ASME Student Design Competition: Remote Inspection Device Final Report

Juan Sebastian Fajardo
Marybel Hernandez
Ryan Manalo

Advisor: Melissa Morris

March 2013

This B.S. thesis is written in partial fulfillment of the requirements in EML 4905. The contents represent the opinion of the authors and not the Department of Mechanical and Materials Engineering.
Ethics Statement and Signatures

The work submitted in this B.S. thesis is solely prepared by a team consisting of Juan Sebastian Fajardo, Marybel Hernandez, and Ryan Manalo and it is original. Excerpts from others’ work have been clearly identified, their work acknowledged within the text and listed in the list of references. All of the engineering drawings, computer programs, formulations, design work, prototype development and testing reported in this document are also original and prepared by the same team of students.

Juan Sebastian Fajardo, Team Leader
Marybel Hernandez, Programmer
Ryan Manalo, Designer

Melissa Morris, Graduate Student Advisor
# Table of Contents

Abstract............................................................................................................ viii
Introduction ........................................................................................................ 1
Problem Statement ............................................................................................. 1
Motivation ........................................................................................................... 1
Project Objectives .............................................................................................. 2
Literature Survey ................................................................................................ 3
Conceptual Design ............................................................................................. 6
  Proposed Design............................................................................................. 7
Final Design ....................................................................................................... 8
Timeline ............................................................................................................. 8
Analytical Analysis.............................................................................................. 9
  Robot Frame ................................................................................................... 9
  Robotic Arm .................................................................................................. 11
Major Components ........................................................................................... 20
  Drive Motor Selection.................................................................................... 21
  Manipulator Motor Selection ....................................................................... 21
Prototype System Description ........................................................................... 22
  Manufacturing of Components ..................................................................... 22
Programming .................................................................................................... 28
Prototype Testing ............................................................................................. 32
  Chassis ......................................................................................................... 32
  Drive ........................................................................................................... 32
  Robot arm .................................................................................................... 33
  Programming / Communication ................................................................. 35
Maintenance..................................................................................................... 35
List of Figures

Figure 1 Competition course ...................................................................................... 2
Figure 2 Packbot ........................................................................................................ 4
Figure 3 Warrior 710 .................................................................................................. 4
Figure 4 Talon ............................................................................................................ 5
Figure 5 Concept 1 ..................................................................................................... 6
Figure 6 Concept 2 ..................................................................................................... 6
Figure 7 Concept 3 ..................................................................................................... 6
Figure 8 Concept 4 ..................................................................................................... 7
Figure 9 First Proposed Design .................................................................................. 7
Figure 10 Second Proposed Design .......................................................................... 7
Figure 11 Final Design ............................................................................................... 8
Figure 12 Timeline...................................................................................................... 9
Figure 13 Force Body Diagram of Robot Arm Design .............................................. 11
Figure 14 Forward Kinematics ................................................................................. 13
Figure 15 Robot Arm Kinematics ............................................................................. 14
Figure 16 Inverse Kinematics ................................................................................... 15
Figure 17 Rotating Drum ......................................................................................... 16
Figure 18 Transformation Rotation ......................................................................... 18
Figure 20 Raw Chassis Material .............................................................................. 22
Figure 20 Milling Machine Stop ............................................................................... 22
Figure 21 Motor Mount Clamping Method ............................................................... 23
Figure 22 Vice Stop.................................................................................................. 23
Figure 23 Plate Set-Up and Hole Saw ...................................................................... 23
Figure 24 Bottom Plate............................................................................................. 24
Figure 25 Scroll Saw ............................................................................................... 24
Figure 26 Caster Spacer ................................................................. 24
Figure 27 Bench Grinder ............................................................... 24
Figure 28 Aluminum Bracket on Bench Vise ............................ 25
Figure 29 Robot Claw with Velcro Grippers ............................ 25
Figure 30 Pan and Tilt GoPro Case ............................................... 26
Figure 31 Modified Top and Bottom Base of the Chassis ......... 26
Figure 32 Third Hand Holding Soldered Servo Cable .............. 27
Figure 33 Battery Eliminating Circuit ......................................... 28
Figure 34 Switch Box ................................................................. 28
Figure 35 Program Flowchart ..................................................... 31
Figure 36 Keyboard Layout ......................................................... 32
Figure 37 Bolt Leg ................................................................. 32
Figure 38 Single Ball Caster ......................................................... 33
Figure 40 Shoulder Spring ......................................................... 34
Figure 40 Elbow Spring ............................................................ 34
Figure 41 Lipo Bag and Fire Extinguisher ................................. 36
Figure 42 Start of Competition ................................................ 37
Figure 43 Placing Sensor in Drop-Off Zone ............................ 37
Figure 44 Picking-Up Sensor ..................................................... 37
Figure 45 Gage Reading ......................................................... 38
Figure 46 Pressing the Button .................................................. 38
Figure 47 Robot Design Drawing ........................................... 44
Figure 48 Top Base Drawing .................................................... 44
Figure 49 Bottom Base Drawing ............................................. 45
Figure 50 Base Spacer Drawing .............................................. 45
Figure 51 DC Motor Bracket Drawing ................................. 46
List of Tables

Table 1 Robot Parameters ................................................................. 12
Table 2 Parameters for Inverse Kinematics ................................. 16
Table 3 Parameter for Transformations ......................................... 19
Table 4 Engineering Hours .............................................................. 39
Table 5 Summary of Cost ................................................................. 40
Table 6 Cost of Electronics .............................................................. 72
Table 7 Cost of Hardware ................................................................. 73
Table 8 Cost of Administrative Expenses ...................................... 73
Abstract

The purpose of this project is to design, manufacture, and build a remote inspection vehicle to compete at the annual 2013 American Society of Mechanical Engineering (ASME) Student Design Competition. The proof-of-concept robot described in this report is being specifically designed to compete in a simulated chemical/radioactive situation. The vehicle will read gages that determine the level of radioactivity at specific location, activate a cooling pump, and have the capability of carrying sensors to and from designated locations.

Our final design concept will be built mostly from scratch. The robot chassis will be built from High Density Polyethylene (HDPE) plastic. An Arduino Mega 2560 microcontroller will be used to carry out all commands received by the operator. Communication between the operator’s laptop and robot will be transmitted using Xbee RF Modules. The program will use a switch operation to select the command to be carried out given a specific input from the operator’s keyboard. The numeric keypad will be used to control the robot’s driving motors and the camera position. The alphanumeric keypad will be used to control the robotic arm, base and claw. In the future, the group plans to improve the prototype design to be able to maneuverer any type of terrain and climb stairs.
Introduction

The tragedy that occurred in March 2011 at Fukushima nuclear facility lead the Nuclear Industry to issue a Request for Proposal to design and build a small, remotely controlled inspection vehicle. The vehicle’s main objectives are to determine the level of radioactivity at specified locations and inspect for damages, without exposing the human operator to high doses of radioactive contamination.

At the time, the Fukushima nuclear facility used heavy duty military defense robots to survey the disaster zone for levels of radiation, and monitor the facility. These robots, though they performed the required tasks, were much larger than needed, equipped with unnecessary equipment, and costly. The proof-of-concept robot described in this report is being specifically designed to compete in a simulated chemical/radioactive situation.

Problem Statement

The objective of this report is to design and build a remote controlled inspection vehicle. The inspection vehicle will consist of a claw used to pick up a small object a camera mounted on the arm. A laptop computer keyboard will be used to input commands such as rotating the camera, opening/closing the claw, raising/lowering the claw arm, and servo speed/directional movements.

Motivation

The main motivation for this project is to represent Florida International University at the 2013 ASME National Student Design Competition. Winning at the district level will gain the team an invitation to compete with other district winners at the International Mechanical Engineering Congress and Exposition (IMECE), showcasing the quality of engineers that FIU produces on a global platform.

The challenge of bringing the theoretical background taught throughout the course of our studies to a practical setting is also an immense motivation for the team to excel in this endeavor. Furthermore, the opportunity to develop an idea that can be used in such a way that it will save a life by limiting human interaction in areas with dangerously high levels of radiation is very encouraging.
Project Objectives

The primary objective of this project is to place Florida International University’s name on the number one spot in ASME’s 2013 Student Design Competition. To obtain this objective, our team must work diligently and efficiently in designing, manufacturing, programming, and testing our robot.

ASME has not placed physical dimensions to the vehicle in order to promote “outside of the box” thinking and ingenuity. The organization does not even require that it be a ground robot; it may be made to fly as long as they follow three simple rules in construction. The vehicle must be powered by rechargeable batteries, the device must only be controlled through wireless communication, and it must have a clearly labeled and accessible master shut off switch.

A “contaminated” area, whose dimensions are 5.00 meters by 7.75 meters (16.4 feet by 25.4 feet), as shown in Figure 1, will be inspected. The operator will not be allowed to view the competition area prior to starting their run. Inside this area, the operator of the vehicle will navigate through a series of obstacles, report a digital pressure gage, drop off a sensor, push a button, pick up a sensor, and return back to the parking area. The sensor is 25.4 millimeters (1 inch) in diameter by 50.8 millimeters (2 inches) long cylindrical wooden dowel. The button is a red 25 millimeters (0.984 inches) diameter button in the middle of a 100 millimeters (3.937 inches) square background.

There is a maximum of five minutes allotted to complete the aforementioned tasks. Each task is rewarded a number of points upon completion. Reporting the correct gage reading is worth 1000 points, dropping off sensor is worth 2000 points, pushing the button is worth 3000 points, and picking up and carrying the second sensor to the designated area is worth 2000 points. The formula for scoring is:

\[ S = \sum (R) - 10s - 200T, \]
where $R$ is the task score, $T$ is the times the device touches the boundary, and $s$ is the seconds it takes to complete the task.

While the robot is maneuvering through the course, there are strict guidelines that must be adhered to. If the vehicle leaves the course, it must re-enter without the help of any team member. If the course is damaged in any way or the vehicle does not meet any of the three requirements stated earlier, the team will be automatically disqualified. If a team member touches the device while it is competing, the team will be penalized with the maximum time of 300 seconds and an $R$ score of 0.

