ELEC 490 FINAL REPORT

CONTROL AND MONITOR OF A ROBOT EXCAVATOR

Submitted By:

Alan Lo, Ryan Murphy, Vuk Zrnic

Faculty Supervisor:

Dr. Keyvan Hashtrudi-Zaad

Executive Summary

The objective of the ELEC 490 project undertaken by group 13 was to remotely control and monitor a robot excavator. The group was successful in achieving these goals.

The excavator is controlled from a PC workstation using dual joysticks. The movements of the excavator are displayed using a real-time 3D model. A wireless link is used for data transfer between the PC and the microprocessor onboard the excavator. The excavator system is self powered, allowing autonomous operation.

A new excavator was purchased for this year's project. Sensors were mounted on the excavator to monitor track and arm movements. A microprocessor and driver board were installed inside the excavator in order to control the motors. The microprocessor was also used to read sensor data and communicate with the PC workstation.

Software was written for the microprocessor to read the sensor data and drive the motors. Software was also written for the PC workstation to interpret the relayed sensor data, take in joystick inputs, send motor commands to the HC11, and display data in a graphical user interface.

The total cost of the project was $520. A $150 increase over the original $400 budget was requested and approved. The increase was needed due to the high shipping cost of the excavator, which was procured from a UK distributor.

Table of Contents

1 Introduction

1.1 Purpose

This report summarizes the work and results of Group 13, which undertook the project "Control and Monitor of a Robot Excavator". The intended audience is project supervisor Dr. Hashtrudi-Zaad, the course instructors, and future 490 groups.

1.2 Background and Objectives

Many situations can be envisioned where an operator may wish to control an excavator from a remote location, sometimes without direct line of sight. For example, the operating environment may be toxic, temperatures may be extreme, or the region may be dangerous due to unstable soil conditions. Therefore, a need arises for a remotely controlled and monitored excavator system.

The objective for this year's project was to remotely control a commercially available toy excavator, and monitor its cabin and bucket position through a real-time GUI display. The project goals were accomplished.

This is the third year this project has been attempted and the group inherited a motor driver board (not fully functional), two encoders, and a wireless RS-232 module from last year's project. All other hardware was purchased, installed and programmed this year. In the past, groups have been successful in mounting hardware, and having individual subsystems operational. A fully functional excavator has never been accomplished.

1.3 Overview of Project Work

A new excavator was purchased this year. Sensors were mounted on the excavator, and the excavator body was disassembled for easier modifications. The hardware and sensors onboard the excavator were powered using a battery pack, enabling autonomous operation.

New HC11 code was written to drive the motors, and read and relay sensor (encoder and potentiometer) information. Software was written to receive information from the serial port,

interpret the data, and display the sensor readings to the user using a 3D model of the excavator. Software was also written which takes in joystick inputs and sends motor commands to the HC11 microprocessor.

A working prototype was demonstrated to faculty during the open house on April 7th, 2006.

2 Background and Motivation

2.1 General

This project can be divided into several subsystems: The hardware mounting and wiring; HC11 coding; PC workstation programming; and creation of a 3D model. The project software components are outlined in table 2-1.

Software Written For	Role of Software
HC11 Microprocessor	Read and interpret sensor data
	Provide logical control of motors through the driver board with data received from the user terminal
PC Workstation	Establish serial connection to the microprocessor
	Translate sensor data and relay to graphical user interface
	Read and interpret joystick data, and send to microprocessor
	Show visual representation of the movement of the excavator through a 3D model
	Display translated sensor data to user

Table 2-1: Major software problems dealt with in project.

2.2 Interface/Performance Specifications

The user controls the speed and direction of excavator motors with two joysticks. One joystick controls the tracks, and the other controls the arm. The user is able to view the excavator movements and sensor information on the GUI.

3 Design and Production Approach

3.1 Division of Labour

Section	Responsibility	Task
Hardware	Vuk Zrnic	Identify and Purchase New Excavator
		Set up Wireless Data transmission
		Test Existing Components
		Mounting of Sensors onto Excavator
		Installation of HC11 and Motor Driver Board
		Achieve Autonomous Powering
		Wire sensors, motors, and power
HC11 Software	Ryan Murphy	Read Sensors
		Communication with PC
		Drive Motors
PC Software		Communicate with HC11
		Translate Sensor Information
		Read Joystick Input
		Joystick Force Feedback
3D Model/GUI	Alan Lo	Create 2D display elements
		Create 3D model of excavator
		Create model Environment
		Develop logic for dynamics of the 3D model
Software/Hardware Integration	Vuk Zrnic / Ryan Murphy	Drive Excavator through microsprossor
		Calibrate Excavator track speeds
		Calibrate Sensor Readings
GUI Integration	Alan Lo / Ryan Murphy	Integrate terminal and OpenGl programs
		Calibrate Excavator Model Movements
Project Website	Alan Lo	Create and Maintain website

Table 3.1.1 – Division of Labour

3.2 System Architecture

To control the excavator motors, the operator uses joysticks connected to the PC terminal. Joystick signals are read by the software onboard the PC workstation, translated to motor commands, and sent to the HC11 microprocessor board via a wireless link. Both joysticks have a USB connection which makes it possible to utilize two joysticks at a time, as most computers have multiple USB ports.

The wireless link between the excavator and CPU is implemented using AIR-Cable DB9 connectors (male and female). Prior to use, the modules were configured for “wireless serial” operation, at 9600 baud. In this mode of operation, the “Air Cables” function just like a RS-232 serial connection (other possible configurations are “master-slave” and “Bluetooth”).

The female module is attached to the HC11 evaluation board and is powered by the battery pack onboard the excavator. The male module is connected to the PC.

The HC11 microprocessor controls all hardware on the excavator. Motor commands are received by the HC11, which sends the appropriate signals to the current driver board. In addition to processing the motor commands, the HC11 also monitors the sensors. Encoder inputs are monitored through Port C (the digital input port), and the analog potentiometer inputs are read by the A/D converter on Port E. When the signals are received by the HC11, they are sent to the PC workstation via the wireless link for further processing.

The "Driver Board" amplifies the waveforms produced by the HC11. Keeping the duty ratio, signals are amplified to the supply voltage (supplied by the battery pack) through the use of H-bridges. The two track motors, the boom motor and the stick motor are connected directly to the middle connector of the driver board. The driver board outputs determine the direction and speed of all the motors on the excavator.

The excavator movements are monitored using four sensors. Each track is equipped with an encoder, which supplies information regarding the displacement and velocity of the excavator. A third encoder is mounted on the base of the boom, and provides information about the boom angle. Lastly, a potentiometer is mounted on the boom/stick joint, and provides information about the angle. Together, the four sensors provide information about every moving component of the excavator, needed for the real – time 3D model.

