Networked Robots Architecture for Multiple Simultaneous Bidirectional Real Time Connections

Sasan Ardalan and Donald Elkins

IONOTRONICS® Corporation
Las Vegas, Nevada, 89102

July 21, 2023

Abstract

The Architecture for Networked Robots presented in this paper, is designed so that entities at the Enterprise level, such as a Java Application can access multiple Robots with real-time, two-way, on-demand reading of sensors and the control over Robot motion (actuators). If an application can simultaneously have access to the sensors of multiple Robots, then sophisticated algorithms can be developed to coordinate the movement of multiple robots. The simultaneous combined full knowledge of all aspects of the Robot's sensors and motion control, open up the capability to make multiple Robots act in a coordinated and purposeful way. In addition, the Networked Robots Architecture allows for multiple Enterprise Entities to have simultaneous access to the same Robot. A significant aspect of this architecture is that multiple independent entities can simultaneously access the Robot through a real time connection. For example, while a Java Application is monitoring and controlling a Robot, another entity such as an HTML5 WebSocket Client can also control and monitor the same Robot through a Web Browser. A multi-threaded WebSocket Server with routing is combined with a separate multi-threaded TCP/IP Server called the Frontline Server. The Robots connect through the Frontline Server which creates a thread per connection and

connects to the WebSocket Server. The WebSocket server accepts connections through Enterprise Applications(e.g. Java based) and Remote Web Based Applications. Each Robot has a unique identification (48 bit represented in hexadecimal) and a truncated WebSocket Session ID that is maintained throughout the connections, including in the Robot's Firmware. Both WiFi LAN and 4G LTE WAN are supported with Robots in both networks accessible through the Internet.

1. Background on Networked Robots

In [1], what exactly is Networked Robotics is explored. In particular, the problem statement for Networked Robots is "given a set of robotics resources and a task which can be broken out into subtasks, how does one compose the resources to create robotic agents that will perform the task?" In [2],"Networked Robots refers to multiple robots operating together in coordination or cooperatively with sensors, embedded computers, and human users". Furthremore [2] states that for Networked Robots "Communication between entities is fundamental to both cooperation and coordination and hence the central role of the network." In this paper, the Network that makes it possible for the robotics resources to create the robotic agents to perform the task is presented. The application (written for example in Java) can access any robot through the network and gain access to its sensor data as well as actuators (motor control) and do this for multiple robots while centralizing all the sensor data and motion control under the control of a single application. Furthermore, the Network allows for multiple applications to simultaneously access the same robot.

In [3], the important role of middlware in Networked Robots is highlighted. In particular, a comprehensive list of applications for Networked Robots is presented. In [3], a survey of middleware is provided. but the survey was prior to the advent of Websockets [4]. According to [5] the WebSocket protocol was standardized by the IETF as RFC 6455 in 2011. The current Websocket API can be found in [4].

A key development to achieve real-time two-way low latency access to the robot through the Internet, has been the development of Websockets. The Apache HTTP Server has implemented Websockets since July 2013 [5]. A Java implementation of the Websocket API was developed by [7] which has been instrumental in the development of Websocket support in Java Applications including Websocket Servers.

The Java Websocket Server in this paper and the Java Applications use the Websocket API in [7].

A key aspect of the Networked Robots Architecture is to allow development of applications using Web Browsers. This is possible since HTML5 [6] supports Websockets. Both Java and HTML5 Web Browser Applications can be developed. Both will be demonstrated in this paper.

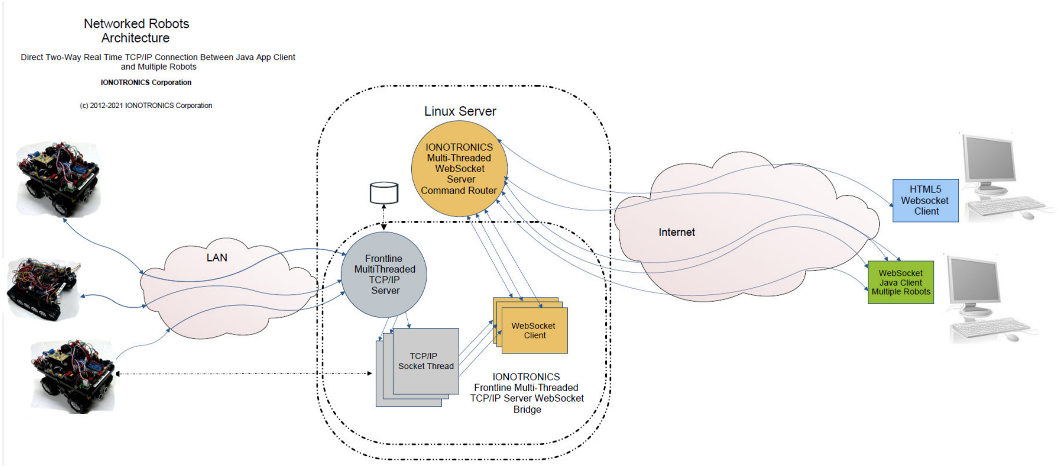

Networked Robots
Architecture
Direct Two-Way Real Time TCP/IP Connection Between Java App Client and Multiple Robots
IONOTRONICS Corporation
(c) 2012-2021 IONOTRONICS Corporation
Linux Server
IONOTRONICS Multi-Threaded WebSocket Server Command Router
LAN
Frontline MultiThreaded TCP/IP Server
TCP/IP Socket Thread
WebSocket Client
IONOTRONICS Frontline Multi-Threaded TCP/IP Server WebSocket Bridge
Internet
HTML5 Websocket Client
WebSocket Java Client Multiple Robots

Figure 1: Networked Robots Architecture

2. Theory of Operation

In the Architecture Diagram in Figure 1, the ”Frontline” Server written in Java is responsible for connecting to the Robot. Each Robot is equipped with a WiFi module. The WiFi Module has a multi threaded server listening to a port. When a connection is made from the Frontline server to the WiFi Module, a TCP/IP socket connection is established. Any data sent by the Frontline Server will appear at the UART (Rx) of the WiFi Module. Likewise any data sent by the Robot to the WiFi’s UART (Tx) will be received by the Frontline Server.

2.1 Frontline Server

The Frontline Server is multi-threaded and, once a TCP/IP connection is established with a Robot (through the WiFi Module), a new thread is created to handle the connection. The thread also establishes a connection to the Websocket Router Server.

