KING MONGKUT’S INSTITUTE OF TECHNOLOGY LATKRABANG
SCHOOL OF ENGINEERING

01006702/01006724/01416316/01416309
PHYSICS - I LABORATORY, 3D DRAWING
COMPUTER PROGRAMMING FOR RAI

LAB TITLE: FINAL PROJECT

INSTRUCTED BY: SUPUN DISSANAYAKA,
DR. KANOKNUCH SONGSUWANKIT,
DR. POOM KONGHUAYROB

NAME: NATCHA, TIPOKE, AKESIT
ID NUMBER: 65011386, 65011607, 65110131
GROUP NO: 5
DATE OF SUB.: 18/12/2022

TABLE OF CONTENT

Purpose

- To apply the knowledge from physics, computer programming, and 3D drawing.
- To practice teamwork while working in a group with friends.
- To comprehend how each component works.

Project Scope

- Capable of moving by referencing the line without manual control by employing an infrared sensor to identify curve, intersection, and straight lines.
- Capable of grabbing and pushing things with a diameter of no more than 10 cm to the appropriate location.
- Able to use the projectile theory to throw objects with a maximum diameter of 10 cm at an angle of 80 degrees.

Duty of members in your team and percentage of working

- Akesit Akkharasaksiri [50%]: Mechanic design and programming

  Doing the mechanic design for both robots. Start from gripper to launcher and body for the manual and how each component for the auto robot assembled. Moreover, programming in c and python languages for both robots. **Signature:**
- Natcha Rungruang [30%]: Mechanic design, make drawing file, do the report.

  Do the mechanical design for the projectile launcher. Then make drawing file in 2D and 3D file in Autodesk inventor. Furthermore, do the report. **Signature:**
- Tipoke Phangkul [20%]: Mechanic design, and make the flowchart for both robots, and then do the calculation part and the list components of the line following robot. **Signature:**

Auto robot (Line-following Robot)

1. Background Research

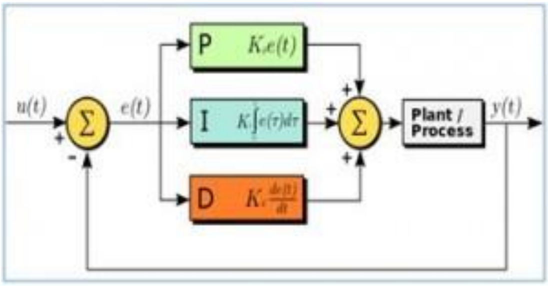
- **PID controller**

  PID controllers are used in a variety of industrial process control applications. PID controllers are used in around 95% of the closed-loop activities in the industrial automation sector. Proportional-Integral-Derivative is referred to as PID. A control signal is created by combining these three controllers in this manner. It delivers the control output at the desired levels as a feedback controller. Analog electronic components were used to implement PID control prior to the development of microprocessors. However, today's PID controllers are all handled by microprocessors. The PID controller instructions are also embedded into programmable logic controllers. Process control applications have always employed PID controllers because of their adaptability and dependability.

  For our project, we have used a PID controller to control the movement of the line following robot with its own error and position.

  - **PID Controller Block Diagram**

    A feedback control system is part of a closed-loop system like a PID controller. To provide an error signal, this system assesses the feedback variable using a fixed point. It modifies the system output in response to that. If the error does not reach zero, this process is repeated until the feedback variable's value is comparable to a fixed point.



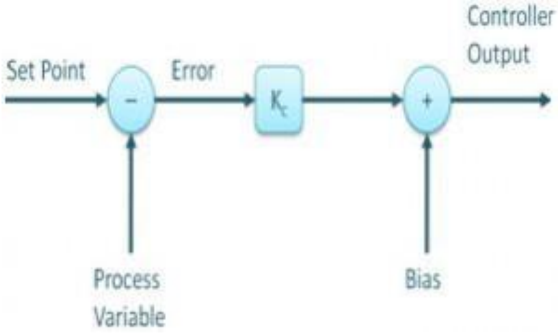
When compared to an ON/OFF controller, this one produce good results. There are just two criteria that can be used to manage the system in an ON/OFF type controller. It will switch ON when the process value falls below the specified point. Similar to that, it will become OFF once the value exceeds a predetermined level. With this type of controller, the output is unstable and frequently oscillates about the fixed point. However, compared to an ON/OFF type controller, this one is more accurate and steadier.

- **Working of PID Controller**

By using closed-loop operations, PID controllers keep the output such that there is no error between the process variable and the setpoint or desired output. PID employs three fundamental control characteristics, which are described here.