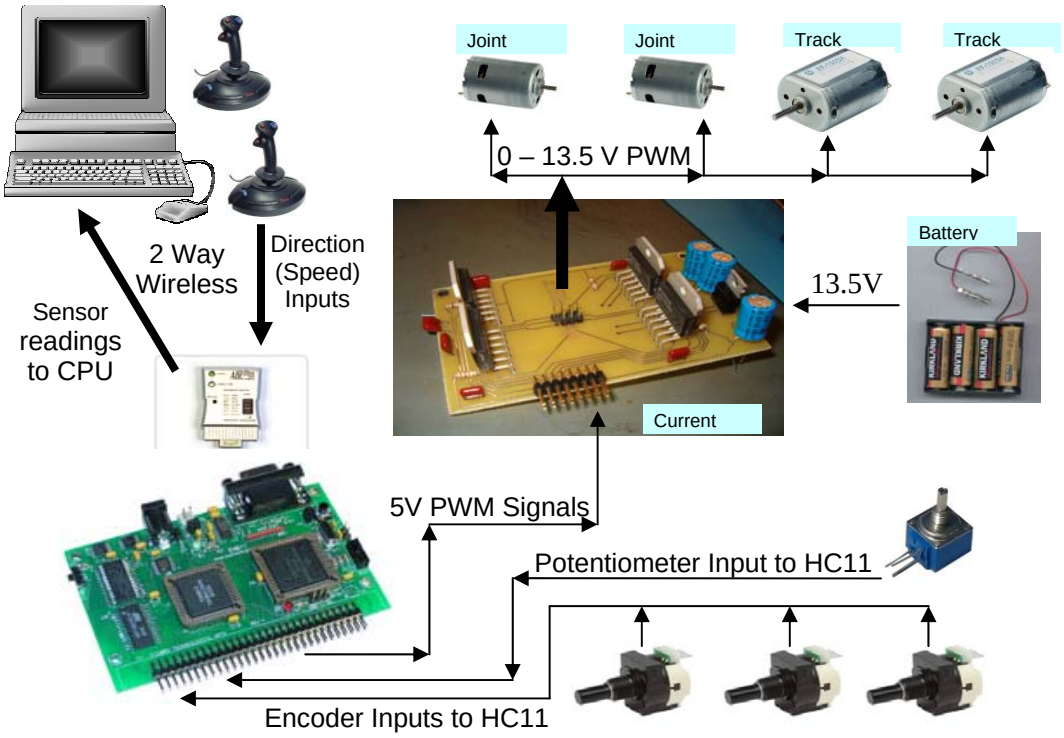


Figure 3.2.1 – System Architecture

3.3 Powering the System

The motors, sensors and all other hardware contained onboard the excavator are powered using ten rechargeable 1.35V NiMH batteries.

The EVB11 (HC11 evaluation board) contains a voltage regulator (equipped with a heat dissipating heat-sink), which brings any input voltage down to the desired 5V voltage. The 13.5 V battery pack output was brought down to 9.45 Volts before being fed to the spliced HC11 power connector.

The full (13.5V) supply voltage is delivered to the motor driver board to power the four H-Bridges.

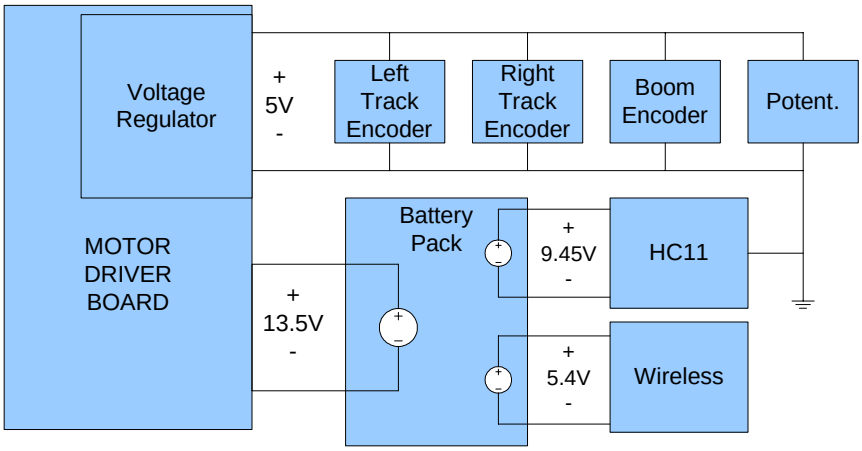


Figure 3.3.1 – Powering of System

A voltage regulator onboard the Motor Driver Board is used to produce a 5V voltage, which powers all the sensors onboard the excavator. The sensor power supplies and ground wires were soldered to a breadboard as two nodes. Those nodes were then supplied with the 5V and 0V (respectively) from the voltage regulator.

The ground node on the breadboard was also used to ground the HC11 and Wireless Air-Cable. Thus, all the systems onboard the excavator had a common ground, which enabled proper propagation of signals from the HC11 to the Motor Driver board, and from the sensors to the HC11 inputs.

3.3.1 Powering Sequence

Upon initial power-up, it was observed that the reset capability of the HC11 was lost. It was discovered the EVB11 (HC11 evaluation board) cannot be reset if the A/D converter is receiving inputs, therefore a specific power up sequence and switch was implemented to ensure proper operation of the individual components.

Firstly, the male wireless module is powered up, and plugged into the CPU. The female wireless module is powered from the battery pack and plugged into the HC11 evaluation board (onboard the excavator). Next, the EVB11 is powered from the battery pack and the program is loaded into the HC11 RAM.

The battery pack 13.5V supply is then connected to the motor driver board via a switch, also powering the encoders and potentiometers. At this point, the wireless connection is lost, because the current change in the battery pack creates a surge in the female wireless module. A simple solution to this problem is to re-power the wireless module connected to the PC workstation and allow the 2 wireless modules to re-sync. When the link is re-established, the system is ready to use.

In an industrial grade system, each of the components onboard the excavator would have a separate power supply, and appropriate interfaces, therefore the power surge problem would be eliminated.

3.4 Sensor Systems

Excavator motion is monitored using a network of sensors. Track movements are monitored using encoders. A third encoder is used to monitor the boom angle, and the boom/stick angle is monitored using a potentiometer.

