MECHENG 312

An Intelligent LabVIEW-Controlled 2D Marking System

William Li (916720181)

Clinton Yiu (741379312)

27/04/2018

Executive Summary

An Intelligent LabVIEW Controlled 2D Marking System (otherwise known as “XY Platform”) is a student project aimed at teaching mechatronics engineering students the principles of real time software, as well as incorporating knowledge from sensors, actuators, and control systems. Working with real-world, non-ideal devices has also given the students a more realistic experience of engineering work.

Real time software is an important aspect of control engineering, its principle lies in the ability to control the environment by receiving and processing data and returning the results fast enough to affect the environment in real time. This has many practical applications in ‘mission critical’ applications such as anti-lock brakes systems, aircraft navigation, and automated stock trading, fields mechatronics engineers can be expected to enter.

Students are issued National Instruments’ “ELVIS II” virtual instrument suite, motor controller and an accompanying motorized XY platform. The initial task is to explore the many features of LabVIEW to generate a program that can control the XY platform’s position, speed and direction while developing a control scheme that prevents the XY platform’s slider from colliding with its borders. Afterwards, students are to use these basic controls to draw circles and rectangles of various sizes and finally, create a friendly user interface.

Clinton contributed mainly to the early stages of the programming and the hardware section of this report. William was responsible for task 2, 4, and 5 of the assignment, along with writing of the software section of this report.

Contents

1.0 Introduction

The objective of this project has been broken down into five parts:

1. Control the speed, direction and position of the 2D marking system, while displaying the number of pulses and their frequency from the motor.
2. Create a protection scheme to prevent the slide rail system from colliding with the borders of the XY platform using limit switches.
3. Accurately draw closed circles and rectangles with the slide rail system.
4. Develop a user-friendly interface for the users of the program

This report will first detail the workings and limitations of the slide rail mechanism on the XY platform, the motor controllers, and the NI ELVIS II board. It will then describe the program we have created, and the steps taken to achieve this. Finally, it will discuss some of the design decisions made to overcome the limitations of the hardware.

2.0 Hardware Overview

2.1 XY Motorised Platform

The XY platform is a custom-made piece of hardware designed for student projects. It moves a pencil across a two-dimensional cartesian plane using two DC motors, two pair of slide rails and a pulley system.

The marking instrument is held within a rigid centre block, sitting on two sets of slide rails. One pair of slide rails allows the pencil holder to move horizontally, the second pair allows vertical movement. Connections between the rails are supported with bearings to minimise friction. See Figure 1 for visual representation of XY Platform.

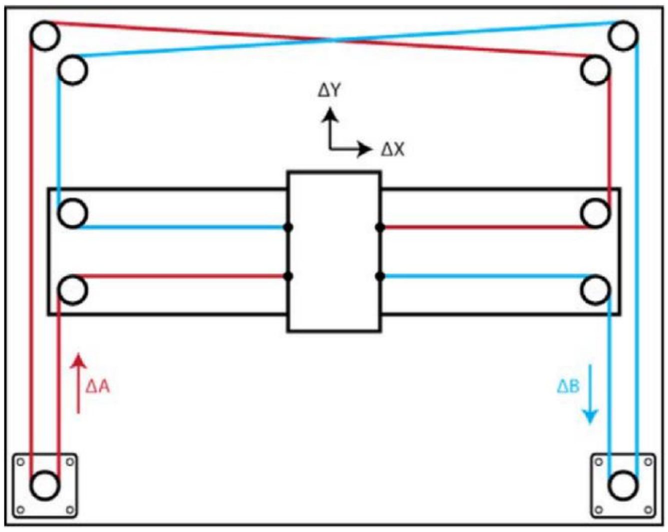


Figure 1 XY Platform

Movement of the pencil is controlled using two DC motors, each connected to the centre block via toothed belts. When powered on individually, the motor moves the holder diagonally across the board, either towards or away from the motor. When motors are turned on together, the resultant movement is the sum of the two independent motions. To see the relationship between motor voltage and pen movement, see Table 1 and Figure 2.

Direction of Pen	Mot-2	Mot-1
H-R	+	+
H-L	-	-
V-U	+	-
V-D	-	+
Dg-RU	+	0
Dg-LD	-	0
Dg-LU	0	-
Dg-RD	0	+

H: Pen moves horz.	V: Pen moves vert.	R: Pen moves right	L: Pen moves left
U: Pen moves up	D: Pen moves down	Dg: Pen moves diag.	
+: Motor turns left (CCW)	-: Motor turns right (CW)	0: Motor stop	

Table 1 Relationship Between Motor Voltage and Movement Direction

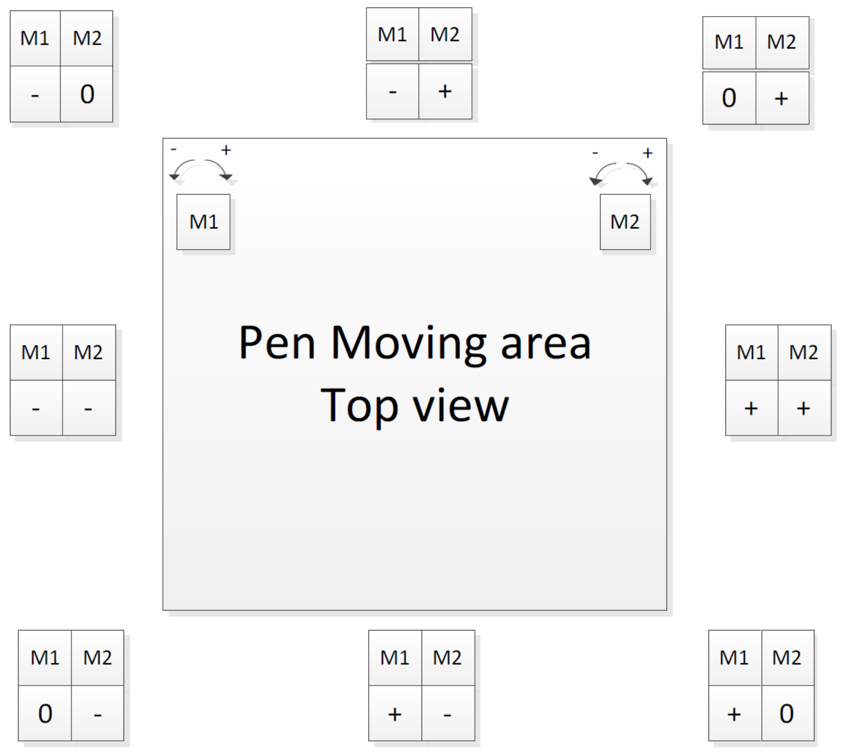


Figure 2 Relationship Between Motor Voltage and Movement Direction

2.2 Limit Switches

The platform is home to four limit switches (Figure 3), marking the extremities of the workspace. When the holder is near the edge of the platform, the appropriate limit switch is triggered, which sends a signal to the LabVIEW program.

