

## Lightweight wireless protocol based on IEEE 802.11 for delay sensitive telerobotic systems

Onur TOKER,<sup>1</sup> Haluk GÜMÜŞKAYA,<sup>2,\*</sup> Cihan ULAŞ,<sup>1</sup> Bora Tamer YILMAZ<sup>3</sup>

<sup>1</sup>Department of Electrical and Electronics Engineering, Faculty of Engineering, Fatih University, Büyükçekmece, İstanbul, Turkey

<sup>2</sup>Department of Computer Engineering, Faculty of Engineering and Architecture, Gediz University, Menemen, İzmir, Turkey

<sup>3</sup>Halk Yatırım, Şişli, İstanbul, Turkey

Received: 07.10.2011 • Accepted: 16.04.2012 • Published Online: 12.08.2013 • Printed: 06.09.2013

**Abstract:** In this paper, we consider wireless telerobotic systems and protocol development for low-delay wireless communication. Telerobotics can be defined as the control of robot arms from a remote location. In a telerobotic system, there is a robot arm to be controlled that is identified as the ‘slave arm’, and a remote operator at a distant location using a robotic manipulator that is called the ‘master arm’. In the control system measurements, actuator delays do degrade the system’s performance. Therefore, communication delays between the master and slave arms, and their minimization, are of extreme importance in telerobotics. In this paper, we first develop a new wireless communication protocol, the lightweight wireless protocol (LWP), designed on top of the 802.11 media access control layer. This low-delay wireless LWP protocol is implemented on an embedded system (AirDrop-LWP) without an operating system and its associated overhead. Finally, 2 AirDrop-LWP–embedded systems running this low-delay wireless LWP protocol are used to build a telerobotic system with a Mitsubishi RV-2AJ industrial robot. The LWP protocol is also tested on a robot car controlled by an AirDrop-LWP card as a slave arm and a standard PC as a master arm. The key features of the LWP are a reduced packet size, simple protocol stack, predictive compression of operator movements, and prediction of lost data packets. The LWP protocol is compared with the user datagram protocol and significant performance improvements are observed: a reduced delay of up to 50% and an additional 20% lower delay via compression. Variation in the packet delay times is also an important parameter for the wireless control system. As the standard deviation of the packet delay times increase, and becomes less and less predictable, the resulting telerobotic system will be more and more difficult to operate. We measured the standard deviation of packet delays and observed that it increases with the packet size, and this increase is faster than the increase in mean packet delay.

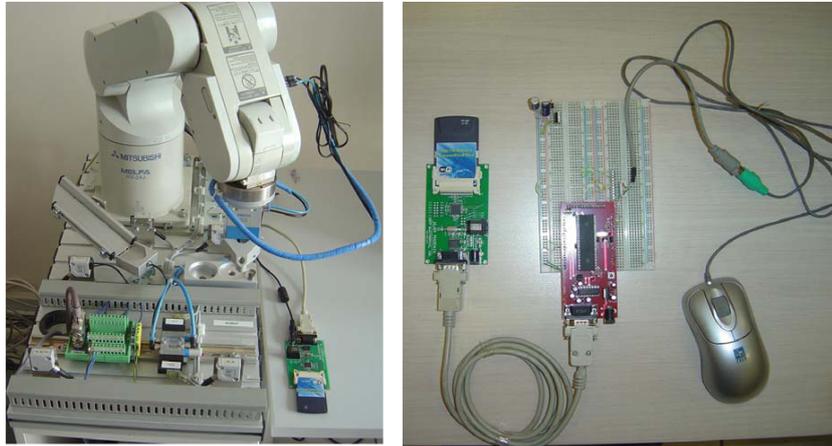
**Key words:** Telerobotics, wireless communication protocols

### 1. Introduction

Telerobotics is basically the control of robotic arms from a remote location, and it has wide-spread use in surgery; teleoperations in space; chemical, biological, and nuclear experiments; mine extraction; robotic bomb disposal systems; etc. In the most common setup, we have a robot arm, called the slave arm, and a robotic manipulator, called the master arm, located at a different point, and a human operator will control the slave arm by manipulating the master arm. In order to provide a more realistic telepresence feeling to the human operator, and hence achieve successful interactive operation, the minimization of communication delays is extremely

\*Correspondence: haluk.gumuskaya@gediz.edu.tr

important. In this paper, we address the design and implementation of low-delay wireless communication protocols, specifically for telerobotic systems. A new wireless communication protocol, called a lightweight wireless protocol (LWP), is developed, implemented on an embedded system called the AirDrop-LWP, and used to build a telerobotic system with an industrial robot Mitsubishi RV-2AJ. The complete system is shown in Figure 1.



**Figure 1.** On the left, the slave arm (Mitsubishi RV-2AJ) and AirDrop-LWP system running the LWP protocol. On the right, the master arm (a PS/2 mouse and its interface electronics) and another AirDrop-LWP system again running the LWP protocol.

The low-delay wireless LWP protocol is designed to minimize communication delays, and its key features are a reduced packet header size, predictive compression of the operator movements, and prediction of lost data packets. The LWP protocol is compared with the transmission control protocol (TCP)/internet protocol (IP) suite of protocols, and when compared with the user datagram protocol (UDP), major performance improvements are observed: without predictive compression, up to a 50% reduction in the delay, and with compression enabled, a reduced delay of up to 70% is observed in LWP-based systems compared to UDP-based systems implemented on the same hardware. Another important point that needs to be stressed is the fact that none of the reported results are pure computer simulations; they are indeed actual experimental measurements/results.

There are several research papers on wired and wireless telerobotic systems using TCP and/or UDP protocols. Similarly, there are several other telerobots based on ZigBee or Bluetooth. However, none of these wireless communication protocols are designed for the specific requirements of telerobotic systems. For example, in telerobotic applications, there is much less sensor and actuator data traffic compared to the traffic observed in large file transfers, and the packet losses are tolerable up to a certain degree. These are not among the design goals of the TCP/IP suite of protocols, and use of them will not result in the best possible performance. Furthermore, in a telerobotic application, the number of nodes is much lower compared to systems that can get a real benefit from ZigBee, and several features of Bluetooth will not be mission-critical in a telerobotic system. Despite all of these points, there are several telerobots built using the above-mentioned standard wireless protocols. The main reasons why these protocols are found in common use in telerobotic systems are the off-the-shelf availability of the required hardware and the abundance of technical documents and qualified engineers in these areas. In summary, commonly used wireless protocols are optimized for completely different

objectives and will result in suboptimal performance. The main goal of this paper is to design a wireless communication protocol optimized for telerobotic systems.

The LWP protocol is built on top of a 802.11 media access control (MAC) layer. Because of this, it uses a 2.4 GHz industrial, scientific, and medical band. A LWP packet has a very simple structure. The payload of a LWP packet is a minimum of 2 bytes or 1 word. An 802.11 packet carrying a LWP packet is much smaller compared to an 802.11 packet carrying a TCP or UDP packet. By reducing the packet size, both the processing time and the probability of collision will be reduced, and hence the communication delay per packet will be improved. By using predictive compression and the prediction of lost packets, further improvements can be obtained. These are the main reasons why the LWP is more suitable for telerobotic systems. On the other hand, for applications where average throughput and reliable communication are the main objectives, such as file transfers and web traffic, protocols with bigger packet sizes and more complex stacks are known to be more appropriate.