The key to the operation of the Networked Robots is the ”unique_id” associated with each Robot. Each Robot has an entry in a Table in the Robotics Database (in this case using MySQL). The Table has multiple columns including three key columns (fields): The unique_id, the IP address and Port of the WiFi module. Note that each WiFi Module is configured with an IP address and Port.

Once a connection to a Robot is established by the Frontline Server, the Server sends heart beats to the Robot on a periodic basis. In this case it sends ”xx” to the Robot. The Heart Beat is very important in that it helps maintain the connection and also is used by the Robot to determine if the link is alive. It is also used by the Robot to do a Hard Stop if the link is lost as described in the sections below.

2.2 Websocket Router Server

When the Frontline Server is started, it makes a connection to the Database Table (Robots) and for each row, the unique_id is obtained and the associated WiFi Module IP/Port is also obtained. The Frontline Server starts a thread and makes a connection to the WiFi Module of the Robot. The unique_id associated with the Robot is provided to the thread. The thread then makes a connection to the Websocket Routing Server and provides the Websocket Routing Server with the unique_id. In this fashion, any enterprise or remote entity can provide the unique_id of the the Robot upon establishing a connection to the Websocket Router Server. The Server then routes any data form the entity to the Robot through the thread established by the Frontline Server. Thus, a direct two way real time connection is established between the entity and the Robot.

2.3 Robots

Robots can be mobile or fixed. Robots have a WiFi Module and a System on a Chip (SoC) and multiple sensors as well as motors for motion in the case of mobile devices or joint actuators in a Robotic Arm. For the purposes of this white paper, a Mobile Rover Robot will be considered, the IONOTRONICS® Microraptor®. The Microraptor® is shown in Figure 2. In this case the SoC is a Microchip® PIC32MX360F512 32 bit MIPS based processor. The Microraptor® WiFi Unit and Sensors are shown in Figure 3.

The Microraptor® is Omnidirectional (with four Mecanum Wheels) and thus has four independent PWM Motor Controllers (on an I2C Bus). It can move Forward/Reverse, Tank (CW/CCW) and move Laterally as well as Diagonally.

The Microraptor® has four Ultrasonic Sensors on each side (since it is Omnidirectional). This way it can detect obstacles in each direction. The Ultrasonic Sensors are made available to the I2C Bus with the 32 bit SoC