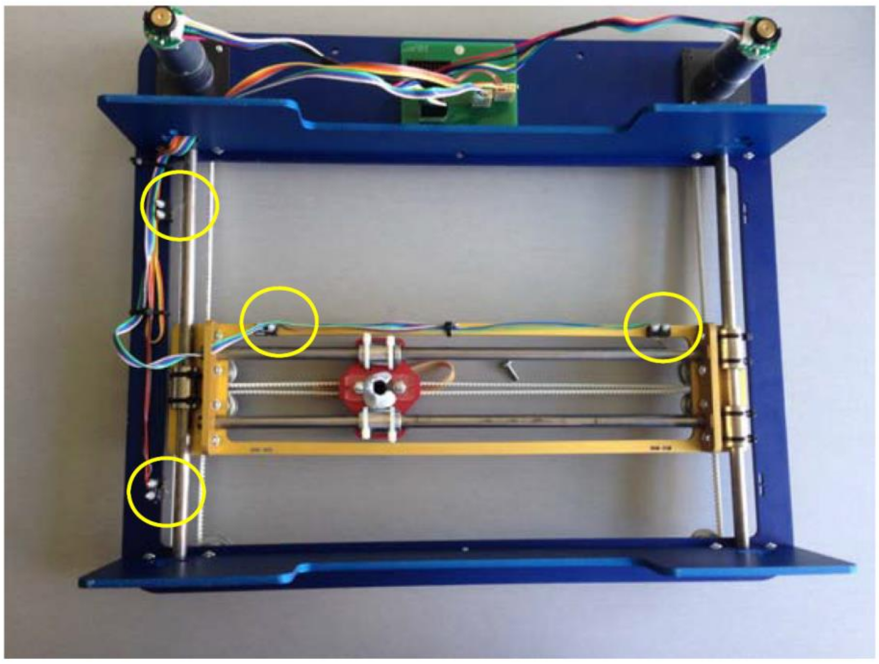


Figure 3 Limit Switch Locations

2.2 Motors

The motors used are POLOLU 2288 brushed DC gearmotor. The motor is intended to be operated at a range between 3 to 9 volts, higher voltages may damage the motors, while lower voltages may not provide enough torque to overcome inertia. This is a main source of limitation, and required a variety of software compensations, these will be discussed in the software portion of the report.

The motor uses two channel hall effect encoders, each reading rising and falling edge signals, allowing for 48 pulse counts per revolution. For this project, we had access only to one channel, and used only rising edge signal, which corresponds to 12 pulses per revolution of the motor shaft. The specific motor used has a gear ratio of 172:1 to step up the torque, this results in an encoder count of 2064 (172*12) per revolution of the output shaft.

2.3 Motor Controller Board

The motor controller is the main interface between the NI ELVIS II board and the XY platform. It is responsible for distributing an externally sources 24 volts DC power to the motor drivers. This is activated through a three-mode switch:

- off (centre position) will not provide any power to the motors;
- test (top position) will supply power to the motor until the switch is released;
- run (bottom position) will continuously supply power to the motors.

The controller board also provides LED indicators for power supply and limit switches, along with a few other features that the students need not worry about. These features are over current and over temperature protection, manual setting of power supplied to motors, and the isolation of power subsystem and instrumentation subsystem.

The motor controller used is V9311-Four Quadrant Linear Output motor controller (Figure 4), it takes in two separate voltages, VE is connected to a 24V DC source for powering the motor, and VREF is set between 10 and -10 to control the motor's speed.

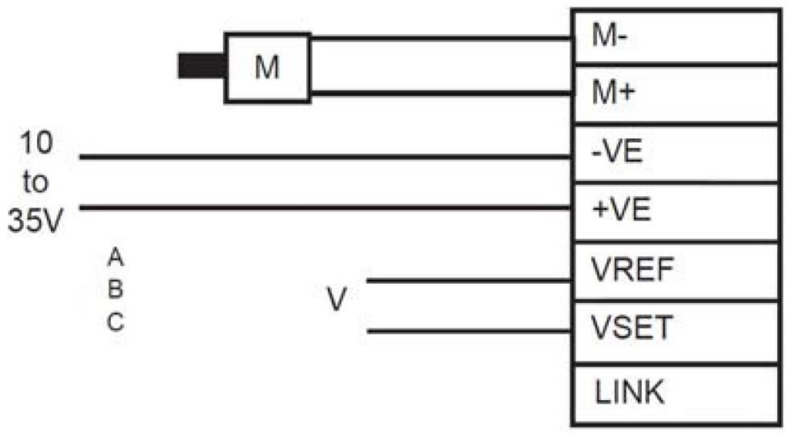


Figure 4 V9311-Four Quadrant Linear Output Motor Controller Schematic

2.4 NI ELVIS II

The ELVIS II board from National Instruments is responsible for the communication between the PC and the motor controller (and by extension the XY platform). Connection to the PC is done with a USB cable, while connection to the motor controller uses a PCMCIA connector. To control the XY platform, NI ELVIS reads signals from the limit switches and encoders, and sends two analogue signals to the motors. It also provides an interface for real time programming of the system using LabVIEW.

3.0 LabVIEW Programme

LabVIEW is the real time software used to control the XY platform. This section has been broken down into 9 sub-components, each with an explanation of functionality, implementation method, problems and areas for improvement. To maintain a level of brevity, the following will assume the reader has a rudimentary understanding of basic LabVIEW blocks.

3.1 Reading and Writing Data

Data communication between the PC and the hardware is completed using a series of Data Acquisition Vis and Functions created by National Instruments. These subVIs communicate with the ELVIS II board receive encoder pulse counts from the motors and sends analogue signal to each motor individually. This is the basis on which all future program is based.