The LWP is implemented on the AirDrop-P card and the resulting embedded system is called the AirDrop-LWP. The AirDrop-P card is a PIC18F6722-based board with a compact flash (CF) interface, RS232, and in-circuit debugger (ICD) ports. The CF interface is used for the 802.11 module, which in turn has the well-known PRISM chipset in it. A simplified implementation of the TCP/IP stack on the AirDrop-P is given in [1], and also see [2] for wired networks.

The AirDrop-P does not have an operating system and the application developer should have a solid understanding of the TCP/IP stack implementation given in [1] in order to build a telerobotic system. The implementation given in [1] does not follow the OSI standard; however, this code was reorganized according to the OSI standard in [3].

The LWP protocol is designed and implemented on top of the 802.11 MAC layer. Therefore, the frequency, bandwidth, modulation, and all of the other physical layer (PHY)-related details cannot be changed. The LWP and the TCP/UDP also have the same MAC layers. Because of this, if a custom protocol is to be built on top of the 802.11 MAC, there is limited room for improvement. Software defined radios (SDRs) can be used to develop custom wireless communication protocols. In this paper, we also present our preliminary results on a particular SDR called the universal software radio peripheral (USRP) [4] and compare software platforms available for the implementation of new ideas, especially new wireless communication protocols. We present a comparative analysis of the available SDR software platforms, and then conclude that they are still at their infancy level compared to the software tools available for AirDrop-P-like systems. However, for future research, SDRs seem to be very promising for the development of much better and further optimized communication protocols.

This paper is organized as follows. In Section 2, we review the relatively recent wireless robotic systems reported in the literature. In Section 3, the AirDrop-P system architecture is presented. In Section 4, the newly developed LWP protocol is described and the implementation details of the telerobotic system shown in Figure 1 are presented. In Section 5, the performance analysis of the LWP is given. In this section, we present our experimental results. In Section 6, an M/M/1 queue model is given to analyze the mean and variance of the packet delay in an ad hoc wireless network. In Section 7, the second case study to test the LWP protocol, a robot car equipped with an AirDrop-LWP, is presented. In Section 8, SDRs and the currently available software platforms are comparatively evaluated. Finally, in Section 9, some concluding remarks are made.

## 2. Related work

There are a tremendous amount of research papers in the literature on wireless robotic systems. In this section, we will sample some of the more recent and relevant results.

In [5], different wireless alternatives, including the low-cost 433 MHz RF modules, IEEE 802.11, and IEEE 802.15.4, are evaluated for the RoboCup robotic soccer application. For such a system to be successful, communication delays between different robot units should be reduced to the minimum possible levels. The results reported in [5] suggest the use of a not-so-well-known wireless communication alternative called Linx, which is designed to support voice transmission. It is well known that in voice communication systems, delays are important and packet losses can be tolerated up to a certain point. Because of this, the results of Liu et al. were not very surprising. The key conclusion here is that a custom or a semicustom wireless communication alternative is optimal. However, a protocol optimized for voice communications need not be optimal for robotic systems.

In [6], the authors considered Aibo robots from Sony and used the wireless TCP/IP suite for communication. In [7], an embedded platform was established for intelligent mobile robots. The proposed system was based on the Windows CE.NET operating system running on a PC104 board and utilized wireless TCP/IP for communication. In [8] and [9], challenges in mobile robot systems were discussed and, again, wireless TCP/IP and ad-hoc wireless local area networks (WLANs) were considered. In [10], a medical telerobotic system architecture was presented, and the UDP over WLANs was the protocol of choice. In [11] and [12], the design and implementation of Bluetooth-enabled robots were discussed, where wireless communication was done using the Bluetooth protocol. In [13], IEEE 802.15.4 (ZigBee) was used for wireless communications on the Mini Car Pro platform. In [14], the use of multimedia protocols real-time transport protocol (RTP) and UDP was discussed.

The first author of the present paper also coauthored several papers on telerobotic systems [15–21], and all of them were based on the use of the standard TCP/IP suite of protocols for communication. In all of the above-mentioned papers, a standard wireless communication protocol, which was designed and optimized for a different application, was used for a robotics related application, and there was no effort to design a custom application specific protocol.

In [22], a new MAC layer protocol was developed for low-power and delay-sensitive wireless sensor networks, and the simulation results showed significant delay improvements. Compared to the available protocols, the custom-developed one again outperformed the existing ones, at least for the specific applications under consideration. In [23], a switched wireless network architecture was proposed that has a central coordinator node, and Nordic nRF24L01+ hardware was used on the mobile robots.

In [24], a different approach was recommended to deal with the communication delays in telerobotic systems. The operator controlling the robot arm sees a graphical environment where both the actual camera image and the simulated image of the robot arm are displayed in an overlapping fashion. This will be a semiclosed loop system, but it is a commonly used technique to deal with large communication delays. In this and similar papers, instead of trying to minimize the communication delays, the communication infrastructure is taken as it is, and alternative ideas are proposed ‘to live with communication delays’.

In [25], a soccer teleoperation system was developed, and again the TCP/IP suite of protocols was used for the wireless communication. It has been reported that the UDP and TCP have comparable average time delays, but that the UDP had a maximum delay that was almost 3 times shorter when compared to the maximum delay observed with the TCP. This has been attributed to the acknowledgment mechanism on the TCP. We would like to stress that, for the same reasons, there is no acknowledgment mechanism in the LWP, either.

In [26], different commercially available wireless networks cards were tested on a prototype system, including a wireless modem. As a performance measure, they used how the average throughput bit rate drops with distance, and they also provided a comprehensive literature review on wireless communication systems. On the other hand, we have adopted the one-way/round-trip time of a packet as a performance measure. In [27], surgical telerobots and different wired and wireless technologies were considered, and as the performance measure, they also considered the time delay.

In summary, the wireless TCP/IP suite of protocols was used in several telerobotic applications. We also see the use of Bluetooth, ZigBee, and RTP/UDP multimedia protocols and a couple of custom approaches where either a new MAC layer protocol was developed or a new centrally coordinated network architecture was proposed, or less widely known custom wireless communication devices were used. Compared to the available results reported in the literature, we propose the use of a custom wireless communication protocol called the LWP. The main design objective of the LWP was the minimization of the delays in telerobotics applications. Instead of using specific hardware, we opted for the use of 802.11 hardware and built a LWP on top of the 802.11 MAC layer. A compromise has been made between the reduction of the delays and the building of the system using commercially off-the-shelf available components. To the best of the authors' knowledge, there is no such balanced optimal approach reported in the literature, and this makes the current paper different and unique in the literature.

In this paper, we also consider SDRs that allow more flexibility in the protocol design, but we do not yet consider these components as commercially off-the-shelf available. However, in the near future, this may completely change as new and more advanced software platforms emerge for SDRs, which is expected to enable the implementation of new ideas less painfully and much faster.

