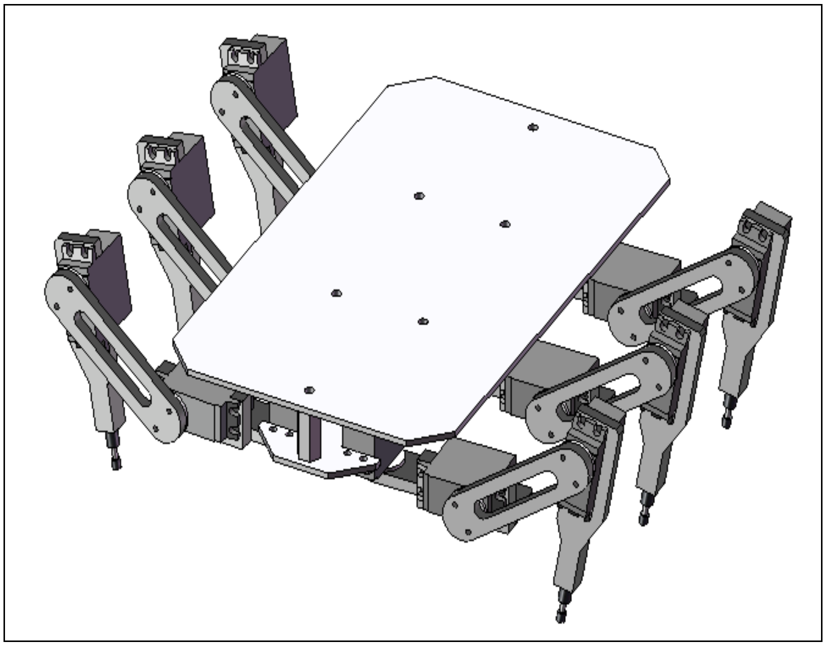
Hexapod Robot



Dalhousie Mechanical Engineering
Senior Year Design Team 2

To
Dalhousie University
Mechanical Engineering Department

November 15, 2008
Rene d'Entremont
Brett MacDonald
Leslie Ssebazza
Seth Stoddart

Abstract

The aim of the project is to build a six-legged walking robot that is capable of basic mobility tasks such as walking forward, backward, rotating in place and raising or lowering the body height. The legs will be of a modular design and will have three degrees of freedom each. This robot will serve as a platform onto which additional sensory components could be added, or which could be programmed to perform increasingly complex motions. This report discusses the components that make up our final design.

Our group has completed the initial drawings for the design, forecasted building costs and created a schedule for the remainder of the project. All these items are included in this report. With the preliminary design of all components finished, we are prepared to start the construction of a mock up leg in order to begin testing the hexapod control software. The completed budget totals $2,320.29 and 24.5 hour of technician time. Team 2 has also begun writing the programs necessary to control the hexapod robot.

Table of Contents

Contents of Appendix

List of Figures

List of Tables

1 Introduction

Team 2 has done research into the elements needed to build a functional hexapod robot in the coming semester. Groundwork includes investigation and design of the two major subsystems; chassis and controls. The group plans to produce a fully functioning robot during the "build" phase of the Design Project next semester.

1.1 Practicality and Complexity of Construction

The design process used in developing the parts of the hexapod considers the level of difficulty associated with building the different sections of the hexapod robot. Cost, complexity and functionality were thoroughly considered during the design in order to keep the built product practical. The scope and design for the project has been scaled to fit the budget that is outlined in the budget section of this report.

The design uses common, available materials for the machined components along with purchased parts such as electronics and robotics available from supply stores. It is an innovative and practical design, which improves on previous hexapod designs. By using shapes that can be easily manufactured we have decreased the machining time needed. Simple fasteners such as glue/epoxy and standard sized screws/bolts will be used to assemble the finished product.

1.2 Technician Consultation

The technician has indicated that the part drawings can be produced in the desired materials. Design changes recommended by technician have been considered and implemented.