3.2 Pulse Counts and Frequency

3.2.1 Cumulative and Net Pulse Counts

Encoder (pulse) counts received from the physical channel is referred to as cumulative pulse count, since its values increase regardless of the direction the motor spins in. Some future applications such as position control requires the direction of movement, therefore a 'net pulse count' was programmed.

Net pulse count logic (Figure 5) first calculates the pulse increments in every iteration of the while loop using a feedback node. It then checks if the motor voltage is negative, if so, the value is multiplied by -1. Finally, it sums each loop's pulse counts together.

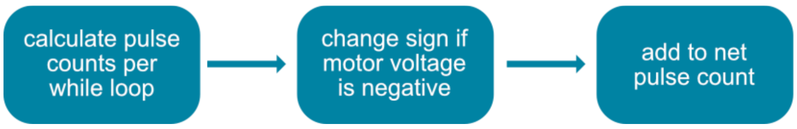

calculate pulse counts per while loop
change sign if motor voltage is negative
add to net pulse count

Figure 5 Net Pulse Count Logic

- This implementation has its flaws, the function will not adjust its values should the voltage change within an iteration of the while loop. However, as each loop iteration runs for only a few milliseconds, the error is marginal and can be ignored.

3.2.2 Frequency

Frequency (Figure 6) is calculated by dividing the number of encoder counts by the time taken. To account for changing speeds, the calculation for pulse frequency is restarted whenever the motor voltage changes. To implement this function, when the voltage sent to a motor changes, the function stores the last number of cumulative pulses before the change and resets the timer. The function then proceeds to divide the difference between current pulse count and last pulse count by the time.

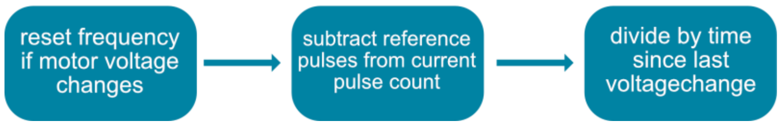

reset frequency if motor voltage changes
subtract reference pulses from current pulse count
divide by time since last voltagechange

Figure 6 Frequency Logic

3.3 Speed Control

As detailed in the hardware section of the report, the motors' speed changes by varying the analogue signal sent to the motor controllers. The relationship between voltage and speed is proportional, but not strictly linear. As precise velocity control is not necessary for this project, the voltage set to the motors can be a proxy for speed without any repercussions.

3.4 Direction

The marking device's direction of movement can be controlled via one of two methods: a pair of X/Y axis sliders, and a dial with exact angle control. The two methods are not mutually exclusive, the user can activate them simultaneously, and the resultant movement will be a vector addition of the two directions.

The sliders apply the correct voltage to move the slider horizontally or vertically across the platform using the information in Table 1. Both sliders have a voltage range of -10 to 10 volts and the two can be used together to achieve angular movements.

We have also implemented an angular direction control (Figure 7) for those who require movement in precise directions. The user can choose the exact angle (from the positive x axis) and voltage to apply. The angle is then converted to its vertical and horizontal components using sine and cosine functions, and subsequently applied to the motor in the same fashion as the sliders.

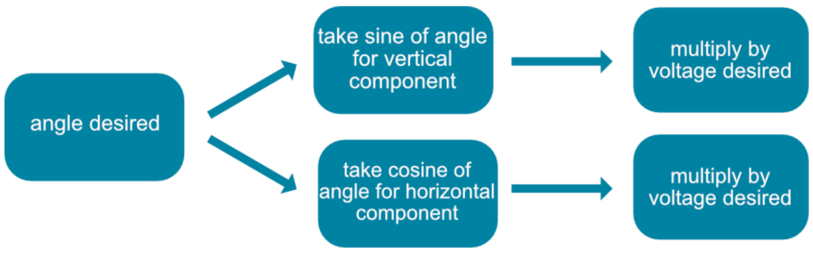

angle desired
take sine of angle for vertical component
multiply by voltage desired
take cosine of angle for horizontal component
multiply by voltage desired

Figure 7 Angular Direction Control

3.5 Position Control

Position control is a function that employs a proportional controller (Figure 9) to allow the user to define a location on the platform and have the marking device reach that point with high precision. The resultant program can move the marking device to any points on the platform with an accuracy of ±1mm.

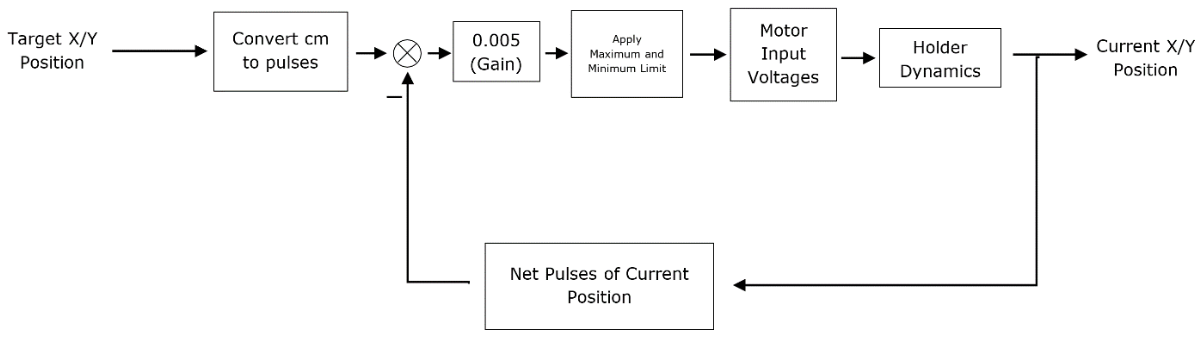


Figure 8 Position Control Diagram

The control scheme first converts the target position from metric to the number of pulses required (Figure 9). It then actively calculates the difference between the desired number of pulses and the current number of pulses, this is the error out. The program then applies a small gain to convert the error into a reasonable number for voltage input and applies them to the motors. For full conversion between pulses and distance, refer to section 5.1.