### 3. The AirDrop-P system

In this section, we present a brief overview of the AirDrop-P system architecture. The AirDrop-P card shown in Figure 2 has a PIC18F6722 microcontroller on it. It is a simple board with a very small number of components on it [1]. The board has a CF interface for the 802.11 module, a RS232 level shifter integrated circuit (IC) (SP3232), a female RS232 port, and an ICD port for programming the flash memory of the microcontroller and for other debugging tasks. The PIC18F6722 IC of the AirDrop-P has a 128 KB program memory (flash), 3936 bytes of RAM, and 1024 bytes of data EEPROM, and it is in 64-pin thin quad flat packaging. The microcontroller itself can operate from 2 V to 5.5 V and can be used with clock speeds of up to 40 MHz. On the AirDrop-P board, the LM1086CS-3.3 is the 3.3 V voltage regulator. Because of the lower operating voltage level (3.3 V), the clock speed is reduced and chosen as 20 MHz to be in a safe operating region.



**Figure 2.** AirDrop-P card [1]. When the LWP protocol is loaded into the flash program memory, the overall embedded system is called the Airdrop-LWP.

There is no operating system running on the AirDrop-P, and so the code running on the microcontroller has to directly communicate with the 802.11 module. This module on the AirDrop-P has the so-called PRISM chipset on it, and there is a part of the code that is PRISM chipset-specific. This part may need to be modified if a different chipset-based 802.11 module is used. This part of the code can be viewed as the ‘driver’, although there is no operating system running on the board. In [1], a simplified implementation of the TCP/IP stack for the AirDrop-P system was developed. The wireless LWP protocol code developed in this paper does not use the TCP/IP stack in [1]. The TCP/IP stack in [1] is used only for performance comparison purposes. However, we greatly benefitted from this code during the implementation of the LWP protocol.

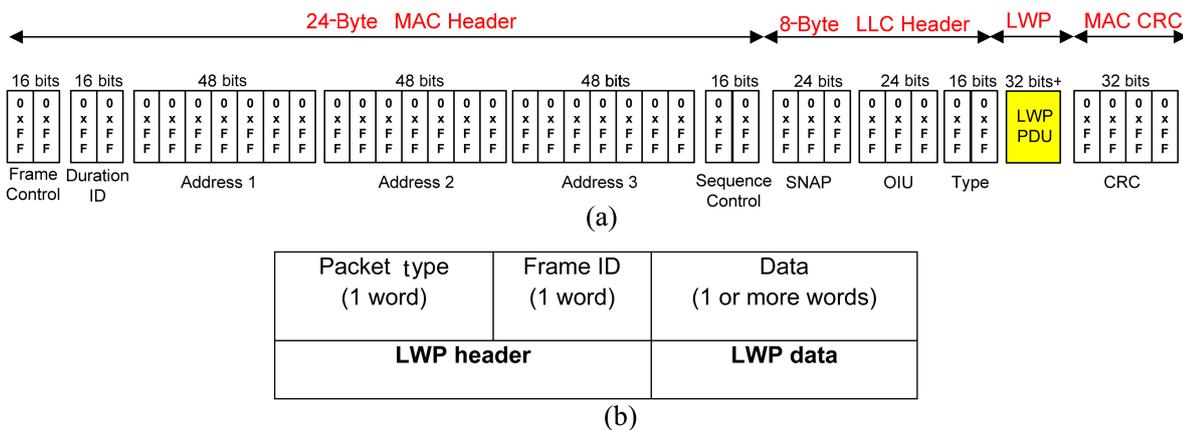
The AirDrop-P card running the LWP protocol code will be called the AirDrop-LWP. In a telerobotic application, the RS232 port of an AirDrop-LWP can be used for communication with the slave arm (Mitsubishi RV-2AJ in Figure 1, on the left), or a master arm (PS/2 mouse used as a robotic manipulator in Figure 1, on the right). In a typical telerobotic setup, at least 2 AirDrop-LWP units will be required, the first connected to the slave arm and the other connected to the master arm.

#### 4. Lightweight wireless protocol

In this section, we present the LWP protocol design details. A typical TCP or UDP socket-based telerobotics application depends on the IP and the UDP or TCP protocols. This results in overhead in protocol headers and protocol processing time. We have chosen to utilize the flexibility of embedded systems by developing a custom-made packet structure of the LWP just above the MAC layer. The LWP sits on top of the logical link layer of the OSI model, right where an IP packet header starts. The applications using the LWP do not have the overhead of the IP and the TCP or UDP headers and their processing time. For specific implementation details of the LWP, we refer to [3].

##### 4.1. LWP packet protocol data unit

An LWP packet has a very simple protocol data unit (PDU), as shown in Figure 3. In the LWP header, there is 1 word used for the packet type, which is fixed and is equal to 0x1001, and there is another word called the Frame ID, which is used for sequencing purposes. Because of the packet-type field, a LWP packet can be distinguished from an address resolution protocol, IP, or internet control message protocol packet, and hence the LWP protocol can coexist. After the LWP header, the data include one or more words depending on the application.



**Figure 3.** The LWP packet PDU. (a) The LWP packet embedded in an 802.11 packet structure. (b) LWP header and data.

The 802.11 packet starts with a standard 24-byte MAC header. The bulk of these 24 bytes, i.e. 18 bytes, are allocated for the communicating stations on the network. Since on an infrastructure-based wireless network, an access point (AP) needs to be present by design forcing every single packet to pass through the AP, a third address field is required. Depending on the instance, an 802.11 packet in the air can be destined either from a station/node to the AP or vice versa.

Following the MAC header, 8 bytes are allocated for the logical link layer requirements. Three bytes are for the subnet information. These 3 bytes never change, as the destination and source access point fields are always marked “0xAA”.

The last 5 bytes of the LLC header are for the router to distinguish what type of a packet is being processed. The 3 bytes are similar to the first 3 bytes and are fixed at “0x00”, indicating an encapsulated Ethernet. The 2 bytes at the very end of the LLC header determine the type of the encapsulated protocol and we have chosen “0x9999” for the LWP.

The LWP PDU sits where an IP header starts on a regular IP communication. As an integral part of Wi-Fi communication, the last 4 bytes are allocated for the MAC cyclic redundancy check.

#### 4.2. Addressing and packet length in LWP

The LWP header does not carry any information about source and destination port and IP addresses, which is not the case for the TCP/IP suite of protocols. Similarly, there is no field about the packet length in the LWP header. The elimination of the address and packet length information from the LWP header will both reduce the packet size and simplify the protocol implementation. However, the LWP addressing will be limited to what kind of addressing can be done via the lower layers. For the wireless communication system proposed in this paper, there will be one ad hoc network for all of the LWP communications, and addressing is limited to this domain. This will be the price paid for a smaller packet header. As mentioned above, there is no field about packet length in the LWP header. One alternative, which is against the OSI model, is to use the packet length information of the 802.11 MAC header, but in most telerobotic applications, the packet length can be determined from the packet type; therefore, omission of the packet length field will not be a major issue.

#### 4.3. Packet types in the LWP

The LWP protocol is implemented on top of the 802.11 MAC layer and has a simple packet structure, as shown in Figure 3. The LWP protocol is designed to coexist with the TCP/IP suite of protocols in a wireless environment, and its packet header consists of 2 words. Packet Type is one word, and it is fixed to 0x1001 in our telerobotic system shown in Figure 1. For more complex telerobotic systems based on the LWP protocol, we reserve the packet types and interpretations shown in Table 1.