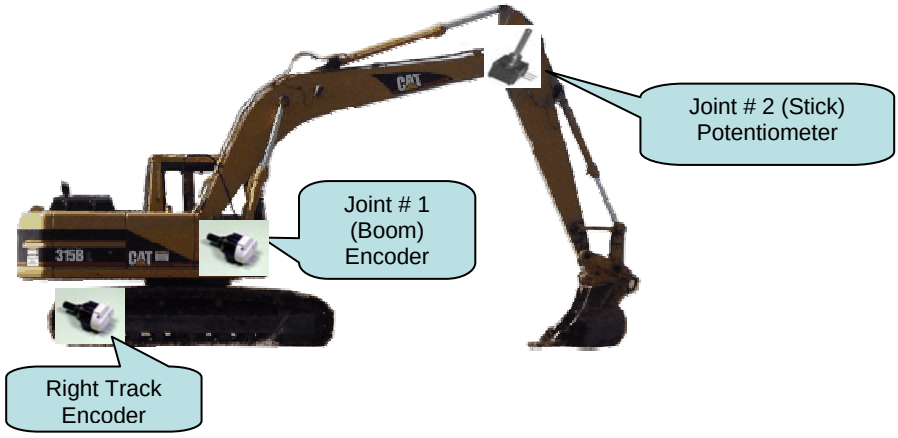


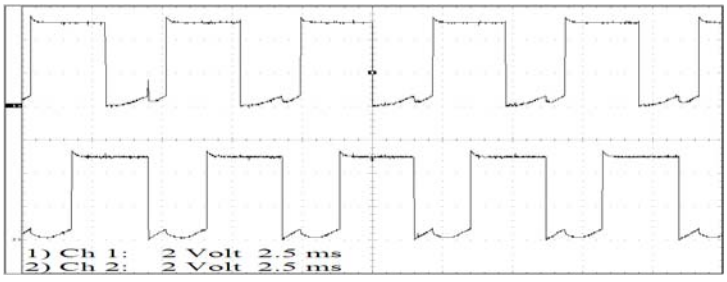
Figure 3.4.1 – Placement of Sensors

3.4.1 Encoders

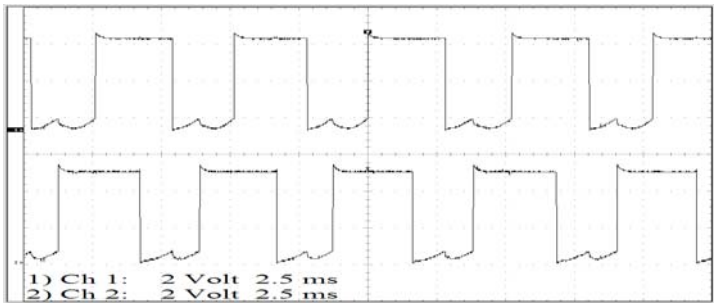
There are three encoders onboard the excavator, one on each track, and an encoder to monitor the changes in boom angle.

An encoder has 4 pins. Pin 1 receives a 5V supply voltage, and pin 4 is the grounding pin. The two middle pins (pin 2 and pin 3) contain information about the speed and direction of movement. Pin 2 outputs waveform A, and pin 3 produces waveform B. In a properly functioning encoder, waveforms A and B are square waves (see Figure 3.4.2). In a dormant state, both A and B are logic lows (about 0V). When the encoder shaft starts rotating, A or B will toggle to a logic high. Which waveform toggles to a logic high first depends on the direction of rotation. A phase shift of 45 degrees is always present between waveforms A and B.

It should be noted that different encoders produce pulses at varying frequencies. Thus, the left track encoder and the right track encoder did not produce the same amount of pulses for one revolution of the track wheel. This discrepancy was compensated for in the software contained on the PC workstation.



Forward Pulse



Reverse Pulse

Figure 3.4.2 – Encoder Waveforms

3.4.1.1 The Problem of Deteriorating Encoder Outputs

After extensive use, the encoders left over from last year started to deteriorate and the outputs were no longer square waves. The outputs had a quick drop in voltage during the high period.

This caused the HC11 to see a logic low when it should not, and in turn treat this dip as an extra pulse in the wrong direction.

Since installing new encoders would be costly and too time consuming, the problem was fixed in the PC software. The correct number of pulses could be taken as half of the number determined by the microprocessor, and the direction was determined using the current joystick direction.

3.4.2 The Potentiometer

A potentiometer is a device which produces a variable voltage output, depending on the position of the shaft. Pin 1 is a 5V input, pin 3 is the grounding pin, and pin 2 produces a variable voltage output. The only limitation of the potentiometer is that the range of motion is limited to 270 degrees. This was not a problem as the stick/boom angle only varied by 50 degrees.

The analog to digital converter on the HC11 is an 8 bit converter, yielding 256 possible values. For a 270 degree range of motion, it is possible to obtain a resolution of 1.05 degrees (270 degrees / 256 values).

3.4.3 Mounting of Sensors

When mounting the track encoders, great care was taken to ensure that the hole into which the encoder shaft was inserted was drilled in the very middle of the track wheel. This was accomplished by connecting the opposing ridges of the wheel with pencil, and drilling though the intersection of all the lines (refer to APPENDIX E). The diameter of the drill bit was chosen to be slightly larger than that of the encoder shaft in order to provide enough space for the epoxy. To ensure that the encoder was in the centermost position with respect to the wheel, the tracks were rotated continuously while the epoxy was setting, ensuring that the final encoder position would be the most natural one. A plastic bracket was attached to the encoder, and track frame, in order to prevent the encoder body from rotating when the excavator was in motion, thus forcing the shaft to rotate with any track movements.

The boom encoder proved to be the most difficult sensor to mount because the boom axis (the shaft about which the boom rotates) was not easily accessible. To achieve proper operation, it is imperative that the encoder be mounted directly on the rotation axle. A cylindrical hole was drilled in the side of the excavator, exposing the boom motor, and a non-protruding, geared-down axle only 4mm thick. A problem arose of how to attach this very thin, non-protruding axle to the encoder shaft. To solve this problem, a 2mm hole was drilled in the centre of the encoder shaft, as well as through the boom axle. A 1.5mm thick copper wire, coated with epoxy was then inserted into the axle from one end, and the encoder shaft from the other, and left to settle. When dried, the copper wire provided a physical bond between the encoder and the boom gear axle. An aluminum bracket was then created to restrict the movement of the encoder body, forcing the shaft of the encoder to rotate when the boom was raised or lowered (please refer to Appendix E).