2 Progress

A summary of progress up to and including Wednesday, November 12, 2008:

2.1 Chassis

Design of the chassis components are complete and are shown in Appendix E. The drawings available now will be used to build the preliminary mock-up of a single leg of the robot using the rapid prototyping machine. On the assumption that no design flaws appear during testing, the final pieces can be fabricated shortly thereafter.

2.2 Hardware

Servo motors will provide the moving force for this robot. Four potential servos have been tested to determine their suitability. Weight, torque, and power consumption have been measured and compared. The ideal servo motor for our application is readily available and has been selected to be sourced.

To set servo pulse width for positioning, a Devantech SD-21 has been procured and tested. This board is able to control all 18 servos and has been successfully interfaced with the preferred microcontroller. The selected micro controller, an Arduino Decimilla board has been procured and tested. This board has been connected successfully to Matlab using a serial connection, as well as to the Devantech SD-21 using I2C ports.

2.3 Software

The necessary electronic components have already been procured for proof of concept testing. Simple, single servo control has been achieved using Matlab. Work on the inverse kinematic program needed to control the robot leg has started. The aim of the program is to use a Jacobian Inverse matrix method to define angles that can then be used to control the leg positions.

Angle positioning testing has been performed in order to map the pulse width vs. angle of the selected servo. A Matlab function has been written in order to convert angle output to pulse width high and low bytes.

3 Component Details

A brief explanation of the robots key components is provided below.

3.1 Chassis

3.1.1 Frame

The frame of the robot consists of two 1/8” aluminum plates that are bolted together using six commercially available standoffs, as illustrated in Figure 1. The smaller plate is used for mounting the legs onto while the larger plate is used for mounting electronic equipment such as the microcontroller boards. As well, there is extra space on the large plate so additional electronic equipment and sensors could be added to the robot in the future, in accordance with the design requirements.

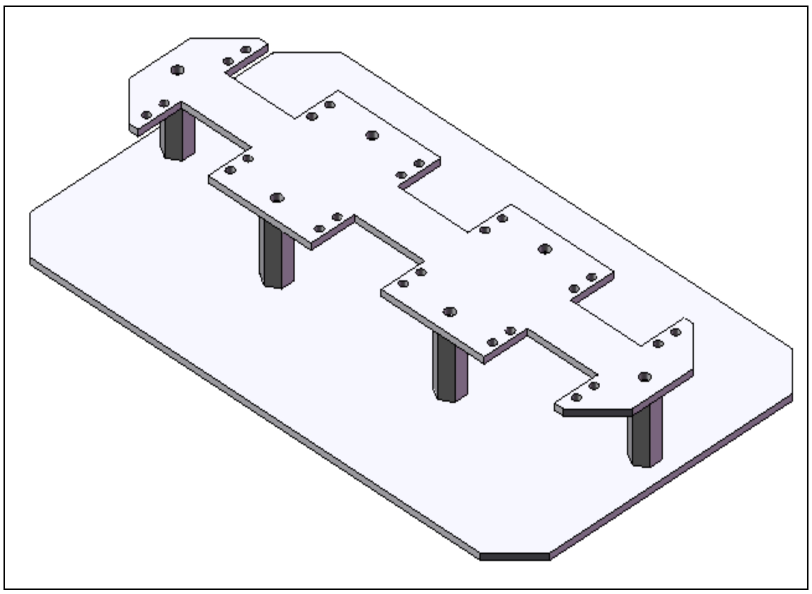


Figure 1: Bottom View of Frame

3.1.2 Legs

Each of the six legs will contain three servomotors that are connected by three leg linkages as shown in Figure 2. There are also small shock absorbers (commercially available) that thread into the outer leg piece. Each leg section will be machined from 1/4” or 1/2” PVC (or ABS) plate. The inner leg section will require two pieces to be separately machined then bonded together. The type of bonding agent has not been finalized yet. Also, the inner and outer leg sections will come in a left and right handed variety while the middle leg section will be the same on both sides. The leg sections will be bolted to the servomotors using small (approx. M4) bolts at the servo mounting flanges and at the servos mounting disk (attached to output shaft).