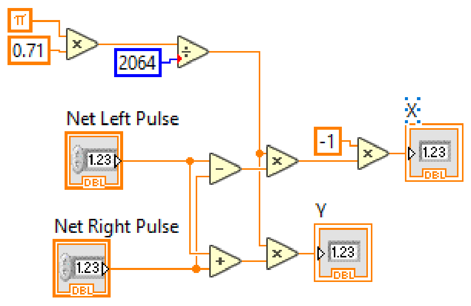


Figure 9 Conversion Between Pulses and Distance

- There are a few considerations when applying this feedback to the voltage, namely the maximum and minimum voltages. Voltage larger than the 10V limit can damage the motors. This problem is solved using an in range and coerced block which limits the output to between -5 and 5 volts. Small voltages, on the other hand, often do not generate enough torque to overcome inertia. This was solved by setting a minimum absolute voltage of 1 volt to ensure the motor always moves when intended to.

3.6 Limit switch

An important aspect of the programme is to ensure the platform does not damage itself, this is done through use of the limit switches. When a limit switch is hit, the program will only allow the motors to move in directions away from or parallel to it. To achieve this, the program checks if the voltage that the user tries to apply will cause the slider to move towards an already depressed limit switch. If this is true, both motors will receive zero volts until the direction of movement is changed.

3.7 Drawings

3.7.1 Rectangle

The rectangle function takes two inputs from the user, width and height, and moves the marking device to draw a rectangle of the intended size. The final rectangle drawn using our program has the correct size and all edges are at right angle to each other. Refer to Figure 11 in section 5.2 for a picture of the output.

The program first converts the inputs into pulse counts, then moves in all four directions for the appropriate number of pulses to form the four sides of a rectangle. Each time the desired point is reached, the marking device changes direction, until finally, it reaches its origin and ceases to move.

- One physical limitation we came up against is that the two motors do not start at the same time, this resulted in a slight uptick on the first stroke of rectangle. This could potentially be countered using a close looped feedback control that constantly checks if the marking device is on its correct trajectory. However, as this only slightly affected the rectangle’s final shape, we deemed it an acceptable error.

3.7.2 Circle

The circle function takes one input, the radius, and is drawn using the precise angular direction control detailed earlier. It was recognised that iterating through 0 to 360 degrees at constant voltage will create a shape that resembles a circle. Doing this will essentially be applying the tangential velocity at each point on the circle’s circumference. Figure 9 shows a visual representation of the logic using iteration size of 45 degrees, the arrows represent the angle of voltage applied. Refer to Figure 11 and 12 in section 5.2 for a picture of the output.

To program this, we used a case structure, where every time it is run, the angle of movement is incremented by one degree, and stops after it reaches 360. The amount of time spent in each iteration determines the size of the circle, therefore the radius input is what dictates the timer of the loop.

We have also encountered the problem of the cut off voltage mentioned earlier, this resulted in part of the circle's circumference being flat. To counteract this problem, we applied an offset of 1 degree like our position control. The result is a closed circle with smooth curves.

- We have considered a few alternatives to this implementation, one of which is to calculate each point on the circle's circumference and use closed loop position control to move along them. This proposal was abandoned for its increased complexity and tendency to create less smooth circles.

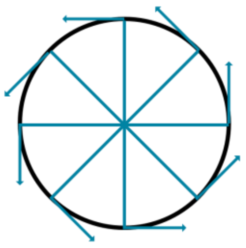


Figure 10 Visual Representation of Circle Logic

3.8 Home Function

To aid with the position control function, we created a helper function called 'home'. Home button, when pressed, returns the marking device to the bottom left of the platform and sets that as origin (0, 0) for future movements. This function is by default turned on when the program is run for the first time to allow the slider to set its origin.

This function first moves the slider down the platform, when the limit switch is hit, the slider moves left. When both bottom and left switches are depressed, home position is reached, and both net pulses are set to zero.

3.9 User interface

The user interface is an important part of the program, for most users it is the only part of the program they will see. Therefore, we needed to design it to be as user friendly as possible, this applies to both our code (block diagram) and our user interface (front panel). Our block diagram needs to be uncluttered enough such that alterations to our program can be easily performed without the help of the original programmer. Our front panel needs to be sufficiently user friendly such that users without prior knowledge can still easily learn how to operate the slide rails.

3.9.1 Block diagram

From the beginning, we realised that if left unintended, LabVIEW's block diagram can quickly become cluttered. Therefore, we set out to design our block diagram to be tidy and understandable. Each major function is separated into its own 'cluster', separated using local variables and subVIs. Inspiration was taken from the principles of object orientated programming, where each object is independent to another. Section 5.3 of the appendices shows the block diagrams of the individual functions.

The benefits of this layout are vast and plenty, reducing the number of wires across the block diagram improves the readability of the program. This not only makes life easier for us while we designed the program, but more importantly allows future programmers to easily understand the code being written and modify it without the help of the original programmer.

3.9.2 Front panel

The front panel is the main user interface for this program and is a place where we spent considerable effort designing. It provides the user with enough controls and indicators to achieve all that he/she needs, while being easy to use so as not to intimidate first time users. On top of this, it should have a coherent design and colour scheme.

The result is a control panel with a consistent colour scheme that has well defined sections for different functions (Figure 11). The front panel offers the user the controls to use all of the aforementioned functions with the addition of a graphic display of the pencil's current position. This feature was included as it allows the user to easily gauge the slider's position in the workspace should he/she be away from the actual platform, this can become useful in cases of remote operation.

