Modbus RTU protocol and arduino IO package: A real time implementation of a 3 finger adaptive robot gripper

Amirul Syafiq Sadun, Jamaludin Jalani and Jumadi Abdul Sukor
Faculty of Engineering Technology, Universiti Tun Hussein Onn Malaysia (UTHM), Batu Pahat, Johor, Malaysia

Abstract. Recently, the Modbus RTU protocol has been widely accepted in the application of robotics, communications and industrial control systems due to its simplicity and reliability. With the help of the MATLAB Instrument Control Toolbox, a serial communication between Simulink and a 3 Finger Adaptive Robot Gripper can be realized to demonstrate a grasping functionality. The toolbox includes a “to instrument” and “query instrument” programming blocks that enable the users to create a serial communication with the targeted hardware/robot. Similarly, the Simulink Arduino IO package also offers a real-time feature that enabled it to act as a DAQ device. This paper establishes a real-time robot control by using Modbus RTU and Arduino IO Package for a 3 Finger Adaptive Robot Gripper. The robot communication and grasping performance were successfully implemented and demonstrated. In particular, three (3) different grasping mode via normal, wide and pinch were tested. Moreover, the robot gripper’s feedback data, such as encoder position, motor current and the grasping force were easily measured and acquired in real-time. This certainly essential for future grasping analysis of a 3 Finger Adaptive Robot Gripper.

1 Introduction

Nowadays, many robotic hardware has a built in function of the Modbus controller unit. An example of this function can be found in a 3 Finger Adaptive Robot Gripper developed by Robotiq. The Modbus protocol was developed in 1979 by Modicon where fundamentally the protocol is mostly applied in manufacturing/automation industries. Many recent industrial products such as PLC, I/O devices and instrument are equipped with the protocol due to its user friendly and easy to be used for hardware integrations [1]. The physical layer for the serial communications is normally, USB, RS-232, RS-422 and RS-485. The fundamental block diagram for Modbus RTU protocol is shown in Figure 1 [2].

Modbus RTU (Remote Terminal Unit) is a type of Modbus protocol that uses serial communication to create a Master - Slave or Server- Client communication between intelligent devices [3]. Based on Figure 1, the basic concept of the Modbus RTU protocol is "query" and "response" between master and slave in which this process continuously occurs throughout the operation of the robot. The master devices send a query to slave devices while the slave devices send a response to the query with a specified code. Table 1 generally summarized the communication structure which consists of Slave ID, Function Code, Data, and Error Code (CRC).

Table 1. General explanation of Modbus RTU communication structure.

<table>
<thead>
<tr>
<th>Structure</th>
<th>Purposes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slave ID</td>
<td>Used to assign the address of Slave devices</td>
</tr>
<tr>
<td>Function Code</td>
<td>The function code tells the slave device what kind of action to be executed at particular “query”.</td>
</tr>
<tr>
<td>Data</td>
<td>The data of the slave devices which are united into a network (i.e gripper status, &amp; finger position request).</td>
</tr>
<tr>
<td>Error Code (CRC)</td>
<td>Cyclic redundancy check is used to check the error of the data transmission over the physical layer.</td>
</tr>
</tbody>
</table>

Figure 1. Modbus RTU Fundamental.
setup for a robot hand with a built in Modbus RTU function:

![Figure 2. Connection for a Robot Gripper with Built in Modbus RTU Function.](image)

The 3 Finger Adaptive Robot Gripper as shown in Figure 2 supports the Modbus RTU and TCP/IP communication protocol. However, the robot comes with a default GUI that is only suitable for demonstrations. To further use the robot in the research environment (i.e. closed loop control system), a communication platform (i.e. by using MATLAB Simulink) is needed for future experimental and analysis. Moreover, to add the element of external sensory feedback to the robot system (i.e. force sensor), a DAQ device is compulsory. Referring to [4], the results for use of Arduino IO package in MATLAB Simulink showed a great potential whereas the Arduino hardware is capable of acting similarly to the DAQ device in real-time. This paper introduces a real-time robot control by using Modbus RTU platform with the integration of Arduino IO Package for a 3 Finger Adaptive Robot Gripper. The robot control is programmed mainly by using the MATLAB Instrument Control Toolbox. Additionally, the integration of robot control with the Arduino IO Package is possible by installing the supported library package to MATLAB Simulink. The study focused on developing a Simulink program to control and acquire the robot available feedback data in real-time such as finger position, actuator motor current and grasping force. It is important to establish the programming platform for the future study related to the 3 Finger Adaptive Robot Gripper (i.e grasping analysis or compliance control). The available input and output data of the robot gripper was recorded individually for each finger utilizing all the available robot modes (i.e. normal, wide and pinch). Furthermore, a graphical data analysis was done to observe the robot operation in the “open loop system”.

