Project Overview:

An interactive MATLAB GUI was created that allowed a user to interact with a virtual robotic arm by adjusting its joint angles to achieve a certain position and orientation for the arm's end effector, a gripping mechanism. As the user adjusted the joint angles, the main MATLAB program *robotArm.m* sent commands to an Arduino Mega ADK via serial communication. The Arduino Mega ADK then interpreted these commands and used them to drive the robotic arm's servos to achieve the same position and orientation of the arm's end effector specified by the user in the MATLAB GUI.

Hardware:

The hardware consisted of a Dagu robotic arm with six servos capable and five degrees-of-freedom (Figure 1) and an Arduino Mega ADK (Figure 2).

Figure 1. Dagu robotic arm.

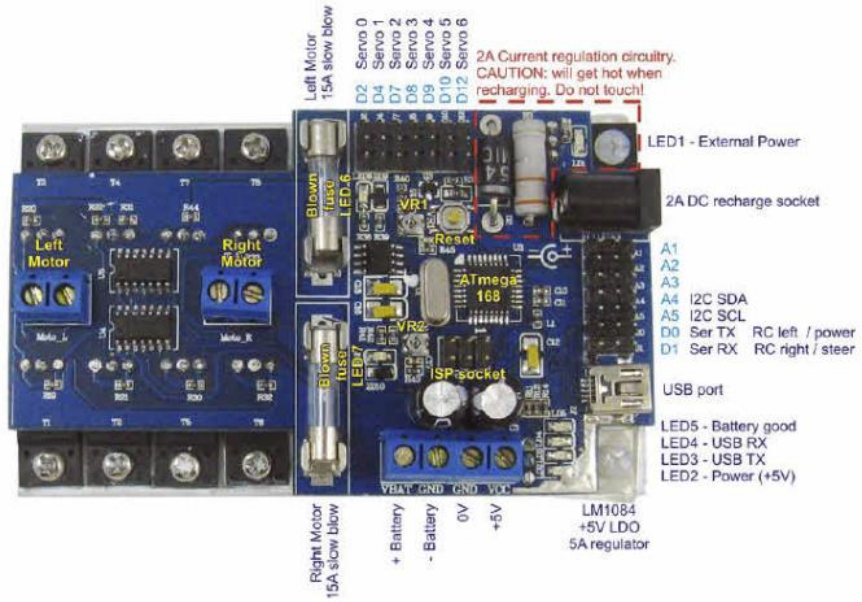

Figure 2. Arduino Mega ADK board for controlling the robotic arm.

MATLAB Files:

MATLAB GUIDE was used to create an interactive GUI for the robotic arm (Figure 3).