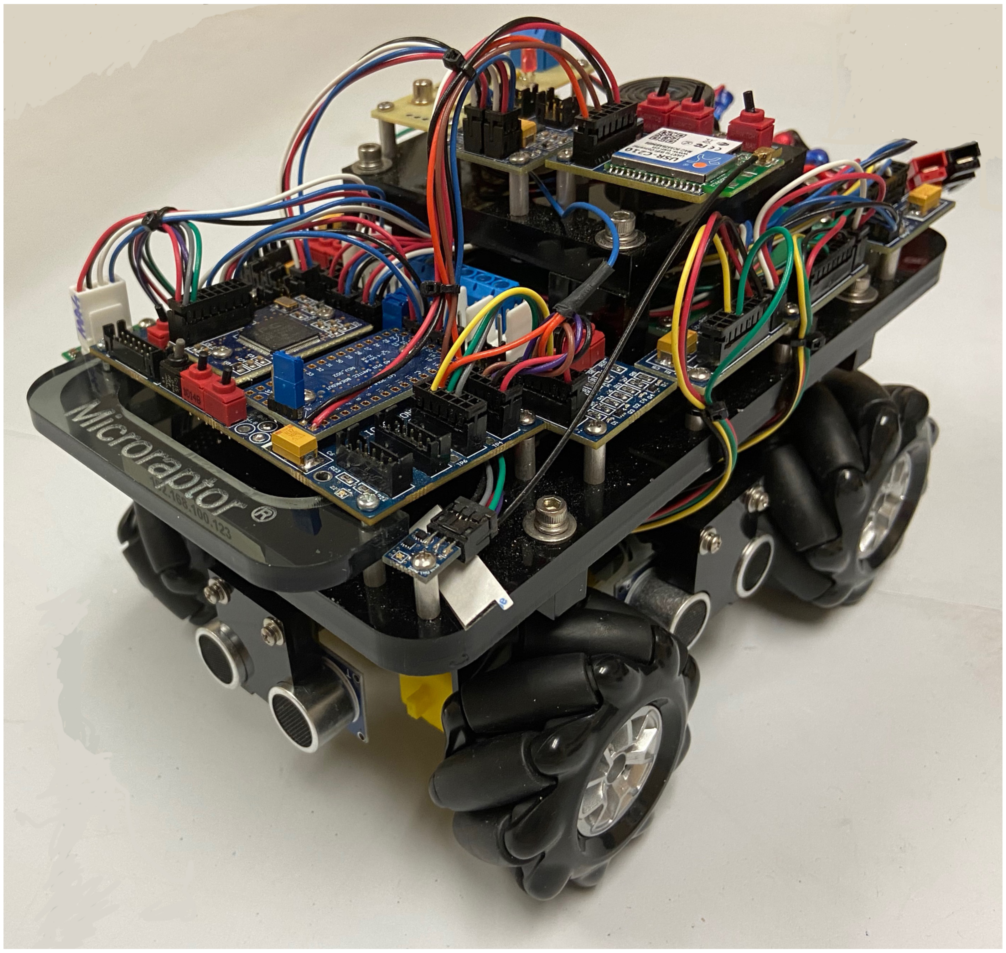


Figure 2: Microraptor® Rover

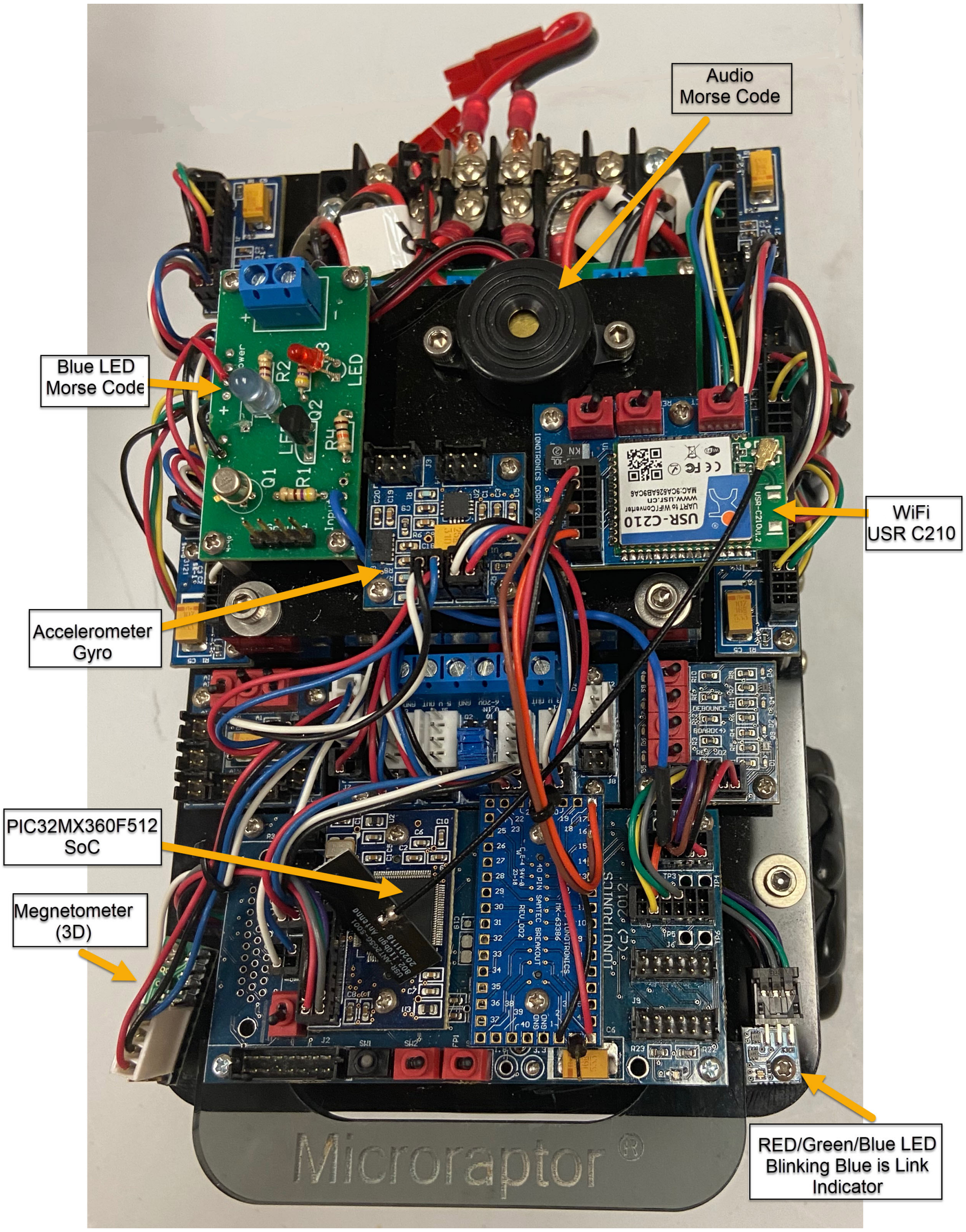


Figure 3: Microraptor® WiFi Module and Sensors

as I2C Master. An 8-bit Microchip® PIC16F1825 SoC is programmed as a slave that processes each Ultrasonic Sensor making it available as a I2C Slave n the I2C bus.

2.4 Remote Robot Command (R^2C) Firmware

Each Robot has an embedded C firmware program called R^2C. This program runs on a 32 bit System on a Chip (SoC) and accepts commands and provides responses, in particular sensor measurements. Commands are also supported which activate actuators. Commands are provided for reading a wide range of sensors. The actuators can be a mobile Robot's motors (in the case of the Omnidirectional Microraptor® four motors) or actuators on a Robotic Arm for each joint. The R^2C firmware also supports the interrupt driven logic to support link alive awareness and the capability to stop a mobile Robot if the link is lost. The R^2C Firmware also supports keeping track of the truncated Websocket Session ID (WSSID) for each command. It can, therefore, send the response back with the WSSID so that the message is routed to only the requester through the Networked Robots Architecture. This is important so that the Robot can handle simultaneous requests from separate Enterprise Entities (Java Applications and HTML5 Web Clients). In the case of 4G LTE based connections, the R^2C supports storing the Robot unique id in an EEPROM. The Frontline Server IP and Port are also stored in the EEPROM in the case of WAN connections. For local WiFi connections, the WiFi Modules IP and Port are stored in a database table at the Enterprise Level. Each Robot's unique id is paired with the IP and Port of the WiFi unit. The WiFi unit has a multithreaded server so that the Frontline Server can connect to the WiFi Unit and send commands and receive response via TCP/IP. The WiFi unit converts TCP/IP traffic to serial using it's UART. The R^2C Firmware through the SoC UART core interface, receives the traffic from the WiFi Unit. The following WiFi modules have been tested and are recommended: the USR WIFI232 B2 [8] and the USR C210 [9] both from PUSR IOT.

2.5 Link Blue LED Indicator and Hard Stop Red LED Indicator

When a link is established with the Rover, the Blue LED of the RGB LED will blink at a fixed time interval. The Firmware running on the Microraptor® is called R^{2}C. The R^{2}C supports a timer in the SoC that causes an interrupt on a specified time interval. The Rover receives a code (”xx”) from the Frontline Server on an interval basis. If this code is received, a global flag is set. In the Interrupt Service Routine of the Timer, if this global flag is set, then the RGB LED Blue is toggled. If it is not set, then the Rover is Hard Stopped. This indicates that the Rover is out of range or the link has been disrupted. So a blinking Blue LED shows an established and maintained link. A red LED indicates loss of link. If the link is re-established the Blue LED will flash. No need to reboot the Robot. To see all this in action see the video in [12].

2.6 Remote Control of Robots Through a 4G LTE Wide Area Network

To support Robots with a Wide Area Networks (WAN) for 4G LTE or 5G networks, a Frontline Multithreaded TCP/IP Server has been developed that accepts incoming connections. This is in contrast to the Frontline Server for WiFi based Robots which makes a TCP/IP connection to the Robots WiFi unit. In the case of a WAN, the Robot initiates a connection to the Frontline Server. The IP and Port of the Frontline server are stored in the Robot’s EEPROM. The IP and Port are transferred to the 4G LTE Modem using AT Commands. When the 4G LTE Modem is powered up, it will make a TCP/IP connection to the Frontline Server using the stored IP and Port number. The Front line server then issues a command to the Robot’s R^{2}C Firmware to obtain the Robot’s unique_id. Using the unique_id, the Frontline Server starts a thread that makes a connection to the Websocket Server announcing the unique_id. From this point on, Enterprise Entities can now connect via the WAN to the Robot.

