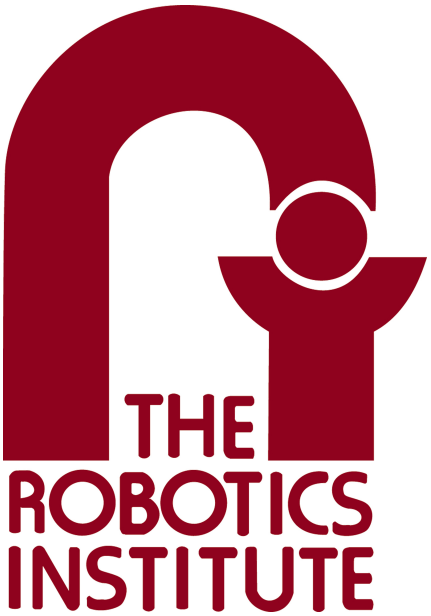

MRSD Project Course

Team I – AIce

Autonomous Zamboni Convoy

Individual Lab Report 1

Team

Rathin Shah

Nick Carcione

Yilin Cai

Jiayi Qiu

Kelvin Shen

Author

Rathin Shah

Feb 10, 2022

Contents

1. Individual Progress

1.1. Sensors and Motor Lab

For the Sensors and Motors lab, my responsibility was to create a subsystem that comprised a 10 kOhms 1 turn trimmer pot and a servo motor. Based on the degree of rotation of the knob of the potentiometer, the servo would rotate proportionally from its rest position. The larger the rotation, the greater the deflection of the servo from its rest position. The control of the position of the motor was in both directions based on the direction of the rotation of the potentiometer. Also additionally, I included LEDs, which would glow up when the servo motor used to reach its extreme position (180 Degrees). The circuit is shown below in Figure

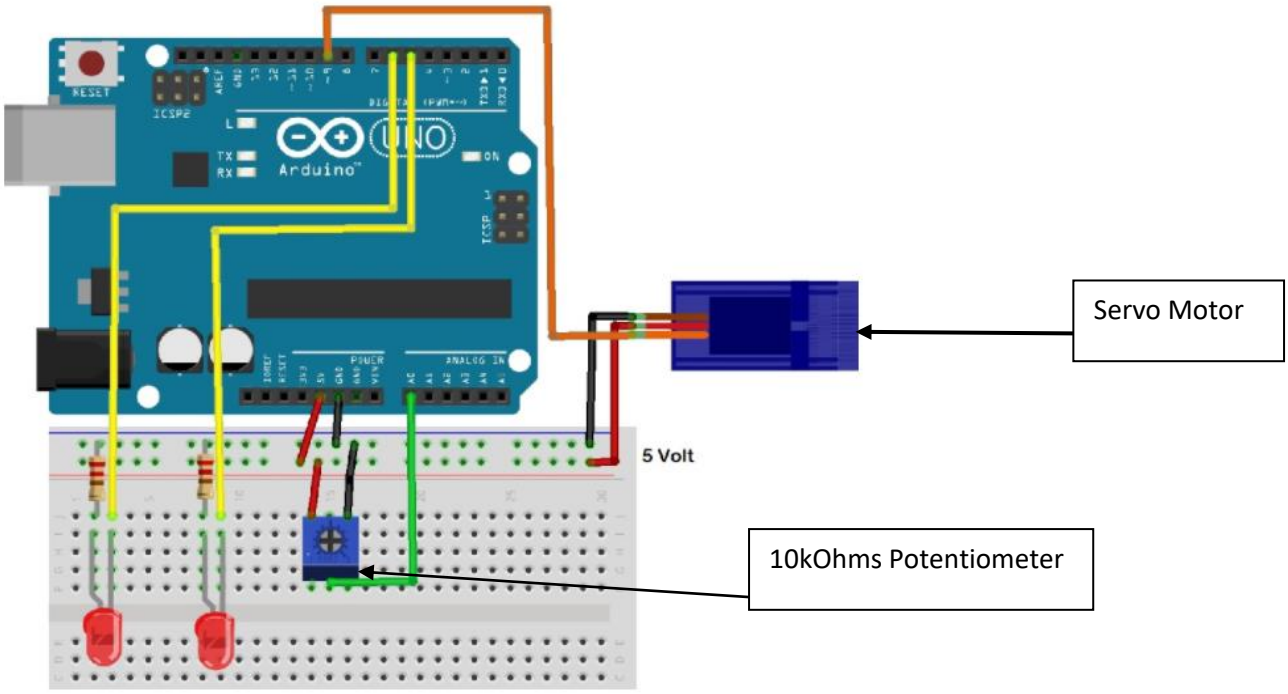

Figure 1: Schematic to interface potentiometer and servo motor to Arduino

I also coded a provision that would allow the user to move the motor to the desired position, via the GUI. Additionally, I helped with hardware integration of the complete system as well as debugging the DC motor and Servo Motor integration with GUI.