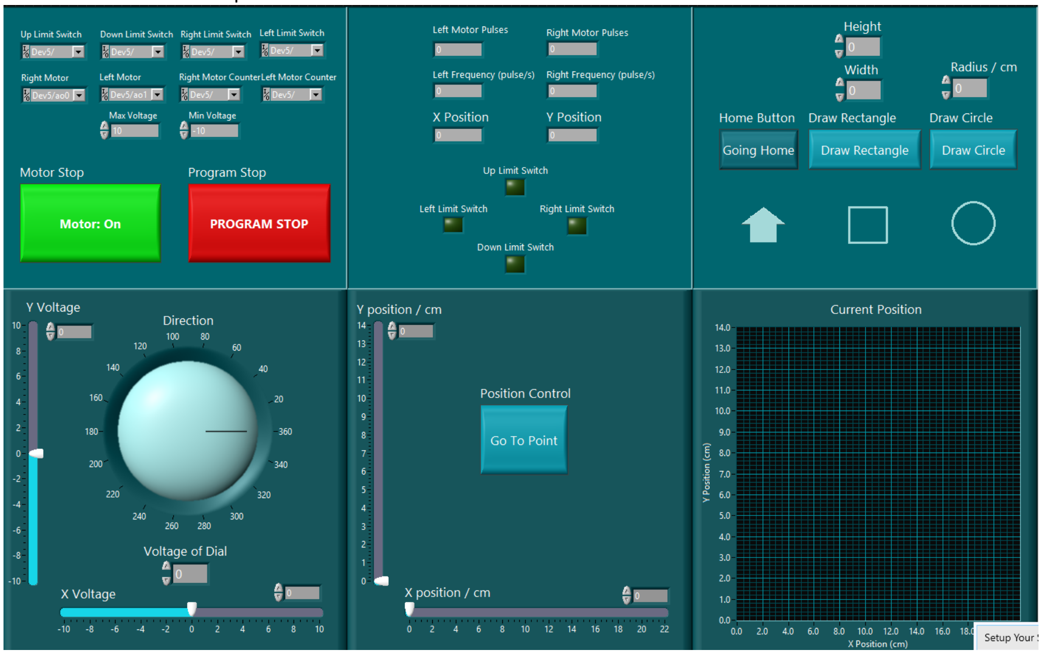


Figure 11 Front Panel

4.0 Conclusion

This project required students to design a program in LabVIEW that can accurately control the trajectory of a 2D marking device. Our final program can control the slider's speed; give multiple forms of direction control; move its position with high precision; protect it from colliding with the platform's borders; draw closed circles and rectangles of variable size; and present it all in a user-friendly manner. We have thus met all the requirements we initially set out to meet.

5.0 Appendices

5.1 Calculations

Pulses to cartesian calculation

P1 = (1032*(X-Y)/(pi*0.8)

P2 = (1032*(X+Y)/(pi*0.8)