- P- Controller

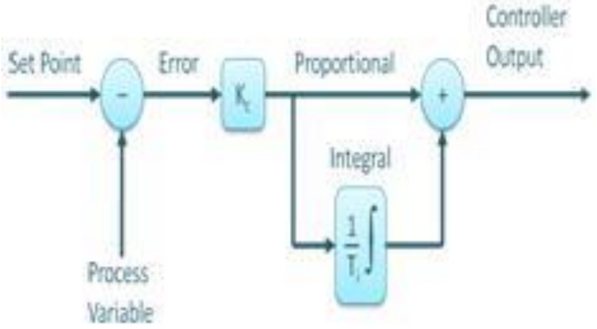
An output from a proportional or P-controller is proportionate to the present inaccuracy (t). It contrasts the actual value or feedback process value with the desired or set point. To obtain the output, the resulting error is multiplied by a proportional constant. This controller's output is zero if the error value is zero.



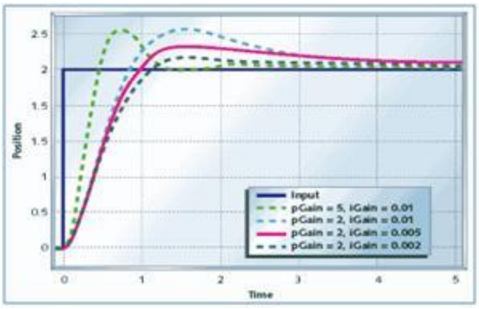
When used alone, this controller needs biasing or a manual reset. This is because the steady-state condition is never reached. Although it offers stable functioning, the steady-state error is constantly maintained. When the proportional constant Kc rises, the response speed rises as well.

- I-Controller

I-controller is required to perform the appropriate action to remove the steady-state error due to the limitation of p-controller, where there always exists an offset between the process variable and setpoint. Until the error value is zero, the error is integrated over a period. It stores the value for the last control mechanism before the error is zero. When a negative mistake occurs, integral control reduces its output. It restricts response time and compromises the system's stability. Decreased integral gain, Ki, accelerates the response time.



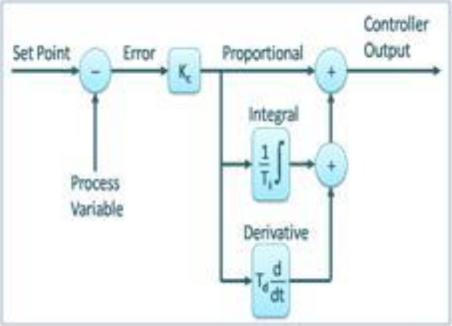
In the picture above, the steady-state error similarly lowers as the I-gain controller's does. The PI controller is typically employed, especially when a high-speed reaction is not necessary.



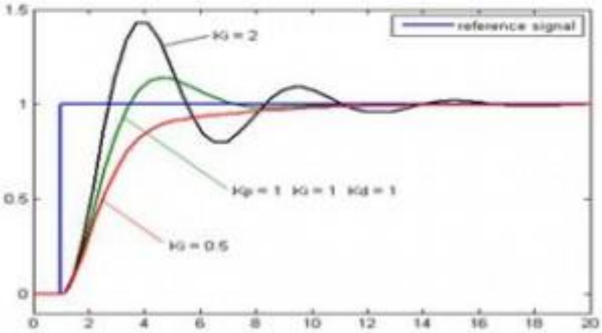
In order to avoid integral wind up scenarios, where the integral output keeps increasing even at zero error state due to nonlinearities in the plant, the PI controller's output is limited to a certain range.

- D-Controller

I-controller doesn’t have the capability to predict the future behavior of error. Once the setpoint is altered, it responds as expected. By foreseeing how the mistake would behave in the future, the D-controller solves this issue. Its output is determined by the derivative constant multiplied by the error's rate of change over time. It provides the catalyst for the output, boosting system responsiveness.



In the above figure, D's reaction, the controller is larger than the PI controller, and the output settling time is also sped up. It enhances system stability by addressing the phase lag brought on by the I-controller. The response time is accelerated by increasing the derivative gain.



