

Grand Valley State University
Padnos School of Engineering

Anti-Sway Compensation Project
EGR 345: Dynamic Systems Modeling and Control
Fall 2003

Team #11:
Chris Hough
Lance King
Bryan Pruess
Nathan Schout
Tyler DeVos
Steve Johnson

December 4, 2003

Executive Summary

The purpose of this project was to design and build a cart that modeled a crane with a suspended mass. The cart was controlled by the Motorola 68HC11 and driven along a 2x4 using a motor and drive wheel. The angular position of the suspended mass and the linear position of the cart were relayed to the 68HC11. The mass was 1 kilogram in size and was suspended 40 centimeters below the top of the 2x4. The program controlling the cart allowed the user to input a desired linear distance for the cart to move. The program also dampened the sway of the mass allowing for quick movement with minimal sway.

Calculations were made from the free body diagrams of the suspended mass to generate equations for a simulation of the movement in Scilab. Stress calculations were performed to verify the dimensions used in the various load-bearing components. Output motor speed and distance traveled per revolution was also calculated.

The two preliminary tests of the cart were both failures. The program was incomplete for the first test and the cart functioned only with a direct voltage supply. The program and cart were complete for the second test, but an encoder failure prevented the cart from running properly. The potentiometer was not used in either of the preliminary tests as the anti-sway compensation portion of the program was not complete. The cart did function for the final test, but not as well as planned. The cart travel was shorter than intended, which meant it did not take full advantage of the tolerance that was available for the test. Also, the potentiometer gain was set to high which caused excessive oscillation at the cart's final position. The final settling time of 9.87 sec was much higher than preliminary runs over the same distance.

Overall, the physical design of the cart was very robust and the program functioned well. However, implementation of the design created many unforeseeable difficulties. The results were directly impacted by the planning and time spent on each part. Our failure to have a fully functioning program for the first test cascaded into the subsequent deadlines and caused us to be one step behind for much of the project. This created a rush of activity before the final deadline. Although the final test was successful, the team would have benefited from a more robust program and more preliminary testing.

Table of Contents

| | |
|---|------------|
| Mechanical/Electrical Design Analysis..... | 3 |
| Program Control Scheme and Schematic | 4-5 |
| Project Budget, BOM, and Weight Inventory | 6 |
| Motion Equations..... | 7-8 |
| Simulation | 9 |
| Motor Parameters..... | 9-10 |
| Actual Test Results and Final Design..... | 10-12 |
| Drawing Summary and Design Modifications | 12-13 |
| Conclusion | 14 |
| Drawings | Appendix A |
| Controller C Program..... | Appendix B |
| Simulation Program | Appendix C |
| Receipts and Cost Evidence..... | Appendix D |
| Stress Calculations..... | Appendix E |

Mechanical/Electrical Design Analysis

Factors considered in choosing the material for the cart included: cost, practicality, strength, machineability, availability, size, and aesthetics. The material chosen for the top plate was Acetal (high density plastic). The plastic available in the Keller Engineering Lab was not thick enough and did not machine as well as was desired. The Acetal satisfied all of our requirements. After the top plate material choice was made, the rest of the material decisions were made and the remaining components were purchased. When choosing the other components, the factors stated above and the factors impacting our final score (see Equation 3) were taken into consideration. A bill of materials (BOM) can be found in Table 1, and displays all component costs and weights. The final design is shown below in Figure 1.

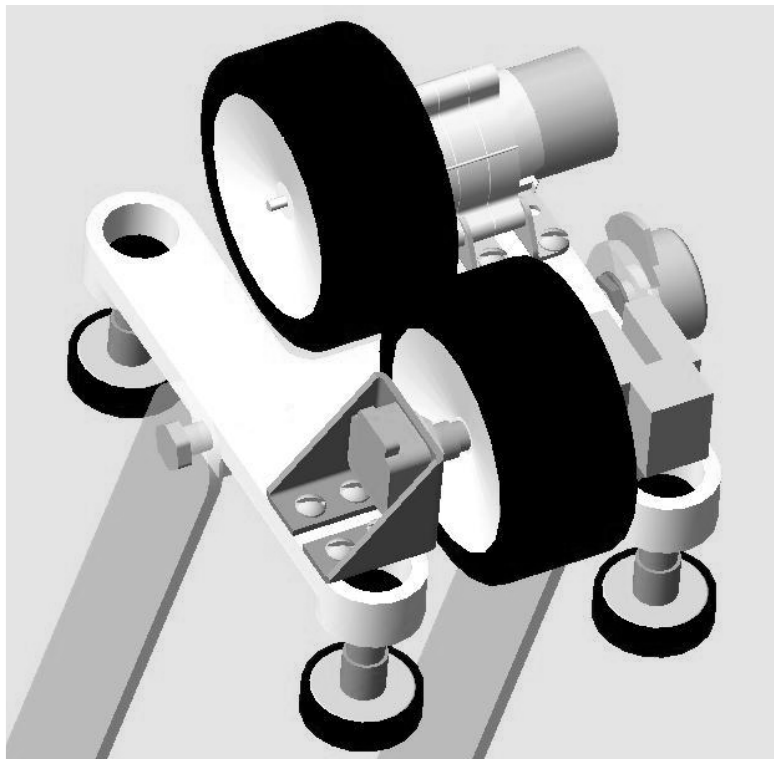


Figure 1: Final Design of Cart

This design was chosen because it was very simple and was easy to design, build and rebuild as needed. A lightweight, inexpensive, mechanical encoder was used and was mated directly to the output wheel through a press fit hole, serving as the axle. The potentiometer was used to output the angular position of the arms. It was fixed directly to one of the swing arms, which were cut from polycarbonate plastic sheet. The polycarbonate plastic helped to minimize the cost and weight of the arms. The motor used was a RC-260 Motor with up to 4 planetary gear assemblies, chosen because of its light weight, low cost and adjustable gear ratio. A gear ratio of 100:1 was used for the final design. The motor drove the cart and functioned as the axle for the drive wheel. The wheels were chosen because of their light weight and gripping characteristics.