X = (-0.8*pi)*(P1+P2)/2064

Y = (0.8*pi)*(P1-P2)/2064

5.2 Drawings

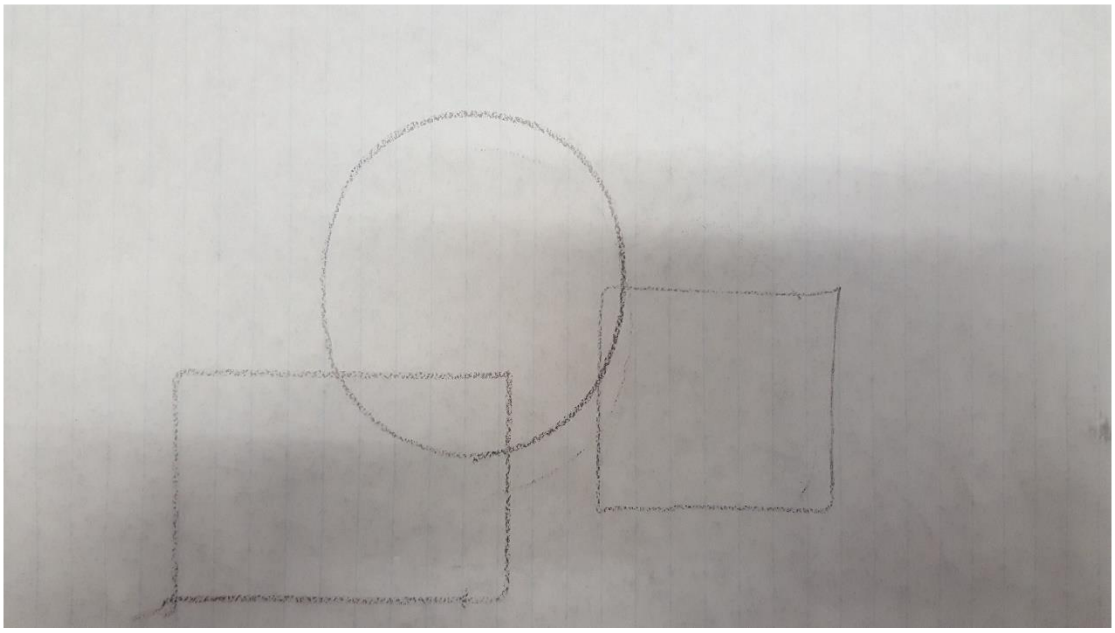


Figure 12 Picture of Drawings 1

