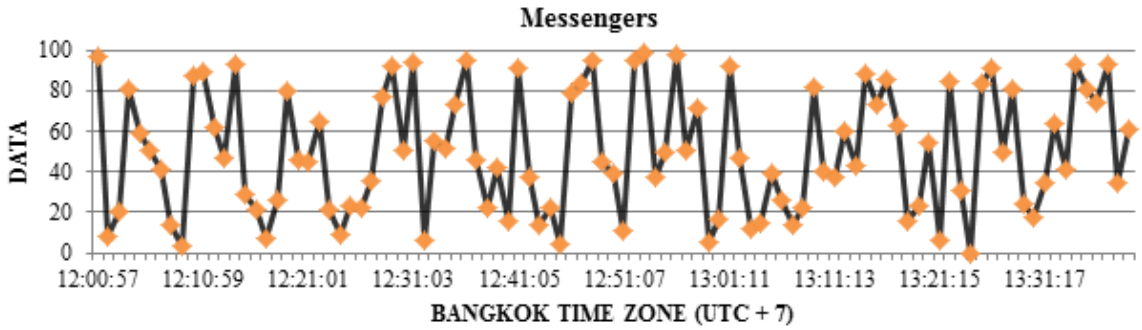
Fig. 8 Transmission data delay time between devices

Fig. 8. shows the transmission data delay between devices, clearly, in the case of a shared internet connection significant variation of speed will occur based on usage and is reflected in the maximum and minimum delay time variation. However, the

Standard Deviation values are small, indicating the minimal variation of the MEAN delay time. The Minimum delay time indicates the maximum data transmission speed possible. From this data transmission test Thailand to Japan, it can be seen that the VPN has an average speed that is faster compared with transmission via PHP, SQL in the magnitude of 1.45 times mean delay time and increased QoS [13]. But PHP, SQL does have the ability to send data faster based on the minimum delay times, in the case of sending information, from Thailand to Japan (For PHP,SQL the server is in Thailand which uses IPv4 unlike Microprocessors X and Y). The transmission times were measured by setting up databases in both the sending and receiving Microprocessors (RPi3B), the above times were based on Network Time Protocol (NTP) of RPi3B (debian.pool.ntp.org). Data trials were carried out by transmitting 24 bits at one minute intervals for one week.



(a) Information received from the device located in Thailand



(b) Information sent from the device located in Thailand

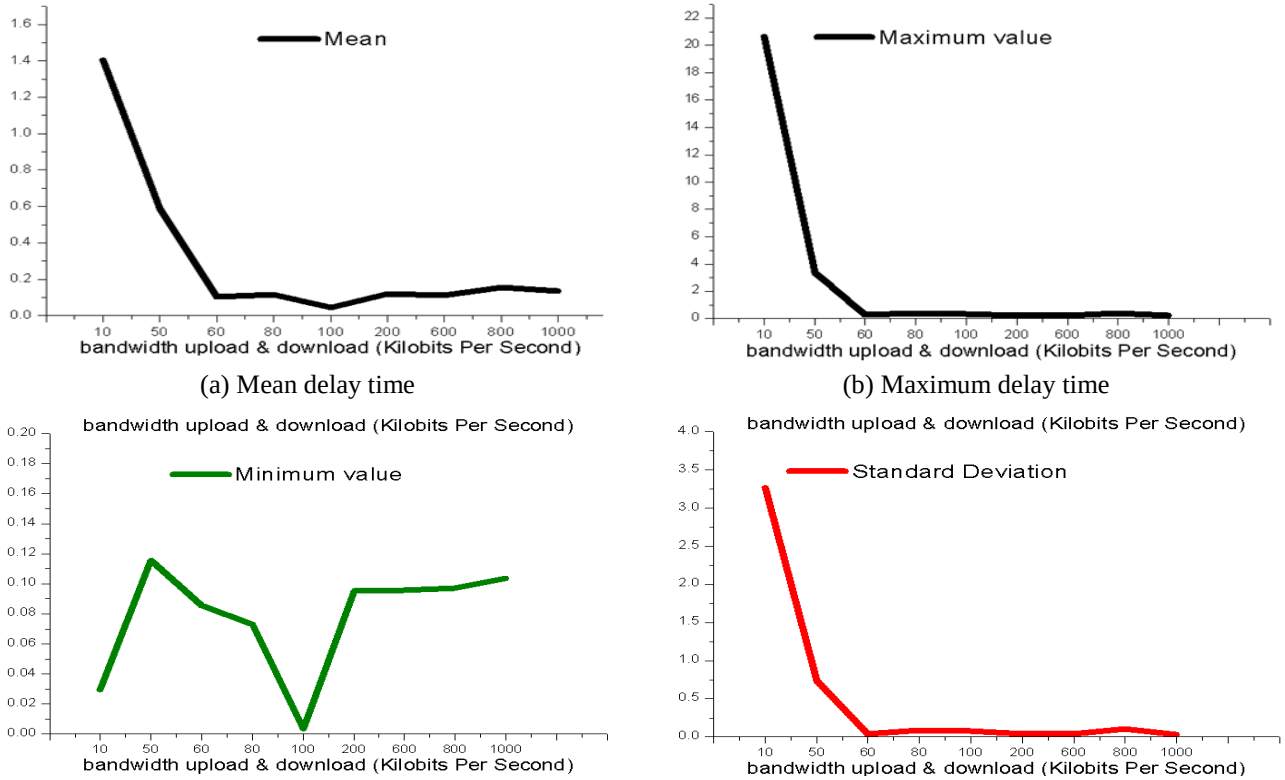
Fig. 9 Random data transmission experiments using VPN

From Fig. 9, the data sent from Thailand to Japan (time zone variation of 2 hours), shows that the random data sent was received without any errors.