Program Control Scheme and Schematic

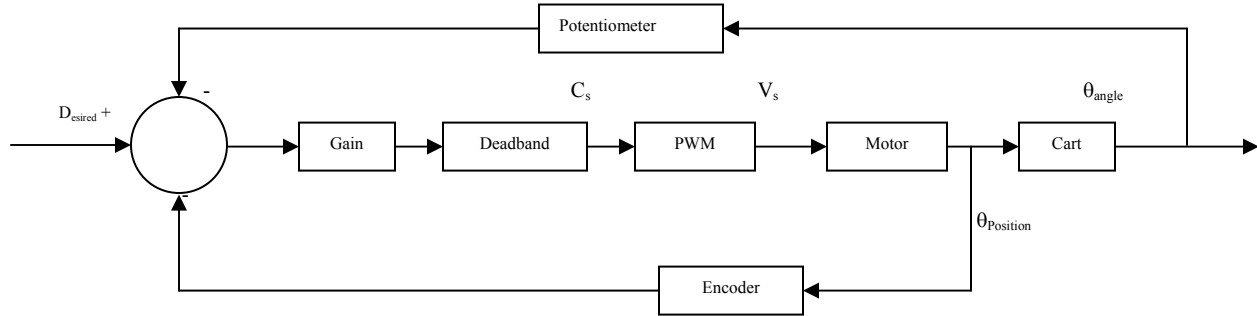


Figure 2: Sway Compensation Block Diagram

The C-Program, included in Appendix C, follows the system block diagram shown above in Figure 2. A desired value is input to the program by the user. This value is magnified by the gain function and sent through the deadband routine to compensate for friction in the motor. It is then sent through the PWM function to generate the proper voltage to drive the system. The 68HC11 micro-controller sends the voltage value to the motor, which moves the cart. While the cart is moving, the encoder sends a return signal to the 68HC11 to update the position of the cart. The potentiometer measures the sway angle between the cart and the mass and returns this value to the 68HC11. This angular position is used along with a back and forth motion of the motor to dampen the sway of the arms when the cart reaches its desired position.

The motion profile shown in Figure 3 is the desired movement of the cart. The profile begins with smooth acceleration, levels off at maximum velocity, and stops after a smooth deceleration. The encoder and set point table generate the acceleration of the cart to maximum velocity. They also generate a deceleration of the motor as the cart reaches the desired position.

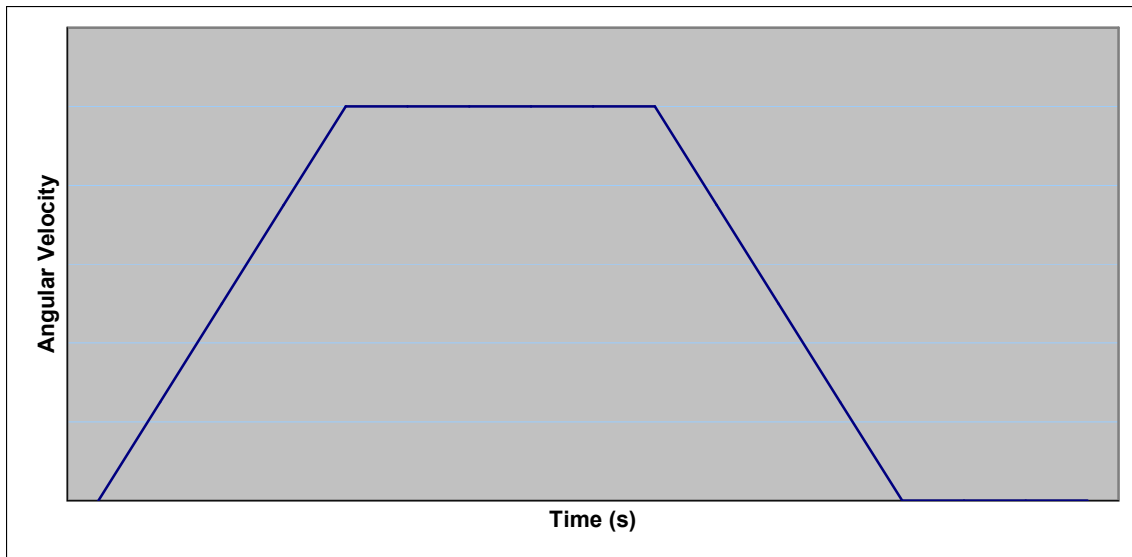


Figure 3: Desired Motion of the Cart

The schematic in Figure 4 shows the wiring of the various electrical components. A fixed voltage supply of 5V was used to power the motor while the potentiometer was powered by the 68HC11. Capacitors were used to reduce noise in the system. The output from the potentiometer was feedback into an analog input of the 68HC11 (PE1). The Grayhill encoder was connected to the PD4, PD5, and GRD pins of the 68HC11. An L293D chip was used to drive the motor.