1.1.1. Servo Motor

Servo motors are commonly used in robotics for precise control of angular positions. The shaft of the servo motor can typically be made to move between 0-180˚. The servo motor used is the HS-311. The electrical connections are 5V, ground, and a PWM output connected to the Arduino. Since the Arduino has a 5V output pin, no external power supply is needed for the servo. The servo motor is controlled using the Arduino “Servo.h” library by providing a value ranging from 0 to 180 to the “write” function. The input value for the function correlates to the physical angle that the servo moves to. Since the servo motor is connected

to a PWM Pin (analog output) of the Arduino, it can be moved by varying angles, proportional to the rotation of the sensor.

The goal of RC servo task was to implement a position control method to map sensor readings to servo angular position. By investigating into the specs of our RC servo, I connected the servo following the way as shown in Figure 2. I powered the servo directly with an Arduino 5V pin, which was also the way many online resources suggested to do. Since Arduino already had a servo library, for position control, I only needed to input desired servo angle in degrees, which was then sent to servo hardware through the PWM pin. I also had to map the pwm signal voltage to 0 to 180 degrees because of an issue I found that when I input a number larger than 180 or smaller than 0, the servo would turn to its limit and begin vibrating. To avoid the vibration causing damage to servo hardware, I limited the acceptable input range from 0 to 180 degrees, while the servo would not respond and the program would return an error message if an integer outside [0, 180] range was received.

1.1.2. Potentiometer

The potentiometer used is 3386F-1-103LF, a 10 kOhms 1 turn trimmer pot. The potentiometer has three electrical connections - 5V, ground, and an analog output connected to the Arduino.

The working of a circuit with a potentiometer is voltage division in practice.

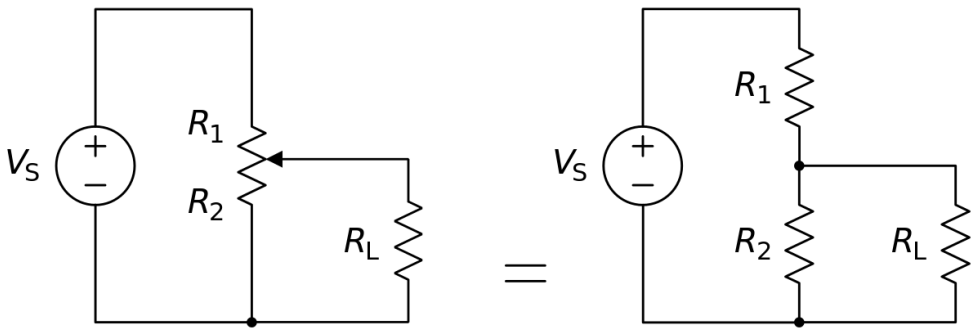

Figure 2: Potentiometer Circuit

In the schematics above, the resistors R1 and R2 are part of the pot itself. RL is the resistance of the load (a motor, a LED, or whatever). When you turn or slide the pot, you will change the values of R1 and R2. Moving the pot one way will reduce R1 and increase R2, and vice versa. This results in various voltage values on the wiper (the one that goes to RL in the schematics above). The voltage value itself can be calculated by using the following equation:

where VL is the voltage difference between the wiper and the output pin, and VS is the voltage difference between the input and output pin

$$V_{\mathrm{L}} = \frac{R_2}{R_1 + R_2} \cdot V_s.$$

The potentiometer knob angle can be anywhere between 0 and 300 degrees. When the knob is at 0 degrees, the analog value is 0. When the knob is at 300 degrees, the Analog value is 1023. The servo motor's movement is constrained to be within 0 and 180 degrees. Mapping 0-300 to 0-180 allows the position of the servo motor to be proportional to the position of the knob.