Figure 4 shows a Tri-Track Tank Robot with a USR-G781 4G LTE Modem [10]. The 4G LTE Modem is connected to the Robot SoC via an RS-232 serial port. Commands to and from the Frontline Server (via the TCP/IP connection) are received and transmitted using the UART on the 4G LTE Modem.

2.7 Robotics Database Table for Robots

The list of Robots in the Robotics Database in the Robots Table (using mySQL Workbench) is shown in Figure 5. The Unique ID for each Robot

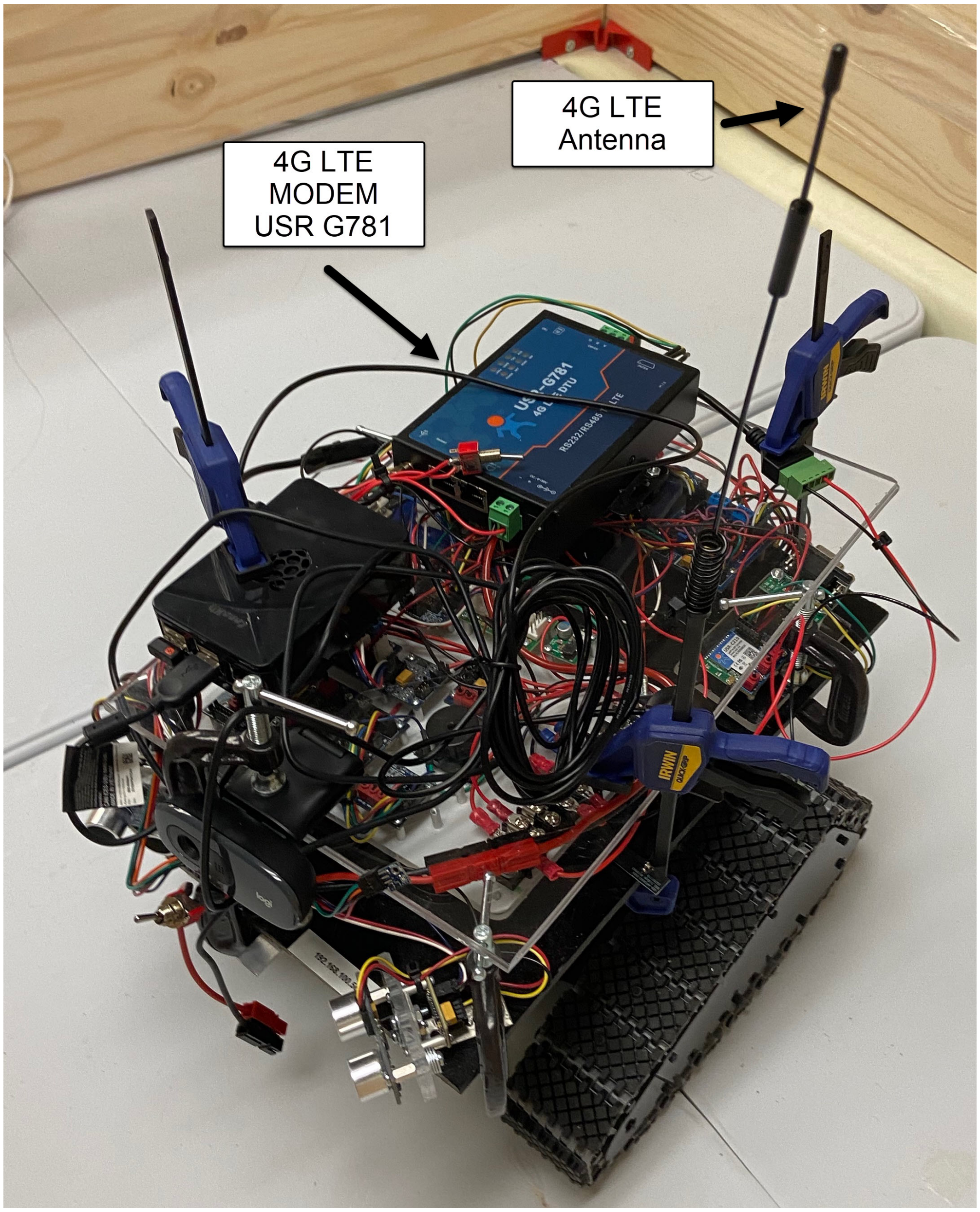
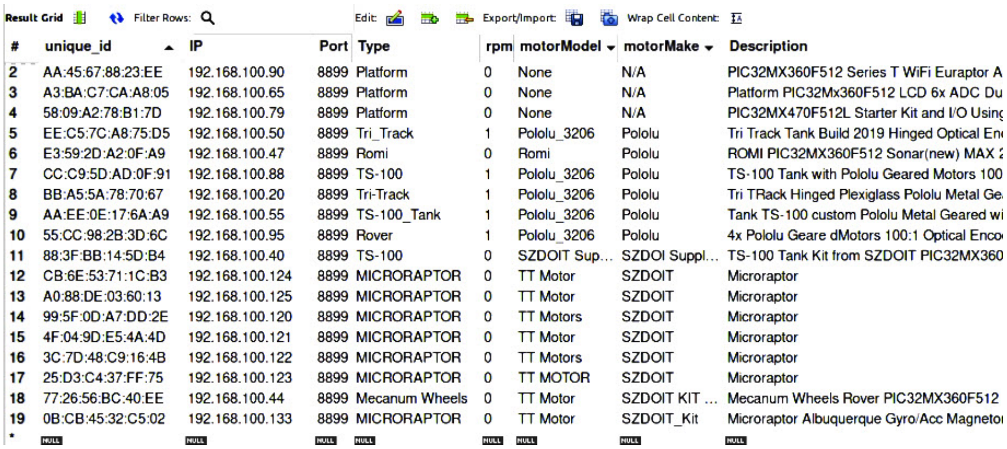


Figure 4: Tri-Track Rover Robot with 4G LTE Modem and Antenna

is the Hexadecimal representation of a 48 bit code with ":" between 8 bit values. As can been seen in the table, associated with each Unique ID is the IP address and Port of the WiFi Unit. The Robot table has a field for the type of Robot as well as type of Motor and a description along with other fields.