**Literature Survey**

Most of the search and rescue robots available on the market are heavy duty robots designed to search for victims after a catastrophic disasters, such as an earthquake, a collapsed building, or hurricane affected zone. There are a few robots specifically designed to handle chemical disasters, but they were not in great demand until the March 2011 Tohoku earthquake and tsunami that hit the Fukushima nuclear facility in Japan. This event lead the Nuclear Industry to issue a request for proposal to build a robot that is designed for this type of situation (ASME, 2012).

An article from Popular Science states that the Fukushima nuclear facility is using modified defense robots from the American company iRobot and British company QinetiQ to “conduct radiation and oxygen-level survey,” monitor the facility, and remove debris. The article continues to describe that although the robots are providing a certain level of assistance by allowing the operators to assess the situation without being in harm’s way, robots are limited in what they can do. Eventually, workers will have to enter the facility and manually fix structural damages, electrical wiring and the reactors cooling system, in order to get the nuclear facility up and running again (Boyle, 2011).

The two robots that were used at the Fukushima nuclear facility are the 510 Packbot, shown in Figure 2, and the 710 Warrior, shown in Figure 3. These robots are manufactured by the American company iRobot. As previously stated, both are classified as defense robots; not purpose designed for chemical disasters. The Warrior and Packbot were modified and equipped with radiation monitoring
equipment before entering the facilities but are inherently built to handle war time missions such as explosive disposal.

The Packbot, iRobot’s first government funded defense robot, was designed to protect soldier’s lives by observing from a safe distance, detecting explosives in a building, and disposing of bombs. It is approximately 16 to 21 inches in length and 27 to 35 inches wide depending on how its flippers are positioned. This robot is portable, lightweight (about 25 pounds), and can be set up and ready to operate in under two minutes. It is equipped with global positioning system that allows the operator to know exactly where it is all times. The robot camera has three degrees of freedom, and relaying quality real time video to the operator to control the robot’s movement. The robot consists of two manipulators. The first arm has eight degrees of freedom (approximately 73 inches when fully extended) and able to lift from 10 to 30 pounds, depending on how far the arm payload is from the center of mass. The second arm is much smaller with only four degrees of freedom (approximately 40 inches when fully extended), and able to lift from 5 to 15 pounds, depending on how far the arm payload is from the center of mass. This robot is powered by two BB-2590/U lithium-ion rechargeable batteries that provides about 4 hours of continuous use on one charge and can go a maximum of 5.8 miles per hour. Therefore, this robot should stay within a 10 mile radius in order to return to the operator before it discharges (iRobot, iRobot 510 PackBot, 2012).

The Warrior is bigger, tougher and heavier than the Packbot. This robot was designed for surveillance on rougher terrain. It can climb stairs and easily recover from roll over. It is approximately 35 inches in length and 21 to 30 inches wide depending on whether the flippers are attach. This robot can weight from 365 to 500 pounds. The robot manipulator extends a maximum of 75 inches and can lift 70 to 300 pounds, depending on the distance the payload is from the center of mass. Warrior is powered by 12 BB-2590/U lithium-ion rechargeable batteries that can run from 4 to 10 hours. The
maximum speed is 8 miles per hour. The wireless range is approximately half a mile (2600 feet) (iRobot, iRobot 710 Warrior, 2012).

The British company, QinetiQ’s Talon robot is also a military defense robot, shown in Figure 4. It was designed to overcome any terrain, weather conditions and/or combat situation thrown at it, including heavy rain, desert storm, steep rocky mountain, ice and snow. This robot is approximately 23 inches in width, 34 inches in length, and weights 115 to 157 pounds, depending on the equipment installed. This robot can be equipped with global positioning system, up to seven cameras, including night vision, thermal imaging and zoom control; and sensors to detect explosives, chemical material, and temperature, depending on the mission. It travels an average of five miles per hour. It has one manipulator with three degrees of freedom that extends up to 82 inches and can lift 10 to 25 pound. Yet, the robot can tow up to 1500 pound. The Talon runs on one lithium battery that lasts about an hour (Esenturk, 2010).

Each of the previously mentioned robots has their downside, primarily in the cost of producing them. The iRobot Packbot’s average cost is $402,874. While the 710 Warrior’s exact cost is not yet available, it can be assumed that it will not be cheaper than the Packbot. The Talon, while not as expensive as the Packbot, still comes in at a staggering $112,184 (Army Guide Magazine, 2008).
Conceptual Design

All conceptual designs have a basic function of being completely remote controlled as the ASME competition requires. The first design concept that our team developed is a four wheel remote vehicle with rear steering for sharp turns and a robotic arm located in the middle of the platform on the right side. This robotic arm design was developed with the intention of stopping along the side of the item that needs to be picked up. The camera is attached to a pan and tilt assembly for easy maneuverability of camera. The drawing of this concept design is shown in Figure 5.

The second concept design is a four wheel drive vehicle whose platform is lower to the ground to reduce drag. It is equipped with a robotic arm located in the front center and dual cameras for a wider view. One of the cameras is located in the robotic arm and the second below the platform in the front. This design is wifi enabled as well. The drawing of this concept design is shown in Figure 6.

The third concept design is a four wheeled drive remote vehicle equipped with a robotic arm also located in the front center. Two cameras located in the front on both sides of the robotic arm. The platform in this design is slightly raised to increase ground clearance for faster maneuverability. This vehicle is wifi enabled as well and has indoor and outdoor application as well. The drawing of this concept design is shown in Figure 7.
The fourth concept design is a caterpillar tracked vehicle that has a robotic arm located toward the front center and a front camera and controlled via wifi to a remote host. Caterpillar tracks are mostly seen on vehicles for used in outdoor applications such as construction. The drawing of this concept design is shown in Figure 8.

**Proposed Design**

The first proposed design agreed upon is similar to the fourth conceptual design discussed in the previous section (Figure 8). The fourth concept design has been modified to include a single camera that moves with the robotic arm for easy view all around. Also, this design will use radio communication instead of wifi. This design was chosen because of its versatility for indoor and outdoor use, able to turn in sharp corners, and rugged form. Further analysis and simulation will be applied and all of necessary components will be integrated as the project progresses. A 3D drawing of the proposed design is shown in Figure 9.

After further considerations, a new design was proposed, shown in Figure 11. Since, the course will not have any rough terrain where the robot is at risk of sinking or loosing traction, using caterpillar tracks proved to be unnecessary. The new drive system will be two wheels at the rear and either a ball caster or a swivel caster at the front. This design not only increases the efficiency of the motors, but it gives flexibility of changing the speed of the robot by simply putting larger or smaller diameter wheels on it. These options in turn give the motors some flexibility in their performance output.
The wheels used are BaneBots medium 40 Shore A orange tread wheels. The sizes acquired are 2.875 inches and 3.875 inches diameter with a hex hub adapter to fit into a 3 millimeters (0.117 inches) diameter shaft of the motor. They are 0.8 inches thick allowing a greater area to maintain contact with the floor and prevent slipping while quickly changing direction or speed.

Final Design

The robot was initially built according to the second proposed design (Figure 10). After testing the design of the robot arm with all of the brackets and motors, it was decided the camera was too heavy to mount at the end of the arm. Therefore, modifications were made to the chassis to accommodate the new design of the pan and tilt camera.

Timeline

The timeline for our project, shown in Figure 12, is constrained by the date of the competition. The competition will be held at the University of Alabama in Tuscaloosa, Alabama from April 5 through 7. The approach the team has taken to accomplish the major tasks is mainly a divided team effort. Each team member was responsible for a major component of the robot: Body-Sebastian; Robotic Arm-Ryan; and Programming-Marybel. Yet, each task required some degree of team input, so everyone always knew what the current and future plans were throughout the entire process. Therefore, much of the work was done independently outside of school. At the team’s weekly meeting, the project’s progress was discussed and modifications of ideas were considered. Communication of in-process ideas and sharing of literature was done through email.
Analytical Analysis

As with any project, some extent of analysis must take place in order to choose parts and ensure that all constraints are met. For this section, the analysis will be broken down into two parts: the robotic arm and the robot frame.

Robot Frame

Before beginning the selection of a motor, a few key factors must be known or properly estimated; Size, weight, speed, and terrain that must be traveled by the robot. A spreadsheet was created where the only inputs are the weight, the maximum desired speed, time required to reach maximum speed, and the wheel diameter. Each of these inputs vary the minimum requirements of the DC motors by quite a bit so a careful selection process is necessary.

The weight of the robot was estimated leaning toward the heavy side of the scale. It is better to plan for a heavier robot and get motors that will surpass the required specifications than to aim low and run out of available power supplied by the motors. In this case, the robot was estimated to be around 20 pounds.

Since the course is roughly 25 feet long and it was arbitrarily decided that the robot would, if needed, travel from one end of the course to the other without obstruction in about 15 seconds, or roughly 90 feet per minute. Furthermore, to
minimize the amount of torque required for the motors, a time of two seconds was selected as the amount of time for the robot to obtain full speed. This time and speed was selected after the spreadsheet was complete and several different hypothetical numbers were tested.

To obtain the minimum required specifications for the motors, the following formulas were used.