1.1.3. Full Circuit

The full circuit assembled and tested by all team members can be seen below

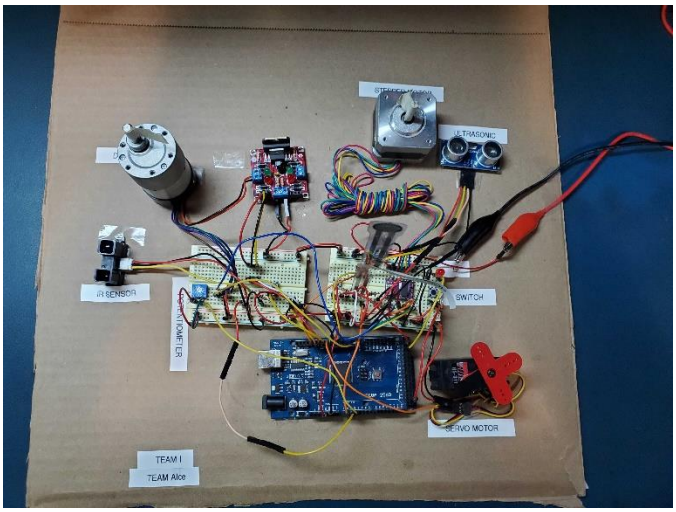

Figure 3: Complete Integrated Circuit

Appendix A shows the final code for this lab. I wrote the code pertaining to the servo motor and potentiometer. I also worked with Kelvin to integrate the GUI with the microcontroller and Yilin to integrate the hardware. As such, I also developed the code for overall program flow and state switching.

1.2. MRSD Project - Autonomous Zamboni Convoy

For MRSD Project, In the past month I worked majorly on two subsystems:

1.2.1. Developing Steering Controller using Pure Pursuit Controller in Simulink

As per our cyber-physical architecture, based on waypoints generated by the path that needs to be followed, the steering controller should generate the steering angle command to make the vehicle follow the waypoints.

To get the desired functionality, I developed a Pure Pursuit Steering Controller in Simulink for an Ackermann-based geometry, which can communicate with ROS Framework, on which our complete navigation stack is implemented. The Simulink model is shown in the figure below. Here I assumed we have the waypoints data which is getting published on ROS Node, to which my Simulink is subscribing. Based on this information, the controller published the front wheel steering angle (for each wheel) command on the required ROS Node.

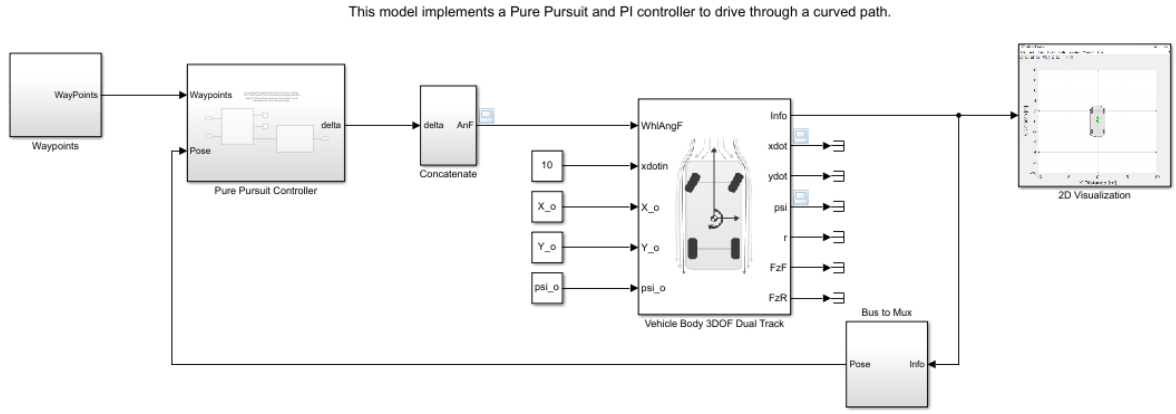

Figure 4: Pure Pursuit Controller

1.2.2. Developing Algorithm to fuse the odometer and IMU data for vehicles localization

Another important subsystem in our project is having a system that can fuse the data from the IMU that is on the Zamboni and the data from the wheel odometry to get accurate localization information of the vehicle. I used the Robot_Localization ROS package to carry out the task. Here I implemented an EKF (Extended Kalman Filter) to fuse the data from the sensors. This package subscribes to the topic odom/ where the wheel odometry data is published. It also subscribes to the topic /imu and listens to the IMU data getting published. It then implements the Kalman filter to fuse the appropriate data. The data is fused as follows:

Odometry: Robot Pose (x,y,z, and yaw)

IMU: x velocity, y velocity, and yaw velocity.

The covariances used in EKF for both sensors are to be tuned with the simulation. However, for now, they are the default values by the package.

1.2.3. Project Management

Other than this, I am responsible for project management, where I am making sure that the project is on a timeline concerning the schedule. Along with, managing and coordinating all the meetings with the Sponsor, Faculty and other stakeholders. I have prepared a project

management system where a Macro enabled Excel sheet is used to keep track of progress, requirements, and inventory.