2. List of the component

Component	Name	Specification	Reference
	IR sensor	tcrt5000	https://robotsiam.blogspot.com/2016/10/ir-infrared-obstacle-avoidance-sensor.html

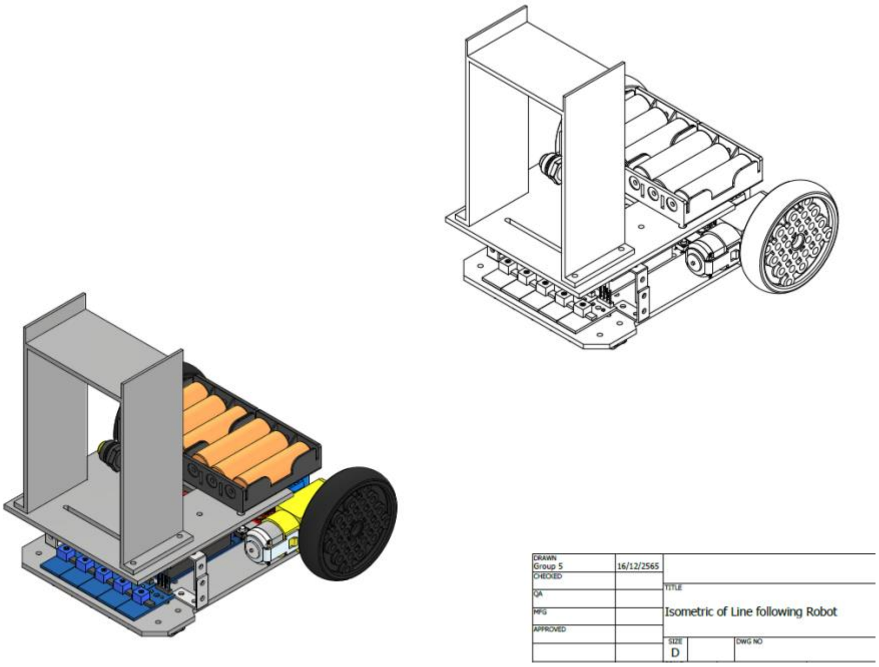
	DC gearbox motor	6.0v 75 rpm (without load) torque 4 kg.cm	https://einstronic.com/product/tt-motor-yellow-geared-dc-motor/
	Battery holder	3 AA battery	https://th.rs-online.com/web/c/batteries-chargers/battery-charger-accessories/battery-holders/
	Push button Switch		https://thai.alibaba.com/product-detail/60mm-plastic-push-button-switch-electric-1600070601327.html
	arduino uno + arduino sensor shield		https://www.arduino4.com/product/325/arduino-uno-r3-sensor-shield-v5-expansion-board
	L298N motor diver module	double H bridge drive,5V-35V drive voltage	https://www.tido.tech/index.php/product/l298n-motor-driver-module/
	Makeblock wheel	63.9mm diameter	http://docs.makeblock.com/diy-platform/en/mechanical-parts/wheels/slick-tyre-64-16mm.html
	Phillips head screw	M3x52, M3x10, M3x15, M3x25, M2x15	https://th.misumi-ec.com/en/vona2/detail/221000547315/
	Angle Strut Bracket	3mm hold	https://www.mcmaster.com/strut-channel-angle-brackets/