Input:

\[ \text{Weight} = w = 20 \text{ pounds} \rightarrow \text{Mass} = m = 0.623 \text{ slugs} \]

\[ \text{Maximum Speed} = 90 \frac{\text{feet}}{\text{minute}} = 1.5 \frac{\text{feet}}{\text{second}} \]

\[ \text{Time for Maximum Speed} = 2 \text{ seconds} \]

\[ \text{Wheel Diameter} = 3.875 \text{ inches} \]

Outputs:

\[ \text{Acceleration} = (\text{max speed}) \backslash (\text{time to achieve max speed}) = 0.75 \frac{\text{feet}}{\text{second}^2} \]

Given that

\[ \text{Torque} = \text{Force} \times \text{Radius} \]

\[ \text{Force} = \text{Mass} \times \text{Acceleration} \]

Therefore

\[ \text{Torque} = \text{Mass} \times \text{Acceleration} \times \text{Wheel Radius} \]

\[ \text{Torque} = 0.623 \frac{\text{pound force} - \text{second}^2}{\text{feet}} \times 0.75 \frac{\text{feet}}{\text{second}^2} \times \frac{3.875 \text{ inches}}{2} \frac{1 \text{ feet}}{12 \text{ inches}} \]

\[ = 0.075274 \text{ pound force} - \text{feet} \]

In all of the motor catalogs, the torque of the motor is given in ounce-inches force. In order to convert pound-feet to ounce-inches, the following relation was used:

\[ 1 \text{ pound force} - \text{feet} \times \left( \frac{16 \text{ ounces}}{1 \text{ pound force}} \right) \times \left( \frac{12 \text{ inches}}{1 \text{ feet}} \right) = 192 \text{ ounce} - \text{inches} \]

So the total torque required to move the robot given the inputs is
0.75274 pound force – feet * 192 ounce – inches = \frac{14.4567 \text{ ounce – inches}}{2 \text{ motors}}

= 7.22633 \text{ ounce – inches/motor}

The more motors that are on the robot, the more the torque gets distributed, taking away stress from individual motors and making the motor selection requirements less expensive.

The next thing needed for the calculations is the required revolutions per minute (RPM) the motors need to achieve maximum speed.

\[
\text{RPM} = \text{max speed} \times \frac{\text{wheel diameter}}{12} \times \pi
\]

\[
= \left(90 \text{ feet/seconds}\right) \times \left(\frac{3.875 \text{ inches}}{12}\right) \times \pi
\]

= 88.71 \text{ revolution/minute}

The final motor choices are discussed in the next section, Major Components.

**Robotic Arm**

Schematic diagram of our robot arm design is shown in Figure 13.

The force calculations required for motor selection is shown below. This data was used to select a motor that can support its own weight along with the added

![Figure 13 Force Body Diagram of Robot Arm Design](Image)
payload. This particular design has three degrees of freedom (DOF) and the center of mass of each linkage is assumed to be length divided by two.

Using these data:

<table>
<thead>
<tr>
<th>Table 1 Robot Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>L₁</td>
</tr>
<tr>
<td>L₂</td>
</tr>
<tr>
<td>L₃</td>
</tr>
<tr>
<td>W₁</td>
</tr>
<tr>
<td>W₂</td>
</tr>
<tr>
<td>W₃</td>
</tr>
<tr>
<td>W₄</td>
</tr>
<tr>
<td>W₅</td>
</tr>
<tr>
<td>W₆</td>
</tr>
<tr>
<td>W₇</td>
</tr>
</tbody>
</table>

Torque about Joint 1:

\[ M_1 = \frac{L_1}{2} W_2 + L_1 W_3 + W_4 \left( L_1 + \frac{L_2}{2} \right) + W_5 \left( L_1 + L_2 \right) + W_6 \left( L_1 + L_2 + \frac{L_3}{2} \right) + W_7 \left( L_1 + L_2 + L_3 \right) \]

\[ M_1 = 3.48 \text{ pound} - \text{inches} \]

Torque about joint 2:

\[ M_2 = \frac{L_1}{2} W_2 + L_1 W_1 + \frac{L_2}{2} W_4 + L_2 W_5 + W_6 \left( L_2 + \frac{L_3}{2} \right) + W_7 (L_2 + L_3) \]

\[ M_2 = 1.79 \text{ pound} - \text{inches} \]

Torque about joint 3:

\[ M_3 = \frac{L_2}{2} W_4 + \frac{L_3}{2} W_6 + L_2 W_3 + W_2 \left( L_2 + \frac{L_1}{2} \right) + W_1 (L_1 + L_2) + L_3 W_7 \]

\[ M_3 = 0.304 \text{ pound} - \text{inches} \]

The servo that we chose for this application for the robot arm is the Hitec 422 servo motors. It has a torque capacity of 3.5625 pound-inches and weighs 0.1 pounds. As you could see in the above computation, torque at joint 1 is the highest as because of all the combined weight are being considered at this point, therefore,
increasing the required torque. At joints 2 and 3 both required minimal torque as shown in the formula. These could be well be minimized by designing a shorter arm length to gain smaller torque requirements. Our design is still within the range of the servo that we chose for this application.

**Forward Kinematics**

<table>
<thead>
<tr>
<th>link</th>
<th>angle</th>
<th>x pt loc</th>
<th>y pt loc</th>
</tr>
</thead>
<tbody>
<tr>
<td>J0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>J1</td>
<td>20</td>
<td>4.23</td>
<td>1.54</td>
</tr>
<tr>
<td>J2</td>
<td>80</td>
<td>3.13</td>
<td>7.74</td>
</tr>
</tbody>
</table>

*Figure 14 Forward Kinematics*

Forward kinematics is the technique for finding the orientation and position of the end effector manipulator, given the joint angles and link lengths of the robot arm.

Here, the end effector location was calculated given the joint angles and link lengths. For this approach, a joint angle of 20 and 80 degrees is given for both joints respectively. For joint 1, obtain an X and Y point location of 4.23 inches and 1.54 inches, and for Joint 2, X at 3.13 inches and Y at 7.74 inches. Below is an explanation of this approach.

**Robot Arm Kinematics**

Assume that the base is located at x=0 and y=0. The first step would be to locate x and y of each joint.

Joint 0 (with x and y at base equaling 0):
Figure 15 Robot Arm Kinematics

$x_0 = 0, \ y_0 = L_0$

Joint 1 (with x and y at J1 = 0):

\[
\cos \varphi = \frac{x_1}{L_1} \rightarrow x_1 = L_1 \cos \varphi
\]

\[
\sin \varphi = \frac{y_1}{L_1} \rightarrow y_1 = L_1 \sin \varphi
\]

Joint 2 (with x and y at J2 = 0):

\[
\cos \theta = \frac{x_2}{L_2} \rightarrow x_2 = L_2 \cos \theta
\]

\[
\sin \theta = \frac{y_2}{L_2} \rightarrow y_2 = L_2 \sin \theta
\]

**End Effector Location computation:**

\[
x_0 + x_1 + x_2, \ or \ 0 + L_1 \cos \varphi + L_2 \sin \theta
\]

\[
y_0 + y_1 + y_2, \ or \ 0 + L_1 \sin \varphi + L_2 \cos \theta
\]

Note that $z$ equals $\alpha$, in cylindrical coordinates

The angle of the end effector, in this example, is equal to $\theta + \varphi$. 
Inverse Kinematics:

Inverse kinematics is the opposite of forward kinematics. The desired end effector position is determined by finding first the joint angles for this approach. Here, a desired position of X at 3.56 inches and Y at 6.43 inches is given. Using the equations below, joint 1 angle of 2.54 degrees and joint 2 angle at 95.99 degrees is obtained.

Provided below is an equation to our design.

\[
\varphi = \cos^{-1}\left(\frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1L_2}\right)
\]

\[
\theta = \sin^{-1}\left(\frac{y(L_1 + L_2c_2) - L_2s_2x}{x^2 + y^2}\right)
\]

Where:

\[
c_2 = \frac{x^2 + y^2 - L_1^2 - L_2^2}{2L_1L_2}
\]

\[
s_2 = \sqrt{1 - c_2^2}.
\]

\[
x = dx\cos\theta
\]

\[
x^2 = dz^2\cos^2\theta
\]

\[
y = dx\sin\theta
\]
\[ y^2 = dz^2 \sin^2 \theta_1 \]
\[ x^2 + y^2 = dz^2 \]
\[ dz = \sqrt{(x^2 + y^2)} \]
\[ \tan \theta_1 = \frac{y}{x} \]
\[ \theta_1 = \tan^{-1} \left( \frac{y}{x} \right) \]

Inverse kinematics involves non-linear simultaneous equations, multiple numbers of solutions and approach, making it difficult to utilize. There is the possibility of zero solutions. Maybe the location is outside the workspace, or maybe the point within the workspace must be gripped at an impossible angle. Singularities, a place of infinite acceleration, can blow up equations and/or leave motors lagging behind. And lastly, exponential equations on a microcontroller are extensive. Therefore, there is no point in having advanced equations on a processor that cannot keep up.

Another example of an inverse kinematics is a rotating drum and the frames and links are specified and marked on Figure 17 and Table 2.

![Figure 17 Rotating Drum](image)

<table>
<thead>
<tr>
<th>Frames</th>
<th>Rotate Angle</th>
<th>Length</th>
<th>Offset</th>
<th>Joint variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>i</td>
<td>ai-1</td>
<td>ai-1</td>
<td>di</td>
<td>\theta_i-1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>\theta_1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>d2</td>
<td>0</td>
</tr>
</tbody>
</table>
Transformation matrix

Our robotic arm has 5 DOF and a lot of ways to do an analysis. One of them is the transformation matrices. These are used to describe the position and orientation of one coordinate frame with respect to another. The matrix is made of a position vector and a rotation matrix.

Position Vector

A position vector is simply the coordinates of one frame compared to another. In robotics cartesian coordinates are generally used to describe position.

Rotation

Rotation transformation is the outcome of the orientation of the initial space compared from the orientation of the final space. The basis vectors of the space do not change how they are oriented relative to one another, but relative to the destination coordinate system, they are pointed in different directions than they were in their own coordinate system. Rotations are known to be difficult of the basic transformations, mainly because of the math involved in computing the transformation matrix. In general, rotations are viewed at as an operation, such as rotating around a particular basis vector or some such. A rotation matrix is just a transform that shows the basis vectors of the input space in a different orientation. The length, origin and angle between the basis vectors will not change. All that changes is the relative direction of all of the basis vectors.

\[
\begin{bmatrix}
    1 & 0 & 0 & dz_1 \\
    0 & 1 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 \\
\end{bmatrix}
\begin{bmatrix}
    C_1 & -S_1 & 0 & 0 \\
    S_1 & C_1 & 0 & 0 \\
    0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
    C_1 & -S_1 & 0 & dzC_1 \\
    S_3 & C_3 & 0 & dzS_1 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1 \\
\end{bmatrix}
\]
Therefore, a rotation matrix is not really considered as a “rotation” matrix; it is an orientation matrix. It details the orientation of one space relative to another space.