2. Challenges

2.1. Sensor Motor Lab

The major challenges were faced during the integration phase.

The servo was connected to the pin(9) of Arduino, which is PWM Enabled. For DC motor, we used the PWM Pin(10), and the PWM capability on pins 9 & 10 is disabled, both the motors were not working simultaneously. We spent some time debugging the issue, and eventually realized the mistake and rectified it. Other than this, the major challenges faced while doing this circuit are mentioned below:

- Not enough memory on Arduino UNO to host rosserial_arduino due to which had to switch to Arduino mega
- Lack of GPIOs on Arduino UNO (We had 3 switches), and so had to switch to Arduino mega
- Sensor readings from IR and ultrasonic sensors were noisy, so we had to test and tune the noise filter like moving average for better reading

2.2. MRSD Project

One major challenge was integrating the MATLAB-Ros communication setup, especially synchronizing the master and client simulation clock. I had to spend some time debugging the communication lag. Another challenge was tuning the localization package's Kalman filter for better localization estimation of the Zamboni. I had to tune the covariance matrices of the sensors – IMU and wheel encoder to get better localization results. As we don't have access to Zamboni, and it can be delayed indefinitely due to covid, we are facing the challenge of validating and verifying our sensors as well as algorithms on the actual hardware

3. Teamwork

Kelvin Shen

Sensor and Motor Lab

Kelvin worked on GUI implementation and GUI-Arduino interface design.

Project Autonomous Zamboni Convoy

Kelvin's major contribution is that he Generated a mesh file of a board of ArUco markers with an appropriate size and tested the ArUco wall with Zamboni model inside the Gazebo environment

Nick Carcione

Sensor and Motor Lab

Nick worked on developing code for DC motor control using IR sensor where he implemented position and velocity control of the motor

Project Autonomous Zamboni Convoy

Nick Researched methods and packages for fusing wheel encoder and IMU data to obtain accurate velocity estimation of follower

Yilin Cai

Sensor and Motor Lab

Yilin worked on controlling the stepper motor with an Ultrasonic sensor where he implemented a switch with debouncing to select ultrasonic or flex sensor as a control input to the motor.

Project Autonomous Zamboni Convoy

Yilin’s major focus was on setting up and maintenance of the simulation environment as well as URDF (XACRO) file development of the Ackermann steering Zamboni with sensor and controller plugins. He also implemented vehicle motion command with keyboard teleoperation

Jiayi Qui

Sensor and Motor Lab

Jiayi worked on controlling the stepper motor with the Flexforce sensor. She also plotted the transfer function for the flex sensor by measuring the force that is applied to the sensor and logging the corresponding voltage.

Project Autonomous Zamboni Convoy

Jiayi investigated vehicle simulation in Gazebo and also helped to build an ice rink simulation environment

4. Future Work/Plans

In the coming weeks, my focus will be majorly on developing and verifying the Follower Zamboni's localization in the world. Once that is done, I will focus on establishing a robust and consistent communication between MATLAB and Simulink which can be compatible with hardware as well. As the work of generating waypoints and path is done in parallel, I will verify and validate the performance of the steering controller with these packages. Another important task will be finalizing a platform on which we can validate the algorithms whenever we don't have access to Zamboni. This is a very important risk mitigation plan as there are chances that the access to Zamboni will be delayed by a few months.

5. Appendix A

Arduino code for the servo motor and potentiometer integration

```
#include <Servo.h>
int LowLed = 5;
int HiLed = 6;
Servo servo1;
int pot = A0;
int val;

void setup() {
  servo1.attach(9);
  pinMode(LowLed, OUTPUT);
  pinMode(HiLed, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  val = analogRead(pot);
  val = map(val, 0, 1023, 0, 180);
  servo1.write(val);
  Serial.println(val);
  if (val == 0)
    digitalWrite(LowLed, HIGH);
  if (val == 180)
    digitalWrite(HiLed, HIGH);
  if (val > 0)
    digitalWrite(LowLed, LOW);
  if (val < 180)
    digitalWrite(HiLed, LOW);
  delay(10);
}
```

Figure 5 Arduino Code

6. SENSORS AND MOTORS LAB QUIZ

1. ADXL335 Accelerometer

o What is the sensor's range?

• The sensor has a range of **±3.6g**

o What is the sensor's dynamic range?

• The sensor's dynamic range is **7.2g. (Minimum is 6 g)**

o What is the purpose of the capacitor CDC on the LHS of the functional block diagram on p. 1? How does it achieve this?