3. Design

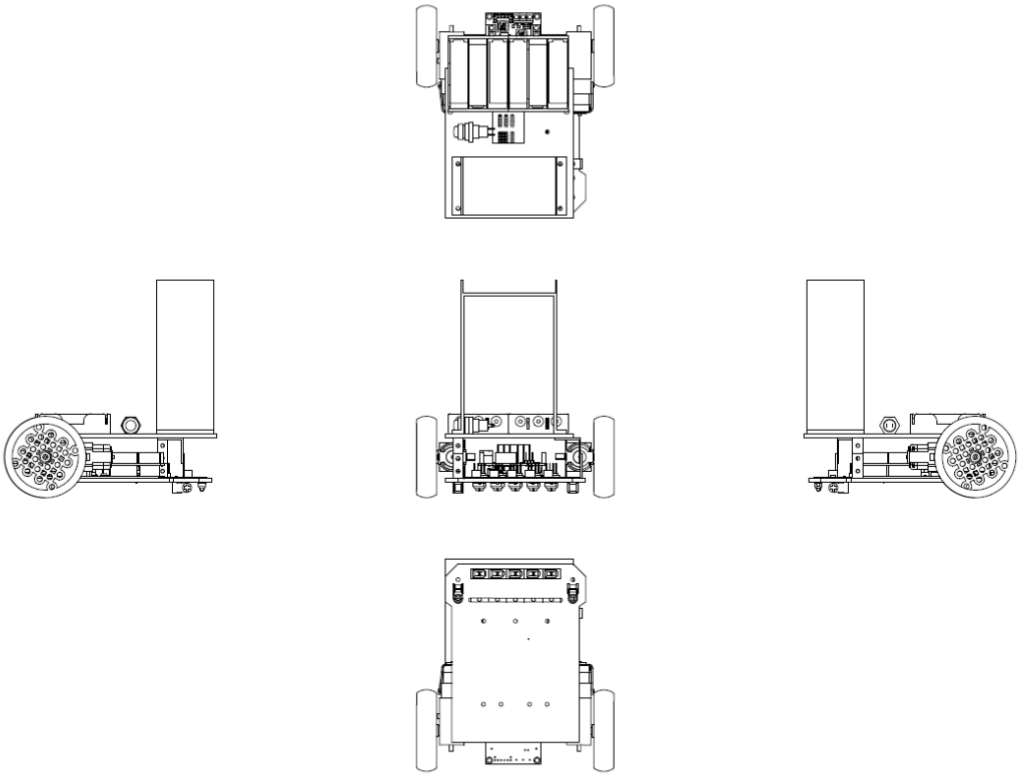
- **Design Concept**

  When we are aware of the pieces we possess, we make an effort to assemble them in accordance with the pictures in order to examine the model and produce the fundamental designs. When the design is finished, we bring the file to be 3D printed with the box holder and laser cut onto 3mm acrylic. Because it is useful and compact, this robot was developed.