**Table 1.** Packet types in the LWP protocol.

Packet type (1 word)	Interpretation
0x1001	LWP packet carrying the position data
0x1002	LWP packet carrying the force feedback data
0x1003	LWP data carrying the urgent ALARM code

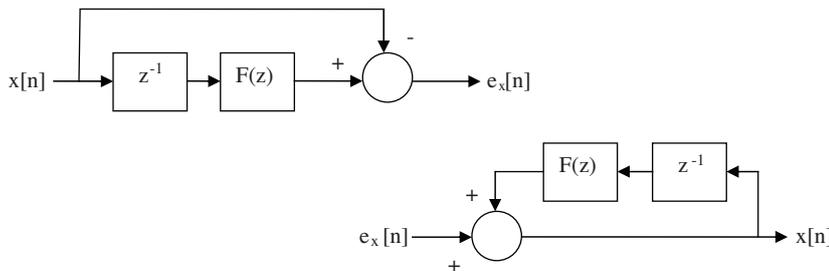
In several high-end telerobotic systems, force feedback and alarms may be used. The LWP packets carrying alarm codes will not be compressed, as they are expected to be rarely used. However, the LWP packets carrying force feedback data as well as position data should be compressed to further reduce the packet size and associated communication delays.

**4.4. Frame ID, sequencing, and acknowledgment in the LWP**

LWP packets basically carry position and/or force data between master and slave arms. Each LWP-enabled embedded system (in our case, AirDrop-LWP) collects position and/or force measurements and transmits this information to other LWP-enabled embedded system units. All of these tasks are done in a periodic fashion, but there is no flow control mechanism to adjust the period, and period selection is considered to be the responsibility of the system operator. Similarly, there is no formal acknowledgment mechanism in the LWP aside from the ones provided by the lower layers, i.e. the limited acknowledgment mechanism available in 802.11. The elimination of the flow control and acknowledgment mechanisms are quite radical decisions but this will definitely reduce delays, provided that the period selection is done properly. During our performance tests, we also noticed that the channel selection does affect the system’s performance, as well. The LWP does not define a formal channel selection procedure, either, and is considered as another responsibility of the system operator.

**4.5. Compression in the LWP**

The fundamental design goals of the LWP are simplicity and the use of the smallest possible packet sizes. Measured data, whether of position or force, are passed through a prediction-based compression unit. This is a well-known technique in signal processing and data compression.



**Figure 4.** The predictive compression block diagram (top left) and the decompression block diagram (bottom right).

The basic idea in the compression technique given in Figure 4 is to find a filter,  $F(z)$ , that can be used to predict  $x[n]$  from its past values. Here  $x[n]$  will be either a position or a force measurement,  $F(z)$  is the prediction filter, and  $e_x[n]$  is the prediction error. The number of bits required to represent the prediction error  $e_x[n]$  will be less than the number of bits required for  $x[n]$ , provided that the prediction filter  $F(z)$  is chosen properly. Hence, instead of sending  $x[n]$  via a LWP packet, the transmission of  $e_x[n]$  is better, as long as both parties agree on the use of compression and the same prediction filter  $F(z)$ .

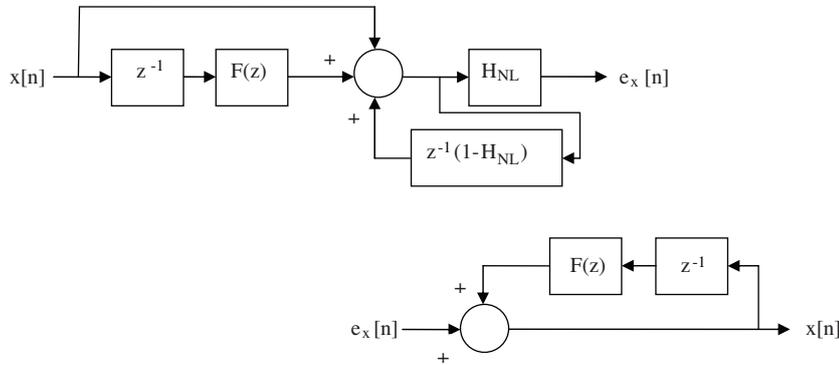
In the telerobotic system shown in Figure 1,  $x[n]$  is a 2-dimensional signal corresponding to the relative movements of the PS/2 mouse in the X and Y directions. These relative movements are compressed with the prediction filter:

$$F(z) = [1 + \alpha_1(1 - z^{-1}), 1 + \alpha_2(1 - z^{-1})]^T, \tag{1}$$

where  $\alpha_1$  and  $\alpha_2$  are constants in  $[0, 1]$ .

In our implementation, we reserved a fixed number of bits (7 bits) for  $e_x[n]$  and incorporated saturation nonlinearity  $H_{NL}$  to guarantee that the error information that will be sent does not exceed this predetermined number of bits. The remaining portion of the error that could not be sent is added to the error generated in the next sampling time; see Figure 5. In this case, the  $x[n]$  output of the decompression block is not necessarily

equal to the  $x[n]$  input of the compression block, because of the potential saturations in the error signal. We observed that while this system guarantees a bound on the number of bits for  $e_x[n]$ , it also quickly recovers from occasional saturations because of the delayed residual error feedback. Ideally, both of the  $x[n]$ s in Figure 5 should be equal to each other, and the delayed residual error feedback tries to restore the equality each time saturation occurs.



**Figure 5.** The residual predictive compression block diagram (top left) and the decompression block diagram (bottom right).

**4.6. Prediction of lost packets in the LWP**

There is no acknowledgment mechanism in the LWP; therefore, on the receiver side, there will be a hardware timer (in our implementation, TIMER3 of PIC18F6722) that will be reset after the reception of each new packet. If the timer exceeds a certain predefined threshold, which is 20 ms in our implementation, the packet will be assumed to be lost, and its data will be predicted from the previously received packets. In our implementation, we have used a first-order filter for the prediction of the lost data; however, as long as both parties agree, more complex prediction mechanisms can be used, as well. Finally, the packet that is assumed to be lost because of an expired timer may indeed arrive after the prediction phase, but it will be simply ignored and the timer will not be reset for this particular packet.

There is a standing assumption that the frame ID data can be used to identify individual packets. In principle, during the transmission of a packet, the frame ID field may get corrupted, and the whole LWP protocol stack may malfunction. Here again, we rely on the error checking mechanism provided by the lower layers and choose to not incorporate further any error checking/correction mechanism in the LWP layer.

**4.7. Packet combination in the LWP**

The minimum payload of a LWP packet is 16 bits. If in a particular system, the payload turns out to be 8 bits or less after the compression, which is the case in our implementation, more than 1 prediction error datum may be loaded to a single LWP packet. This will cause the second error data to be delayed more, but all of the LWP payload will be utilized in this approach, and the overall packet traffic will be reduced. This is expected to reduce the collision probability, as well.

**4.8. Flow diagram of the LWP protocol**

The flow diagram of the LWP protocol running on 2 AirDrop-LWP cards on the master arm and slave arm sides is shown in Figure 6. The algorithm running on the master arm side is basically responsible for sending the data.