• The capacitor CDC is a decoupling capacitor, it decouples the accelerometer from power supply noise. It filters out voltage spikes and passes only the DC component, thus making it smooth.

o Write an equation for the sensor's transfer function.

- The transfer function: **1.5V + (0.3** $\frac{V}{g}$**)** $*$ **a**

o What is the largest expected nonlinearity error in g?

- The largest expected nonlinearity error in g is : **0.3%** $*$ **7.2g = 0.0216g**

o What is the sensor's bandwidth for the X- and Y-axes?

- The sensors bandwidth is 0.5 Hz to 1600 Hz

o How much noise do you expect in the X- and Y-axis sensor signals when the sensor is excited at 25 Hz?

- We apply the following equation to determine the RMS noise:

$$RMS\ Noise = Noise\ Density \times \sqrt{BW \times 1.6}$$

The RMS noise at 25Hz is $(150 \times \sqrt{(25 * 1.6)}) = \mathbf{948.68\ \mu g}$

o If you didn't have the datasheet, how would you determine the RMS noise experimentally? State any assumptions and list the steps you would take.

Following steps will be taken:

1. I will place the sensor on a flat surface.
2. Make sure the surface is disturbance free
3. Collect certain number of readings

4. Use RMS $= \sqrt{\frac{1}{n}\sum_i x^i}$

n: number of recorded samples

x^i : ith sample measurement

. The following assumptions I have made:

- There is negligible change in temperature
- Noise with power source is negligible
- Sensor is on a flat surface

2. Signal conditioning

A. Filtering:

- **Name at least two problems you might have in using a moving average filter.**
  1. In order to filter out significantly high frequency noise, the window size of the filter needs to be large
  2. These filters use convolution which is a slow algorithm, so it can be challenging using this filter for real-time applications
- **Name at least two problems you might have in using a median filter**.

  1. The median average does not reflect changes in measurements instant

  2. This filter has a high computational cost

B. OP-AMPS

- **In the following questions, you want to calibrate a linear sensor using the circuit in Fig. 1 so that its output range is 0 to 5V. Identify in each case: 1) which of V1 and V2 will be the input voltage and which the reference voltage; 2) the values of the ratio Rf/Ri and the reference voltage. If the calibration can't be done with this circuit, explain why.**
  - **Your uncalibrated sensor has a range of -1.5 to 1.0V (-1.5V should give a 0V output and 1.0V should give a 5V output).**

    Given: -1.5V is mapped to 0V; 1V is mapped to 5V.

    Using the equation of op-amp $V_{out} = (V_2 - V_1) * \frac{R_f}{R_i} + V_2$

$$5 - 0 = \left(1 + \frac{R_f}{R_i}\right) * (1.5 + 1.0)$$

$$\text{So } \frac{R_f}{R_i} = 1$$

V_2 is the input voltage and V_1 is the reference voltage.

V_{ref} is -3V

- **Your uncalibrated sensor has a range of -2.5 to 2.5V (-2.5V should give a 0V output and 2.5V should give a 5V output).**

  Given: -2.5V is mapped to 0V; 2.5V is mapped to 5V.

  Using the equation of op-amp $V_{out} = (V_2 - V_1) * \frac{R_f}{R_i} + V_2$

$$5 - 0 = \left(1 + \frac{R_f}{R_i}\right) * \left(2.5 - (-2.5)\right)$$

$$5 - 0 = \left(1 + \frac{R_f}{R_i}\right) * (5)$$

$$\text{So } \frac{R_f}{R_i} = 0$$

  This implies that Ri is infinite, which is not possible, so calibration for this case cannot be done with this circuit

3. Control

o If you want to control a DC motor to go to a desired position, describe how to form a digital

input for each of the PID (Proportional, Integral, Derivative) terms.

The digital input for P term should be the error between current motor position and desired motor position at each time stamp.

The digital input for I term should be the accumulated error between current motor position and desired motor position for a period.

The digital input for the D term should be the rate of change of error (error between current motor position and desired motor position) between adjacent time stamps.

o If the system you want to control is sluggish, which PID term(s) will you use and why?

I will use the P teem. The larger the p's gain is, the quicker the response time will be, as we use P to control the response time

o After applying the control in the previous question, if the system still has a significant steady-state error, which PID term(s) will you use and why?

I will use the I term because the I term is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously, thus proportional to the magnitude of the error. Increasing the I term gain will lead to less steady-state error.

o After applying the control in the previous question, if the system still has overshoot, which PID term(s) will you apply and why?

I will use D term because the D term is calculated by determining the slope of the error over tim. Increasing the gain of D term will lead to less overshoot.