The final sensor mounted onto the excavator was the potentiometer, located on the boom-stick joint. The potentiometer shaft was epoxied directly to the axis of rotation. A plastic bracket was attached to the potentiometer from one end, and bolted to the excavator stick from the other, ensuring that with any rotation, the potentiometer body would rotate with respect to the potentiometer shaft. The potentiometer was mounted so that in a fully retracted position, the output is Vmax = 5V, and as the stick is extended, the voltage output decreases linearly. The relationship between the potentiometer voltage and the boom/stick angle is included in Appendix H.

In mounting the track encoders and stick/boom potentiometer, “nylox” nuts were used to secure the bolts to the excavator. These nuts have a nylon coating on the inside of the nut, making them immune to vibrations of the excavator.

3.5 The Microprocessor

A Motorola 68HC11A (HC11) was mounted onboard the excavator to relay sensor signals to the PC, accept motor commands from the PC, and send commands to the motor drivers.

3.5.1 Sensor Readings

The encoder signals are read by continuously polling the 6 appropriate digital input pins, two of which belong to each encoder as mentioned in section 3.4.1. The programming logic used to determine when a pulse occurs is to compare the waveform values with the previous values. A pulse gets recorded only when the previous values were both logic lows and the current values are a logic low and logic high. Direction is determined by which pin is high.

3.5.2 Motor Outputs

The HC11 controls the motors by sending PWM duty ratio, direction, and a brake signals to the H-bridges. This is discussed in detail in section 3.6.1.

The PWM signals are created using the HC11's timer interrupt system. During each interrupt, the outputs on Port A are toggled high or low, and the number of clock cycles until the next interrupt is set depending on the desired duty ratio. Since the clock operates at a 2MHz frequency, a 200Hz PWM signal is created using a period of 10,000 clock cycles.

3.6 The Motor Driver Board

The driver board is populated with 4 H-bridges necessary for motor control, input and output connectors, a 5V voltage regulator as well as capacitors to smooth voltage spikes. Refer to Appendix B for the driver board pin configuration and.

3.6.1 The H-bridges

The H-bridges on the motor driver board amplify the signals from the microprocessor to levels required to drive the excavator motors. Refer to Appendix D for the H-Bridge pin configuration.

Three signals are required to control the operation of an H-bridge. The PWM INPUT (pin 5) takes a PWM signal from the HC11. It is through this pin that the duty ratio of the output waveform is determined. The DIRECTION INPUT (pin 3) determines the direction of the motor. A logic high (5V) indicates forward movement and a logic low (0V) reverses the

direction of the motor. The BRAKE INPUT (pin 4) is the third control signal. A logic low (0V) allows normal operation and a logic high (5V) stops the motor.

The H-bridge produces two output signals, OUTPUT 1 (pin 2) and OUTPUT 2 (pin 10). The difference of these signals (OUTPUT 1 – OUTPUT 2) is the final output that drives the motor (please refer to Appendix C)

3.7 PC Terminal Software

The PC software was written in C++ using the Visual Studio development environment. The 3D model was created using the OpenGL library. The 2D graphical user interface (GUI) was created using a Microsoft Foundation Classes (MFC) dialog. The joystick inputs were read using the Simple DirectMedia Layer (SDL) library, and the serial communication used the Microsoft Communications ActiveX Control. Microsoft Visual Studio 6.0 is available on all computers in the ILC laboratory.

3.7.1 OpenGL 3D Model

OpenGL was selected to create the 3D model due to its ease of use as an environment to create both 2D and 3D applications. It is also open source, thus lowering the overall cost of the project.

3.7.1.1 Creating the 3D Model of the Excavator

OpenGL renders 3D objects through the creation of polygons in space. The glBegin(GL_POLYGON) -> glEnd() construct was used to create a polygon in 3D space. Vertices for the polygon are defined in between glBegin and glEnd. An example of the creation of a four sided polygon in 3D space is shown in Figure 3.7.1.

```
glBegin(GL_POLYGON);
        glVertex3f(1.5, -2.3, -1.5);
        glVertex3f(0.8, -2.3, -1.5);
        glVertex3f(0.8, -2.3, 2.0);
        glVertex3f(1.5, -2.3, 2.0);
glEnd();
```

Figure 3.7.1 – Creating a four sided polygon in 3D space.

For simplicity, the model was created with minimal detail, as manually creating a large number of polygons becomes difficult. The 3D model is shown below in figure 3.7.2.

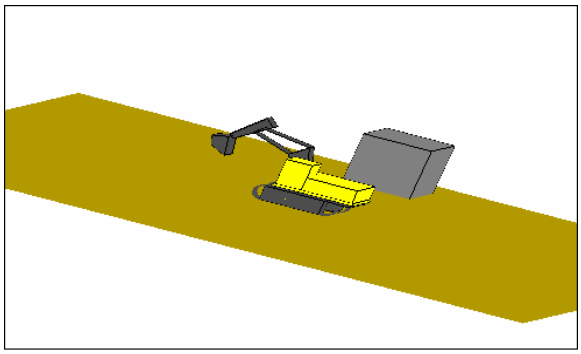


Figure 3.7.2 – 3D rendering.

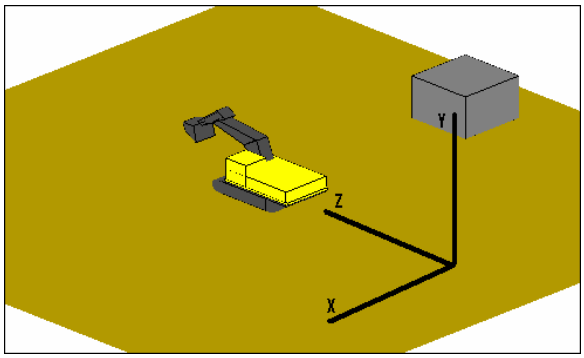


Figure 3.7.3 – Axes for the 3D model.

3.7.1.2 Moving the 3D Model of the Excavator

The difficulty in creating the 3D model was providing movement to excavator as well as the individual parts of the arm. Movement in OpenGL is performed by transformations being applied to individual vertices of the polygons. The use of glTranslate() and glRotate() is used to translate and rotate vertices respectively. As the model became more complex with the increasing number of polygons, applying individual transformations to the vertices became difficult due to the fact that not only could the excavator move as a whole, but the arm joints could move independently as well.

A system was devised which uses a hierarchical model to provide order to the transformations. The excavator was broken down into parts that could move independently of each other, and their respective relationships were defined. The relationships are shown in Table 3.7.1, based on the axes shown in Figure 3.7.3.