In telerobotic applications, there will be a sensor or multiple sensors on the master arm side. In the LWP, as soon as a master arm movement is detected, a timer is started with a recommended timer expiration/overflow value of 10–100 ms. Instead of sending several small movements as separate packets, the LWP protocol basically tries to combine several small movements that are detected until the timer expires, and then sends the combined vector movement as a single packet. This does introduce a performance penalty in the form of a slightly increased delay for some of the detected movements, but when the overall system overhead is taken into account, the interactive performance of the system is improved. The combined movement is passed to an optional prediction filter for

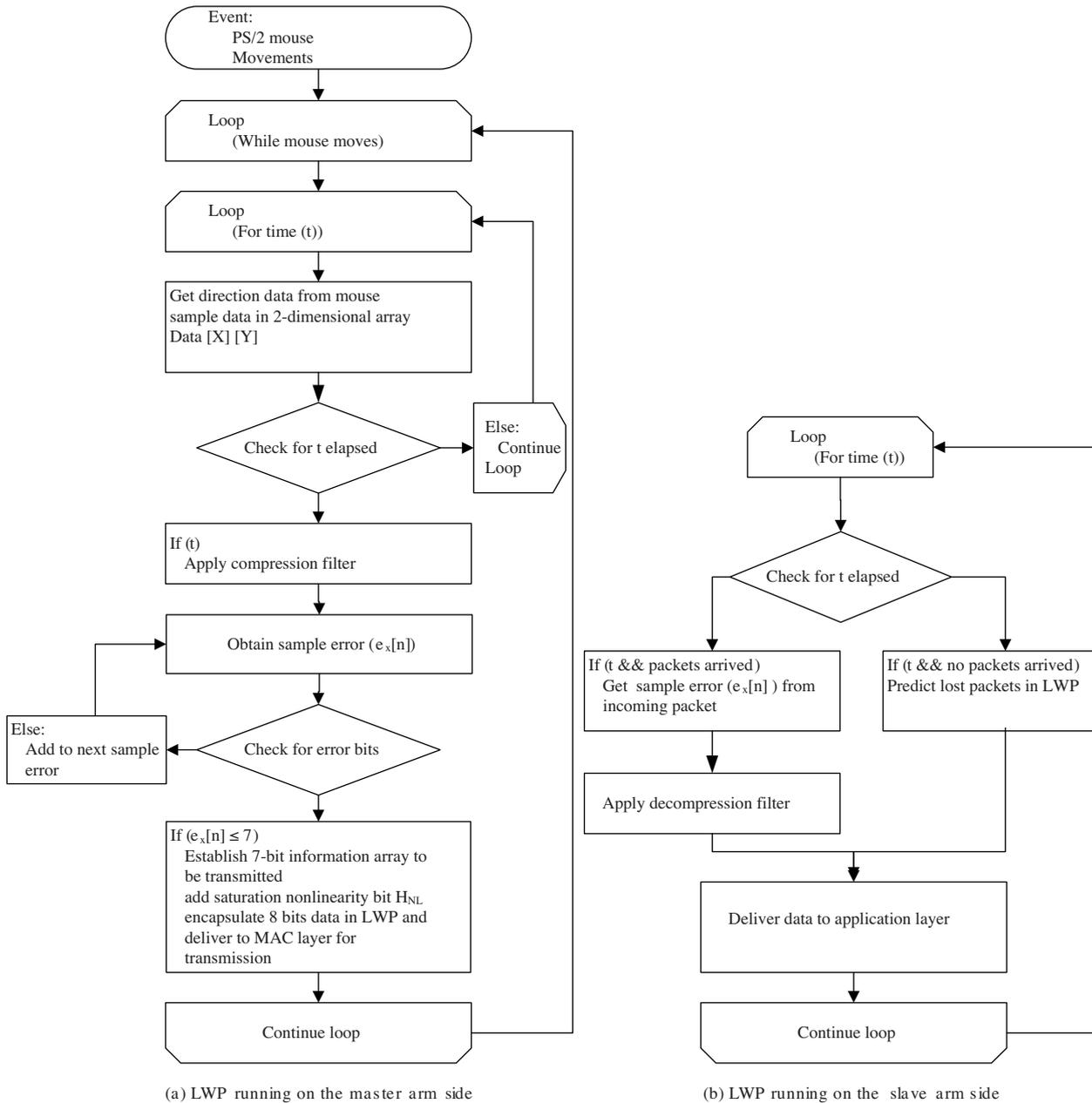


Figure 6. Flow diagram of the LWP Protocol running on the master arm (a) and slave arm (b) sides.

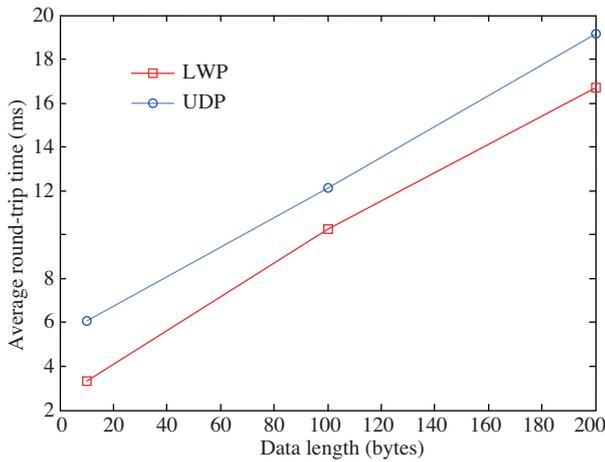
the data compression, which is intended to reduce the data size. There are certain PHY-based minimums on the packet size; therefore, the overall effect of the prediction filters/compression blocks may not be very significant for a single sensor system, but it will be quite noticeable for a master arm with multiple sensors. Moreover, for applications where a couple of high resolution sensors are used, again the effect of the compression will be noticeable. After the compression block, the resulting prediction error and/or saturation data are used to form the LWP packet, which is then passed to the MAC layer.

On the slave arm side, after each received packet, the decompression algorithm is used to extract the raw data, which is the actual master arm movement, and they are then passed to the robot arm acting as the slave unit. If no packet is received within a specified period of time (the recommend waiting time values are a couple of 100 ms), a separate prediction process is started to predict the lost packet. In the LWP, we recommend the use of the past 2 or 3 samples and predict the lost packet using a linear extrapolation.

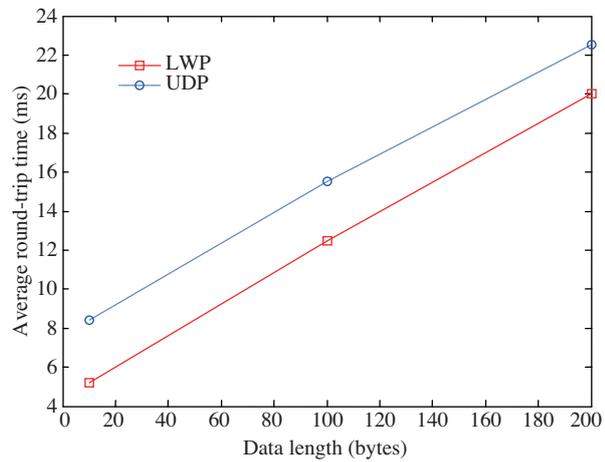
**5. Performance analysis of the LWP**

In this section, we present the performance test results of the LWP. There are a total of 3 tests performed, and for each case, the LWP protocol is compared with the UDP protocol, which is also connectionless and does not have an acknowledgment mechanism. The same ad hoc network is used for all of the tests. To assess the protocol performance, we conduct the standard echo test for different data sizes. Namely, for a given data size, we generate random data having this predetermined size and load these random data onto a LWP or UDP packet, and then measure the average round-trip time for the standard echo test. Averaging is done over 10,000 packets, and the results shown in Figures 7, 8, and 9 are obtained.