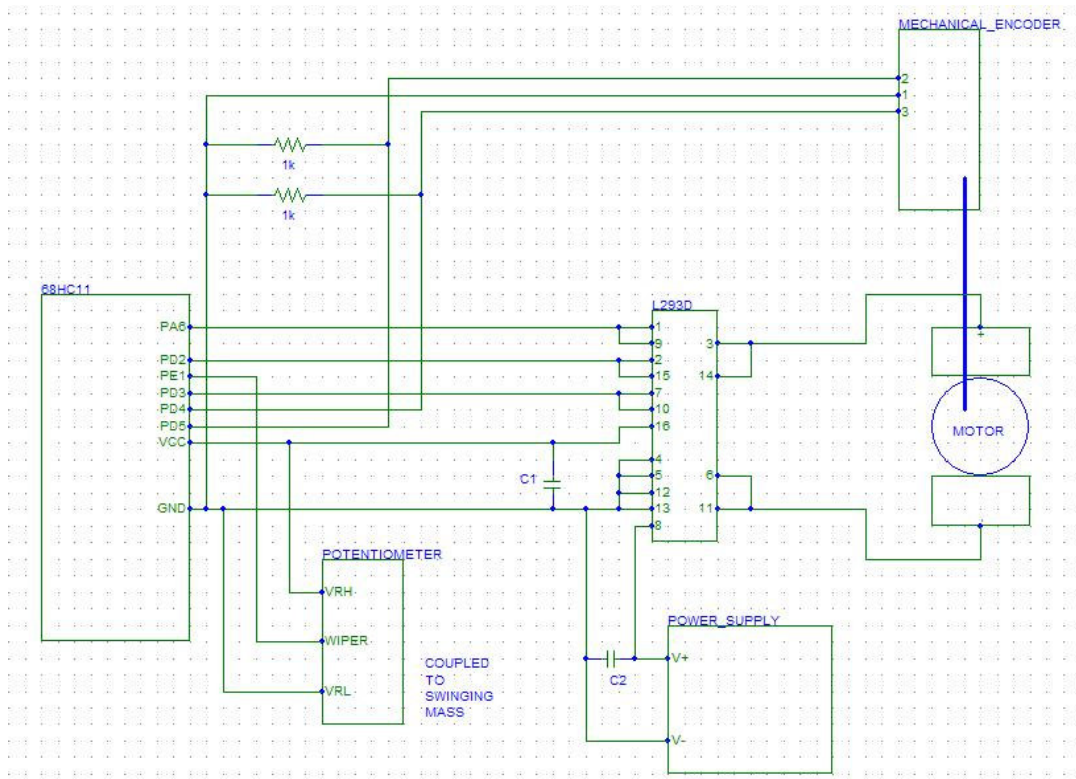


Figure 4: Project Wiring Schematic

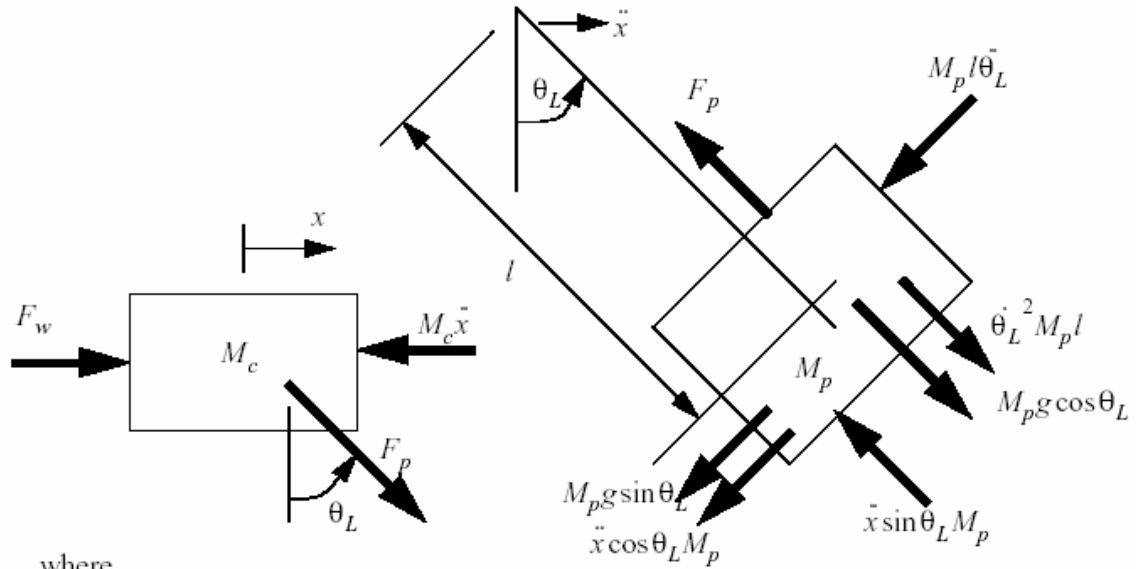
Project Budget, BOM, and Weight Inventory

The Bill of Materials (BOM) is shown in Table 1, and was used for the design and construction of the cart. It itemizes each part with a weight and cost. The receipts for the materials that were purchased can be found in Appendix E (note: receipts are not included with the online report). The prices for other purchased materials were estimated using online sources for similar components. All materials were individually weighed using a digital scale.

Table 1: BOM and Weight Inventory

| Item # | Part Name | Material | Supplier | Cost | Quantity | Cost Sub-Total | Weight Sub-Total(g) |
|---------------|-----------------------|-----------------|--------------------|-------------|-----------------|-----------------------|----------------------------|
| 1 | Top Plate | Acetal | GHSP | \$4.00 | 1 | \$4.00 | 105.6 |
| 2 | Swing Arm | Plastic | Keller | \$1.00 | 2 | \$2.00 | 34.2 |
| 3 | Roller Blade Bearings | NA (steel) | Hobby Store | \$0.85 | 4 | \$3.40 | 64.7 |
| 4 | 8 mm Bolts | Steel | Chris | \$0.16 | 4 | \$0.64 | |
| 5 | Bearing Spacers | Aluminum | Chris | \$0.12 | 4 | \$0.48 | 22.8 |
| 6 | 6V Gear Motor | NA | Rider's | \$17.00 | 1 | \$17.00 | 94.3 |
| 7 | Axiom Board | NA | Keller | \$89.00 | 1 | \$89.00 | 0.0 |
| 8 | RC Wheels | Plastic | Rider's | \$7.00 | 2 | \$14.00 | 62.9 |
| 9 | Encoder | NA | Digi-Key | \$3.00 | 1 | \$3.00 | 15.2 |
| 10 | 1/4" Swing Arm Bolt | Steel | Keller | \$0.70 | 1 | \$0.70 | 10.3 |
| 11 | Potentiometer | NA | Radio Shack | \$0.20 | 1 | \$0.20 | 20.5 |
| 12 | Bearing Cushions | Foam Rubber | Breakaway Bicycles | \$0.08 | 4 | \$0.32 | 4.2 |
| 13 | Cat5 Jack | NA | Radio Shack | \$4.99 | 2 | \$9.98 | 7.7 |
| 14 | Cat5 Cord | NA | Radio Shack | \$7.99 | 1 | \$7.99 | |
| | | | | | Totals | \$152.71 | 442.4 g |