For any two spaces, the orientation transformation between them can be expressed as rotating the source space by some angle around a particular axis (specified in the initial space). This is true for any change of orientation.

The axis of rotation is expressed in terms of the initial space. In 2D, there is only one axis that can be rotated around and still remain within that 2D plane: the Z-axis. In 3D, there are many possible axes of rotation. It does not have to be one of the initial space’s basis axes; it can be any arbitrary direction. It is not possible to represent a rotation of coordinate systems with a vector, because the commutatively rule will not hold. Figure 18 is an example similar to our project.

The Denavit and Hartenberg convention, also known as the DH parameters is a commonly used for selecting frames of reference in robotics applications is which was introduced by Jacques Denavit and Richard S. Hartenberg. In this convention, coordinate frames are attached to the joints between two links such that one transformation is associated with the joint, \([Z]\), and the second is associated with the link \([X]\). The coordinate transformations along a serial robot consisting of \(n\) links form the kinematics equations of the robot,

\[
[T] = [Z_1][X_1][Z_2][X_2] \ldots [X_{n-1}][Z_n],
\]

where \([T]\) is the transformation locating the end-link.
In this problem, we drew the schematics first of the serial manipulator, and then assigned the frames. Next, we create a table for the parameters as shown in Table 3.

### Table 3 Parameter for Transformations

<table>
<thead>
<tr>
<th>Frames</th>
<th>Rotate Angle</th>
<th>Length</th>
<th>Offset</th>
<th>Joint variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( a_{i-1} )</td>
<td>( a_{i-1} )</td>
<td>( d_i )</td>
<td>( \theta_{i-1} )</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>( \theta_1 )</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>( a_1 )</td>
<td>( d_2 )</td>
<td>( \theta_2 )</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>( a_2 )</td>
<td>( d_3 )</td>
<td>( \theta_3 )</td>
</tr>
<tr>
<td>5</td>
<td>-90</td>
<td>( a_3 )</td>
<td>( d_4 )</td>
<td>( \theta_4 )</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>( \theta_5 )</td>
</tr>
</tbody>
</table>

For revolute joints, the \( a_i \) parameters are constants, and the \( \theta_i \) parameters are variables. Next step is to do the transformation, from beginning of the frame to the next.

\[
^0T_i = ^0T_z(\theta_1) = \begin{bmatrix} C_1 & -S_1 & 0 & 0 \\ S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\]

The transformation \(^0T_i\) expresses the difference between the body frame one (1) and the body frame of frame zero (0)

\[
^1T_2 = ^1T_z(90) \begin{bmatrix} 3/2R_x(\theta_2) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_2 & -S_2 & 0 & 0 \\ S_2 & C_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\]

The transformation \(^1T_2\) expresses the difference between the body frame one (1) and the body frame of frame two (2)

\[
^2T_3 = ^2T_z(a_1) \begin{bmatrix} 3/2R_x(\theta_3) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} C_3 & -S_3 & 0 & a_1 \\ S_3 & C_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_3 & -S_3 & 0 & a_1 \\ S_3 & C_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\]

The transformation \(^2T_3\) expresses the difference between the body frame two (2) and the body frame of frame three (3)
The transformation \( ^3T_4 \) expresses the difference between the body framethree (3) and the body frame of frame four (4)

\[
^3T_4 = ^3T_x (a_2) ^3R_z (\theta_4) = \begin{bmatrix}
1 & 0 & 0 & a_2 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix}
C_4 & -S_4 & 0 & 0 \\
S_4 & C_4 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} = \begin{bmatrix}
C_4 & -S_4 & 0 & a_2 \\
S_4 & C_4 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

The transformation \( ^4T_5 \) expresses the difference between the body frame four (4) and the body frame of frame five (5)

\[
^4T_5 = ^4R_x (0) ^5R_z (\theta_5) = \begin{bmatrix}
1 & 0 & 0 & a_2 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} \begin{bmatrix}
C_5 & -S_5 & 0 & 0 \\
S_5 & C_5 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} = \begin{bmatrix}
C_5 & -S_5 & 0 & 0 \\
S_5 & C_5 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

The transformation \( ^5T_6 \) expresses the difference between the body frame five (5) and the body frame of frame six (6). We just finish writing the transformation matrix for a serial manipulator with 6 degrees of freedom.

These equations were used to get some insight on the theoretical part of this project. The actual process for the position and the angles proved to be much more trivial, especially for the preset positions. For this case, the servos were told to output their angular position after every movement. Once the arm was in a location that the team seemed suitable, the angles were recorded and pre-programmed into the microcontroller.

**Major Components**

The key components in this project are the Arduino Mega microcontroller, the chassis, the robotic arm, the video system of the robot, the radio communication, and the motors. There was a significant amount of time required to learn how to wire, program, and carefully calibrate the hardware to work seamlessly with the Arduino Mega 2560. Since the team is starting from scratch, there is a lot of trial-and-error that took place.
The chassis must be lightweight enough to reduce the size of motors required to move the robot, yet heavy enough to anchor it when the robot arm moves. In addition, the plan is to have a very neat looking robot which does not leave wires exposed and cluttered; so plans will be made to take that into consideration.

The arm is one of the main drivers in how big the rest of the vehicle was. The placement of the arm along the chassis depended how heavy the arm physically was due to the danger of tipping when it reaches for the button and sensor.

In order to efficiently navigate the course, it was imperative that the team employed a camera with quality video output and a large viewing angle. Since the competition is limited to five minutes, any time spent getting lost within the course due to poor quality video or too narrow of viewing angle could cost the team first place.

For communication between the operator and robot, the team has chosen to use radio communication. Radio is reliable and consistent regardless of where and under what conditions the robot is being used. More information about the major component can be found in Appendix F: Component Specifications.

**Drive Motor Selection**

After all the necessary calculations were carried out and taking into consideration to use motors that are 50% higher than calculated for the DC motors (servomagazine.com), the motors selected to drive the robot are a pair of FingerTech Spark 100:1 gear motor. This motor has an output of 94 revolutions per minute and an output torque of 52.39 ounce-inches. These values more than meet the specifications of the robot under its heaviest condition.

**Manipulator Motor Selection**

For the robot arm, the motor selection was narrowed down to two types of motor: one for the joints and the other for the base. For the robot arm joint, the HS-422 Standard Servo motor and for the robot base, the HS-485HB Deluxe Ball Bearing Servo was chosen for its design and sturdiness. For the robot arm torque requirement, the 57 ounce-inches at 6 Volts is more than is needed, as the initial calculations just run below the capacity of this servo motor so it is simply reasonable to pick this servo motor in accelerating and driving the joints at desired speeds.
Prototype System Description

As the ASME competition specified in their requirements, a mobile robotic arm will be controlled wirelessly and has to travel to a specific path and able to read and capture data via a wireless camera and able to retrieve back and forth a light weight instrument in an assigned area.

The prototype will consist of a rover vehicle, which has a pivot steer system that will ease navigation and reduce turn radius. Another part is the robotic arm that could lift and carry parts from one point to another and an integrated camera that will provide as the visuals in controlling the vehicle and obtaining data.

Using a microcontroller for the prototype provides control of the vehicle, robotic arm and the camera. A computer is used to actively display the data coming from the camera as well as driving both the vehicle and moving the robotic arm. Radio communication will be used in order to control the vehicle, since it is reliable all around the world under any conditions as long as it is in range.

Manufacturing of Components

The chassis of the robot is made of High Density Polyethylene (HDPE) plastic with aluminum brackets for motor mounts. The top and bottom base was originally a 14 inches by 36 inches piece of HDPE and the motor brackets were aluminum ‘L’ stock 36 inches in length. Figure 20 shows the raw materials used to build the robot.

All components were machined on a Bridgeport vertical milling machine. In order to have a consistent size on the overall length and width of the top and bottom base, a stop was placed on the milling machine table which ensured that all final cuts would be in precisely the same location, shown in Figure 20.
Milling the motor mounts to length was a bit tricky due to their profile. They are not square and clamping down on an edge would distort the part. The solution to this was to use a piece of HDPE and clamp it between the motor mount and the vice. This ensured that the mount was firmly clamped on one of its flat faces and there would be no distortion. The motor mount clamping method can be seen in Figure 21.

To drill the motor mounts, a vice stop was used to establish a known zero datum point, shown in Figure 22. This way, once a bracket was finished, the next can be butted up to the stop and the machining can continue. If this method was not to be used, one would have to change tools, put in an edge finder (as the name implies, it helps find edge of the work piece to establish the datum), and re-locate a zero point. This wastes precious time that could be spent actually machining the parts.

Since the vice was not big enough to fit the top and bottom base of the chassis, they had to be drilled while clamped to the table itself. A dial indicator was used along the machined edges and using a small plastic mallet, the plate was aligned to be square within .001 inch. Below, in Figure 23, is a picture of the plate clamped to the table (on the left) as well as a hole saw being used to create the clearance hole for the rotating base of the robot arm (on the right).
All the machining went smoothly except for the bottom plate. In this plate, the error was attributed to a lack of digital read out on the milling machine and operator error. A dimension was read wrong for one of the holes and for the other, not enough turns on the handle were accounted for and the location was not double checked prior to drilling. This mistake does not affect the assembly in any way. However, it simply makes it a little bit lighter. The bottom plate of the chassis is shown in Figure 24.

A spacer, shown in Figure 26 was made to go underneath the caster wheels in the front in order to make up the difference in height from the 3.875 inches diameter drive wheels and keep the robot level. If the 2.875 inches diameter drive wheels are to be used, removing this block will also maintain the robot level.

The robot arm links are consists of polycarbonate sheet material and aluminum brackets. Polycarbonate material is strong and light and is readily available in any hardware store. The aluminum bracket is made out of an aluminum strip cut into pieces. This aluminum strip is available as well in any hardware store.

Using a scroll saw, shown in Figure 25, the polycarbonate sheet was measure and the templates drawn. Before starting the cut, plastics tend to melt due to heat cause by the blade, so the blade was lubricated first with some cutting oil.

After cutting the templates, the bench grinder was used on the rough edges on the plastic pieces, as shown in Figure 27, and using...
the wire wheel, the excess plastics was removed carefully and on the other side the grinding wheel was used to remove any sharp corners.

