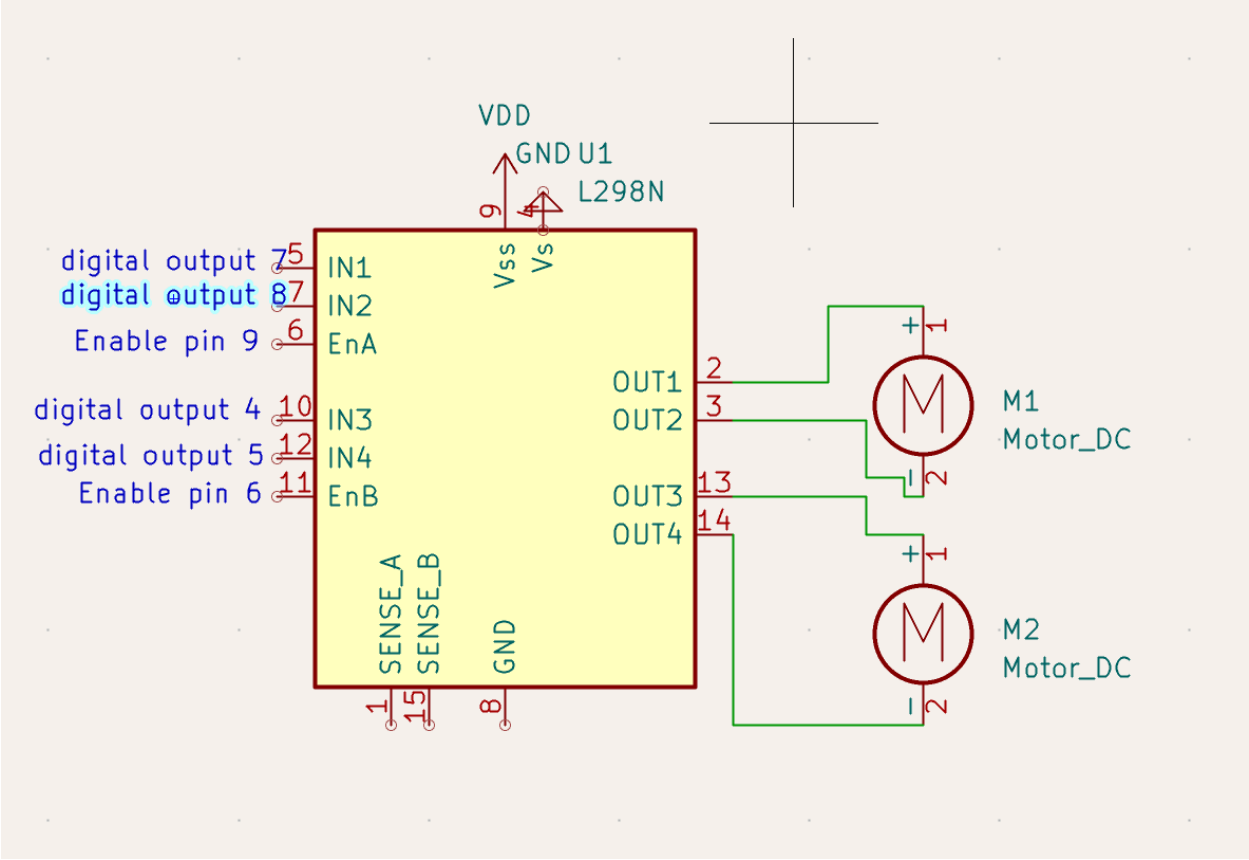
Circuit schematic for testing flywheel shooter/drivetrain:



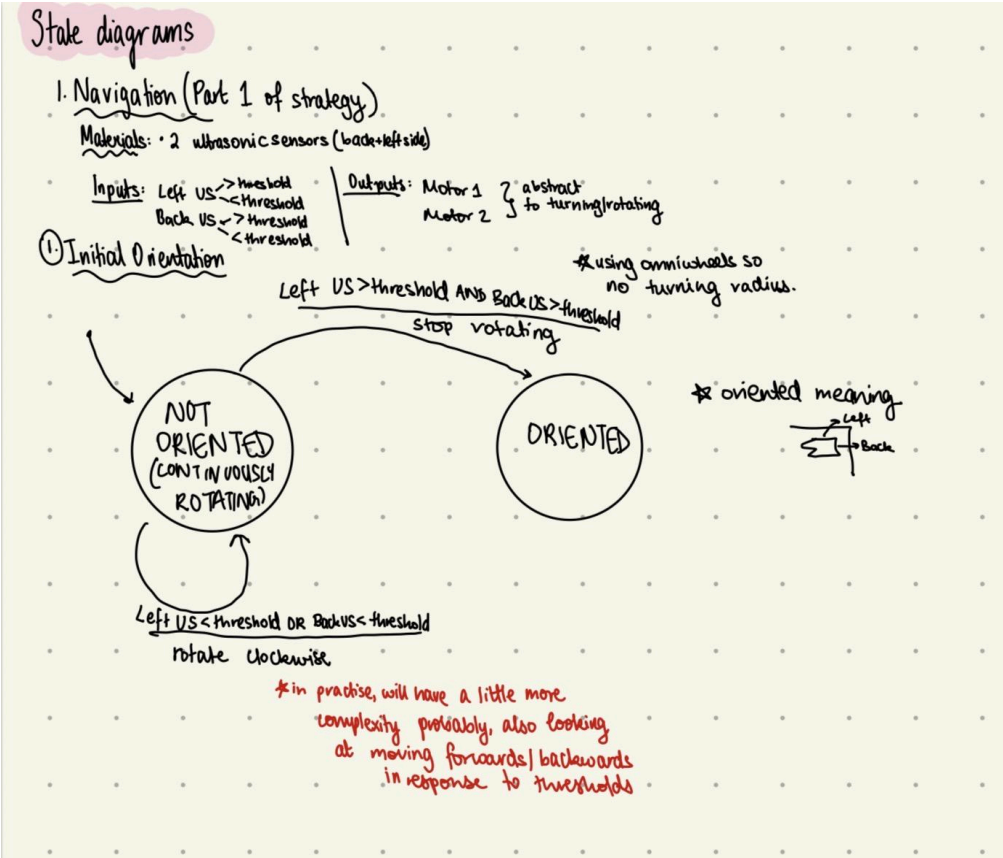
Corresponding arduino code:

```
// Motor A pins
const int enableA = 9;
const int in1 = 8;
const int in2 = 7;

// Motor B pins
const int enableB = 6;
const int in3 = 5;
const int in4 = 4;

void setup() {
 // Set all pins as outputs
 pinMode(enableA, OUTPUT);
 pinMode(in1, OUTPUT);
 pinMode(in2, OUTPUT);
```

```
 pinMode(enableB, OUTPUT);
 pinMode(in3, OUTPUT);
 pinMode(in4, OUTPUT);

 // Set Motor A direction (forward)
 digitalWrite(in1, HIGH);
 digitalWrite(in2, LOW);

 // Set Motor B direction (forward)
 digitalWrite(in3, HIGH);
 digitalWrite(in4, LOW);

 // Enable both motors at full speed
 digitalWrite(enableA, HIGH);
 digitalWrite(enableB, HIGH);
}

void loop() {
 // Motors run continuously
}
```

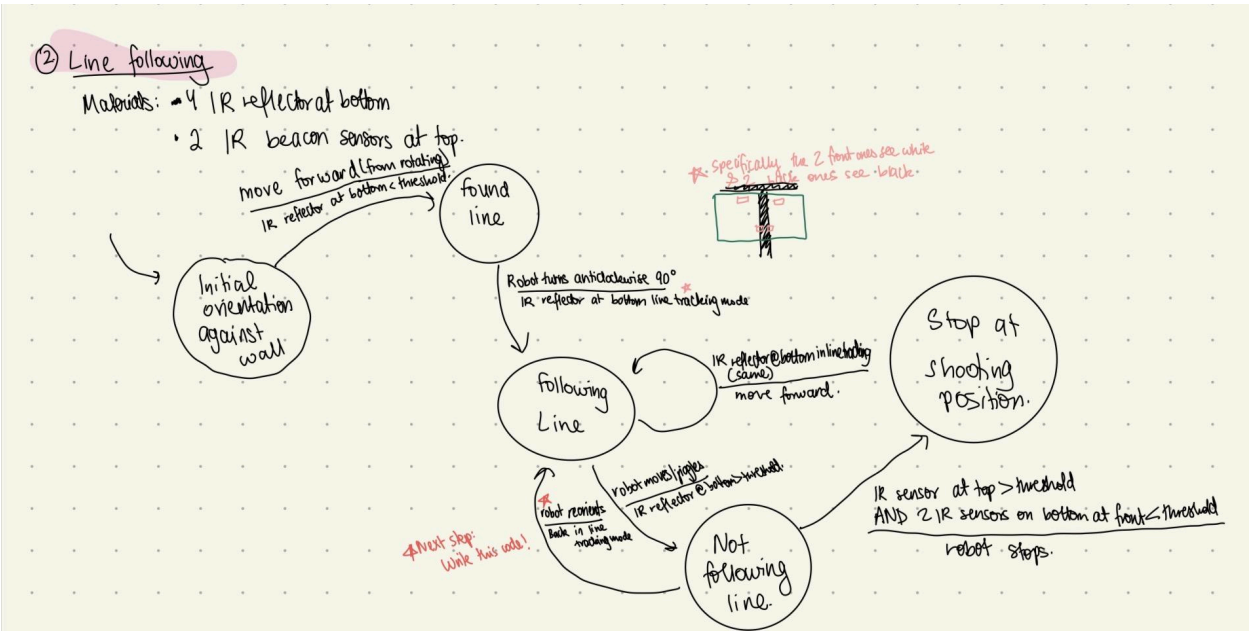
State diagrams:

To plan out the pipeline based on our initial brainstorming we included in the preliminary presentation, we split up our process into 3 distinct parts and represented them through state diagrams. The first was for orienting the robot initially using 2 ultrasonic sensors. During testing, we checked whether ultrasonic sensors work well for this, and the tests indicated they are a good choice for getting the robot oriented around the back and left wall. Still need to decide whether in addition to rotating based on

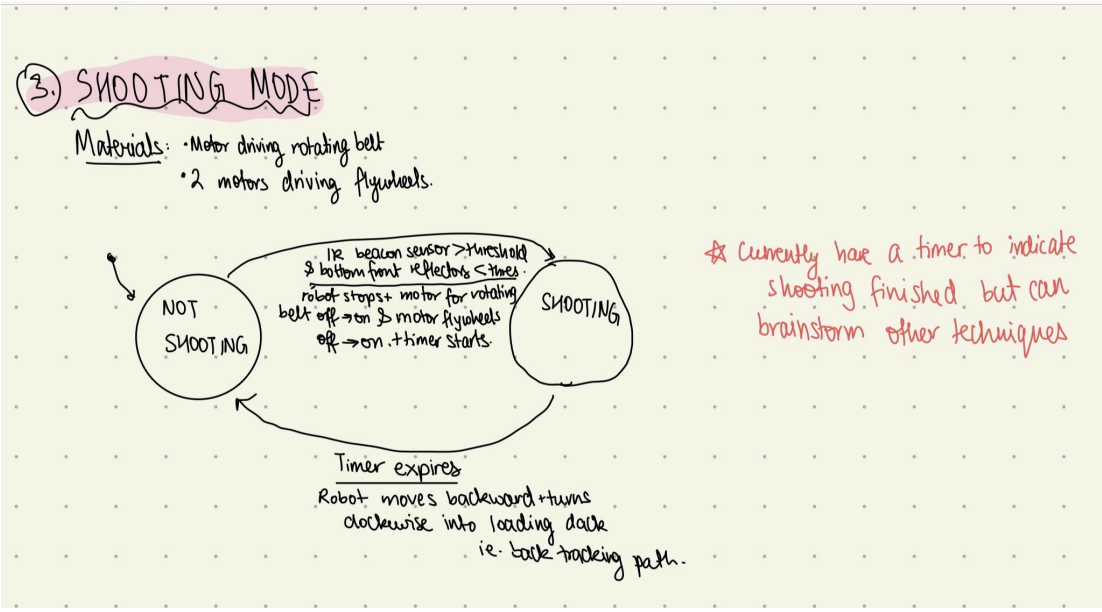


sensor reading, we also want to move forwards/backwards.

The second part is the line following which we are planning to do using a mix of IR sensors for the beacon and reflectors for the tape. Our initial testing focused on testing out omniwheels and deciding their orientation/number. Next steps are integrating the IR reflectors and beacon sensors.