2 Preliminary setup

2.1 Hardware setup

![Figure 3. General Hardware Setup.](image)

Figure 3 shows the general hardware setup for the experiment. The Modbus RTU communication protocol was done by using MATLAB Simulink where the robot is connected to a computer (laptop) via USB cable. The setup also consists of Arduino UNO which acts as a DAQ device. The analog input from sensors are connected to Arduino ADC (Analog to Digital) pin and the data were then acquired by using Simulink Arduino IO Package. The robot control and data acquisition were run in the same Simulink program by using 2 different COM Ports.

2.2 Force Sensor Setup

Previous studies for FSR sensors showed that that the FSR sensors are sufficient and reliable to measure and record the data once the force is detected even though a minor drawback has been identified regarding the fluctuating readings (sensor accuracy) [5]. In general, the FSR sensor is measured in voltage (V) which can be converted to Force (N) by using the linear model equation as presented in [6]. The force range detection is between 1N to 100N. However, a proper method of sensor placement has to be carefully designed and positioned so that the contact surface is adequately measured [5]. The better force reading can be obtained by spreading the pressure over a greater area. For this experimental purposes, the researcher has developed a 3D printed plastic cover for the FSR sensor as shown in Figure 4.

![Figure 4. Connection Block Diagram.](image)

3 The control and communication scheme

3.1 Cyclic redundancy check (CRC)

In Modbus RTU communication, it is compulsory to perform a Cyclic Redundancy Check (CRC) to ensure the data integrity in term of errors and abnormalities. During the robot gripper’s communications, the data is sent and received as a string of 8 binary characters framed with a start bit, and a stop bit, making each byte 10 bits [7]. Thus, every byte in the data instructions/message is used to calculate the CRC either during the “request” or
“query” of the robot gripper [8]. Figure 5 shows the flow chart of the CRC.

![Flow Chart of CRC](image)

Figure 5. Cycling Redundancy Check.

Referring to Figure 5, the process is repeated for every new data transmission (send/receive) throughout the Modbus RTU protocol. If the results of the CRC are incorrect, communication errors occur between the robot gripper and MATLAB Simulink. Thus, the robot will stop responding to the requested instruction.

### 3.2 Programming design structure

![Programming Design Structure](image)

Figure 6. Simulink Programming Design Structure.

The programming design structure for the Modbus RTU communication protocol and the Simulink Arduino IO package can be divided into 4 stages. The first step is to send the instructions to set the grasping mode and the finger’s individual setting parameter. At this step, it is important to identify the correct parameter setting based on the robot gripper’s user manual. Step 2 is to receive the robot gripper response. This “handshake” stage works as an initial indication to identify any abnormalities and to indicate the previous instruction data is well received and correct. Step number 3 was carried out to request the robot gripper’s data in real-time (i.e. the encoder position changes). At this stage, the data of the robot gripper is being requested for every 1 millisecond. Next, the robot gripper completed the request in step 3 by returning the data of the requested parameters (in step 4). These 4 steps are repeated during the program runs. Simultaneously, the analog data of the force sensor can be acquired by using the Arduino IO Package. Figure 7 shows the Simulink programming blocks.

![Programming Simulink Blocks](image)

Figure 7. Programming Simulink Blocks.

### 4 Experimental results

#### 4.1 Robot grasping

This section demonstrates the robot gripper’s grasping by using the developed MATLAB Simulink program. The purpose of the test is to control the robot in “open loop system” while capturing the desired and actual finger position of the robot gripper. The test was implemented by using the 3 available modes of the robot gripper namely a normal, wide and pinch modes. Figure 8 (a), (b) and (c) shows the acquired data for desired versus the actual for all 3 modes.