We first consider the ‘noncongested traffic’ case and conduct the performance tests when there is not much external traffic in the 2.4 GHz band. In Figure 7, we see that as the size of the data is increased, the echo times increase almost linearly, but there is more or less a constant gap between the LWP and UDP round-trip times. Basically, the advantage of using the LWP instead of the UDP is clearly demonstrated.



**Figure 7.** The LWP and UDP round-trip times (‘noncongested traffic’ case).



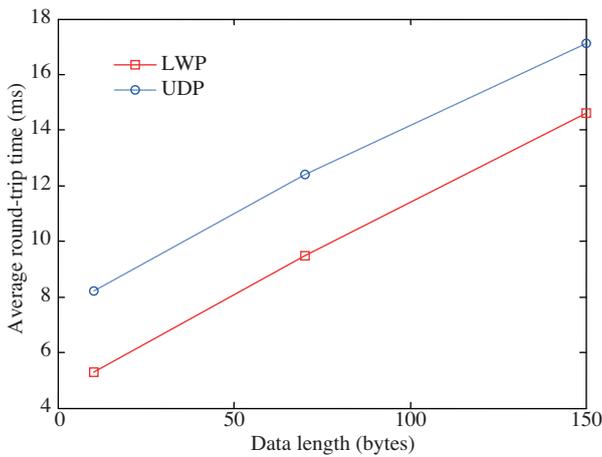
**Figure 8.** The LWP and UDP round-trip times (‘low received signal strength’ case).

For the second experiment, we consider the ‘weak signal’ case and conduct the performance tests when the signal levels are low. To make sure that the received signal strength (RSS) values are low, we use aluminum foil to partially cover the 802.11 antennas. In Figure 9, we see that as the size of the data is increased, the echo

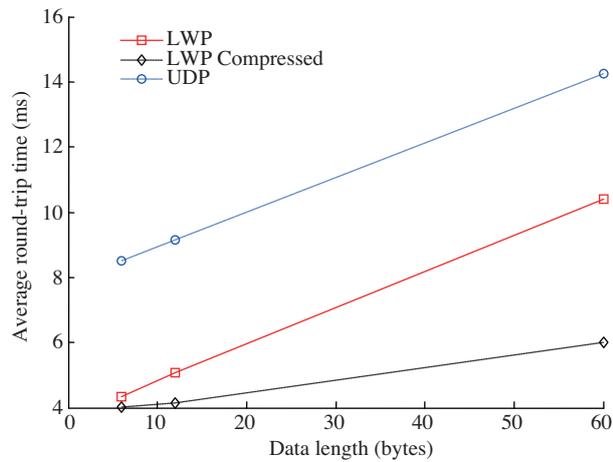
times increase as well, but for the ‘weak signal’ case, differences between the LWP and UDP round-trip times become more significant.

In the third test, we consider the ‘congested traffic’ case and conduct the performance tests during the daytime when there is more traffic in the 2.4 GHz band. In Figure 9, we see that as the size of the data is increased, the echo times increase as well, but for the ‘congested traffic’ case, differences between the LWP and UDP round-trip times become more significant.

In the fourth and the final test, we again consider the ‘congested traffic’ case and compare the round-trip times of the UDP, LWP, and LWP compressed. In Figure 10, we see that as the size of the data is increased, the echo times increase as well, but the advantages of the compression are clearly visible. Since the data compression reduces the payload proportional to the data size, the ‘LWP compressed’ in Figure 9 has a smaller slope.



**Figure 9.** The LWP and UDP round-trip times (‘congested traffic case’).



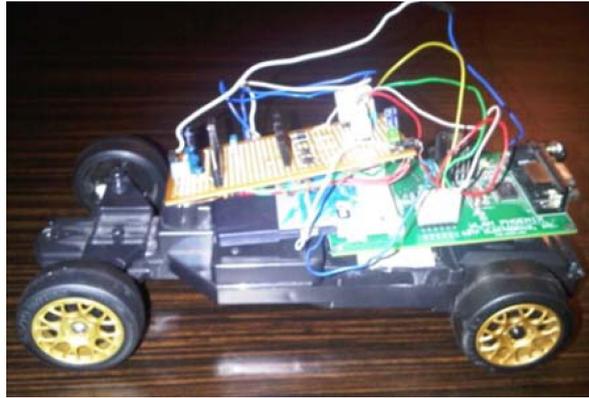
**Figure 10.** The UDP, LWP, and LWP with compression round-trip times.

In summary, our performance tests show that for the same data size, the LWP has a smaller communication delay compared to the UDP. Communication delays do increase under increased traffic congestion or reduced RSS; nevertheless, the LWP has less of a communication delay compared to the UDP. Although we did not conduct tests for the LWP versus the TCP, the TCP, being a connection-oriented reliable communication protocol with an acknowledgment mechanism, will show even larger differences in the communication delays when compared to the LWP.

Finally, we see similar patterns when we consider the minimum round-trip time over 10,000 trials or the maximum round-trip times over 10,000 trials.

**5.1. The second test study: a robot car equipped with an AirDrop-LWP**

The LWP protocol was also tested on a robot car to develop basic motion planning mechanisms to operate a telerobotic system in 4 directions. A complete wireless robot car using the proposed LWP protocol was designed and developed; performance tests and communication delay measurements were conducted in [28] and [29]. These studies use the protocol and do not present details of the protocol and theoretical work presented in this paper. The developed system has a model robot car equipped with an AirDrop-P card as a slave arm, as shown in Figure 11, and a standard PC as a master arm.



**Figure 11.** The wireless robot car equipped with an AirDrop-P card-slave arm.

In order to measure the performance of the LWP protocol, we placed a man at the middle test computer between the master arm PC and the robot car equipped with the AirDrop card to capture the wireless traffic exchanged between both of the nodes. The tests are performed using both the ad hoc and infrastructure modes of operation. The test computer is equipped with a regular wireless NIC in monitor mode to ‘listen’ to the packets in the air and uses the Wireshark program, which is a well-known and widely used network packet analyzer tool. The implementation details of the robot car system and performance measurements are found in [30].