Part	Possible Movement	Relationships
Bucket	Rotation around the bucket-stick joint in the YZ plane.	None.
Stick	Rotation around the stick-boom joint in the YZ plane.	Rotating the stick also rotates the boom.
Boom	Rotation around the boom-body joint in the YZ plane.	Rotating the boom also rotates the stick and the bucket.
Body	Translation in the ZX plane.	Translation of the body also

		results in translation of boom, stick and bucket.
	Rotation about the Y axis.	Rotation of the body also results in rotation of boom, stick and bucket.

Table 3.7.1 – Relationships between the parts of the 3D model.

A hierarchical tree of the 3D model was built and is shown in figure 3.7.4. The tree defines the hierarchical relationship between the various parts. Movement of a parent in the tree moves all of its children. The excavator components were encapsulated in their own individual draw methods in the software. Each method could be called individually which would then draw the individual part.

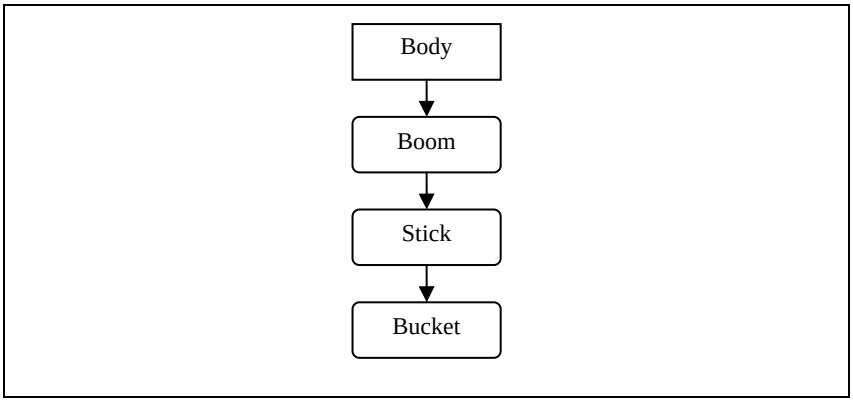


Figure 3.7.4 – Hierarchical tree of the 3D model.

Each component was drawn with its pivot point at the origin in 3D space (0, 0, 0). Placing the pivot point at the origin allows for easy rotation of the object without involving any translation of the object. The glRotate() function only rotates about the origin, therefore rotating an object which is not at the origin requires an additional translation to get the rotation point for the object to the origin, and another rotation to move it back to the original position.

The 3D model was built using OpenGL's stack-based principle. The excavator body was placed at the bottom of the stack. The rest of the parts (boom, stick, bucket) were then pushed onto the stack. At each push, a snapshot of the cursor position is captured, with the cursor position being at the origin if no transformations have occurred. As each new element is

pushed onto the stack, the cursor position is changed relative to the cursor position of the part below. Thus the stick is defined relative to the position of the boom, which in turn is defined relative to the position of the excavator body.

Separate methods were built and placed accordingly in the stack to move the individual parts of the excavator. Since they were placed in the stack along with where the parts were drawn, the hierarchy was preserved.

During the first build of the 3D model, there was no hardware integration, therefore there was no sensor data being sent to the model. Keyboard inputs were used to ensure that the excavator moved properly, and that all components moved properly relative to each other.

3.7.2 Graphical User Interface

In addition to the 3D model, additional data needed to be displayed for the operator. This includes track displacement and velocity, and the boom, stick and bucket angles.

A Microsoft Foundation Class (MFC) dialog was used to create the graphical user interface (GUI). Microsoft Visual Studio IDE provides an easy to use interface for dialog creation where the programmers can drag-and-drop the desired controls such as text fields, and buttons.

The 3D model was incorporated into the MFC dialog by creating a custom MFC control derived from a standard text window. This method allows the model to be easily sized and positioned in the drag-and-drop interface by simply positioning a standard window. It also allowed the use of the standard MFC events such as OnCreate, OnPaint and most importantly the OnTimer event for refreshing the display. The refresh timer can be set arbitrarily when the 3D model object is created.

3.7.3 Joystick Inputs

Using the Microsoft API to read joystick inputs is too complicated, so a library class needed to be used.

The initial library chosen was DirectInput from the DirectX SDK. DirectInput was chosen for its support of force feedback since an initial project extension involved force feedback on the arm joystick. DirectInput was abandoned because of program portability and reuse concerns. Visual Studio 6.0, which is installed on the ILC computers, does not support the current DirectX SDK or DirectInput.

The library used for the final build was “Simple DirectMedia Layer” (SDL) library, a simple, more portable alternative. Unfortunately, this library does not support force feedback.

3.7.4 Serial Communication

3.7.4.1 Serial ActiveX Control

Using the Microsoft API for serial communication directly is much to complex, so an additional library needed to be used. The Microsoft Communications ActiveX control (MCA) is an easy to use library with native C++ binding.

Using MCA, it was easy to set the port settings such as port number, baud rate, byte size, and input size. This control also supports event based handling when receiving data, so the port did not need to be polled continuously.

One of the problems with the Microsoft Communications ActiveX control is that it is not included by default on laptops which not have serial ports. If a laptop is to be used as the operating terminal using a USB-Serial converter, the ActiveX control can be copied from a PC and registered.

3.7.4.2 Serial Protocol

For receiving sensor data from the HC11, a fixed sequence protocol was used. The sequence starts with a special start character (~) and is followed by sensor data in a specific order. This method is simple, however data cannot be sent out of order. For example, if the stick potentiometer input is read, the data cannot be sent immediately but must wait its turn.

For sending motor instructions, the methods considered were: sending a single character which contains all the information, such as ‘Q’=Left Track Forward 50% Duty Cycle; or to send one byte with information on desired track and direction, and a second byte with the desired duty cycle. The first solution is the simplest, yet the second method provides a greater set of possibilities for the duty cycle.

3.7.5 Sensor Data Translation

The sensor data received from the HC11 is a value which represents either the pulse count since the last transmission for an encoder, or the decimal output of the analog to digital converter for the potentiometer.

The track displacement was calculated using the equation $\Delta d=(2\pi r)/360*\Delta degrees$, where r is the track radius. The encoder pulse counts are converted to degrees by multiplying them by the appropriate conversion factor (degrees per pulse). This factor is a property of the specific device and was confirmed to be accurate on average for a multiple trials.

Since the boom encoder is mounted on a geared down axle, the encoder degree value does not directly equal the change in boom angle. The value relating encoder pulse count to change in boom degrees was determined experimentally by measuring the boom angle displacement manually, and comparing the measurement to the reported pulse count. This process was repeated multiple times and average was used.

The potentiometer angle was calculated using a linear, angle vs. voltage relationship. The relationship was derived by measuring the angle manually in intervals, and comparing the measurement to the reported decimal value. The linear equation was then computed as an approximation from these values. This relationship is shown in Appendix H.