Next, the handheld band saw cut the aluminum brackets. After measuring and marking the amount needed, the aluminum strip was placed in a bench vise, as shown in Figure 28. Carefully, the length was cut as marked. After cutting the brackets into pieces, the bench grinder was used again to carefully remove the sharp edges and excess materials on the side.

Forming a shape of two angle plates, the aluminum brackets were attached to the polycarbonate sheet. Also, they were measured and marked a hole to secure it in place. A nut and bolt application was used for this part.

The claw or end manipulator was modified for the servo by attaching a bigger gripper in the ends of the claw, so it would be able to grasp and hold materials much better. Using the same polycarbonate and aluminum strip, the aluminum was measured and cut in to the bracket and modified gripper. After attaching the modified gripper, industrial grade Velcro was cut it the same size as the modified gripper. The Velcro was attached to the claw. This prevents the object being picked up by the robot arm, from slipping. Figure 29 shows the robot claw with the Velcro grippers attached.

After the initial design of the robot was built, some flaws in the design we discovered and adjustments were made. The first issue was the GoPro camera. The camera was supposed to be attached to the robot arm. But after built the robot it was evident that the camera was too heavy for the robot arm. Using a regular off the shelf
camera was not an option due to its size and changing the camera was also not an option either due to financial reasons. Therefore, a bracket was made for the GoPro case that was attached to a servo in order to provide the panning motion. Another servo was then attached to the back of the GoPro case with a wire (a paper clip) attached to the base. This provides the tilting motion of the pan and tilt system, shown in Figure 30.

The next issue was how to make the GoPro sit as low as possible on the chassis, so the robot arm could clear it when reaching for the sensor. The solution was to create custom spacers that allowed the servo to be below the surface of the bottom chassis plate. In order to accommodate this new set up, a cut-out was made to the bottom chassis plate for clearance of the bracket and go pro case. To allow the maximum viewing angle for the camera, the top plate was also cut. This removed the obstruction and allowed the operator to have almost 250 degrees of viewing angle without rotating the robot. Figure 31 shows the modified top base (on the left) and modified bottom base (on the right) of the chassis.
After the robot was built, it was time to wire it all together. Since the servo for each motor comes with pre-determined length of wire, extensions had to be made to make the wires long enough to reach the microcontroller. A tool called “the third hand”, shown in Figure 32, was extremely helpful to get through all of the soldering that was needed. This tool has an iron base with two clamps attached to it, helping the user hold two wires steady. Temporary sticky putty was also helpful in holding the cables together while the final lengths of the servo wires were determined. To minimize the chances of having a wire get snagged on any of the joints, flexihose was used. This hose has a slit along its entire length which allows wire to be introduced into it at any point. Ideally, something like a wire mesh should be used but due to time constraints, flexihose was the ultimate decision.

The batteries that were chosen were lithium polymer batteries. They were chosen primarily for their efficiency and size. These batteries are extremely dangerous to work with and every precaution should be taken to reduce the risk of fire and injury. A fireproof “lipo guard” was purchased and a fire extinguisher was on hand at all times. The batteries were never left un-attended while charging and electrical tape was used on the leads when the batteries were not being used. Each major component has its own power source. This ensures that no battery is overloaded, and it allows enough power to reach the components. All of the servos in the arm and the pan and tilt system are run off a single 11.1 volt 3200 milliampere-hour battery. Since the servos get damaged if run at higher than 6 volts, a switch voltage regulator was needed. This regulator is more efficient than a typical 5 volt linear regulator found at local electronics stores. Switch regulators allow more current to pass will maintaining constant voltage. The unit used was called a Battery Eliminating Circuit (BEC) from Castle, shown in Figure 33. These units are normally used in model airplanes but it was an inexpensive solution to allow more current to pass through the regulator. The drive motors and microcontroller are each powered by two different 7.4 volt, 1300 milliampere-hour lithium polymer batteries while the
video transmitter and receiver use one 11.1 volt, 1000 milliampere-hour each. It was quickly realized that there was not enough room on the chassis of the robot to accommodate the required switches and emergency stop button. The solution was a plastic hobby box sold at the local electronics store, modified to fit the switches and chassis, shown in Figure 34.

There are many wires that also need to be connected on the user end of the robot. Firstly, there is the dongle with the transmitting RF communication module. Then, there is the video converter that connects to the video receiver. All of these devices also need to be powered by the battery, adding an extra set of wires. The solution for this was a modified USB soda can cooler. The internal components were removed and replaced with all of the before mentioned components, making for a very neat way to contain all of the cables. A switch was added to the top to quickly power the video receiver on. More pictures showing the robot construction and assembly are shown in Appendix B: Assembling components.

**Programming**

The program opens a command window in order to accept input commands from the keyboard. The program uses a switch operation that selects the commands to be carried out given a certain input. The driving servos and camera will be controlled using the number key pad as follows:

To control the driving motors:

- 2- Move backward
- 4- Turn Left
6- Turn Right  
8- Move forward

To tilt the camera:
5- Tilt Upward  
7- Pan left  
9- Pan right  
/- Tilt down

The robot arm base, joints and claw will be controlled using the alphabetic keyboard as follows:

To rotate the robot arm base
l (lowercase L) - Rotates counterclockwise  
; (semicolon) - Rotates clockwise

To control the shoulder joint:
a- Raise joint  
q- Lower joint

To control the elbow joint
d- Raise joint  
e- Lower joint

To control the wrist joint
g- Raise joint  
t- Lower joint

To control the claw
, (comma) - Opens the claw  
. (period) - Closes the claw

The robot commands also include some preset position that will save time during the competition.

These position commands are as follows for the robot arm:
These position commands are as follows for the camera:

0- Home position
1- Left position
3- Right position

The pick-up position is activated by pressing SHIFT + 1. This position is used when ready to pick-up an object directly in front of the robot. The robot end-effector is placed with the claw open, close to the ground in front of the camera. Therefore, with only minimal adjustment, the object can be pick-up quickly. The home position is activated by pressing SHIFT + 3. This position is the robot arm default position. It takes the robot arm from any position and retracts the arm close to the base out of sight from the camera. The travel position is activated by pressing SHIFT + 6. This position is used after an item has been picked up. It positions the claw right above the camera. That way when taking an object from one place to another, the object is always in view. In the case that the claw drops the object, the operator will know immediately. The drop-off position is activated by SHIFT + 8. This position is used to drop of the object that has been collected. Once the robot has been positioned correctly to drop the object in the designated area, this preset position lowers the robot arm to a position close to the ground and then, opens the claw to release the object.

After any of these operations have been carried out the program will loop back to original window ready to accept the next alphanumeric keyboard command. A flow chart for the program described above is shown in Figure 35. A layout of the keyboard commands are shown in Figure 36. Note that the actual program can be found in Appendix E: Program.
Prototype Testing

Chassis

The HDPE chassis is made from an old cutting board. It is very solid and appears to be able to withstand all the weight of the robot just fine. Since it sat under a few things in a garage, it developed a slight warp that is only noticeable when the drive motors are attached. The shafts of the motors sit at a small upward angle. This did not appear to cause any sort of negative effect.

Drive

The motors that were calculated to work were FingerTech goldspark 50:1 gearhead motors. These motors are rated to perform best at less than 7.4 volts, but can be driven with as high as 22.2 volts. The battery that was initially driving these motors was 11.1 volts. The calculations showed that these motors would outperform what the robot required as far as torque and speed goes; that was not the case. What was failed to be taken into account is the event of encountering a small step, in our case, a rug. When the robot touched the rug, the motors could not handle it and the drive gear actually stripped out. Two motors were ruined in this fashion, leaving the team with a robot using a bolt as a second wheel, as shown in Figure 37.
The new motors that were purchased were FingerTech silver spark gearhead motors with a ratio of 100:1. These motors have a much coarser drive gear pitch and are rated to output much more torque. Of course, this comes at the expense of speed.

The amount of voltage being supplied to the motors was also dropped from 11.1 volts to 7.4 volts. This further reduced the torque and speed of the motors but they were noticeably smoother when activated. One extra set was purchased just in case they fail before the competition.

The drive wheels were also changed from 3.87 inches in diameter to 2.875 inches in diameter. The half inch in radius difference means the motors need to output less torque to move the robot from full stop. The downside is the robot will travel 9.02 inches per revolution instead of 12.167 inches but it was determined that the current speed is still a competitive. A physical test showed the robot can travel 25 feet in 20.22 seconds and reaches maximum velocity in approximately 2.87 seconds.

The wheel casters were also replaced as they were a source of burden when changing directions. Since they need to pivot before aligning with the direction of travel, they made the actual direction very unpredictable and extra directional inputs were needed to make the robot move where the operator wanted. The new design is a single ball caster rated for 75 pounds. This ball caster performs excellent and makes the steering much more predictable. The 1 inch ball is big enough to not get hung up on small cracks or rugs. A special bracket was made for this using perforated steel and nylon spacers. Figure 38 shows the single ball caster in the robot assembly.

**Robot arm**

The robot arm took a frustrating trial and error approach to get right. Firstly, the motors that were originally chosen were not sufficiently strong to hold the weight so they burnt out. Once the right servo was purchased, there didn’t seem to be
enough battery power that could get it to work when all the other servos were wired in. Modified wall chargers were used to try and figure out why the robot arm would not work. Finally, an old router DC wall plug as used and it worked; all motors would move, though strained. It was finally determined that once all of the servos were wired and the robot arm was extended, too much current was being drawn, causing all the motors to simply flop and stall.

The router wall plug had an output of 2 amps, which seemed to be enough but just to be safe, a 3 amp, 11.1 volt battery was purchased. This would allow for plenty of power and time of usage before needing to be recharged. The servos can only withstand 6 volts. Therefore, a switch voltage regulator was used to drop the voltage down. A simple linear regulator would not work because it can only output 1 amp, wasting all the energy in the form of heat. Through the testing, this battery was used total time of approximately 45 minutes and it still had plenty of charge left for another 30 minutes.