Fig. 10 shows how bandwidth affects transmission delays. Clearly, reduced bandwidth will result in greater data transmission delays. However, it can be seen that a bandwidth of more than 60 kbps (Upload and Download) did not result in further significant reductions of delay time. Regarding bandwidth on a shared network, while maximum transmission speeds may be engineered, however, on account of the shared nature of the networks, actual bandwidth performance will be somewhat random. This is also most likely the reason for the lack of correlation between minimum delay time and bandwidth in the Fig. 10 minimum delay (instantaneous peak) values. While there are numerous factors that effect delay times. For example, some routers may check regular URL traffic but pass VPN traffic unchecked.

Based on the above results, communication between devices is more efficient as VPN usage results in faster average data transmission speeds of 1.45 times, reduced mean delay time Thailand to Japan and minimal average data transmission delays. Using VPN also offers many other advantages, such as making it simpler to edit programs and provide for remote system management.

However, data transmission by VPN connection and URL are both affected by network distance and bandwidth. As shown in Fig. 9, the delay time is directly affected by the type of data transmission, that is VPN or URL. Fig. 10 shows that the bandwidth directly affects the data transmission speed between devices.



(a) Mean delay time

(b) Maximum delay time

(c) Minimum delay time

(d) Standard deviation

Fig. 10. Bandwidth controlled data transmission test

4. Conclusion

This research focused on the selection of a protocol that would optimally suit remote control of such as robotic devices. Based on the results of the experiment, it can be concluded that using a VPN provided higher average data transmission rates, although using a regular URL resulted in minimum (instantaneous peak) delay time. Factors that impact internet transmission speeds are many and varied, such as the pathway taken by specific packets on an IP network from source to destination, Quality of Service (QoS), bandwidth management, specific Operating Systems, programming and algorithms used.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] R. Cohen-Almagor, “Internet history,” International Journal of Technoethics, University of Hull, vol. 2, no. 2, pp. 45-64, 2011.

[2] B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, and S. S. Wolff, “The past and future history of the internet,” Communications of the ACM, vol. 40, no. 2, February 1997.

[3] N. Chivarov, Y. Paunski, V. Ivanova, V. Vladimirov, G. Angelov, D. Radev, and N. Shivarov, “Intelligent modular service mobile robot controllable via internet,” IFAC Proceedings Volumes, vol. 45, no. 10, 2012, pp. 149-153.

[4] W. M. H. W. Kadira, R. E. Saminb, and B. S. K. Ibrahim, “Internet controlled robotic arm,” Sciverse ScienceDirect, Procedia Engineering, vol. 41, 2012, pp. 1065-1071.

[5] R. Dhanabal, Bh. S. R. P. Varma, and V. Bharathi, “Remote access of FPGA robot via internet,” 2014 International Conference on Electronics and Communication System (ICECS 14), February 2014.

[6] D. Portugal, S. Pereira, and M. S. Couceiro, “The role of security in human-robot shared environments: a case study in ROS-based surveillance robots,” 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), August 2017

[7] C. Xia, Y. Zhang, L. Wanga, S. Coleman, and Y. Liu, "Microservice-based cloud robotics system for intelligent space," Robotics and Autonomous Systems, no. 110, pp. 139-150, October 2018.
[8] A. A. Jaha, F. B. Shatwan, and M. Ashibani, "Proper virtual private network (VPN) solution," IEEE Xplore, January 2009
[9] A. Veglis, M. Leclercq, V. Quema, and J. B. Stefani, "PHP and SQL made simple," IEEE Distributed Systems Online, vol. 6, no. 8, pp. 1541-4922, August 2005
[10] H. Martínez-García, "Implementation of a remote laboratory for distance training in robotic applications," IEEE 13th International Conference on Industrial Informatics (INDIN), July 2015.
[11] S. H. Masovic, M. H. Saračević, P. Stanimirovic, and P. V. Krtolica, "Computing triangulations of the convex polygon PHP/MYSQL environment," Series Mathematics Informatics, vol. 34, no. 1, pp. 137-147, 2019.
[12] O. Goldstain, I. Ben-Gal, and Y. Bukchin, "Remote learning for the manipulation and control of robotic cells," European Journal of Engineering Education, vol. 32, no. 4, pp. 481-494, August 2007.
[13] S. H. Masovic, M. H. Saračević, P. Stanimirovic, P. V. Krtolica, and A. R. Gallon, "QoS-classifier for VPN and non-VPN traffic based on time-related features," ScienceDirect, Computer Networks, vol. 144, pp. 271-279, October 2018.