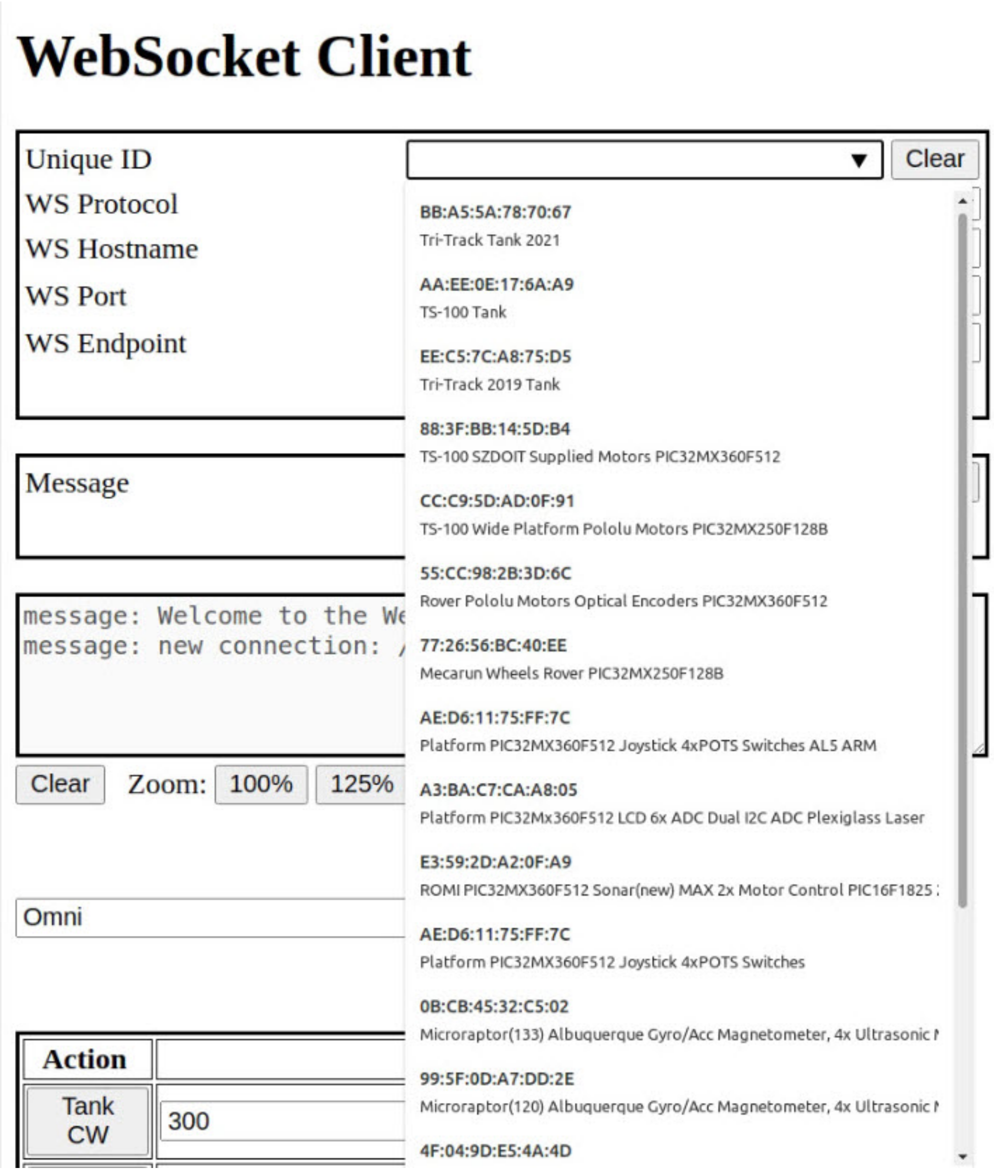
#	unique_id	IP	Port	Type	rpm	motorModel	motorMake	Description
2	AA:45:67:88:23:EE	192.168.100.90	8899	Platform	0	None	N/A	PIC32MX360F512 Series T WiFi Euraptor A
3	A3:BA:C7:CA:A8:05	192.168.100.65	8899	Platform	0	None	N/A	Platform PIC32Mx360F512 LCD 6x ADC Du
4	58:09:A2:78:B1:7D	192.168.100.79	8899	Platform	0	None	N/A	PIC32MX470F512L Starter Kit and I/O Using
5	EE:C5:7C:A8:75:D5	192.168.100.50	8899	Tri_Track	1	Pololu_3206	Pololu	Tri Track Tank Build 2019 Hinged Optical En
6	E3:59:2D:A2:0F:A9	192.168.100.47	8899	Romi	0	Romi	Pololu	ROMI PIC32MX360F512 Sonar(new) MAX
7	CC:C9:5D:AD:0F:91	192.168.100.88	8899	TS-100	1	Pololu_3206	Pololu	TS-100 Tank with Pololu Geared Motors 100
8	BB:A5:5A:78:70:67	192.168.100.20	8899	Tri-Track	1	Pololu_3206	Pololu	Tri TRack Hinged Plexiglass Pololu Metal Ge
9	AA:EE:0E:17:6A:A9	192.168.100.55	8899	TS-100_Tank	1	Pololu_3206	Pololu	Tank TS-100 custom Pololu Metal Geared wi
10	55:CC:98:2B:3D:6C	192.168.100.95	8899	Rover	1	Pololu_3206	Pololu	4x Pololu Geare dMotors 100:1 Optical Enco
11	88:3F:BB:14:5D:B4	192.168.100.40	8899	TS-100	0	SZDOIT Sup...	SZDOI Suppl...	TS-100 Tank Kit from SZDOIT PIC32MX360
12	CB:6E:53:71:1C:B3	192.168.100.124	8899	MICRORAPTOR	0	TT Motor	SZDOIT	Microraptor
13	A0:88:DE:03:60:13	192.168.100.125	8899	MICRORAPTOR	0	TT Motor	SZDOIT	Microraptor
14	99:5F:0D:A7:DD:2E	192.168.100.120	8899	MICRORAPTOR	0	TT Motors	SZDOIT	Microraptor
15	4F:04:9D:E5:4A:4D	192.168.100.121	8899	MICRORAPTOR	0	TT Motor	SZDOIT	Microraptor
16	3C:7D:48:C9:16:4B	192.168.100.122	8899	MICRORAPTOR	0	TT Motors	SZDOIT	Microraptor
17	25:D3:C4:37:FF:75	192.168.100.123	8899	MICRORAPTOR	0	TT MOTOR	SZDOIT	Microraptor
18	77:26:56:BC:40:EE	192.168.100.44	8899	Mecanum Wheels	0	TT Motor	SZDOIT KIT ...	Mecanum Wheels Rover PIC32MX360F512
19	0B:CB:45:32:C5:02	192.168.100.133	8899	MICRORAPTOR	0	TT Motor	SZDOIT_Kit	Microraptor Albuquerque Gyro/Acc Magneto
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 5: Robots Table in the Robotics Database