Motion Equations



where,

- M_c = mass of cart
- M_p = mass of payload
- F_p = force in suspension arm
- F_w = force from wheels
- θ_L = angle of payload from vertical
- l = length of suspension arm
- r_w = radius of cart wheel

$$\nearrow \sum F = -M_p g \sin \theta_L - M_p l \ddot{\theta}_L - \ddot{x} \cos \theta_L M_p = 0$$

$$l \ddot{\theta}_L = -g \sin \theta_L - \ddot{x} \cos \theta_L$$

$$\nwarrow \sum F = F_p - M_p g \cos \theta_L - \dot{\theta}_L^2 M_p l + \ddot{x} \sin \theta_L M_p = 0$$

$$F_p = M_p g \cos \theta_L + \dot{\theta}_L^2 M_p l - \ddot{x} \sin \theta_L M_p$$

$$\rightarrow \sum F_x = -M_c \ddot{x} + F_p \sin \theta_L + F_w = 0$$

$$-M_c \ddot{x} + F_p \sin \theta_L + F_w = 0$$

$$-M_c \ddot{x} + (M_p g \cos \theta_L + \dot{\theta}_L^2 M_p l - \ddot{x} \sin \theta_L M_p) \sin \theta_L + F_w = 0$$

Figure 5: FBDs and equations for the suspended mass

$$\begin{aligned}
\dot{\omega}_w + \omega_w \left(\frac{K^2}{JR} \right) &= V_s \left(\frac{K}{JR} \right) - \frac{F_w r_w}{J} \\
\therefore F_w &= V_s \left(\frac{K}{r_w R} \right) - \dot{\omega}_w \left(\frac{J}{r_w} \right) - \omega_w \left(\frac{K^2}{r_w R} \right) \\
\text{given } x &= \theta_w r_w \\
\therefore F_w &= V_s \left(\frac{K}{r_w R} \right) - \ddot{x} \left(\frac{J}{r_w^2} \right) - \dot{x} \left(\frac{K^2}{r_w^2 R} \right) \\
(1) \text{ ---} &\rightarrow \\
-M_c \ddot{x} + (M_p g \cos \theta_L + \dot{\theta}_L^2 M_p l - \ddot{x} \sin \theta_L M_p) \sin \theta_L + V_s \left(\frac{K}{r_w R} \right) - \ddot{x} \left(\frac{J}{r_w^2} \right) - \dot{x} \left(\frac{K^2}{r_w^2 R} \right) &= 0 \\
-\ddot{x} \left(M_c + \frac{J}{r_w^2} + (\sin \theta_L)^2 M_p \right) + \cos \theta_L \sin \theta_L M_p g + \dot{\theta}_L^2 (M_p l \sin \theta_L) + V_s \left(\frac{K}{r_w R} \right) - \dot{x} \left(\frac{K^2}{r_w^2 R} \right) &= 0 \\
\ddot{x} \left(\frac{M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2}{r_w^2} \right) &= (\cos \theta_L \sin \theta_L) M_p g + \dot{\theta}_L^2 (M_p l \sin \theta_L) + V_s \left(\frac{K}{r_w R} \right) - \dot{x} \left(\frac{K^2}{r_w^2 R} \right) \\
\ddot{x} &= (\cos \theta_L \sin \theta_L) \left(\frac{M_p g r_w^2}{M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2} \right) + \dot{\theta}_L^2 \left(\frac{M_p l \sin \theta_L r_w^2}{M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2} \right) + \\
&V_s \left(\frac{K r_w}{R (M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2)} \right) + \dot{x} \left(\frac{-K^2}{R (M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2)} \right)
\end{aligned}$$

Figure 6: Differential Equations for the Motor

State Equations:

$$\begin{aligned}
\dot{x} &= v \\
\dot{v} &= v \left(\frac{-K^2}{R (M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2)} \right) + (\cos \theta_L \sin \theta_L) \left(\frac{M_p g r_w^2}{M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2} \right) \\
&+ \dot{\theta}_L^2 \left(\frac{M_p l \sin \theta_L r_w^2}{M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2} \right) + V_s \left(\frac{K r_w}{R (M_c r_w^2 + J + (\sin \theta_L)^2 M_p r_w^2)} \right) \\
\dot{\theta}_L &= \omega_L \\
\dot{\omega}_L &= \frac{-g}{l} \sin \theta_L - \frac{\dot{v} \cos \theta_L}{l}
\end{aligned}$$

Figure 7: Final State Equations for the Motor

Figures 5, 6 and 7 were used with Scilab to run a simulation of the cart motion.

Simulation

Figure 8 displays a motion profile of the cart simulated using Scilab and the equations given in Figures 5-7. The profile shows the cart moving smoothly to the twenty inch mark and then oscillating to compensate for the sway of the arms. The simulation shows the cart reaching the allowed tolerance within 4.5 seconds.

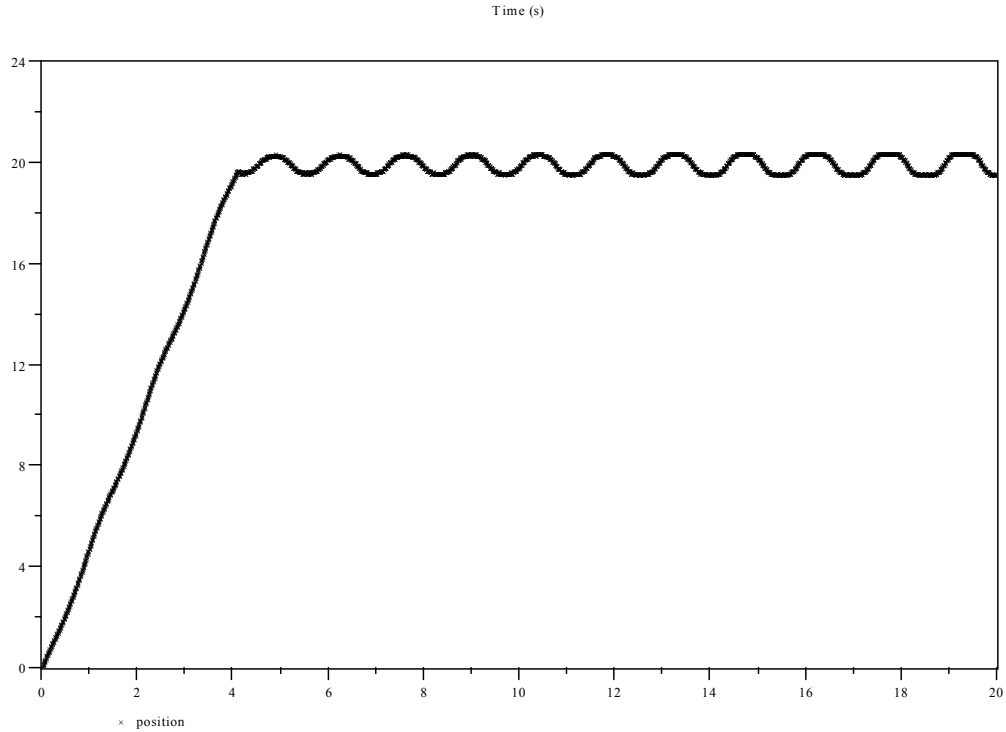


Figure 8: Scilab Simulation

The Scilab code for this simulation is included in Appendix C.