![Finger Position Data](image)

Figure 8 (a). Finger Position Data for Normal Mode.
Based on Figure 9, the data for actuator motor current during grasping and ungrasping indicates a constant reading (no load/no object condition). For all 3 fingers, the motor current reading has an average of 4.5 mA during the motor movement, and once the desired position is reached, the current value drops to 0 mA (motor stops). The higher resistance during the grasping (i.e grasping a high stiffness object), the higher value of motor current can be expected.

4.2 Grasping with force sensing

This section demonstrates the grasping test with force sensing element by utilizing the Arduino IO package. The purpose of this test is to observe the capability of the program and the hardware setup to capture the data during the grasping operation (i.e finger position and force sensor value).

Based on Figure 9, the data for actuator motor current during grasping and ungrasping indicates a constant reading (no load/no object condition). For all 3 fingers, the motor current reading has an average of 4.5 mA during the motor movement, and once the desired position is reached, the current value drops to 0 mA (motor stops). The higher resistance during the grasping (i.e grasping a high stiffness object), the higher value of motor current can be expected.

This section demonstrates the grasping test with force sensing element by utilizing the Arduino IO package. The purpose of this test is to observe the capability of the program and the hardware setup to capture the data during the grasping operation (i.e finger position and force sensor value). The desired position was set from 65 to 80 (decimal) for all 3 fingers and grasping test was carried out with the robot grasping a low stiffness ball in wide mode (see Figure 10).

It is known that the robot gripper has the capability to grasp with the force ranging 15N to 60N. Thus, for this experiment, the force was set to maximum (60N) to observe data recording for finger position and force sensor. Figure 11 (a), (b) and (c) shows the data for the grasping test.

Based on Figure 9, the data for actuator motor current during grasping and ungrasping indicates a constant reading (no load/no object condition). For all 3 fingers,
Figure 11 clearly shows that the data for finger position and force sensor has successfully been recorded during the grasping. The data plot for force sensor indicates that the force gradually increases as the finger moving towards the desired position (grasping the object). The force values vary for each finger ranging from 40N to 58N. There are many factors contributing to the variation of the forces, the main factor is due to the underactuated mechanism of the robot gripper. Furthermore, the study also able to capture the data for actuator motor current versus force sensor as shown in Figure 12. The simple test is executed to see the fluctuation of motor current during the grasping contact with the plastic bottle.

Based on the results, the motor current for each finger increases as the resistance become higher (ranging from 3.8 mA to 5.8 mA) and the motor current drops to 0 mA when the motor stops. Additionally, the ungrasping operation indicates a low motor current values. Further discussion is needed on the topic of grasping analysis, however, it is not included in this study.

5 Conclusion

This paper established a real-time robot control by using Modbus RTU via MATLAB Instrument Control Toolbox and Arduino IO Package for a 3 Finger Adaptive Robot Gripper. The robot communication was successfully implemented and demonstrated for an individual finger using all the available robot grasping modes. Based on the results of this study, Modbus RTU protocol was proven to be a reliable and stable platform for the robot control that integrates with the Arduino IO package. Moreover, the hardware setup for the FSR force sensor was able to detect the increment of force value during the object grasping. With the accuracy of 1 millisecond, the acquired force data are reliable for the analysis of an active compliance control for the future study. On the other hand, the study also able to record the actuator motor current during the grasping process. Based on the results, the motor current has an average of 4.5 mA during “no load” condition and fluctuated up to 5.8 mA during the object grasping. Nevertheless, for future study, it is important to know the accurate angular proportional of the robot finger joint. An actual joint angular position should be used instead of the decimal input and output (0 to 255). Finally, the platform that has been created in this study can be used for the future study involving the 3 Finger Adaptive Robot Gripper especially for the application in the “closed loop system” environment.

Acknowledgement

This topic is part of a project funded by the Fundamental Research Grant Scheme (FRGS), Vot 1480. The authors also wish to thank the Faculty of Engineering Technology, Universiti Tun Hussein Onn Malaysia for providing a platform to carry out the research activities.

References
5. A. S. Sadun and J. Jalani, “Force Sensing Resistor ( FSR ): Preliminary Results for Active Compliance Control of the Multifingered Robot Hand Faculty of Engineering Technology, Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Batu,” vol. 6, no. 0, pp. 2–5.