3. Control of Robots Using an HTML5 Websocket Client on a Web Browser

Using an HTML5 Client with Javascript, a Websocket connection can be made to the Websocket Routing Server. In this case the unique_id of the Robot is provided to the Websocket Routing Server and a direct two-way real-time connection is established from a Web Browser (Chrome in this case or Brave) to the Robot. Figure 6 shows the Websocket client with a pulldown menu to select a Robot. Once a Robot is selected, its unique_id will populate the field. The IP and Port of the Websocket Server will also be populated. The "Connect" button is then clicked and a connection is made to the Websocket Server. At this point, commands can be sent to the Robot from the Web Browser. See Figure 7. As shown in Figure 8, the motion of the Robot can be controlled from the Web Browser Form.

A video [13] shows the control of multiple Robots using an HTML5 client through a Web Browser.



WebSocket Client

Unique ID
WS Protocol
WS Hostname
WS Port
WS Endpoint

Message

```
message: Welcome to the We
message: new connection: /
```

Omni

300

BB:A5:5A:78:70:67
Tri-Track Tank 2021

AA:EE:0E:17:6A:A9
TS-100 Tank

EE:C5:7C:A8:75:D5
Tri-Track 2019 Tank

88:3F:BB:14:5D:B4
TS-100 SZDOIT Supplied Motors PIC32MX360F512

CC:C9:5D:AD:0F:91
TS-100 Wide Platform Pololu Motors PIC32MX250F128B

55:CC:98:2B:3D:6C
Rover Pololu Motors Optical Encoders PIC32MX360F512

77:26:56:BC:40:EE
Mecarun Wheels Rover PIC32MX250F128B

AE:D6:11:75:FF:7C
Platform PIC32MX360F512 Joystick 4xPOTS Switches AL5 ARM

A3:BA:C7:CA:A8:05
Platform PIC32Mx360F512 LCD 6x ADC Dual I2C ADC Plexiglass Laser

E3:59:2D:A2:0F:A9
ROMI PIC32MX360F512 Sonar(new) MAX 2x Motor Control PIC16F1825

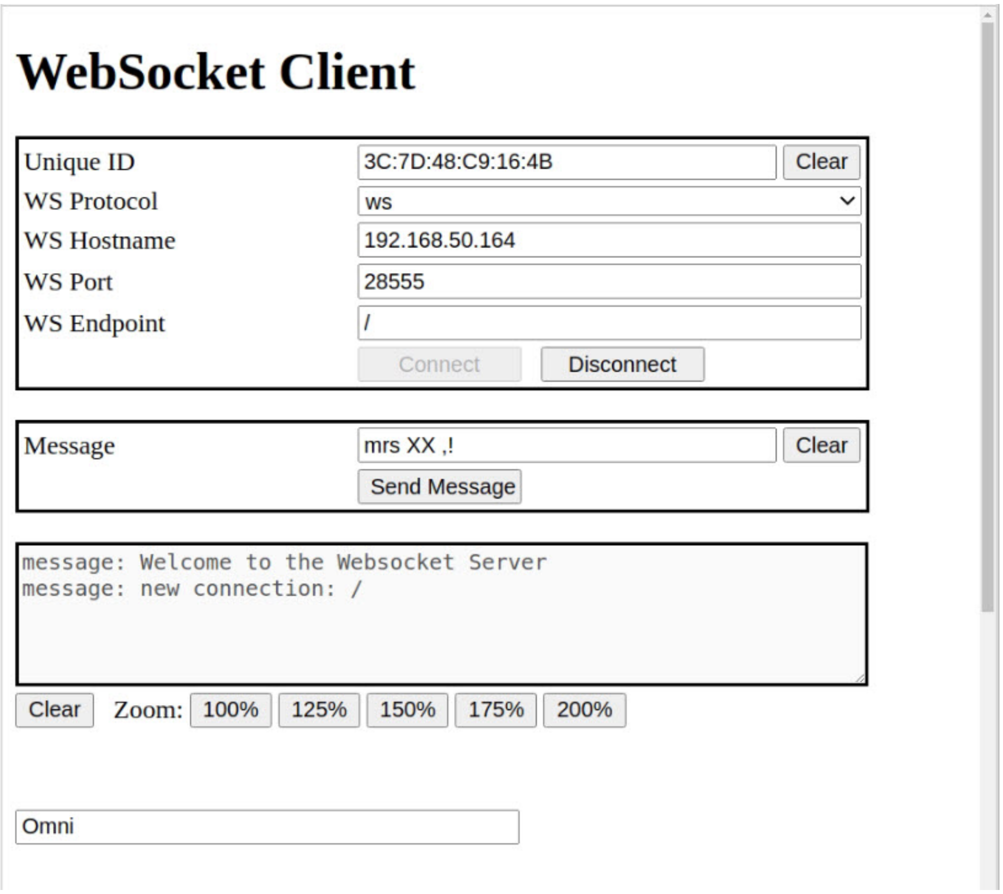
AE:D6:11:75:FF:7C
Platform PIC32MX360F512 Joystick 4xPOTS Switches

0B:CB:45:32:C5:02
Microraptor(133) Albuquerque Gyro/Acc Magnetometer, 4x Ultrasonic

99:5F:0D:A7:DD:2E
Microraptor(120) Albuquerque Gyro/Acc Magnetometer, 4x Ultrasonic