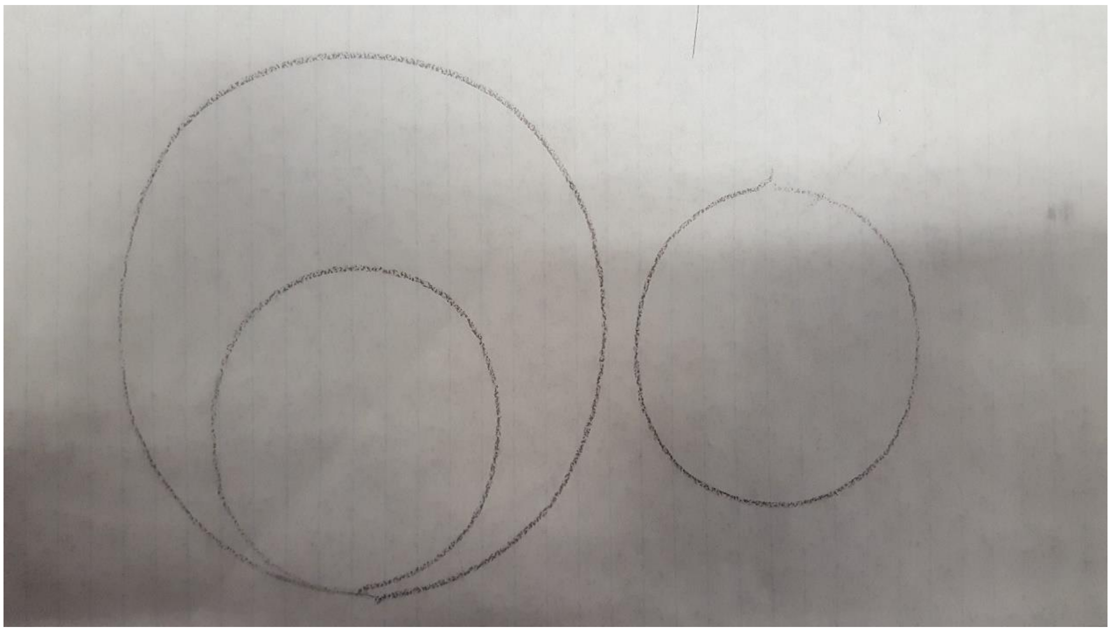
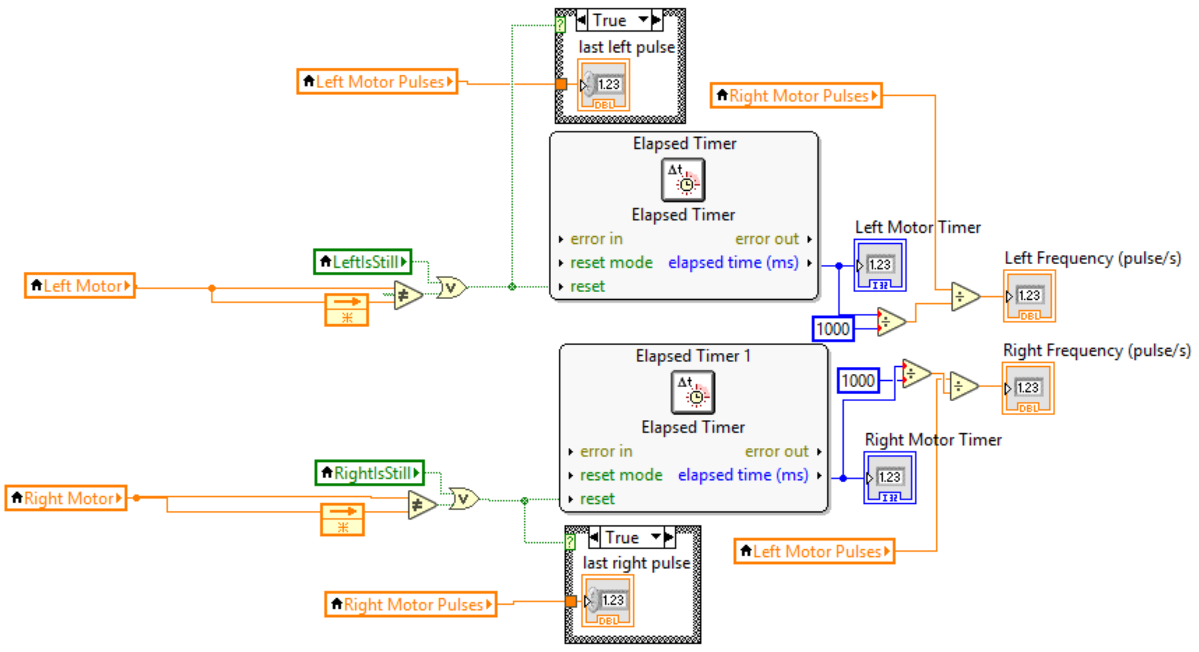
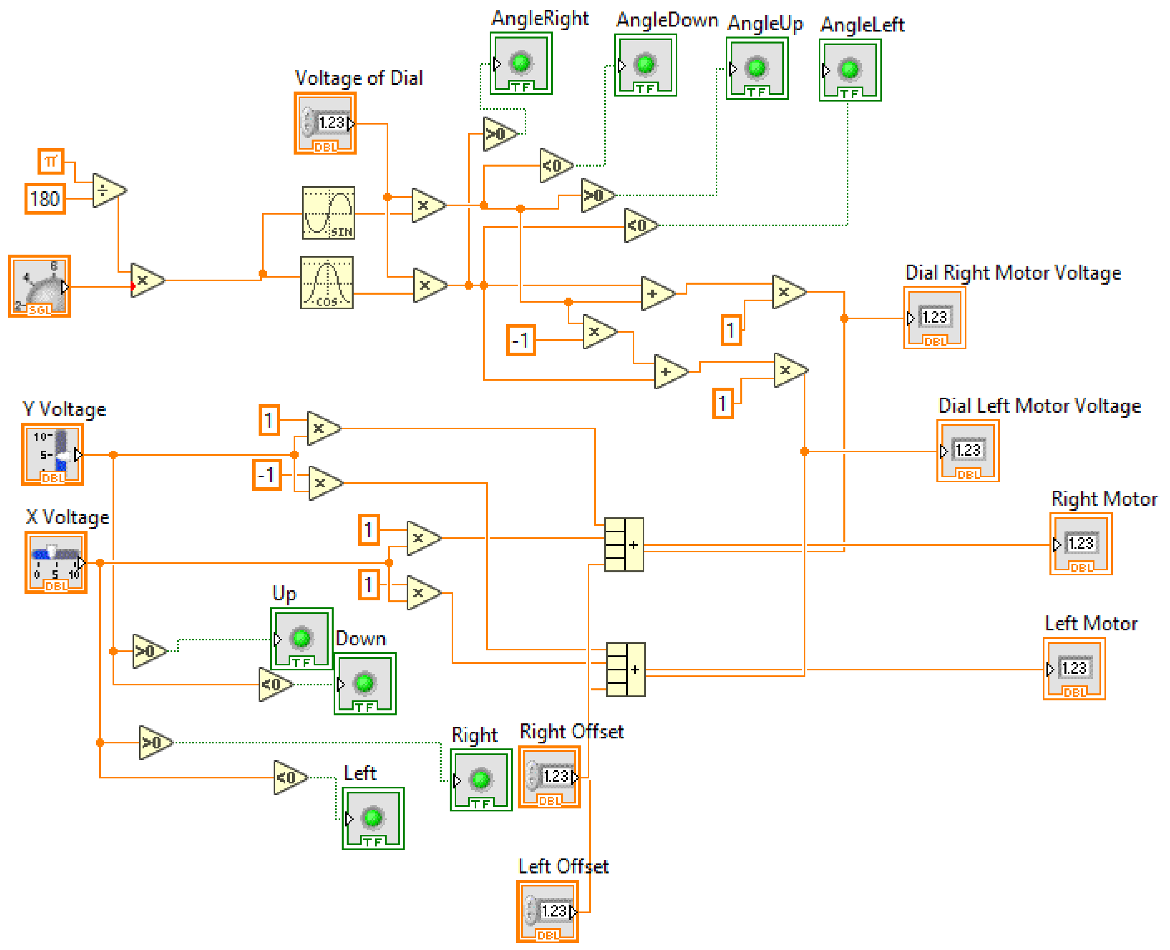
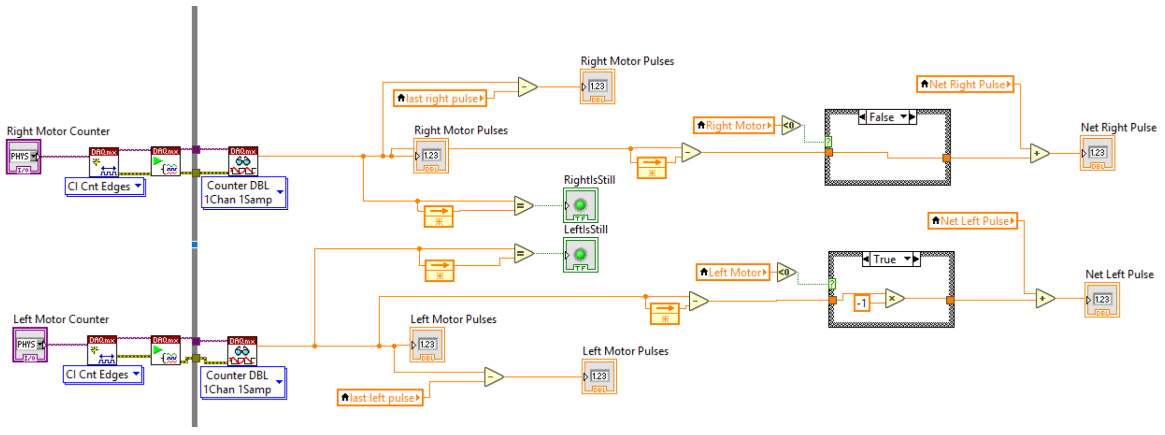