Finally, for shooting. We tested out a simple prototype of the flywheel mechanism this week, which seemed to indicate the flywheel was a solid shooter design, but we want to constrain the flywheels and conduct a more rigorous test in the upcoming days,



Calculations to decide on motor:

Radius of Wheel (m)	Down force (Robot Wt.)	Torque needed (N.m)	Torque needed (kg.cm)	Torque Needed Per Wheel
0.08255	9.8	0.80899	8.249402192	2.062350548

Motor Tests

Motor	Link	Max Torque	Max RPM	Notes
Geartisan (12V, 200 RPM)	https://toytooth.com/prod	2.2kg-cm	200	Strongest (3x)
Geartisan (12V, 550 RPM)	https://ibspot.com/us/pro	0.7kg-cm	550	Slightly Stronger (4x)
Jameco ReliaPro	https://www.jameco.com	180g-cm	220	Weakest
Zhengke (12V, 550 RPM)	https://www.direnc.net/z	0.7kg-cm	550	Discontinued

Code for chassis control (6 movements got checked off):

```
const int speed = 185; //one speed for all now

// Motor R pins
const int enableA = 9; //white
const int in1 = 12; //yellow //+
const int in2 = 7; //orange //-

// Motor B pins
const int enableB = 6; //blue
const int in3 = 13; //grey //+
const int in4 = 4; //green //-

// Motor L pins
const int enableA_ = 10;
const int in1_ = 2; //+ //we lied
const int in2_ = 8; //-

// Motor F pins
const int enableB_ = 11;
const int in3_ = 3; //we lied
const int in4_ = 5;

void setup() {
 // Set all pins as outputs
 pinMode(enableA, OUTPUT);
 pinMode(in1, OUTPUT);
 pinMode(in2, OUTPUT);

 pinMode(enableB, OUTPUT);
```