4 Testing, Evaluation and Results

4.1 Excavator Motion

Excavator motion was calibrated during the testing process. Duty cycles sent to the right and left track motors were adjusted in order to achieve pure forward movement. The results of this

calibration were confirmed by positioning the excavator parallel to a straight line, and measuring the drift of the excavator over a 10ft test run. The average drift recorded was 9 inches. This constitutes a (9/120 = 0.075) 7.5% drift error. Same testing method was used to measure reverse driving, and yielded identical results.

4.2 Arm Sensor Readings

Sensor reading errors were determined by moving the excavator components (stick and boom), and recording the difference between the GUI display values and physical measurements.

The boom/stick angle readings were accurate to 2.5 degrees. This error was expected. The potentiometer has a 270 degree range, and the HC11 A/D converter uses an 8 bit register with 256 possible values. The error due to the potentiometer limitation was therefore equal to (256/270) 1.05 degrees. Tolerances in the stick and boom construction contributed an additional 1.5 degree of error.

The boom encoder is attached to a geared down axle (1 degree change in boom angle = 20 degree encoder shaft rotation) and produced 150 ticks per revolution. Thus the (360/150 = 2.4) 2.4 degree resolution in encoder readings was reduced by a factor of 20 when the obtained values were translated into actual boom angles. A major source of error was the poor toy construction (1.5 degrees, as above), and the degraded boom encoder waveforms, which contributed an additional 2 degrees to the angle error. The total boom angle error was therefore 3.5 degrees (+/- 1.75 degrees).

4.3 Future Work (Possible Project Extensions)

The coordinated motion algorithm was not implemented, but is a natural extension to this year's work. Joystick inputs controlling boom and stick motors can be put through an inverse Jacobian matrix to produce x-y directional velocities. This is a software modification, and requires no additional hardware.

Force readings were not implemented this year due to the poor quality of signals produced by the “current sense output” pin (pin 8) of the H-bridges. Currents can be translated into bucket forces simply by observing the increase in motor currents as a load is applied. Although the group demonstrated the ability to read and interpret analog signals (by reading the potentiometer values), the current sense signal had too much electrical noise which was generated by the motors, and could not be read by the A/D converter.

5 Conclusion

The primary goal of creating a remotely controlled and monitored robot excavator was completed successfully. Sensors were mounted to provide feedback to the HC11 microprocessor. All required hardware and battery packs were mounted inside of the excavator, creating a robust, autonomous system. HC11 code was written to drive the motors and gather and relay the sensor information to the PC workstation. A 3D rendering of the excavator was created, allowing the user to monitor excavator movements and all relevant data in real time.

The system was demonstrated to be fully functional during the final presentations and open house, providing the basis for future work on the project.

Future groups can engage in implementing a force feedback mechanism, using the force feedback joystick provided by Dr. Hashtrudi-Zaad.

Another possible extension would be track-slippage detection. A position sensor can be used and compared to the encoder readings. If the two displacement readings are not within an acceptable tolerance, track slippage is detected. This modification requires additional hardware mounting and extensive HC11 programming.

Replacing the current driver board with a PWM to DC voltage converter would improve the operations of the motors. In the current design, H-bridges amplify the HC11 PWM waveforms, and the PWM output is sent to the motors. If the motors were supplied with a variable DC voltage, the excavator operation would be considerably smoother.

Currently, motors are controlled using duty ratios. The problems with this control method is that motor speeds decrease as the battery packs are drained, and perfect track calibration is difficult. Implementation of speed feedback control would maintain desired track speeds even with decreasing reference power. Furthermore, calibration would be much more accurate than the current straight driving calibration.

The primary focus for improving the graphical user interface lies in the detail of the model. The 3D model is modularized, allowing improvements in detail without making changes to the functional elements of the code. Future work can be done to improve the model based on more accurate physical measurements of the actual excavator. Also, a better physics model will allow the model to move more accurately based on sensor readings.

6 References

[1] Vuk Zrnic, Alan Lo, Ryan Murphy, (2005). *ELEC490 Proposal – Remote Control of An Excavator Robot, Third Generation,*
Queen's University, Kingston, Canada.

[2] Vuk Zrnic, Alan Lo, Ryan Murphy, (2005). *ELEC490 Blueprint – Remote Control of An Excavator Robot, Third Generation,*
Queen's University, Kingston, Canada.

[3] Shamir Charania, Noel Johnston, Tavis MacCallum, (2004-2005). *ELEC490 Project – Remote Control of An Excavator Robot, Second Generation,*
Queen's University, Kingston, Canada.

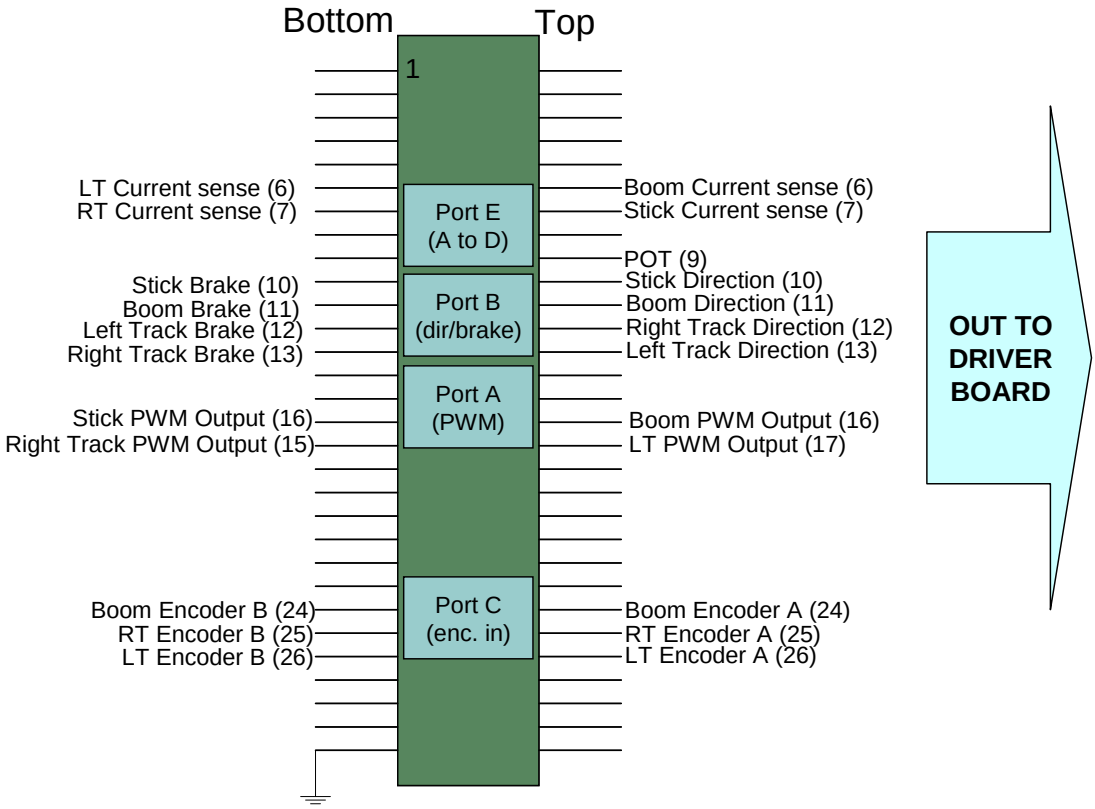
[4] Honeywell Sensing and Control Website
http://content.honeywell.com/sensing/products/resistors