Motor Parameters

The parameters for the motor were found using a tachometer. The input voltage and RPMs were recorded to correlate the voltage and the steady state velocity of the motor. From this, three motor parameters were found. The resistance (R) was measured directly as 7 ohms. The motor speed constant (K) was calculated using Equation 1 and was equal to $0.537 \frac{\text{V}}{\text{rad}}$. Lastly, the moment of inertia (J) was calculated using Equation 2 and was equal to $0.00025 \text{ kg} \cdot \text{m}^2$.

$$\omega_{SS} = \frac{1}{K} V_{SS} \quad (1)$$

$$J = \frac{K^2}{R \left(\frac{1}{\tau} \right)} \quad (2)$$

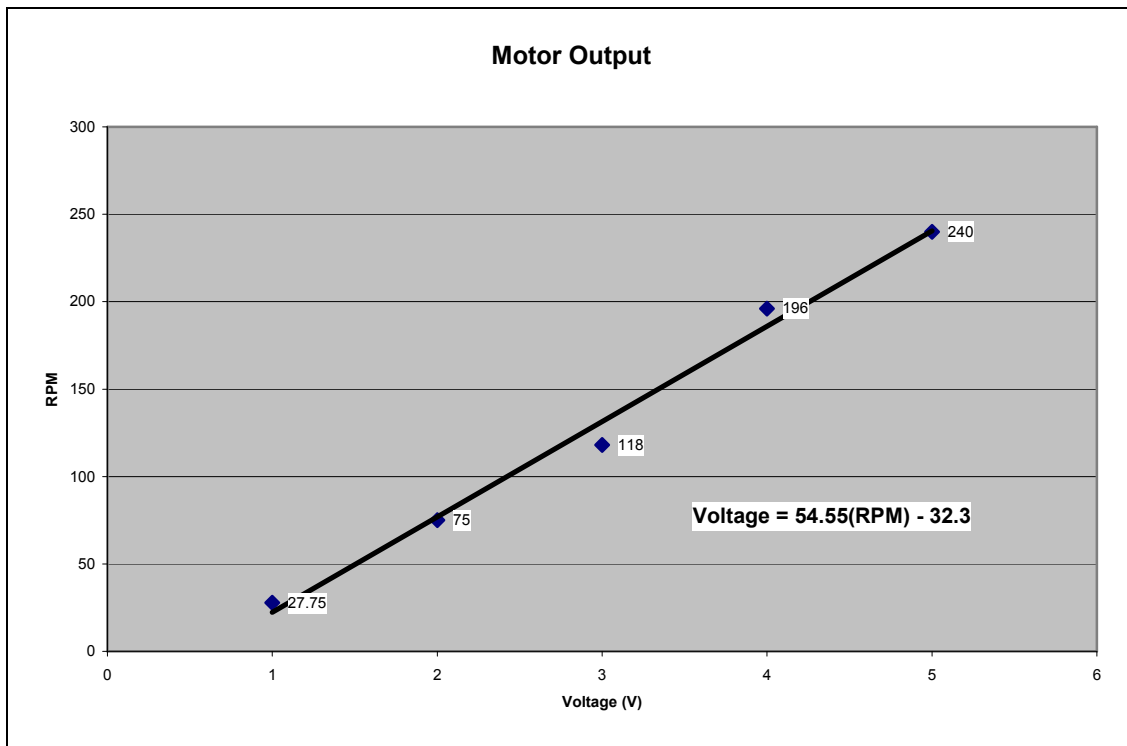


Figure 9: Motor RPM vs. Voltage

The motor speed vs. applied voltage was graphed as shown in Figure 9 (using MS Excel). The graph displays a linear relationship between voltage and motor speed.

Actual Test Results and Final Design

First Formal Test

The C Program was not working at the time of testing, so direct voltage was applied to move the cart. There was also no encoder to check the cart position. The guides seemed to bind in some areas of the 2x4 and had excessive play in other areas. This affected the stability the cart.

Second Formal Test

Encoder, potentiometer, and some minor C program problems prevented the cart from running. A bad encoder was to blame for a lack of position feedback from the cart. A new encoder was wired up and the program functioned properly. At this point, the relationship between the actual distance traveled and the distance read by the encoder was accurate enough for the requirements of the project.

Final Test

The cart and program were both functional for the final test. However, the test was not as successful as previous unofficial runs. The official time for our final test of the cart was 9.87 sec, which was slower than our theoretical time of around 4.5 sec. The major problem during the final test was a potentiometer gain that was set to high. Also, the actual position of the cart was about a quarter of an inch shorter than the desired position of twenty inches. This caused the cart to travel just short of the optimum position, and the high potentiometer gain made the cart oscillate forward and back across the timing sensor. With a few minor changes in the program, the cart would have run successfully and the final time would have been much faster. The difference between the theoretical and actual settling times was a result of these two problems.

Final Score

Once the final test was complete, the final score was calculated using equation 3.

$$\text{Score} := \left(\frac{ts}{d}\right)^2 \cdot 4^{\frac{C}{200}} \cdot 10^B \cdot 10^T \cdot 2^{\frac{M}{0.2}} \quad (3)$$

where,

t_s = the time to settle (s)

C = total cost of part (\$)

d = distance moved in test (m)

B = build quality score assigned by judges (0 to1)

T = theory quality score assigned by judges (0 to1)

M = mass of the apparatus (Kg)

Computed in MathCAD, our team score was 18,562 points. The high score was due mostly to the length of time it took our cart to settle. Had our time been as low as simulated, or closer to our preliminary tests, our score would have much lower.

$$ts := 9.87 \quad d := .2 \quad C := 152 \quad B := .535 \quad T := .2 \quad M := .460$$

$$\text{Score} := \left(\frac{ts}{d}\right)^2 \cdot 4^{\frac{C}{200}} \cdot 10^B \cdot 10^T \cdot 2^{\frac{M}{0.2}}$$

$$\text{Score} = 1.869 \times 10^5$$

Table 2 shows how our team score compared with the other 11 teams.