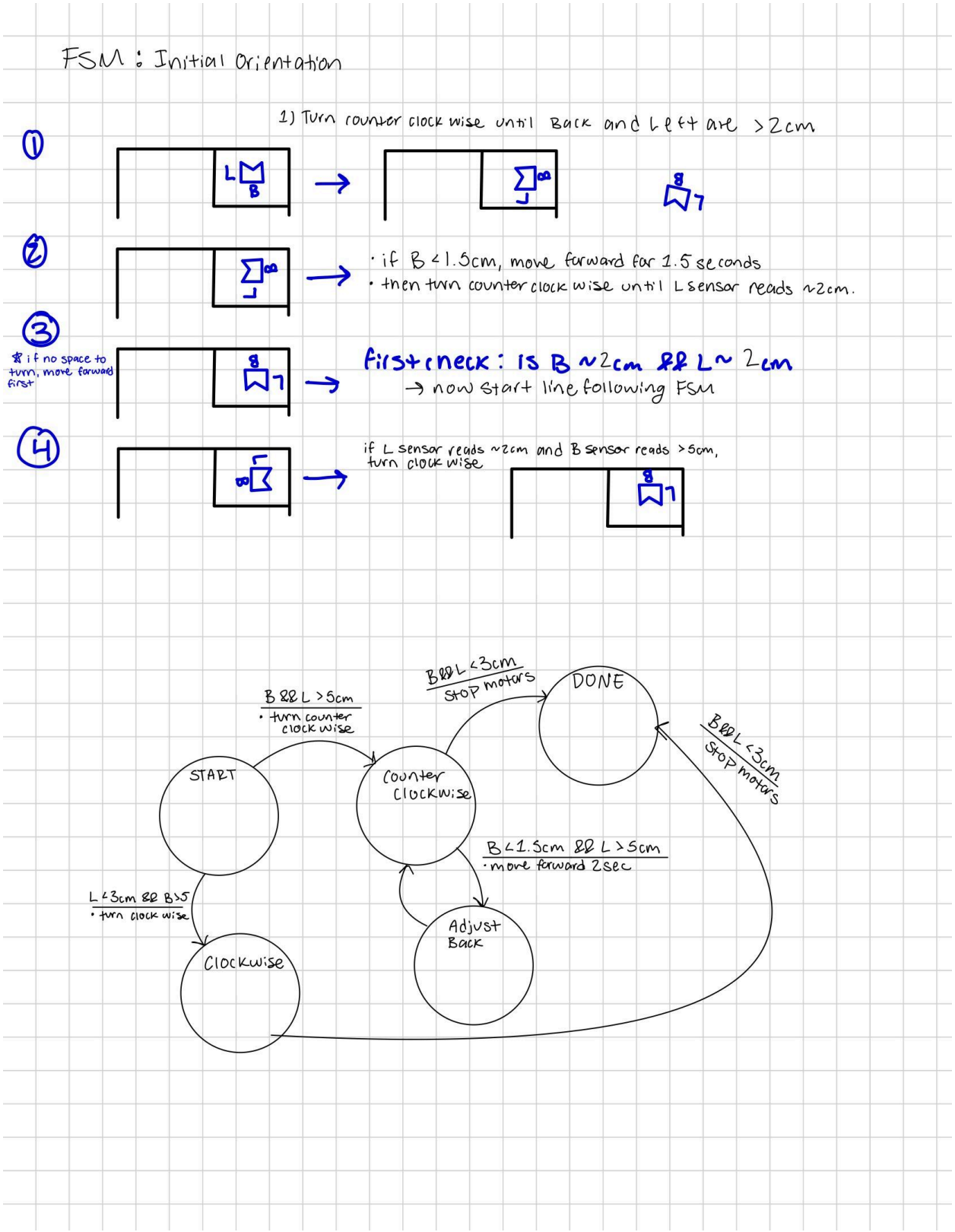
```
pinMode(in3, OUTPUT);
pinMode(in4, OUTPUT);

pinMode(enableA_, OUTPUT);
pinMode(in1_, OUTPUT);
pinMode(in2, OUTPUT);

pinMode(enableB_, OUTPUT);
pinMode(in3_, OUTPUT);
pinMode(in4_, OUTPUT);

// Enable both motors at speed
analogWrite(enableA, speed);
analogWrite(enableB, speed);
analogWrite(enableA_, speed);
analogWrite(enableB_, speed);

botStop();
delay(2000);
botForward(speed);
delay(2000);
botStop();
delay(2000);
botBackward(speed);
delay(2000);
botStop();
delay(2000);
botLeft(speed);
delay(2000);
botStop();
delay(2000);
botRight(speed);
delay(2000);
botStop();
delay(2000);
botAnticlockwise(speed);
delay(2000);
botStop();
delay(2000);
botClockwise(speed);
delay(2000);
botStop();
```

```
 delay(2000);

}

void moveForward(int high_pin, int low_pin){
 digitalWrite(high_pin, HIGH);
 digitalWrite(low_pin, LOW);

}

void moveBackward(int high_pin, int low_pin){
 digitalWrite(low_pin, HIGH);
 digitalWrite(high_pin, LOW);

}

void motorStop(int high_pin, int low_pin){
 digitalWrite(low_pin, LOW);
 digitalWrite(high_pin, LOW);
}

void botForward(int speed){ //motorR&L
 analogWrite(enableA, speed);
 analogWrite(enableA_, speed);
 moveForward(in1, in2); //R
 moveForward(in1_, in2_); //L
}

void botBackward(int speed){
 analogWrite(enableA, speed);
 analogWrite(enableA_, speed);
 moveBackward(in1, in2); //R
 moveBackward(in1_, in2_); //L
}

void botLeft(int speed){
 analogWrite(enableB, speed);
 analogWrite(enableB_, speed);
 moveForward(in4, in3);
 moveForward(in4_, in3_);
}
```

```
void botRight(int speed){
 analogWrite(enableB, speed);
 analogWrite(enableB_, speed);
 moveBackward(in4, in3);
 moveBackward(in4_, in3_);
}


void botStop(){
 motorStop(in1, in2);
 motorStop(in1_, in2_);
 motorStop(in1_, in2_);
 motorStop(in3_, in4_);
}

void botAnticlockwise(int speed){
 analogWrite(enableA, speed);
 analogWrite(enableB, speed);
 analogWrite(enableA_, speed);
 analogWrite(enableB_, speed);
 moveForward(in1, in2); //R
 moveForward(in3, in4); //B
 moveBackward(in1_, in2_); //L
 moveBackward(in3_, in4_); //F
}

void botClockwise(int speed){
 analogWrite(enableA, speed);
 analogWrite(enableB, speed);
 analogWrite(enableA_, speed);
 analogWrite(enableB_, speed);
 moveBackward(in1, in2); //R
 moveBackward(in3, in4); //B
 moveForward(in1_, in2_); //L
 moveForward(in3_, in4_); //F
}

void loop() {


}
```

Additional FSMs

FSM for initial orientation.



FSM for line following.

FSM Line Following

Tape Reader reads 400→500→600 at white part and ~900→1000 at the black tape

2)

move left until middle line sensor reads ~1000, and left + right sensors read > 800
↳ subject to more testing