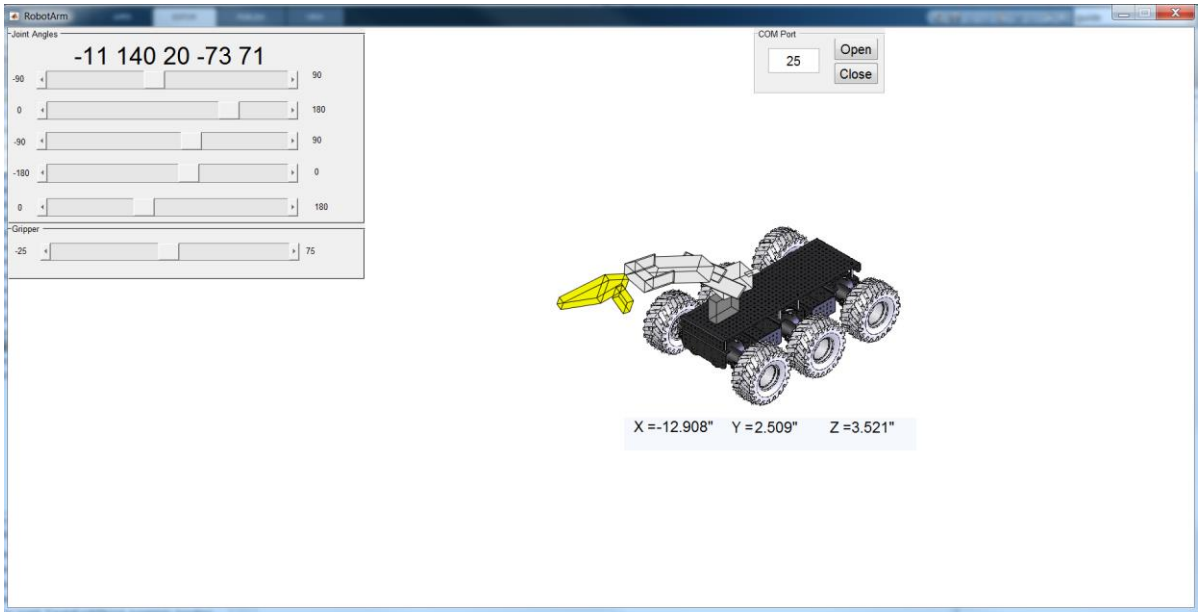

Figure 3. A MATLAB GUI that allows a user to adjust the joint angles of the robotic arm and also view a virtual representation of the resulting position and orientation of each link and the end effector.

The sliders in the top left of the GUI allowed the user to adjust the joint angles between the links and end effector of the robotic arm; the five joint angles were displayed directly above the sliders. The COM Port display at the top center of the GUI was used to open a serial connection to the Arduino Mega ADK for communication to the robotic arm. The *addImageToAxes.m* file was used to add an image of the Wild Thumper chassis to the middle of the GUI. The *madeLink0.m, madeLink1.m, madeLink2.m, madeLink3.m, madeLink4.m,* and *madeLink5.m* files were used to create the images of the links and end effector and place them into the GUI to display the robotic arm. The coordinates displayed directly below the robot are the x-, y-, and z-coordinates of the arm's end effector.

Forward kinematic analysis was used to determine the position and orientation of the end effector by using the joint angles. The Denavit-Hartenberg (DH) convention was used to accomplish this and also select the initial coordinate frames for each joint. The *makeHomogenousTransformation.m* file was used to create the DH homogeneous transformation matrices by passing along the DH parameters of the robotic arm to the *homogenousTransformations.m* file, which creates a DH homogeneous transformation matrix from the given DH parameters.

NOTE: If you have MATLAB and would like to play around with the GUI, open up the *RobotArm.m* file and run it. To make changes to the GUI via MATLAB GUIDE, simply type the following command into MATLAB's command window: guide('RobotArm.fig').

Arduino Libraries:

Several libraries were used with the Arduino board for communication and hardware control; some of these libraries came installed with the Arduino IDE, others were custom-built. Below is a list of the libraries, with descriptions of each:

1) RobotAsciiCom Library – Custom library used on the Arduino Mega ADK for serial communication between it and MATLAB. Communication is through messages that contain ASCII characters. The library parses messages received from MATLAB, interprets them, and then the main code *MatlabRobotArm.ino* performs the appropriate actions.
2) Arduino Servo Library – Built-in Arduino servo library that provides methods for moving a servo a certain number of degrees.

3) ArmServos Library – Custom library that converts servo angles into Denavit-Hartenberg representation angles. It is a subclass of the Arduino Servo library and inherits all of its methods.

4) ArmServosSpeedControlled Library – Custom library that is a subclass of both the ArmServos and Arduino Servo libraries; it inherits all of their methods and contains additional methods that allow for better control of the speed of the robot's servos. It uses an ease-in ease-out algorithm to move the joints slower at the beginning and end of a movement, resulting in less jerk over the course of the robot's movement.

5) LiquidCrystal Library – Built-in Arduino library that is used for controlling an LCD and displaying messages on it, in this case controls the LCD on the Arduino Mega ADK 2560.

The *MatlabRobotArm.ino* Arduino file and the aforementioned libraries were uploaded to the Arduino Mega ADK and allowed the user to control the robotic arm through the MATLAB GUI.