[5] Discussions with Faculty Supervisor: Dr. Keyvan Hashtrudi-Zaad

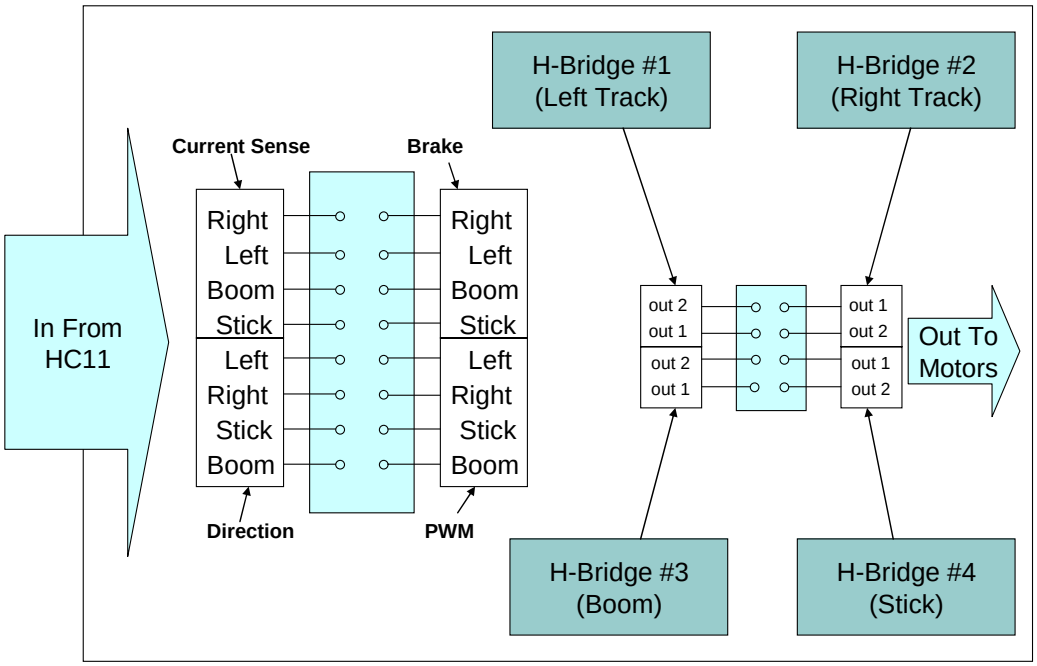
[6] Motorola, *MC68HC11A8 Programming Reference Guide,* Rev. 1