To further reduce the amount of stress on the servos when the arm was extended, a spring was placed on the shoulder joint and the elbow joint. This spring minimizes the amount of weight the shoulder and elbow servo “see,” allowing for less power consumption and smoother operation. Figure 40 and Figure 40 show the spring for the elbow and shoulder, respectively.
Programming / Communication

The Arduino Mega 2850 is the brains driving the robot. It uses the Xbee wireless radio frequency modules to transmit the data from the computer to the robot. Since this robot uses the keyboard to control it, there are many different options to controlling the robot. Presets were made for quicker movements of the claw. This was done by reading the angle of the servos, writing the angle down, and making a list of the order of movement. The benefit of this is speed. Since the robot arm moves in increments of 4 degrees when being controlled by the operator, it is slow when trying to move 120 degrees. When a preset is made, it can be brought to that position at a much faster rate. Also, since the camera cannot see what is going on behind it, there would be no way for the operator to know where the robot arm is.

Maintenance

This robot is very low maintenance and upkeep takes no time at all. Most of mechanical parts are screwed tightly using screws and lock washers. Medium strength thread locker has been applied to any component that does not have to be removed for maintenance. The top plate of the chassis is easily removed by disconnecting the video transmitter and unscrewing (4) screws.

The only components in the robot which need to be checked regularly are the batteries. Due to their nature, if they are run too low, they will incur permanent damage and will not charge. A lithium polymer battery cell checker is used to make sure the batteries’ don’t fall below the 3.2V per cell the manufacture recommends.

Environmental Impact

This robot does not have a major impact on the environment. In fact, it’s purpose helps the environment by being able to enter radioactive spaces and measure their safety. The entire robot, camera system, and controls are run using rechargeable lithium polymer batteries, eliminating waste from their non-rechargeable counterparts.
**Risk Factors**

A very good precaution to take when dealing with lithium polymer batteries is to have a fireproof bag and a fire extinguisher, shown on Figure 41. While attempting to change the leads on the 11.1 volt battery, without thinking, both leads were cut at the same time, shorting out the battery. In a matter of seconds, the battery let out a puff of smoke and fire quickly came after. A simple mistake nearly set the apartment on fire. Other than a fire hazard, this robot does not pose any other risks.

![Lipo Bag and Fire Extinguisher](image)

**Competition**

The competition was held in Tuscaloosa, Alabama on April 6\textsuperscript{th}, 2013. A total of 30 teams competed, making it one of the biggest turn outs compared to years past. The course that the robot ran is said to have been the most difficult course of all the districts. From the uploaded videos of other districts, this seems to be true due to the minimalistic course builds.

This course was built using tall sheet rock and plywood. The layout was in a basketball court, with blue tape marking the boundaries of the course. Water bottles were scattered about and towels were placed as obstacles. This added an extra level of difficulty.

The following are screen shots of the actual competition run. The left view is what the operator saw and the right is what the assisting team member saw. The two team members were isolated and not allowed to speak to each other until the run was over. The competition was started with the sensor already in the grasps of the claw, and within the boundary of the tape, as seen in Figure 42. This was also the location the robot had to return before five minutes were up.
The drop off zone was also designated by a square box. It was in the farthest part of the course and somewhat obstructed by the towel. This is shown in Figure 43.

The sensor to be picked up was in the corner of two walls. In order to reach this location, the robot had to navigate around three water bottles, as shown in Figure 44.
Figure 45 shows the pressure gage reading. This number had to be read to a judge sitting in the booth. The judge would not tell the operator whether the reading was correct or not, so having good video feed was crucial.

![Figure 45 Gage Reading](image)

The button, shown in Figure 46, was supposed to be about 2 inches off the ground but it was made almost 4 inches high. This task proved difficult for some of the teams that only had a 2 inch reach. For these teams, as long as the robot touched the wall, the task had been accomplished.

![Figure 46 Pressing the Button](image)

Out of the 30 teams that entered the competition, Florida International University came in 9th place. This is extraordinarily well considering we were up against teams that had as many as 18 engineers working on a single robot. The average number of people on a team was 8.
Cost Analysis

In addition, to the cost of materials for the prototype, there is time invested by each of the engineers in developing ideas; researching competitor’s design, materials, and pricing; sketching conceptual design and drawing the final design; building the prototype; programming and debugging; testing the prototype; and preparing for competitions and class presentations. For this project, the group put in approximately 40 hours in meeting discussing potential projects, researching current designs, discussing design concepts and distributing the workload; 10 hours preparing PowerPoint presentations, rehearsing as a group and presenting in the classroom; 9 hours doing researching individually; 9 hours sketching various design concepts; and 45 hours writing, proofreading and editing the report and other documents related to the project. Table 4 shows the breakdown of each member’s contribution to the project and the total hours projected for fall and spring for each member.

<table>
<thead>
<tr>
<th>Task</th>
<th>Hours for Fall</th>
<th>Projected Hours for Spring</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Marybel</td>
<td>Ryan</td>
</tr>
<tr>
<td>Meeting</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>Research</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Presentation</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Design</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Writing</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>Building</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Programming</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Debugging/Testing</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Competition</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Total per member</td>
<td>28</td>
<td>27</td>
</tr>
<tr>
<td>Total per semester</td>
<td>83</td>
<td></td>
</tr>
</tbody>
</table>
Prototype Cost Analysis

This project was generously funded by Southern Gear and Machine Inc. in Hialeah, Florida. They are a gear manufacturing company whose president, Allan Arch, is heavily involved with Florida International University College of Engineering and eager to see future engineers reach their full potential.

With this funding, the team was able to purchase the right equipment to obtain the best possibility of succeeding. Building of the robot cost a total of $1235.30. This was quite a bit higher than was originally budgeted for. The price of electronics drove the cost very high, as one can note from the pivot table summary shown in Table 5. An expanded table can be found in Appendix G: Prototype Expenses. There were factors such as shipping costs and likelihood of breaking components that made it necessary to purchase multiple quantities.

<table>
<thead>
<tr>
<th>QTY Purchased</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electronics</td>
<td>$782.51</td>
</tr>
<tr>
<td>Hardware/ Misc.</td>
<td>$207.79</td>
</tr>
<tr>
<td>Administrative</td>
<td>$245.00</td>
</tr>
<tr>
<td>Grand Total</td>
<td>$1,235.30</td>
</tr>
</tbody>
</table>

This table does not include travel costs to the competition. The competition was held in Tuscoloolsa Alabama, a 768 mile trip one way. The registration fee was $65 and the hotel stay was $302.05 after taxes. When all of these expenses are taken into account, the grand total of the project was about $2,000, including food and travel expenses.
Conclusion and Future Work

Even though, ASME competition guidelines did not provide many design restriction, the design of this robot was constantly evolving. Even after being assembled, the team had to go back and redesign certain parts of the robot, in order to make the robot more efficient. This could be mainly due to the team inexperience in building robots.

Our team was also inexperienced in working with circuits. As mechanical engineers, only two classes that deal with circuits and electrical components are taken. As a result, there was a lot of trial and error involved in calculating and choosing of motors as well as in wiring the system. Therefore, in some cases, motors burned out and new motors had to be purchased.

Although our project received funding, the project exceeded the budget. Originally, we had estimated approximately $500 would be spent on parts, but it turned out to be almost $1000 in electronics and hardware alone. This is not including administrative expenses, such as poster board and final report, and out-of-state competition traveling expenses.

Our team worked almost completely independently. For future projects, we recommend looking for consultant in the team’s weaker area, such as circuit and component selection. This will save time and money in the long run.

In the future, our group plans to improve the prototype so that it can climb stairs and can maneuver any type of terrains. Also, to present a more competitive product, the group will research ways to improve energy consumption in order to maximize efficiency. Finally, the group reviewed the budget and considers upgrading visual, sensory and communication equipment.
References


Appendices

Appendix A: Engineering Drawings

Figure 47 Robot Design Drawing

Figure 48 Top Base Drawing
Figure 51 DC Motor Bracket Drawing

Figure 52 Caster Spacer Drawing
Figure 53  Robot Arm Assembly Drawing
Appendix B: Assembling components

Figure 54 Top Plate

Figure 55 Close-up of Motor Mount
Figure 56 Drive Motor On Mount

Figure 57 Wheel Hub On Drive Motor
Figure 58 Motor Mounts And Spacers

Figure 59 Caster Wheel On Spacer
Figure 60 Drive Motor On Bottom Base

Figure 61 Wheels Assembled To Bottom Base
Figure 65 Chassis With Pan and Tilt

Figure 66 Sticky-Putty Holding Servo Wires For Sizing
Figure 67 Flexihose

Figure 68 Video Receiver
Figure 69 Robot Arm Wiring

Figure 70 Robot Fully Assembled (Left View)
Figure 71 Robot Fully Assembled (Right View)

Figure 72 Robot Fully Assembled (Rear View)
Appendix C: Wiring Diagrams

Figure 73 Xbee Wiring
Appendix D: Software used

Solid Works 2010

This is a 3D software that enables engineers to rapidly create parts, assemblies, and 2D drawings of their ideas. Solidworks was used to bring the ideas of the team into a digital medium which allowed for easier communication of each member’s vision.

Arduino IDE

This is an application written in java which uses C or C++ as the primary language. The ease of using this software comes from the built in libraries which allow for uploading sketches, as arduino calls programs, onto different boards with a simple click of a drop down list.

X-CTU

This is software from the makers of the Xbee Wireless RF Modules, Digi. With this software, and a Xbee USB Explorer from Sparkfun.com, the modules can be easily configured to work with each other. An added bonus to this software is its ability to send serial commands through the Xbees. This makes it much easier to use the keyboard as a remote control for the drive motors. The software can be downloaded from www.digi.com.