Table 2: Final Scores

| Team # | Score |
|--------|-------|
| 1 | 1533 |
| 2 | 2245 |
| 3 | 748 |
| 4 | 900 |
| 5 | 969 |
| 6 | 939 |
| 7 | 575 |
| 8 | 1186 |
| 9 | 478 |
| 10 | 781 |
| 11 | 18562 |
| 12 | 8004 |

We finished last among all the teams. Along with our slow run time, which could have been fixed with minor programming changes, our build quality score was also low. This was due to our simplistic design and poor quality of the plastic mill. Our final top plate had many visible imperfections and our build quality score was not as high as other teams with more complex designs.

Drawing Summary and Design Modifications:

Four drawings were created for this project (Appendix A). The first drawing is an assembly drawing which gives all the part names and the standard views of the assembly. The second three drawings are of the parts that were manufactured for the construction of the cart. The drawing of the top plate of the cart gives the necessary dimensions required to machine of the part. The final two drawings are of the swing arms and give the necessary dimensions to manufacture the parts using a milling machine or manual methods. There are two separate swing arm drawings as the two parts have slight differences in relation to their attachment to the cart. One is simply hung from the cart using a through hole, while the other is attached and oriented to the potentiometer using two through holes.

The assembly drawing of the cart is included in Appendix A and should be referenced for better understanding of the changes made.

Potentiometer Mount:

The original design included a top plate (Item 16 of the Assembly Drawing), which had a feature that allowed for the insertion of the potentiometer (Item 13) into the top plate. However, this design required complex machining of the top plate, and retention of the potentiometer to constrain the potentiometer relative to the shaft movement. It was decided that the potentiometer would instead be mounted to the swing arm and press fit into the top plate. This allowed the top plate to have a simple

interference fit hole to capture the potentiometer shaft, and required a simple secondary hole on the swing arm to eliminate the rotation of the potentiometer base (see Potentiometer Swing Arm drawing).

Belt Drive System:

The original idea for propulsion of the cart was to use a gear and belt system to both drive the cart and turn the encoder. The system would use a drive wheel attached directly to the motor shaft to drive the cart, and a feedback wheel attached to the encoder to output the position of the cart to the computer. These two wheels would be coupled using a belt, which would relate the drive rotation to the encoder rotation. There were two major problems with this system. First, a belt could not be found that would properly mesh with the two gears and provide a suitable grip to the 2x4. And second, when a rubber band was used to test the system, the rubber band tended to creep and eventually come off the gears completely. These problems were eliminated when two RC car wheels (Item 17) were used to both power and control the cart. Also, the second wheel could now be uncoupled from the first wheel, as it would also ride on the 2x4. This eliminated ratio calculations and simplified the corresponding C program.

Swing Arms:

The original swing arms were made of thicker plastic material, which did not flex and was heavier. A material change provided lighter material which reduced the cart weight and allowed the arms to flex inward to better locate the mass directly below the cart.

Guide Bearings:

The original design used eight bearings to guide the cart along the 2x4 and prevent the cart from tipping. A wood prototype cart was made, and it was found that four bearings could isolate the cart movement just as well as eight, with a weight reduction of 64.7 grams. After the first test we also noticed some binding, as the bearings had no compliance and the width of the 2x4 had loose tolerances. It was decided that the bearing spacing would be made wider, and foam cushion would be added to the bearings to compensate for width changes in the 2x4.

Motor and Encoder attachment:

The original design provided no means of attaching the motor and encoder to the top plate. The second design revision used simple straps to retain the motor and encoder. However, this resulted in a height difference between the encoder shaft and motor shaft, and a resulting angled top plate. The new motor included mounting brackets, and an encoder bracket was created to match the height of the encoder shaft with that of the motor shaft.

Top Plate:

The top plate design was changed to reduce the overall weight by eliminating unnecessary material. The thickness was reduced by a 1/4 inch and the weight was reduced by 200 grams, with reduction in strength or stiffness.

Conclusion

The overall performance of the anti-sway crane was a success. The cart evolved successfully from a mere idea to a fully working model. The strengths of the design were the low weight and the simple assembly of the cart. Also, the wheels provided additional strength and speed because of their large contact area and gripping ability. One drawback was our simplistic design which caused a decrease in our build quality score, and in turn, our overall total. The poor results from the plastic mill also had a negative effect on our score. The major detriment to the team was the lack of time to debug and adjust the final working program, which resulted in minor program flaws that had major impact on our overall score.

APPENDIX A

The following drawings can be found at
<http://claymore.engineer.gvsu.edu/~schoutn/index.html>