## 6. Theoretical analysis of the mean and variance of the packet delay

In this section, we consider an M/M/1 queue model to analyze the mean and variance of the packet delay in an ad hoc wireless network. The queue considered in this analysis represents the packet queue of a single node, denoted by  $A_0$ , and in our experimental setup, it corresponds to a single AirDrop-P card. Let  $\lambda$  represent the arrival rate, which is directly related to the packet generation rate at  $A_0$  and will be considered as fixed throughout this analysis. Let  $\mu$  represent the service rate, which of course depends on the packet length, protocol details, and external traffic. Next, the expected value of the sojourn time (mean packet delay) will be as follows [30].

$$\mathbf{E}(S) = \frac{1/\mu}{1 - \lambda/\mu} \quad (2)$$

Consider the node  $A_0$ . If the external traffic level is kept as fixed and the packet generation rate at  $A_0$  is constant, but the packet length and protocol details are considered as the design parameters, it is easy to see that a complicated protocol and/or large packet size requires more processing time and hence decreases the service rate  $\mu$ . Furthermore, if the external traffic level is kept as fixed, but the node  $A_0$  decides to use longer packets, both the packets at this node and the packets at all of the other nodes will experience a stronger competition for the use of the channel. Finally, a longer packet requires more time to transmit. Although the exact relationship among the protocol details, packet size, and service rate  $\mu$  is hard to derive analytically, a simple protocol and short packet size means a high service rate, whereas a complicated protocol with long packets means a low service rate. Based on this argument, we can conclude that a simpler protocol with short packets will result in a smaller mean packet delay.

In an M/M/1 queue, the distribution of the sojourn time is given as:

$$P(S > t) = e^{-\mu(1-\rho)t}, \quad (3)$$

where  $\rho = \lambda/\mu$ , and the associated probability distribution function is given as:

$$p(t) = \mu(1 - \rho)e^{-\mu(1-\rho)t}. \tag{4}$$

Therefore, the standard deviation of the sojourn time (standard deviation of the packet delay),  $\sigma_S$ , can be computed as follows:

$$\mathbf{E}(S^2) = \int_0^\infty t^2 \mu(1 - \rho)e^{-\mu(1-\rho)t} dt = 2 \frac{1/\mu^2}{(1 - \rho)^2}, \tag{5}$$

and using  $\sigma_S^2 = \mathbf{E}(S^2) - \mathbf{E}(S)^2$ , we get:

$$\sigma_S^2 = \left( \frac{1/\mu}{1 - \lambda/\mu} \right)^2. \tag{6}$$

Therefore, when the arrival rate  $\lambda$  is kept constant, or in other words the packet generation rate is kept as fixed, and the service rate is increased using a less complicated protocol with shorter packets, the service rate  $\mu$  will increase; hence, both the mean packet delay,  $\mathbf{E}(S)$ , and the standard deviation of packet delay,  $\sigma_S$ , will decrease, as well. This means smaller packet delays and less variation in the packet delays. This reduced delay level and improved predictability will result in a simpler control problem for the telerobotic operation and hence is expected to improve the interactive performance of the overall system.

Our experimental measurements are parallel with the above theoretical predictions. Although some of the above assumptions may not be applicable to our telerobotic system, we still observed that the mean packet delay, the standard deviation of the packet delay, and the standard deviation-to-mean ratio increase with the packet size, as summarized in Figures 7–9 and Table 2.

**Table 2.** Experimental measurements.

Packet data size (bytes)	Standard deviation of the delay/mean delay
10	7%
40	8%
100	11%

### 7. Software-defined radios for the wireless protocol development

The LWP is built on top of the 802.11 MAC layer. The advantage of this approach will be the usability of the widely available 802.11 modules as hardware units, i.e. there is no need for special hardware; therefore, the existing 802.11 modules are all that we need in terms of the hardware to be able to use the LWP.

One may naturally ask whether it will be possible to do further optimization in the packet delay times if the MAC layer as well as the PHY are reengineered for the specific needs of the telerobotic systems. This is indeed true, and software-defined radios (SDRs) can be used for both research and prototype development. They basically allow modification in the PHY as well as in the MAC layer, but they are much less user-friendly, at least at the time of the writing of this paper. If the implementation of different ideas can be made easier and more practical, SDRs will definitely open new frontiers in wireless communication systems.



Figure 12. The SDR called the USRP.

We evaluated 2 SDRs, more specifically USRPs, as shown in Figure 12. Currently, C++, C#, and MATLAB-based software platforms are still in their infancy. There is another software platform that is Python-based, and it has a graphical user interface as well. This is called the GNU Radio Companion, and it comes with the standard USRP distribution. Programming or system development is done by connecting signal processing blocks as shown in Figure 13, and it is also possible to build custom blocks and use them together with system-defined ones. This is a programming or system development environment to which very few engineers are accustomed, and development does not seem to be as easy as writing sequential code, as in the AirDrop-LWP case.

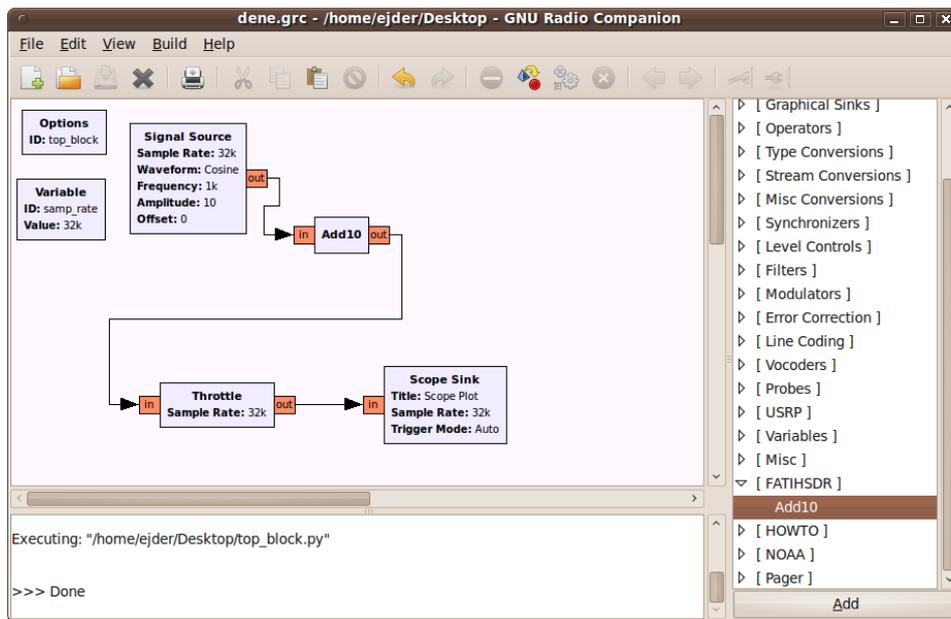


Figure 13. Programming via custom built blocks. Add10 is a custom-built block.

In summary, if proper software development tools do emerge for SDRs and the implementation of different ideas can be made easier and practical, they can be used to develop more efficient wireless communication protocols optimized for a given specific task.

## 8. Conclusion

In this paper, we have presented the results of the research project “Design, implementation and performance analysis of high performance telerobotic systems using real time communication protocols on wireless networks”, which was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK). We have first reviewed the recent progress in wireless communication alternatives for telerobotic systems and stressed the fact that custom-designed hardware and/or protocols are more likely to perform better and result in smaller communication delays. The approach proposed in this paper is a mixed one, where we fixed the hardware to widely available 802.11 modules and developed custom wireless protocols on top of the 802.11 MAC layer. We developed a new wireless protocol called the LWP and demonstrated its performance advantages, not by pure computer simulations but rather via actual tests. We were able to get up to a 50% reduced delay, and an additional 20% lower delay via compression. Our experimental results showed both a reduced mean packet delay and suggested a smaller standard deviation-to-mean ratio for the LWP. Therefore, we conclude that the interactive operation of the telerobotic system will be improved. Finally, the developed protocol was implemented on an embedded system without an operating system and was used to construct a telerobotic system using an industrial robot, the Mitsubishi RV-2AJ.