Fritzing

With Fritzing, built in parts are combined on a breadboard and connections are made to create diagrams as shown in Appendix B. This software is free to download from www.fritzing.org. It can be used to create custom Printed Circuit Boards (PCB) as it automatically routes the diagrams. For this project, this feature was used to simply communicate wiring approaches among team members rather than sending a picture or handwritten sketch.
Appendix E: Program

#include <Servo.h>

//******Servo Motor **********
Servo baseServo;
Servo shoulderServo;
Servo elbowServo;
Servo wristServo;
Servo clawServo;
Servo panServo;
Servo tiltServo;

//****** Drive Motor Constants ********
int motorPin1 = 51;
int motorPin2 = 49;
int ENA = 53;
int motorPin3 = 47;
int motorPin4 = 45;
int ENB = 43;
int driveDelay = 50;
int v = 90;
int slowInc = 2;
int slowDelay = 50;

void setup() {
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(8, OUTPUT);
  pinMode (motorPin1, OUTPUT);
  pinMode (motorPin2, OUTPUT);
  pinMode (ENA, OUTPUT);
  pinMode (motorPin3, OUTPUT);
  pinMode (motorPin4, OUTPUT);
  pinMode (ENB, OUTPUT);
  baseServo.attach(2);
  shoulderServo.attach(3);
  elbowServo.attach(4);
  wristServo.attach(5);
  clawServo.attach(6);
  panServo.attach(7);
  tiltServo.attach(8);
  Serial.begin(9600);
  Serial.println("Ready");
  //homePos();
}

void loop() {
  int baseLoc = baseServo.read();
  int shoulderLoc = shoulderServo.read();
  int elbowLoc = elbowServo.read();
  int wristLoc = wristServo.read();
  int clawLoc = clawServo.read();
  int panLoc = panServo.read();
  int tiltLoc = tiltServo.read();
  char ch;
  int intervals = 2;
  if (Serial.available()) {
    ch = Serial.read();
    switch(ch) {
    case '8':
      digitalWrite(ENA, HIGH);
      break;
    // Add more cases here
    def
digitalWrite(ENB, HIGH);
digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
delay(driveDelay);
digitalWrite(ENA, LOW);
digitalWrite(ENB, LOW);
break;

//***** BACKWARDS *****
case '2':
digitalWrite(ENA, HIGH);
digitalWrite(ENB, HIGH);
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, HIGH);
delay(driveDelay);
digitalWrite(ENA, LOW);
digitalWrite(ENB, LOW);
break;

//***** RIGHT *****
case '6':
digitalWrite(ENA, HIGH);
digitalWrite(ENB, HIGH);
digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, LOW);
digitalWrite(motorPin3, LOW);
digitalWrite(motorPin4, HIGH);
delay(driveDelay);
digitalWrite(ENA, LOW);
digitalWrite(ENB, LOW);
break;

//***** LEFT *****
case '4':
digitalWrite(ENA, HIGH);
digitalWrite(ENB, HIGH);
digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, HIGH);
digitalWrite(motorPin3, HIGH);
digitalWrite(motorPin4, LOW);
delay(driveDelay);
digitalWrite(ENA, LOW);
digitalWrite(ENB, LOW);
break;

//****** BASE ******
case 'l':
if (baseLoc <= 55){
v = 55;
Serial.println("Base Servo Max Travel");
}
else{v = baseLoc - intervals;}
baseServo.write(v);
Serial.println(v);
break;

case ';':
if (baseLoc >= 145){
v = 145;
Serial.println("Base Servo Max Travel");
}
else{v = baseLoc + intervals;}
baseServo.write(v);
Serial.println(v);
break;

//***** SHOULDER *****
case 'q':
if (shoulderLoc >= 165){
v = 165;
Serial.println("Shoulder Servo Max Travel");
}
}  
else{v = shoulderLoc + intervals;}
shoulderServo.write(v);
Serial.print(v);
break;

case 'a':
if (shoulderLoc <= 10){
  v = 10;
  Serial.println("Shoulder Servo Max Travel");
}  
else{v = shoulderLoc - intervals;}
shoulderServo.write(v);
Serial.print(v);
break;

/***** ELBOW *****
case 'e':
if (elbowLoc >= 150){
  v = 150;
  Serial.println("Elbow Servo Max Travel");
}  
else{v = elbowLoc + intervals;}
elbowServo.write(v);
Serial.print(v);
break;

case 'd':
if (elbowLoc <= 35){
  v = 35;
  Serial.println("Elbow Servo Max Travel");
}  
else{v = elbowLoc - intervals;}
elbowServo.write(v);
Serial.print(v);
break;

/***** WRIST *****
case 't':
if (wristLoc <= 60){
  v = 60;
  Serial.println("Wrist Servo Max Travel");
}  
else{v = wristLoc - intervals;}
wristServo.write(v);
break;

case 'g':
if (wristLoc >= 180){
  v = 180;
  Serial.println("Wrist Servo Max Travel");
}  
else{v = wristLoc + intervals;}
wristServo.write(v);
Serial.print(v);
break;

/***** CLAW *****
case ',':
if (clawLoc <= 40){
  v = 40;
  Serial.println("Claw Servo Max Travel");
}  
else{v = clawLoc - intervals;}
clawServo.write(v);
Serial.print(v);
break;

case '.':
if (clawLoc >= 100){
  v = 100;
  Serial.println("Claw Servo Max Travel");
}  
else{v = clawLoc + intervals;}

clawServo.write(v);
Serial.println(v);
break;

//***** PAN *****
case '7':
  if (panLoc >= 170){
    v = 170;
    Serial.println("Pan Servo Max Travel");
  }
  else{
    v = panLoc + intervals;
  }
panServo.write(v);
Serial.println(v);
break;

case '9':
  if (panLoc <= 0){
    v = 0;
    Serial.println("Pan Servo Max Travel");
  }
  else{
    v = panLoc - intervals;
  }
panServo.write(v);
Serial.println(v);
break;

//***** TILT *****
case '/':
  if (tiltLoc >= 175){
    v = 175;
    Serial.println("Tilt Servo Max Travel");
  }
  else{
    v = tiltLoc + intervals;
  }
Serial.println(v);
tiltServo.write(v);
break;

case '5':
  if (tiltLoc <= 35){
    v = 35;
    Serial.println("Tilt Servo Max Travel");
  }
  else{
    v = tiltLoc - intervals;
  }
tiltServo.write(v);
Serial.println(v);
break;

//***** HOME *****
case '#':
  homePos();
  break;

//***** Sets claw within view of camera while holding sensor *****
case '^':
  shoulderSlow(100);
delay(50);
elbowSlow(100);
delay(50);
wristSlow(90);
delay(50);
panServo.write(90);
delay(50);
tiltServo.write(90);
delay(50);
baseSlow(90);
break;

//***** Pick Up Mode ****

/**
case '!' :
  shoulderSlow(40);
  elbowSlow(130);
  shoulderSlow(80);
  elbowSlow(110);
  wristSlow(120);
  shoulderSlow(100);
  clawServo.write(40);
  break;

} */

//***** HOME *****

void homePos() {
  shoulderSlow(80);
  elbowSlow(110);
  wristSlow(100);
  shoulderSlow(10);
  elbowSlow(155);
  panServo.write(90);
  delay(50);
  tiltServo.write(90);
  delay(50);
  baseSlow(90);
}

//***** Wrist movement *****

void wristSlow(int angle)
int x = wristServo.read();
if ( x < angle)[
  while(x <= angle){
    x += slowInc;
    wristServo.write(x);
    delay(slowDelay);
  }
}
else if (x > angle){
  while (x >= angle){
    x -= slowInc;
    wristServo.write(x);
    delay(slowDelay);
  }
}

//*****SHOULDER SLOW *****

void shoulderSlow(int angle)
int x = shoulderServo.read();
if ( x < angle){
  while(x <= angle){
    x += slowInc;
    shoulderServo.write(x);
    delay(slowDelay);
  }
}
else if (x > angle){
  while (x >= angle){
    x -= slowInc;
    shoulderServo.write(x);
    delay(slowDelay);
  }
}

//***** ELBOW SLOW *****

void elbowSlow(int angle)
int x = elbowServo.read();
if ( x < angle){
  while(x <= angle){
    x += slowInc;
  }

63
elbowServo.write(x);
delay(slowDelay);
}
else if (x > angle){
  while (x >= angle){
    x -= slowInc;
    elbowServo.write(x);
    delay(slowDelay);
  }
}

//***** BASE SLOW *****
void baseSlow(int angle){
  int x = baseServo.read();
  if (x < angle){
    while(x <= angle){
      x += slowInc;
      baseServo.write(x);
      delay(slowDelay);
    }
  } else if (x > angle){
    while (x >= angle){
      x -= slowInc;
      baseServo.write(x);
      delay(slowDelay);
    }
  }
}
## Appendix F: Component Specifications

### HP ProBook 4440s Notebook PC

#### Overview
Windows 8 Pro or other operating systems available

Your Business Partner. Optimized for Windows 8 Pro, these notebooks are ideal for SMBs. They offer multimedia tools, easy-to-use security and a sleek, vertical brushed aluminum casing. Options include two display sizes.

<table>
<thead>
<tr>
<th>System Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating system:</td>
</tr>
<tr>
<td>Processor:</td>
</tr>
<tr>
<td>Chipset:</td>
</tr>
<tr>
<td>Featured Model:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dimensions and Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weight:</td>
</tr>
<tr>
<td>Dimensions (w x d x h):</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory:</td>
</tr>
<tr>
<td>Memory slots:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal drive:</td>
</tr>
<tr>
<td>Optical drive:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Graphics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graphics:</td>
</tr>
<tr>
<td>Display:</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Communications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wireless:</td>
</tr>
</tbody>
</table>
Network interface: 10/100/1000

<table>
<thead>
<tr>
<th>Power and operating requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy efficiency: ENERGY STAR® qualified; EPEAT® registered</td>
</tr>
<tr>
<td>Power supply: 65W Smart AC adapter; HP Fast Charge</td>
</tr>
<tr>
<td>Battery type: 6-cell (47 WHr) Li-Ion</td>
</tr>
<tr>
<td>Battery life: Up to 7 hours 30 minutes</td>
</tr>
</tbody>
</table>

(Hewlett-Packard, 2012).

**Arduino 2560 Mega Microcontroller**

**Overview**

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the ArduinoDuemilanove or Diecimila.

Stronger RESET circuit.

Atmega 16U2 replace the 8U2.