4F:04:9D:E5:4A:4D

Figure 6: Robot List Pulldown Menu in the HTML5 Client



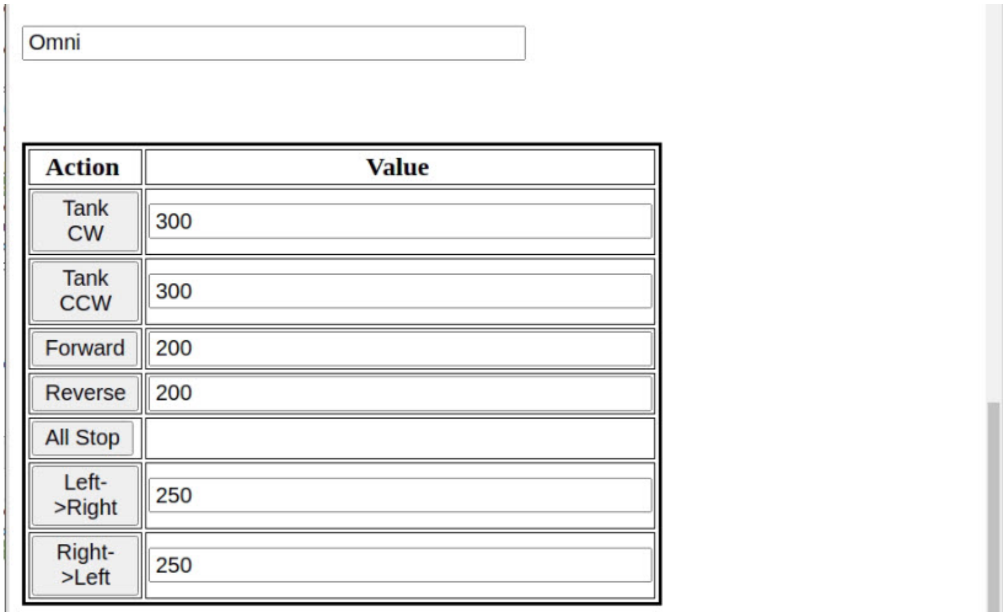
Unique ID: 3C:7D:48:C9:16:4B
WS Protocol: ws
WS Hostname: 192.168.50.164
WS Port: 28555
WS Endpoint: /

Message: mrs XX ,!

```
message: Welcome to the Websocket Server
message: new connection: /
```

Omni

Figure 7: HTML5 Websocket Client Showing Connection to a Robot (Unique ID)



Omni

Action	Value
Tank CW	300
Tank CCW	300
Forward	200
Reverse	200
All Stop	
Left->Right	250
Right->Left	250

Figure 8: HTML5 Websocket Client Zoom In on Motion Control

4. Java Application Control of Multiple Robots

In Figure 9, multiple Microraptor® Rovers are on the Networked Robots Platform. In the video [11] three Microraptor® Rover Robots are simultaneously controlled by a signal Java Application through the Websocket Router Server and Frontline Server. Figure 10 shows a photo from the video. In the video, each Microraptor® has a blinking blue LED showing that it is connected to the network. The three Microraptor® Robots are simultaneously controlled by the Java Application that makes three connections, one per Robot, using the unique_id via the Websocket Server/ Frontline Server. The rovers are moved forward, reverse, lateral to the left and lateral to the right using the Java code.

5. Java Application with Java Beans Displaying and Charting Sensor Data via the Networked Robots Architecture

The reading of sensor measurements from a Robotic Rover equipped with an accelerometer using the Networked Robots Architecture is demonstrated in