## Acknowledgments

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBİTAK) under the research grant EEEAG-108E091. Support from Fatih University is also acknowledged.

## References

- [1] F. Eady, *Implementing 802.11 with Microcontrollers: Wireless Networking for Embedded Systems Designers*, Burlington, MA, USA, Newnes, 2005.
- [2] F. Eady, *Networking and Internetworking with Microcontrollers*, Burlington, MA, USA, Newnes, 2004.
- [3] O. Toker, “Design, implementation and performance analysis of high performance telerobotic systems using real time communication protocols on wireless networks”, Final Report of Project EEEAG-108E091, Turkish National Science and Technology Foundation, 2010.
- [4] J. Elsner, M. Braun, S. Nagel, K. Nagaraj, F.K. Jondral, “Wireless networks in-the-loop: software radio as the enabler”, *Software Defined Radio Forum Technical Conference*, 2009.
- [5] Y. Liu, M. Mazurkiewicz, M. Kwitek, “A study towards reliability- and delay-critical wireless communication for RoboCup robotic soccer application”, *International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 633–636, 2007.
- [6] L. Hohl, R. Tellez, O. Michel, A.J. Ijspeert, “Aibo and Webots: simulation, wireless remote control and controller transfer”, *Robotics and Autonomous Systems*, Vol. 54, pp. 472–485, 2006.
- [7] Z. Fang, G. Tong, Y. Hu, X. Xu, “An embedded platform for intelligent mobile robot”, *Proceedings of the 6th World Congress on Intelligent Control and Automation*, Vol. 2, pp. 9104–9108, 2006.
- [8] F. Zeiger, N. Kraemer, M. Sauer, K. Schilling, “Challenges in realizing ad-hoc networks based on wireless LAN with mobile robots”, *Proceedings of the 6th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks and Workshops*, pp. 632–639, 2008.
- [9] F. Zeiger, F.N. Kraemer, K. Schilling, “Commanding mobile robots via wireless ad-hoc networks — a comparison of four ad-hoc routing protocol implementations”, *IEEE International Conference on Robotics and Automation*, pp. 590–595, 2008.

- [10] F. Babet, L. Jossierand, G. Poisson, Y. Toure, “Closed loop control for a tele-echography system with communication delays”, Proceedings of the 14th IEEE Mediterranean Electrotechnical Conference, pp. 356–361, 2008.
- [11] H. Rissanen, J. Mahonen, K. Haataja, M. Johansson, J. Mielikainen, P. Toivanen, “Designing and implementing an intelligent Bluetooth-enabled robot car”, IFIP International Conference on Wireless and Optical Communications Networks, pp. 1–6, 2009.
- [12] G. Mester, “Wireless sensor-based control of mobile robots motion”, Proceedings of the 7th International Symposium on Intelligent Systems and Informatics, pp. 81–84, 2009.
- [13] K.D. Young, Y.Q. Ou, L.H. Cai, J. Ho, K. Cheng, “Real time embedded control system development for wireless mobile platforms”, IEEE International Symposium on Industrial Electronics, pp. 2022–2027, 2008.
- [14] Y.Y. Li, W.H. Fan, Y.H. Liu, X.P. Cai, “Teleoperation of robots via the mobile communication networks”, 2005 IEEE International Conference on Robotics and Biomimetics, pp. 670–675, 2005.
- [15] M. Al-Mouhamed, O. Toker, A. Iqbal, “Design of a multi-threaded distributed telerobotic framework”, Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems, Vol. 3, pp. 1280–1283, 2003.
- [16] M. Al-Mouhamed, O. Toker, A. Iqbal, “Performance evaluation of a distributed telerobotic framework”, Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems, Vol. 3, pp. 1284–1287, 2003.
- [17] M. Al-Mouhamed, O. Toker, A. Al-Harthy, “A 3D vision-based man-machine interface for hand-controlled telerobot”, IFAC International Conference on Intelligent Control Systems and Signal Processing, pp. 586–591, 2003.
- [18] M. Al-Mouhamed, O. Toker, A. Iqbal, M. Nazeeruddin, “A distributed framework for relaying stereo vision for telerobotics”, ACS/IEEE International Conference on Pervasive Services, pp. 221–225, 2004.
- [19] M. Al-Mouhamed, O. Toker, A. Al-Harthy, “A 3D vision-based man-machine interface for hand-controlled telerobot”, IEEE Transactions on Industrial Electronics, Vol. 52, pp. 1–15, 2005.
- [20] M. Al-Mouhamed, O. Toker, A. Iqbal, M.S. Islam, “Evaluation of real-time delays for networked telerobotics”, Proceedings of the 3rd IEEE International Conference on Industrial Informatics, pp. 351–356, 2005.
- [21] M. Al-Mouhamed, O. Toker, A. Iqbal, “A multi-threaded distributed telerobotic framework”, IEEE Transactions on Mechatronics, Vol. 11, pp. 558–566, 2006.
- [22] C. Ceken, “An energy efficient and delay sensitive centralized MAC protocol for wireless sensor networks”, Computer Standards & Interfaces, Vol. 30, pp. 20–31, 2008.
- [23] V. Srovnal, Z. Machacek, V. Srovnal, “Wireless communication for mobile robotics and industrial embedded devices”, Proceedings of the 8th International Conference on Networks, pp. 253–258, 2009.
- [24] W. Xiaotao, Z. Jun, S. Zengqi, Z. Zhenmin, “Design and implementation of a telerobotic system with large time delay”, Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space, pp. 321–324, 1997.
- [25] Y. Zheng, W. Zheng, N. Xi, Y.C. Wang, “Development of an internet soccer tele-operation system based on pocket PC”, IEEE International Conference on Automation and Logistics, pp. 3035–3038, 2007.
- [26] E.A. Thompson, E. Harmison, R. Carper, R. Martin, J. Isaacs, “Robot teleoperation featuring commercially available wireless network cards”, Journal of Network and Computer Applications, Vol. 29, pp. 11–24, 2006.
- [27] M.J.H. Lum, J. Rosen, H. King, D.C.W. Friedman, T.S. Lendvay, A.S. Wright, M.N. Sinanan, B. Hannaford, “Teleoperation in surgical robotics—network latency effects on surgical performance”, Proceedings of the 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 125–130, 2009.
- [28] B.T. Yılmaz, H. Gümüşkaya, O. Toker, “A lightweight wireless protocol based on IEEE 802.11 for a robot car”, 2nd International Symposium on Computing in Science and Engineering, 2011.
- [29] B.T. Yılmaz, “Development of a network protocol for electronic control over wireless IEEE 802.11 network connection”, MSc, Haliç University, Computer Engineering Department-Management Information Systems Program, 2011 (in Turkish).
- [30] L. Kleinrock, Queuing Systems, Vol. 1: Theory, New York, Wiley-Interscience, 1975.