Schematic, Reference Design & Pin Mapping
EAGLE files: arduino-mega2560_R3-reference-design.zip
Schematic: arduino-mega2560_R3-schematic.pdf
Pin Mapping: PinMap2560 page

**Summary**

Microcontroller    ATmega2560
Operating Voltage  5V
Input Voltage (recommended) 7-12V
Input Voltage (limits) 6-20V
Digital I/O Pins 54 (of which 15 provide PWM output)
Analog Input Pins 16
DC Current per I/O Pin 40 mA
DC Current for 3.3V Pin  50 mA
Flash Memory  256 KB of which 8 KB used by bootloader
SRAM  8 KB
EEPROM  4 KB
Clock Speed  16 MHz

**Power**

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

VIN. The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.

5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.

3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.

GND. Ground pins.
**Memory**

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

**Input and Output**

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial:** 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX). Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega16U2 USB-to-TTL Serial chip.
- **External Interrupts:** 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2). These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM:** 2 to 13 and 44 to 46. Provide 8-bit PWM output with the analogWrite() function.
- **SPI:** 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS). These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Due, and Diecimila.
- **LED:** 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it’s off.
- **TWI:** 20 (SDA) and 21 (SCL). Support TWI communication using the Wire library. Note that these pins are not in the same location as the TWI pins on the Due or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it...
possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

AREF. Reference voltage for the analog inputs. Used with analogReference().

Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

**Communication**

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega16U2 (ATmega 8U2 on the revision 1 and revision 2 boards) on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2/ATmega16U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports TWI and SPI communication. The Arduino software includes a Wire library to simplify use of the TWI bus; see the documentation for details. For SPI communication, use the SPI library.

**Programming**

The Arduino Mega can be programmed with the Arduino software (download). For details, see the reference and tutorials.

(Mellis, 2011)

**ROB-10332 Robotic Claw**

This robotic claw arm is great for all your gripping needs. They are made from metal and are pretty heavy-duty. The claw opens to about 2" and depending on the servo
motor used, it can pick up some relatively heavy objects. Because the arms move parallel to each other, you get a better grip.

These also have a mounting plate on the bottom which accepts standard spacing found on servo mounts (the extra bits that come with our servo motors). These do not come with a servo motor, so check below, the ‘medium servo’ is the one that works well.

(SparkFun Electronics).

**HS-422 Standard Servo**

**HS-422 Deluxe Standard Servo**

---

**Specifications**

<table>
<thead>
<tr>
<th>Motor Type:</th>
<th>3 Pole</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bearing Type:</td>
<td>Dual Cylindrical</td>
</tr>
<tr>
<td>Speed (4.8V/6.0V):</td>
<td>0.21 / 0.16 sec @ 60 deg.</td>
</tr>
<tr>
<td>Torque oz-in. (4.8V/6.0V):</td>
<td>46 / 57</td>
</tr>
<tr>
<td>Torque kg-cm. (4.8V/6.0V):</td>
<td>3.3 / 4.1</td>
</tr>
<tr>
<td>Size in Inches:</td>
<td>1.59 x 0.77 x 1.44</td>
</tr>
<tr>
<td>Size in Millimeters:</td>
<td>40.39 x 19.56 x 36.58</td>
</tr>
<tr>
<td>Weight ounces:</td>
<td>1.60</td>
</tr>
<tr>
<td>Weight grams:</td>
<td>45.36</td>
</tr>
</tbody>
</table>

(HiTec, HS-422 Deluxe Servo Standard, 2012).
HS-485HB Deluxe HD Ball Bearing Servo

GoPro HD Naked Hero Camera

Wearable and gear-mountable the GoPro HD NAKED HERO Camera is waterproof to 197', captures professional 170-degree wide angle 720p video and 127-degree semi-wide angle 1080p video plus 5 megapixel photos and has earned a place in history. Whether you're new to GoPro and want the most affordable way to get started or you're looking for a second GoPro to capture your adventures from additional perspectives, the HD HERO Naked is a world famous camera at an incredibly entry-level price.
GoPro HD NAKED HERO Camera:

- Full HD resolution
- 5MP still images
- Takes pictures at 2/5/10/30/60 second intervals or single shot, triple shot or self timer
- Audio excellence
- Rechargeable 1100mAh lithium-ion battery with 2.5 h battery life
- Integrated battery heating system
- Shockproof, waterproof
- Includes 2 adhesive mounts

(GoPro HD NAKED HERO Camera Value Bundle with 16GB SD Card, 2012).

Appendix G: Prototype Expenses

<table>
<thead>
<tr>
<th>Table 6 Cost of Electronics</th>
</tr>
</thead>
<tbody>
<tr>
<td>QTY Purchased</td>
</tr>
<tr>
<td><strong>Electronics</strong></td>
</tr>
<tr>
<td>1000mAh 11.1v Batteries</td>
</tr>
<tr>
<td>2s 7.4v lipo</td>
</tr>
<tr>
<td>3s 11.1 V lipo</td>
</tr>
<tr>
<td>50:1 drive motor</td>
</tr>
<tr>
<td>80W lipo charger</td>
</tr>
<tr>
<td>Battery Charger</td>
</tr>
<tr>
<td>BEC switch regulator</td>
</tr>
<tr>
<td>breakaway headers</td>
</tr>
<tr>
<td>Breakoutboard for Xbee</td>
</tr>
<tr>
<td>drive motors 100:1</td>
</tr>
<tr>
<td>Female headers</td>
</tr>
<tr>
<td>lipo cell voltage checker</td>
</tr>
<tr>
<td>round red button</td>
</tr>
<tr>
<td>Servos</td>
</tr>
<tr>
<td>spst push on and off</td>
</tr>
<tr>
<td>toggle switch</td>
</tr>
<tr>
<td>Vide RX/TX</td>
</tr>
<tr>
<td>Video Converter</td>
</tr>
<tr>
<td>xbee explorer</td>
</tr>
<tr>
<td>Xbee RF module</td>
</tr>
<tr>
<td><strong>Hardware/ Misc.</strong></td>
</tr>
</tbody>
</table>
## Table 7 Cost of Hardware

<table>
<thead>
<tr>
<th></th>
<th>QTY Purchased</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Electronics</strong></td>
<td>44</td>
<td>$782.51</td>
</tr>
<tr>
<td><strong>Hardware/ Misc.</strong></td>
<td>53</td>
<td>$207.79</td>
</tr>
<tr>
<td>#4 flat head wood screws</td>
<td>1</td>
<td>$1.18</td>
</tr>
<tr>
<td>#8 washer</td>
<td>1</td>
<td>$1.18</td>
</tr>
<tr>
<td>1/4 cap screw</td>
<td>2</td>
<td>$1.64</td>
</tr>
<tr>
<td>10-32 cap screws</td>
<td>4</td>
<td>$4.26</td>
</tr>
<tr>
<td>10-32 flat head screws</td>
<td>1</td>
<td>$1.18</td>
</tr>
<tr>
<td>10-32 nuts</td>
<td>3</td>
<td>$3.54</td>
</tr>
<tr>
<td>18awg wire</td>
<td>1</td>
<td>$4.49</td>
</tr>
<tr>
<td>18g wire</td>
<td>1</td>
<td>$4.49</td>
</tr>
<tr>
<td>5/16 coupling</td>
<td>1</td>
<td>$1.40</td>
</tr>
<tr>
<td>5/16 washer</td>
<td>2</td>
<td>$2.36</td>
</tr>
<tr>
<td>8-32 cap screws</td>
<td>2</td>
<td>$1.84</td>
</tr>
<tr>
<td>8-32 nut</td>
<td>1</td>
<td>$1.18</td>
</tr>
<tr>
<td>Arm bracket</td>
<td>4</td>
<td>$40.00</td>
</tr>
<tr>
<td>ball caster set</td>
<td>1</td>
<td>$5.99</td>
</tr>
<tr>
<td>base rotate kit</td>
<td>1</td>
<td>$40.00</td>
</tr>
<tr>
<td>deans plug</td>
<td>1</td>
<td>$3.25</td>
</tr>
<tr>
<td>female deans plug</td>
<td>1</td>
<td>$2.75</td>
</tr>
<tr>
<td>flex tubing</td>
<td>1</td>
<td>$2.48</td>
</tr>
<tr>
<td>heat shrink</td>
<td>1</td>
<td>$2.39</td>
</tr>
<tr>
<td>Lipo Guard</td>
<td>1</td>
<td>$5.86</td>
</tr>
<tr>
<td>male deans plug</td>
<td>1</td>
<td>$3.25</td>
</tr>
<tr>
<td>male deans plugs</td>
<td>2</td>
<td>$7.50</td>
</tr>
<tr>
<td>nut</td>
<td>1</td>
<td>$1.18</td>
</tr>
<tr>
<td>nylon spacer</td>
<td>9</td>
<td>$4.22</td>
</tr>
<tr>
<td>plastic clamp</td>
<td>2</td>
<td>$3.96</td>
</tr>
<tr>
<td>project box</td>
<td>1</td>
<td>$5.49</td>
</tr>
<tr>
<td>Punched Steel plate</td>
<td>1</td>
<td>$4.88</td>
</tr>
<tr>
<td>sticker</td>
<td>4</td>
<td>$3.20</td>
</tr>
<tr>
<td>Wheels and Hubs</td>
<td>1</td>
<td>$42.65</td>
</tr>
</tbody>
</table>

## Table 8 Cost of Administrative Expenses

<table>
<thead>
<tr>
<th></th>
<th>QTY Purchased</th>
<th>Total Price</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Electronics</strong></td>
<td>44</td>
<td>$782.51</td>
</tr>
<tr>
<td><strong>Hardware/ Misc.</strong></td>
<td>53</td>
<td>$207.79</td>
</tr>
<tr>
<td><strong>Administrative</strong></td>
<td>3</td>
<td>$245.00</td>
</tr>
<tr>
<td>poster</td>
<td>1</td>
<td>$130.00</td>
</tr>
<tr>
<td>Registration fee</td>
<td>1</td>
<td>$65.00</td>
</tr>
<tr>
<td>Printing</td>
<td>1</td>
<td>$50.00</td>
</tr>
</tbody>
</table>