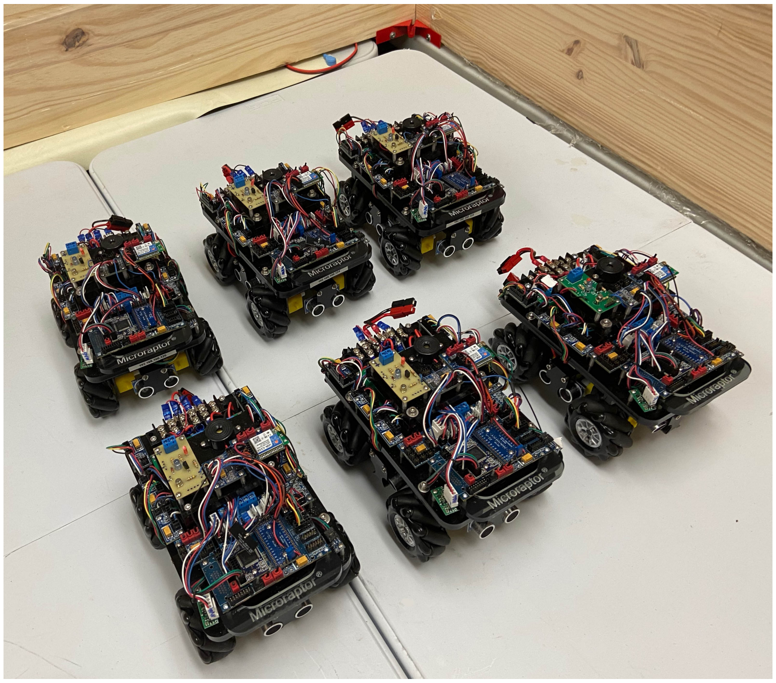


Figure 9: Networked Microraptor® Rovers

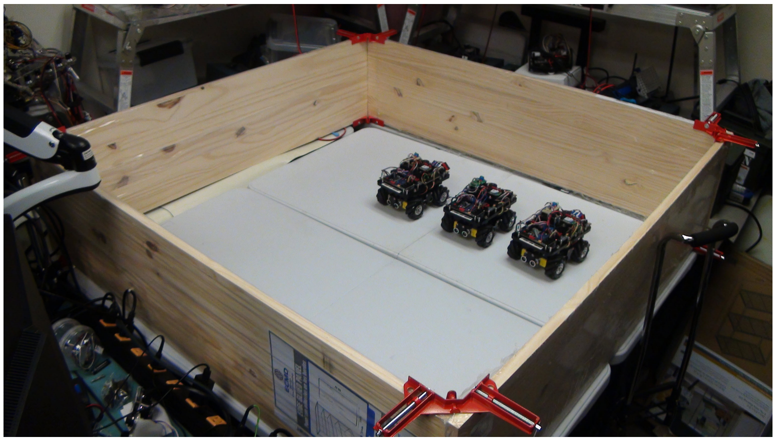
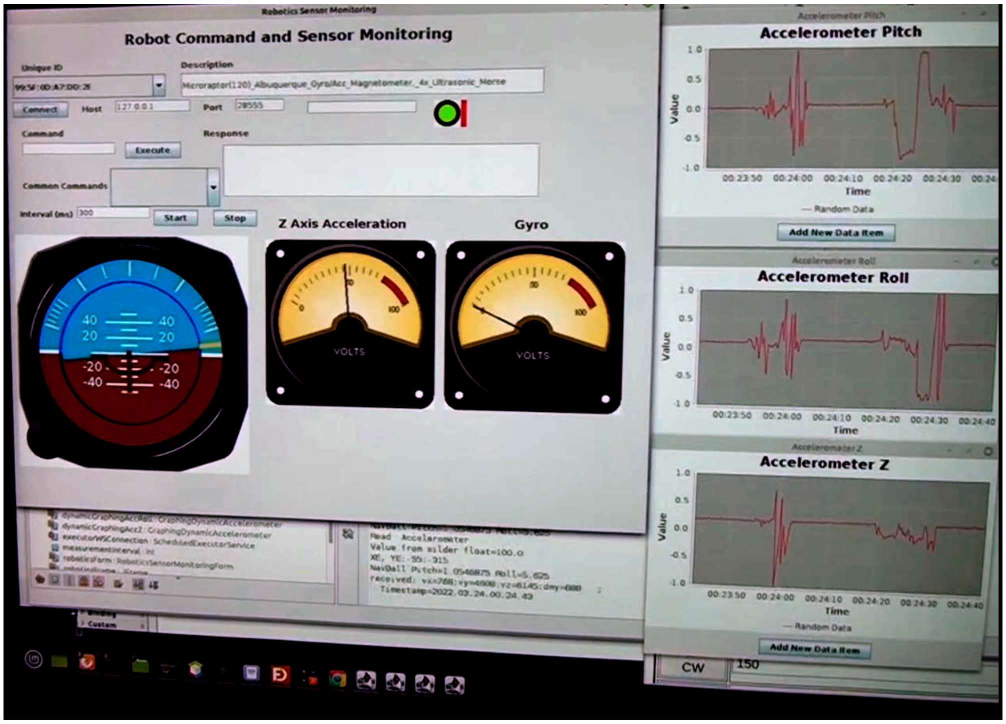


Figure 10: Java Application Control of Three Networked Microraptor® Rovers

the video in [14]. Figure 11 is a photo from the video. It shows Java Beans for a NavBall and Dials used to graphically show the results of the sensor measurements. These measurements are obtained by the Java Application code from the Robot using the connection to the Robot using the unique_id and the Websocket Router Server. The Robot is connected to the Frontline Server. Figure 11 also shows strip charts which show the history of the accelerometer measurements over time as the Robotic Rover is moved (tilted or rotated). Measurements are made every 300 ms.



Robotics Sensor Monitoring

Robot Command and Sensor Monitoring

Unique ID: 99:5F:0D:A7:DD:2E

Description: Microraptor(120)_Albuquerque_Gyro/Acc_Magnetometer_4x_Ultrasonic_Morse

Host: 127.0.0.1

Port: 28555

Interval (ms): 300

```
Read  Accelerometer
Value from silder float=100.0
XE, YE:-55:-315
NavBall Pitch=1.0546875 Roll=5.625
received: vx=768:vy=4608:vz=6145:dmy=888
  Timestamp=2022.03.24.00.24.43
```

Figure 11: Java Application Reading Sensors from Robot via Networked Robots Architecture

6. Conclusion

In this paper a Networked Robots Architecture is presented that allows for multiple entities (e.g. Java Applications, HTML5 Web Browsers) to access and control multiple Robots. The connection to each Robot is bidirectional and real time. The Robot can provide sensor measurement data on demand

and also respond to motion commands from the entities. A key feature of the Architecture is that separate entities can communicate with the same Robot. Each entity receives a response only from its request, although multiple entities communicate with the same Robot. Links to videos are provided that demonstrate the Networked Robots Architecture in practice.

References

[1] McKee, Gerard, *What is networked robotics?*, Informatics in Control Automation and Robotics: Selected Papers from the International Conference on Informatics in Control Automation and Robotics 2006 Springer, pages 35-45, 2008.

[2] Kumar, Vijay and Bekey, George and Sanderson *Networked robots*, WTEC Panel on International Assessment of Research and Development in Robotics Draft Report, pages 57-72, 2005.

[3] Mohamed, Nader and Al-Jaroodi, Jameela and Jawhar, Imad, *A review of middleware for networked robots*, International Journal of Computer Science and Network Security, Volume 9, Number 5, pages 139-148, 2009.

[4] websockets.spec.whatwg.org, *WebSockets Standard*, https://websockets.spec.whatwg.org/, 2023.

[5] Wikimedia, *WebSocket*, https://en.wikipedia.org/wiki/WebSocket, 2023.

[6] www.websocket.org, *HTML5 WebSocket – A Quantum Leap in Scalability for the Web*, https://www.websocket.org/, 2016.

[7] Nathan Rajlichg, *A barebones WebSocket client and server implementation written in 100% Java*, https://github.com/TooTallNate/Java-WebSocket, 2017, June 6

[8] Jinan USR IOT Technology Limited, *USR-WIFI232-B2 IEEE 802.11 b/g/n WiFi serial device server*, https://www.pusr.com/products/uart-wifi-modules-usr-wifi232-b2.html, 2023

[9] Jinan USR IOT Technology Limited, *USR-C210 IEEE 802.11 b/g/n WiFi serial device server*, https://www.pusr.com/products/low-cost-wifi-module.html, 2023

[10] Jinan USR IOT Technology Limited, *USR-G781 4G LTE Modem Serial and Ethernet Support*, https://www.pusr.com/products/cellular-modem-routers-usr-g781.html, 2023

[11] Ionotronics. (2023, June 11). *Java Application Simultaneous Control Three Networked Microraptors®* [Video]. Youtube. https://www.youtube.com/watch?v=KrMWaPQIwVE

[12] Ionotronics. (2023, May 9). *Networked Robots Stopped With Lost Link TS 100 Tank* [Video]. Youtube. https://www.youtube.com/watch?v=b_tt9sxkcNc

[13] Ionotronics. (2023, May 9). *Networked Robots HTML5 Client Multiple Robots* [Video]. Youtube. https://www.youtube.com/watch?v=35_9gkn1hYc

[14] Ionotronics. (2023, May 9). *Networked Robots Measuring Sensors (Accelerometer) Microraptors®* [Video]. Youtube. https://www.youtube.com/watch?v=RN3oU9fa17Q