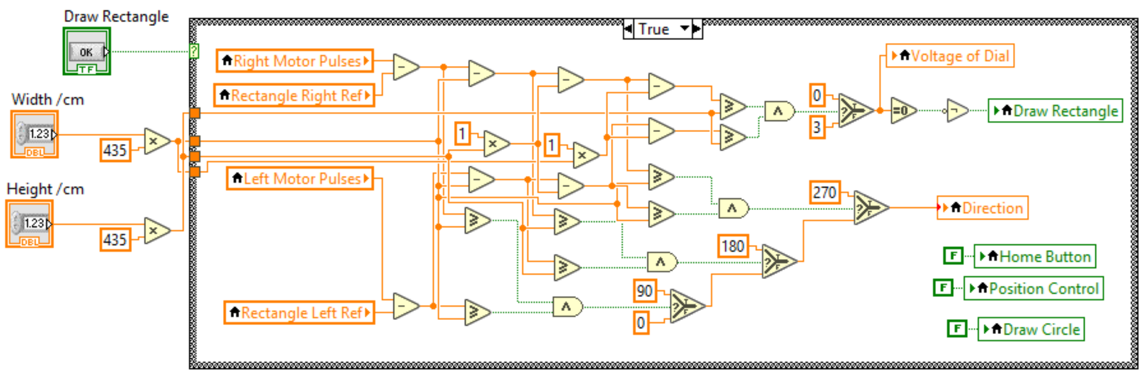
Figure 13 Picture of Drawings 2

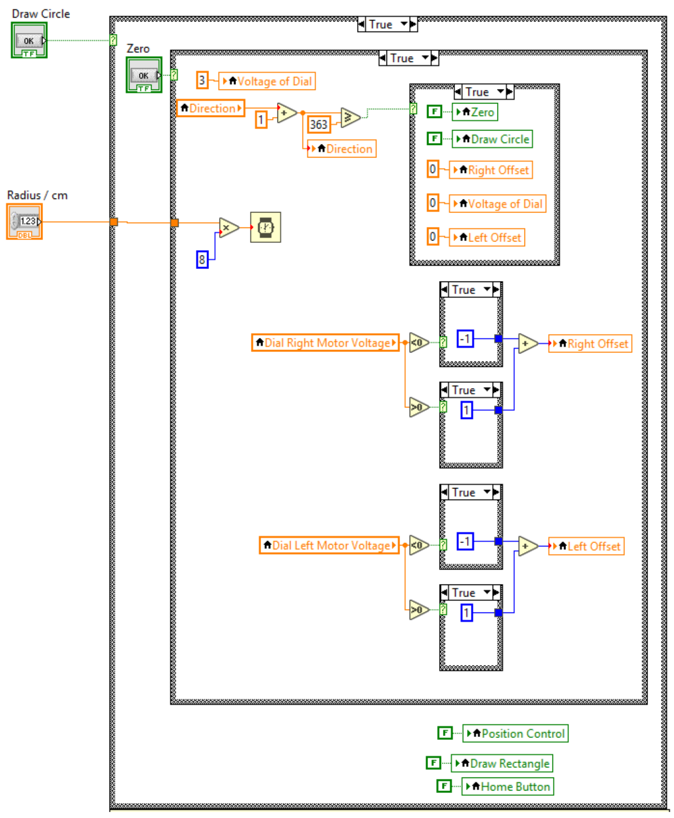
5.3 Block Diagram

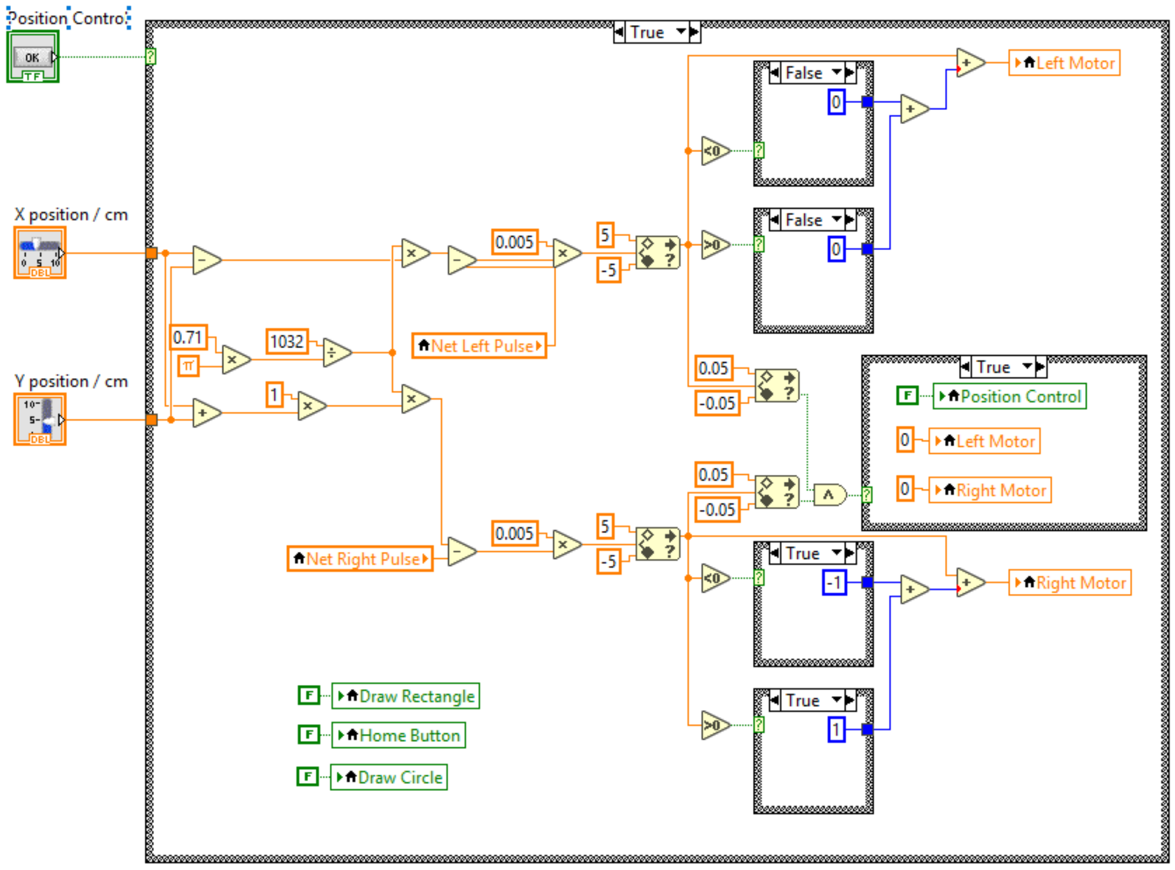




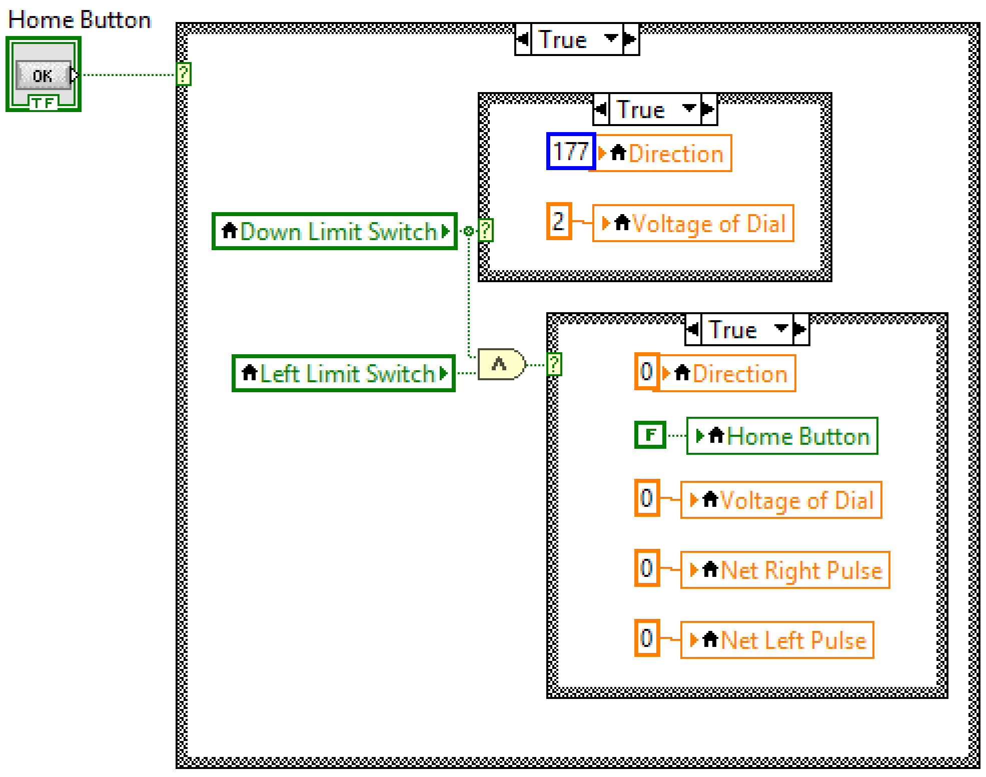

Right Motor Counter
CI Cnt Edges
Counter DBL 1Chan 1Samp
Left Motor Counter
CI Cnt Edges
Counter DBL 1Chan 1Samp
Right Motor Pulses
last right pulse
Right Motor Pulses
RightIsStill
LeftIsStill
Right Motor
False
Net Right Pulse
Net Right Pulse
Net Left Pulse
Left Motor
True
-1
Net Left Pulse
Left Motor Pulses
Left Motor Pulses
last left pulse


Draw Rectangle
True
Width /cm
435
Height /cm
435
Right Motor Pulses
Rectangle Right Ref
Left Motor Pulses
Rectangle Left Ref
1
1
Voltage of Dial
0
3
Draw Rectangle
270
Direction
180
90
0
Home Button
Position Control
Draw Circle


Draw Circle
True
Zero
True
3
Voltage of Dial
Direction
1
363
Direction
True
Zero
Draw Circle
0
Right Offset
0
Voltage of Dial
0
Left Offset
Radius / cm
8
Dial Right Motor Voltage
True
-1
Right Offset
True
1
Dial Left Motor Voltage
True
-1
Left Offset
True
1
Position Control
Draw Rectangle
Home Button



Position Control
X position / cm
Y position / cm
True
False
0.005
5
-5
0.71
1032
1
Net Left Pulse
Net Right Pulse
0.05
-0.05
Position Control
Left Motor
Right Motor
-1
Draw Rectangle
Home Button
Draw Circle



Home Button
True
177
Direction
2
Voltage of Dial
Down Limit Switch
Left Limit Switch
0
Home Button
Net Right Pulse
Net Left Pulse