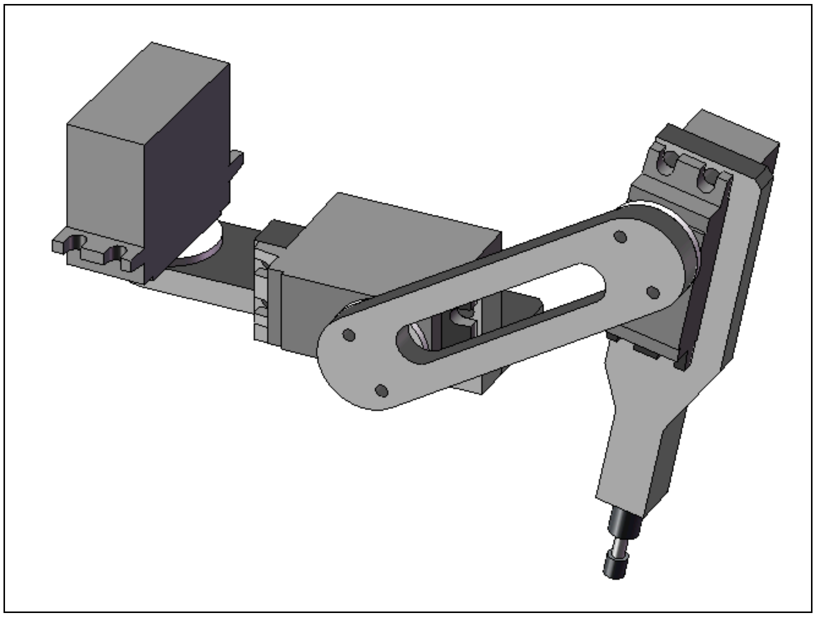


Figure 2: Leg Assembly, Consisting of three servomotors, three leg linkages, and a Shock Absorber.

3.2 Hardware

Electronic hardware used in this robot will consist of 18 HiTec brand servo motors to provide the 18 degrees of freedom required. Four suitable high torque servo motors have been tested to determine the best balance of weight, size, torque, current drawn and price. The current drawn proved to be the limiting factor. During testing it was observed that larger servos (see Table 1) used up to 2 amps of current each when running under no load while the smaller servos used only .75 amps at stall torque.

To minimize the current drawn by the robot, the lowest torque motor suitable for the task had to be chosen based on the findings of our testing which are summarized in Table 1. For the requirements of our design we chose the HS-645MG servo motor.

Table 1: Summary of Servo Motor Testing

Servo Name	Torque (kg*cm)	Weight (g)	Power Consumption
HS-805BB Giant Scale	24.7	152	1.7A No Load
HS-765HB "Sail Arm" Servo Motor	13.2	110	1A No Load
HS-645MG Servo Motor	9.6	55.2	0.75A Stall torque
GWS Heavy Duty S777 6BB Servo Motor	42	190	2A No Load

The servo control board selected for our design is the Devantech SD21 Servo Controller, pictured in Figure 3. This board is cheap and is able to control up to 21 servo motors. It is a simple board used to send and receive data via the I2C connection so as to allow the robot to be controlled by a series of integer numbers stored in its onboard array. The board can then generate the appropriate pulse widths required based on these input numbers.