List of Appendices

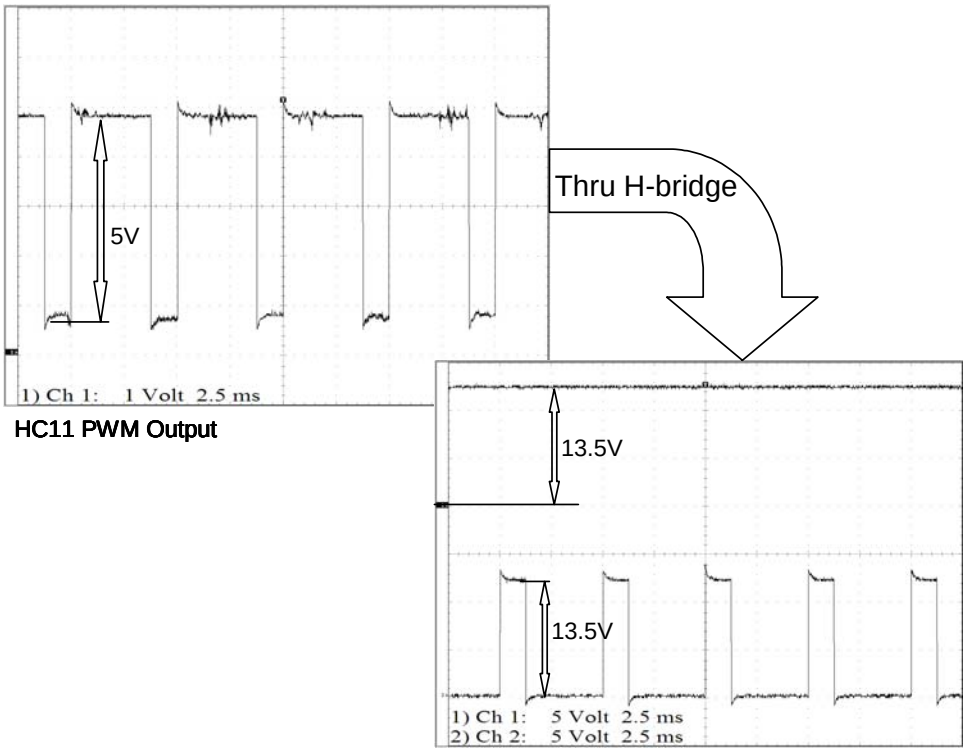
Appendix A



Appendix B



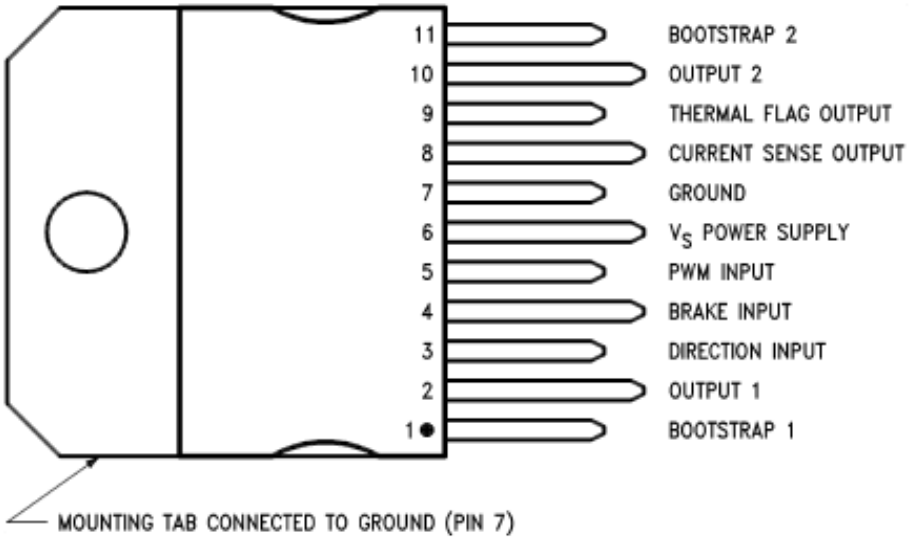
Appendix C



HC11 PWM Output

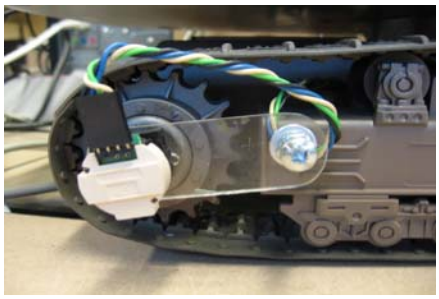
Motor Output

Appendix D



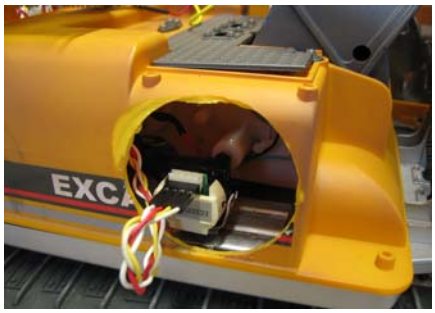
Appendix E

TRACK ENCODERS

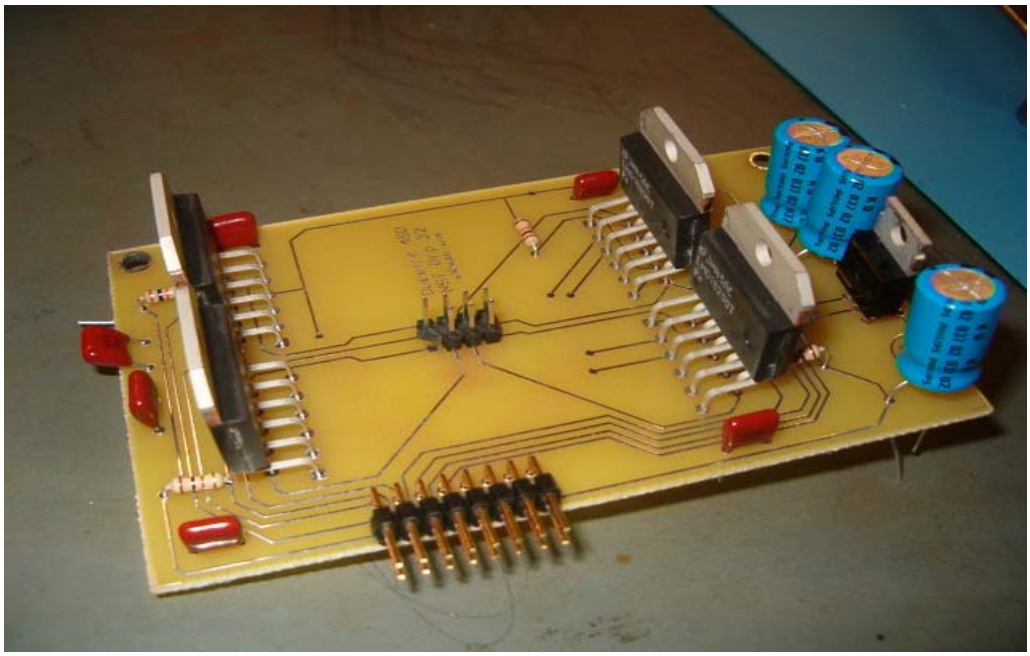


STICK POTENTIOMETER

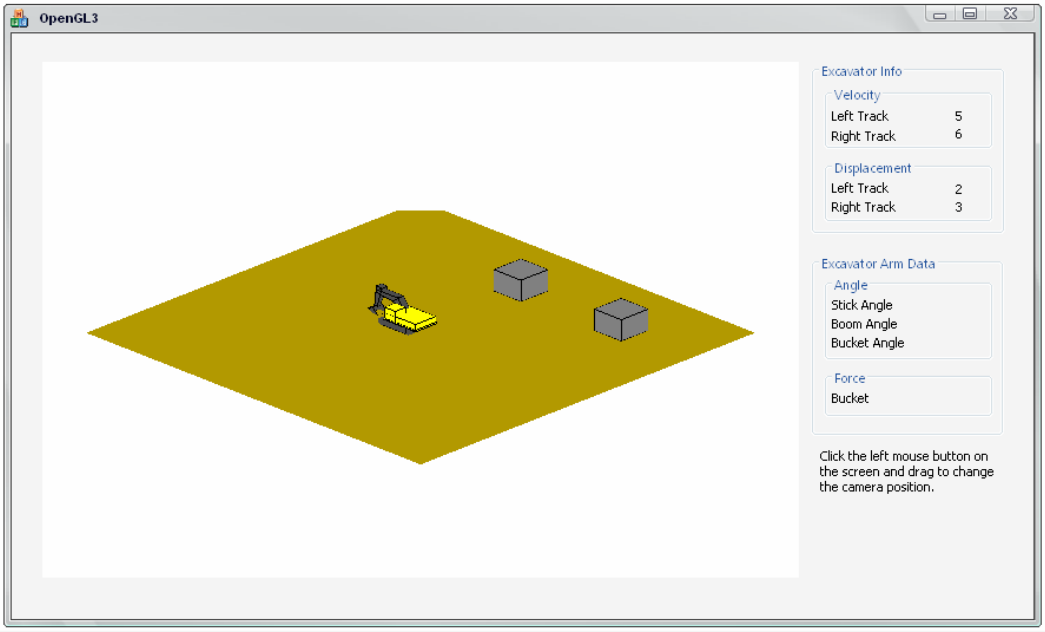
BOOM ENCODER



Appendix F



Appendix G



Appendix H