Assembly Drawing

Top Plate Drawing

Swing Arm Drawing

Potentiometer Swing Arm Drawing

Encoder Bracket Drawing

APPENDIX B

C Program for Anti-Sway Cart Motion

```

/*****
/*
/*           Crane Sway Compensation
/*
/*****
/*
/* Group 11:      Chris Hough, Bryan Pruess, Lance King, Nate Schout
/*
/* Date: November 20, 2003
/*
/*****
/* Program Description:
/*
/*           A setpoint table is generated to move the cart from 2" to 20"
/*           inches in 2" increments. The cart moves when the user selects
/*           a value from '0' to '9'.
/*
/*           An anti-sway routine is used to compensate for the swinging
/*           mass after the cart has moved the appropriate length. The
/*           program reads the potentiometer angle, and then moves the cart
/*           to damp the swinging mass.
/*
/*****

/* Includes files */

#include <hc11e9.h>
#include <buffalo.h>
#include "pwm.h"

/* Defined Variables */

#define c_stick_pos 180
#define c_stick_neg 180
#define c_max 255
#define c_min 255
#define T 4
#define kp 1
#define ke 20
#define TABLE_SIZE 11
#define V0 0
#define V1 12

/* Deadband Values */

/* Potentiometer Gain */
/* Encoder Gain */

/* Global Variables */

unsigned num = 0;
int cwanted = 0;
int array[1000];
int point_master[TABLE_SIZE] = {0,24,95, 206, 345, 500, 655, 794, 905, 976, 1000}; /* Acceleration Table */
int point_position[TABLE_SIZE];
int point_time[TABLE_SIZE];
int point_start_time;
int point_index;
int ticks;
int point_current;
int e_sum = 0;
int c_wanted = 0;
int position = 0;

/* variables to keep a system clock count */
/* gloal variable to track position */
```

```

int theta_vertical = 0;
int theta = 0;
/* Generates Setpoint Table */

void update_table(int start, int end, int duration_sec){
    unsigned i;

    point_time[0] = ticks + 3;
    point_position[0] = start;

    for(i=1; i< TABLE_SIZE ; i++){
        point_time[i] = point_time[0] +
            (unsigned long)i * duration_sec * 250/ (TABLE_SIZE - 1);
        point_position[i] = start + (long int) (end-start) * point_master[i]/1000;
    }
    point_index=0;
}

/* Setpoint Table Check */
void get_setpoint(){
    ticks++;

    if(point_index < TABLE_SIZE){
        if(point_time[point_index] == ticks){
            point_current = point_position[point_index++];
        }
    }
}

int integrate(int e){
    e_sum += e*T;
    return e_sum;
}

/* Feedback Controller */
int controller(int cd, int cf){
    int cw;
    int pe;

    theta = ADR2;
    pe = theta-theta_vertical;

    if((pe < 15) && (pe > -15)){
        cw=ke*(cd-cf);
    }
    else{
        cw=ke*(cd-cf)+pe*kp;
    }

    return cw;
}

/* Deadband Compensation */
int deadband(int c_wanted){
    int c_adjusted;
    if(c_wanted == 0){
        c_adjusted = 0;
    }
    else if(c_wanted < 0){
        c_adjusted = c_stick_pos
            + (unsigned)c_wanted * (c_max - c_stick_pos)/ c_max;
    }
    else {
        c_adjusted = (-c_stick_neg)
            - (unsigned)(-c_wanted ) * (c_min - c_stick_neg)/ c_min;
    }
    return c_adjusted;
}

```

```

/* Voltage for Desired Value */

void v_output(int v_adjusted){
    if(v_adjusted >= 0){
        CCW off*/
        PORTD &=~PD3;
        PORTD=PD2;

        if(v_adjusted > 255){
            RefSignal = 255;
        } else {
            RefSignal = v_adjusted;
        }
    } else {
        off, CCW on */
        PORTD &=~PD2;
        PORTD=PD3;

        if(v_adjusted < -255){
            RefSignal = 255;
        } else {
            RefSignal = -v_adjusted;
        }
    }
}

/* Update Position Variable */

void EncoderUpdate(void){
    static unsigned char state = 0xFF;
    unsigned char new_state;

    new_state = (PORTD & (PD5 | PD4)) >>4;

    if (state != new_state) {
        switch(state){
            case 0x00:
                if (new_state == 0x01)
                    position++;
                else
                    position--;
                break;

            case 0x01:
                if (new_state == 0x03)
                    position++;
                else
                    position--;
                break;

            case 0x03:
                if (new_state == 0x02)
                    position++;
                else
                    position--;
                break;

            case 0x02:
                if (new_state == 0x00)
                    position++;
                else
                    position--;
                break;
        }
        state=new_state;
    }
}

```

```

/* Interrupt Service Routine */

extern void __attribute__((interrupt)) RTI_ISR(void){

    TFLG2 = RTIF;
    __asm("CLI");
    EncoderUpdate();
    get_setpoint();
    v_output(deadband(controller(point_current, position)));

}

/* Initializing Function */

void initial(void) {

    PWMInit();
    PWMEnable();

    *(unsigned *)0x00EC = (unsigned)RTI_ISR;
    TMSK2 |= RTII;
    __asm("CLI");

    OPTION |= ADPU;
    ADCTL = SCAN | CA;

    DDRD &= ~(PD4 | PD5);
    DDRD |= (PD2 | PD3);

    theta_vertical = ADR2;

    ticks = 0;
    point_current = 0;
    point_index=TABLE_SIZE;

}

int main (void)
{
    unsigned mode;
    char key;
    int start;

   _putstr("Enter a value to output \n");
   _putstr("Press 0 for position 2in \n");
   _putstr("Press 1 for position 4in \n");
   _putstr("Press 2 for position 6in \n");
   _putstr("Press 3 for position 8in \n");
   _putstr("Press 4 for position 10in\n");
   _putstr("Press 5 for position 12in\n");
   _putstr("Press 6 for position 14in\n");
   _putstr("Press 7 for position 16in\n");
   _putstr("Press 8 for position 18in\n");
   _putstr("Press 9 for position 20in\n");

   _putstr("Press q to quit\n");

    int t;

```

```

    c_wanted = 0;
start = 0;
    mode = 0;

    initial();

    do{
        key = input();
        switch (key)
        {
            case 0:
                break;

            case '0':
                start = c_wanted;
                c_wanted = 4;
                t = 1;

                update_table(start, c_wanted, t);
                putstr("Moving 2 inches\n");

                break;
            case '1':
                start = c_wanted;
                c_wanted = 8;
                t = 1;

                update_table(start,c_wanted, t);
                putstr("Moving 4 inches\n");

                break;
            case '2':
                start = c_wanted;
                c_wanted = 12;
                t = 1.5;

                update_table(start,c_wanted, t);
                putstr("Moving 6 inches\n");

                break;
            case '3':
                start= c_wanted;
                c_wanted = 16;
                t = 1;

                update_table(start,c_wanted, t);
                putstr("Moving 8 inches\n");
                break;
            case '4':
                start = c_wanted;
                c_wanted = 20;
                t = 1.5;

                update_table(start, c_wanted, t);
                putstr("Moving 10 inches\n");

                break;
            case '5':
                start = c_wanted;
                c_wanted = 24;
                t = 2;

                update_table(start, c_wanted, t);
                putstr("Moving 12 inches\n");

                break;
            case '6':
                start = c_wanted;
                c_wanted = 28;

                t = 2.5;
                update_table(start, c_wanted, t);
                putstr("Moving 14 inches\n");

                break;
            case '7':
                start = c_wanted;
                c_wanted = 33;
                t = 2.5;

                update_table(start, c_wanted, t);
                putstr("Moving 16 inches\n");

                break;
            case '8':