To handle all onboard processing of the serial data output by the controlling computer, an Arduino Decimilla microcontroller is used (Figure 3). This board is relatively cheap and features USB connectivity and a built in I2C library. The available I2C port is connected to the Devantech board through a 4 pin header. The Decimilla is able to receive input from analog or digital sensors, can power the logic portion of the servo control board, and can communicate with the computer using its serial converter for ease of debugging. This board will be suitable for our robot control as well as any future iterations of the robot. Both boards are to be mounted on the upper plate of the body assembly


Figure 3: Devantech SD21 Servo controller (right) and Arduino Decimilla Microcontroller (left)

3.3 Software

Programming is done using Matlab and the compiler supplied for the Arduino board. Both software programs use simple programming language similar to C. Software carried onboard of the robot on the Arduino board will be used to interpret the serial communication from Matlab, and send the appropriate data to the servo control board. The current version of the software onboard the Arduino reads in an integer number from Matlab, breaks it into integer representations of its hexadecimal low and high bytes, and then sends it to the appropriate address in the Devantech in order to set the pulse width for a desired servo position. The pulse width high and low bytes are sent with a speed and servo call integer in order to select the appropriate servo and travel rate.

The Matlab portion of the software will handle the intensive positioning control. A Jacobian matrix will be used in order to calculate the positions of the servos to provide different leg tip positions. A leg travel path is drawn from these positions to achieve the desired motion. The data will be stored in a matrix of leg positions which will be read in by the Arduino and continuously updated on the Devantech.

4 Unknowns

List of Unknowns:

- The primary walking gait has been selected (tripod); however, programming of that gait has not yet been designed. Given programming success to this point, this unknown is deemed acceptable and will be an extension of current programming. A complete chassis will be needed to refine the gait.
- It has not been decided what bonding agent will be used to join the two parts of the inner leg section together.
- The approximate sizes of all fasteners are known, but the exact size, length and style of fastener are yet to be determined.

5 Work Allotment

All fabrication work will be completed by the department. This is limited to the cutting and machining of parts. This will ensure quality fabrication of components. An estimate of the machining time required from the department for this project is provided in Table 2.

Table 2: Estimated Machining Time Required from Department

Part Number	Part Description	Quantity	Total Machining Time (hrs)
HX-0012	Top Plate	1	1.5
HX-0013	Bottom Plate	1	4
HX-0022	First Leg Section	7	6
HX-0023	Second Leg Section	7	3
HX-0024	Third Leg Section	7	10
			24.5

All assembly work will be completed by Team 2. Assembly requires the use of simple hand tools and bonding agents. Advice may be sought from the technicians during the assembly process.

All programming and electronics work will be completed by the team.

6 Budget

The team has prepared a budget for the entire project. The detailed budget is provided in Appendix B and a budget summary is provided in Table 3. The most expensive component of the design is the servo motors since eighteen motors (plus spares) are required.

Table 3: Budget Summary. Taxes and Shipping Included.

Component	Cost
Servo Motors	$1,173.20
Electronics	$399.76
Raw Materials	$262.29
Purchased Parts	$485.04
Grand Total	$2,320.29

Appendix A
Gantt Chart for the Winter Term

Appendix B
Project Budget – Hexapod Robot

Materials	Unit cost	Amount	Cost	Supplier
Electronics				
HS-645MG Servos	$39.02	21	$819.42	robotshop
HS-765HB	$43.78	1	$43.78	robotshop
HS-805BB	$43.36	1	$43.36	robotshop
25' 22g black wire	$2.62	1	$2.62	robotshop
25' 22g red wire	$2.62	1	$2.62	robotshop
Netmedia 6" jumper Wire kit	$8.49	1	$8.49	robotshop
Eneloc 30 pc. Reinforced Jumper wire kit	$18.91	1	$18.91	robotshop
Pulse Witdh Modulator	$63.15	2	$126.30	robotshop
USB Cable	$2.99	1	$2.99	robotshop
Microcontroller	$40.64	1	$40.64	robotshop
Resistors (500 ohm + 2000 ohm)	$1.00	2	$2.00	robotshop
Potentiometer (500ohm)	$7.85	1	$7.85	robotshop
Rocker Switch	$1.39	1	$1.39	robotshop
Protoboard	$15.00	1	$15.00	robotshop
Sensors	$75.00	1	$75.00	robotshop
Raw Materials				
3/8" Hex Standoff 1/8"PL	$1.00	12	$12.00	mcmaster carr
1/4" ABS	$13.29	3	$39.87	mcmaster carr
1/2" ABS	$27.39	2	$54.78	mcmaster carr
1/8" Alluminum Plate 6061	$26.06	3	$78.18	mcmaster carr
Plastic Bonder	$24.99	1	$24.99	mcmaster carr
1-1/4" AL Hex Standoff 10-32 screw	$3.03	6	$18.18	mcmaster carr
Leg Shock Absorbers	$28.13	7	$196.91	mcmaster carr
Fastners	$20.00	1	$20.00	mcmaster carr
Rapid prototyping	$7.00	3	$21.00	In House
Overhead Boom	$99.95	1	$99.95	

Subtotal	**$1,776.23**
10% SF	$177.62
Tax (15%)	$266.43
Shipping	$100.00
Total	**$2,320.29**

Appendix C
Matlab Code for Communication with the Microcontroller

```
%Proof of concept
%Single Servo Control
%V1 - Debugging feedback

clear; clc;
%POSITIONS SENT
   %SERVO 1
   Position1 = 1700
%LowHigh is a function written by group 2 for ease of conversion
%LowHigh reads interger numbers and splits them into low and high
%bytes.

   a = LowHigh(Position1)

%USING COM PORT 4
%opens com port 4, a usb port used to connect to the Arduino
   com = serial('COM4','BaudRate', 9600);
   fopen(com)

%CONTROL OF SERVOS 1,2,3
%for debugging, the program reads the old position in from the arduino
%to print it for comparison with the printed result of the new position
%it has achieved. The board will accept an interger value sent to it as a
%high and low byte.
  ServoVerify = fread (com,2)
  fwrite (com,Position1,'int');
  ServoVerify = fread(com,2)


%CLOSING COM PORT 4
%Close the port.
   fclose(com)
```

Appendix D
Arduino Decimilla Preliminary Code
Comment lines begin with "//"

```
#define SD21 0xc2 //SD21 I2C Address

//Initialize Serial and I2C connections

void setup()
{
  Wire.begin();  // start I2C
  Serial.begin(9600);
}

//Main Void Loop for program. This loop runs constantly

void loop()
{

//Define initial positions. The reset button on the board will return the servos to this position. Low and high bytes of 220 and 5
//represent 1500, a midpoint position on the servo. (256*5 + 220 = 1500)
//Initial Positions on Board Reset

int OneLow = 220;
int OneHigh = 5;

//Jump into second "void" style loop. This loop will run continuously until the reset button breaks the program and jumps back //to
reset the initial position. This is handy for debugging as mistakes in sending Matlab data can be cleared with a button //press. A battery
disconnect will do the same, however this is necessary now that a power supply is used.

int Start = 1;
while (Start>0)
{

//Writes the low and high bytes to be received into a 2 position matrix in Matlab.

   Serial.write (OneLow);
   Serial.write (OneHigh);

//Serial.available checks if there are any available bits in the serial buffer. If there are, they are read in through the serial.read
//commands. The buffer is then flushed to remove unknown bits sent along with the low and high bytes.

  if (Serial.available () > 0)
   {
   OneLow = Serial.read ();
   OneHigh = Serial.read ();
   Serial.flush();
   }

//Sends the fixed servo address (0 is servo 1 in the register), and the speed (0 is max speed). The high and low bytes are sent as //well.
Once the servo is established, the remaining 3 properties are automatically assigned to the correct registry position.

  Wire.beginTransmission(SD21>>1);
   Wire.send(0);
   Wire.send(0);
   Wire.send(OneLow);
   Wire.send(OneHigh);
   Wire.endTransmission();
}}
```

Appendix E
Initial Part Drawings

This appendix contains the initial part drawings for the robot. All units are in millimeters and the drawings are done using first angle projections.