```

```

start = c_wanted;
c_wanted = 37;
t = 3;
                                update_table(start, c_wanted, t);
                                putstr("Moving 18 inches\n");
break;
                                case '9':
start = c_wanted;
c_wanted = 41;
t = 4.5;
                                update_table(start, c_wanted, t);
                                putstr("Moving 20 inches\n");
break;
                                case 'p':
                                case 'P':
                                outint16(position);
                                putstr ("\n");
                                break;
                                case 'a':
                                case 'A':
                                outint16(theta_vertical);
                                putstr ("\n");
                                break;
                                case 't':
                                case 'T':
                                outint16(theta);
                                putstr ("\n");
                                break;
                                case 'q':
                                case 'Q':
                                mode = 1;
                                break;
                                }
} while (mode==0);
PWMDisable();
return 0;
}

```

APPENDIX C

Scilab Simulation Program

```
// Group 11
//
// Chris Hough, Bryan Pruess, Lance King, Nathan Schout
// Sway Project Simulation
//

// System component values
l = 0.40; // 40cm
Mp = 1.0; // 1kg
Mc = 0.463; // 450g
g = 9.81; // gravity
rw = 0.03175; //2.5" diameter
K = 0.537; // motor speed constant
R = 7; // motor resistance
J = 0.00025; // rotor inertia

// System state
x0 = 0; // initial conditions for position
v0 = 0;
theta0 = 0.0; // the initial position for the load
omega0 = 0.0;
X=[x0, v0, theta0, omega0];

// The controller definition

Kenc = 16; // the encoder pulse/rev ratio
Kpot = 0.08; // the angle voltage ratio
Vzero = 2.5; // the voltage when the pendulum is vertical
Vadmax = 5; // the A/D voltage range
Vadmin = 0;
Cadmax = 255; // the A/D converter output limits
Cadmin = 0;
tolerance = 0.5; // the tolerance for the system to settle
Kpp = 20; // position feedback gain
Ksp = 1; // sway feedback gain
Vpwmmax = 12; // PWM output limitations in V
Cpwmmax = 255; // PWM input range
Cdeadpos = 180; // deadband limits
Cdeadneg = 180;

function foo=control(cwanted)
    Cp = (Kenc * X($, 1))/(2*3.1416*rw);
    Cpe = cwanted - Cp;

    Cpc = Kpp * Cpe;
    VL = Kpot * X($,3) + 2.5;
```

```

CL = (VL - Vadmin) / (Vadmax - Vadmin) * (Cadmax - Cadmin);
if CL > Cadmax then CL = Cadmax; end // check for voltages over limits
if CL < Cadmin then CL = Cadmin; end
CLc = Ksp * (CL - (Cadmax + Cadmin)/2);
Cc = Cpc + CLc;
if Cc > 0.5 then // deadband compensation
    Cpwm = Cdeadpos + (Cc/Cpwmmax)*(Cpwmmax - Cdeadpos);
end
if Cc <= 0.5 then
    Cpwm = -Cdeadneg - (Cc/Cpwmmax)*(Cpwmmax - Cdeadneg);
end

foo = Vpwmmax * (Cpwm / Cpwmmax) ; // PWM output
if foo > Vpwmmax then foo = Vpwmmax; end // clip voltage if too large
if foo < -Vpwmmax then foo = -Vpwmmax; end
endfunction

```

```

// define the state matrix function
term1 = Mc*rw^2 + J; // Precalculate these terms to save time
term2 = R*term1;
x = 20;
cw = Kenc*x/(2*3.1416*rw)

function foo=derivative(state,t,h)
    Vs=control(cw);
    term3 = cos(state($,3)); // precalculate the trig functions to save time
    term4 = sin(state($,3));
    term5 = term1 + Mp * (term4 * rw)^2;

    foo = [ state($,2),...
            -state($,2)*(K^2)/(term5*R) ...
            + term3*term4*(Mp)*g*rw^2/term5 ...
            + state($,4)^2 * (Mp)*1*term4*rw^2/term5 ...
            + Vs*K*rw/term5...
            , state($,4)...
            , -g*term4 / 1 - state($,2)*term3 / 1...
            ];
endfunction

```

```

// Integration Set the time length and step size for the integration
steps = 5000; // The number of steps to use
t_start = 0; // the start time - normally use zero
t_end = 20; // the end time
h = (t_end - t_start) / steps; // the step size
t = [t_start]; // an array to store time values
for i=1:steps,
    t = [t ; t($,:) + h];
    X = [X ; X($,:) + h * derivative(X($,:), t($,:), h)]; // first order
end

```

```

// Graph the values for part e)
plot2d(t, [X(:,1)+1*sin(X(:,3))], [-2], leg="position");

```

```
xtitle('Time (s)');

// find the settling time in the results array
ts = 0;
for i = 1:steps,
    xmass = X(i,1) + 1*sin(X(i,3));
    if xmass > (x + tolerance) then ts = i*h + t_start;
    end
    if xmass < (x - tolerance) then ts = i*h + t_start;
    end
end
printf("\nTheoretical settling time %f\n", ts);
```

Appendix D

Receipts

* Receipts were not posted to the web for security reasons (credit card numbers, etc.) If receipts are needed contact any group member

Appendix E

Stress Calculations

The stress calculations were done on the swing arms.

$$\sigma_b = \frac{F}{A} = \frac{2.2}{\frac{1}{8} * \frac{3}{8}} = 46.93 \frac{lb}{in^2}$$

$$\sigma_s = \frac{F}{A} = \frac{2.2}{\pi * \frac{1}{8}} = 11.20 \frac{lb}{in^2}$$

For this type of composite the allowable bearing stress and shear stress is approximately 1-4 Ksi and 40-60 Ksi , respectively. The shear stress and bearing stress both have a factor of safety that is in the hundreds; therefore, there is no concern for any stresses occurring in the swing arms.