- **Drawing**



Isometric view of line following robot



Third-angular-orthographic projection for all side view of the Auto Robot

4. Programming

- **Coding with concept**

```
float error, lasterror , I, output, motora, motorb;
int mmax = 255;
int mmin = 255;
int i = 0;
int light[] = {300, 300, 300, 300, 300};//use array
unsigned long time;
void setup() {
  Serial.begin(9600);
  pinMode(A0, INPUT);  //left to right
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);
  pinMode(A3, INPUT);
  pinMode(A4, INPUT);
  pinMode(9, OUTPUT);  //PWM Motor1 Left
  pinMode(8, OUTPUT);
  pinMode(7, OUTPUT);
  pinMode(4, OUTPUT);  // Motor2 Right
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);  //PWM
  //Right
  m1(255);
  m2(255);
  delay(400);
  PIDC(50, 0, 0, 255, 1);
  right(250, 250);
  PIDR(50, 0, 0, 230, 1);
  right(250, 200);
  PIDR(50, 0, 0, 240, 1);
  right(250, 200);
  PIDC(50, 0, 0, 255, 2);
  left(250, 200);
  PIDC(50, 0, 0, 255, 1);
  left(250, 200);
  PIDC(50, 0, 0, 255, 1);
  left(250, 200);
  PIDC(50, 0, 0, 255, 1);
  m1(255);
  m2(255);
  delay(200);
}

void loop() {
  //check();
  m1(0);
  m2(0);
  delay(50);
}
```

```
void check() {
  struct st_c {
    int sensor0;
    int sensor1;
    int sensor2;
    int sensor3;
    int sensor4;
  };//use structure
  struct st_c c = {analogRead(A0), analogRead(A1), analogRead(A2),
  analogRead(A3), analogRead(A4)};
  Serial.print(c.sensor0);
  Serial.print(",");
  Serial.print(c.sensor1);
  Serial.print(",");
  Serial.print(c.sensor2);
  Serial.print(",");
  Serial.print(c.sensor3);
  Serial.print(",");
  Serial.println(c.sensor4);
}//use pointer
void PIDC (float kp, float kd, float ki, int basespeed, int i) {
  int count = 0;
  while (count < i) { //use loop
    if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // white
      if (lasterror == -5 || lasterror == -4) {
        error = -5;
        Serial.println("error = -5");
      }
      else if (lasterror == 5 || lasterror == 4) {
        error = 5;
        Serial.println("error = 5");
      }
      else {
        error = 0;
        Serial.println("error = 0");
      }
    }
    if (analogRead(A0) > light[0] && analogRead(A1) > light[1] &&
    analogRead(A2) > light[2] && analogRead(A3) > light[3] &&
    analogRead(A4) > light[4]) { // black
      error = 0;
      Serial.println("error = 0");
      count++;
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) > light[4]) { //4
      error = 4;
      Serial.println("error = 4");
    }
```

```
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
analogRead(A2) < light[2] && analogRead(A3) > light[3] &&
analogRead(A4) > light[4]) { // 3,4
  error = 3;
  Serial.println("error = 3");
}
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
analogRead(A2) < light[2] && analogRead(A3) > light[3] &&
analogRead(A4) < light[4]) { // 3
  error = 2;
  Serial.println("error = 2");
}
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
analogRead(A2) > light[2] && analogRead(A3) > light[3] &&
analogRead(A4) < light[4]) { // 2,3
  error = 1;
  Serial.println("error = 1");
}
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
analogRead(A2) > light[2] && analogRead(A3) < light[3] &&
analogRead(A4) < light[4]) { // 2 center
  error = 0;
  Serial.println("error = 0");
}
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] &&
analogRead(A2) > light[2] && analogRead(A3) < light[3] &&
analogRead(A4) < light[4]) { // 1,2
  error = -1;
  Serial.println("error = -1");
}
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] &&
analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
analogRead(A4) < light[4]) { // 1
  error = -2;
  Serial.println("error = -2");
}
else if (analogRead(A0) > light[0] && analogRead(A1) > light[1] &&
analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
analogRead(A4) < light[4]) { // 0,1
  error = -3;
  Serial.println("error = -3");
}
else if (analogRead(A0) > light[0] && analogRead(A1) < light[1] &&
analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
analogRead(A4) < light[4]) { // 0
  error = -4;
  Serial.println("error = -4");
}//use loop
I = I + error;
output = (kp * error) + (kd * (error - lasterror)) + (ki * I);
```

```
    //how quick it reach the set point + slow down the speed before reach
    the set point + quickness of response of the system to the change from
    the set point
    lasterror = error;
    Serial.println(lasterror);
    motora = basespeed + output;
    motorb = basespeed - output;
    Serial.println(motora);
    Serial.println(motorb);
    m1(motora);
    m2(motorb);
}//use function
void PIDL (float kp, float kd, float ki, int basespeed, int i) {
  int count = 0;
  while (count < i) {
    if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // white
      if (lasterror == 5 || lasterror == 4) {
        error = 5;
        Serial.println("error = 5");
      }
      else {
        error = -5;
        Serial.println("error = -5");
      }
    }
    if (analogRead(A2) < light[2] && analogRead(A0) > light[0]) { // black
      error = 0;
      count++;
      Serial.println("error = 0");
      m1(120);
      m2(120);
      delay(100);
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) > light[4]) { //4
      error = 4;
      Serial.println("error = 4");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) > light[3] &&
    analogRead(A4) > light[4]) { // 3,4
      error = 3;
      Serial.println("error = 3");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) > light[3] &&
    analogRead(A4) < light[4]) { // 3
      error = 2;
      Serial.println("error = 2");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) > light[2] && analogRead(A3) > light[3] &&
    analogRead(A4) < light[4]) { // 2,3
      error = 1;
      Serial.println("error = 1");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) > light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 2 center
      error = 0;
      Serial.println("error = 0");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] &&
    analogRead(A2) > light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 1,2
      error = -1;
      Serial.println("error = -1");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 1
      error = -2;
      Serial.println("error = -2");
    }
    else if (analogRead(A0) > light[0] && analogRead(A1) > light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 0,1
      error = -3;
      Serial.println("error = -3");
    }
    else if (analogRead(A0) > light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 0
      error = -4;
      Serial.println("error = -4");
    }

    I = I + error;
    output = (kp * error) + (kd * (error - lasterror)) + (ki * I);
    lasterror = error;
    Serial.println(lasterror);
    motora = basespeed + output;
    motorb = basespeed - output;
    Serial.println(motora);
    Serial.println(motorb);
    m1(motora);
    m2(motorb);
}
```

```
void PIDR(float kp, float kd, float ki, int basespeed, int i) {
int count = 0;
  while (count < i) {
    if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // white
      if (lasterror == 5 || lasterror == 4) {
        error = 5;
        Serial.println("error = 5");
      }
      else  {
        error = -5;
        Serial.println("error = -5");
      }
    }
    if (analogRead(A4) > light[4]) { // black
      error = 0;
      count++;
      Serial.println("error = 0");
      m1(120);
      m2(120);
      delay(100);
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) > light[4]) { //4
      error = 4;
      Serial.println("error = 4");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) > light[3] &&
    analogRead(A4) > light[4]) { // 3,4
      error = 3;
      Serial.println("error = 3");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) > light[3] &&
    analogRead(A4) < light[4]) { // 3
      error = 2;
      Serial.println("error = 2");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) > light[2] && analogRead(A3) > light[3] &&
    analogRead(A4) < light[4]) { // 2,3
      error = 1;
      Serial.println("error = 1");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
    analogRead(A2) > light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 2 center
      error = 0;
      Serial.println("error = 0");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] &&
    analogRead(A2) > light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 1,2
      error = -1;
      Serial.println("error = -1");
    }
    else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] &&
    analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
    analogRead(A4) < light[4]) { // 1
      error = -2;
      Serial.println("error = -2");
    }
    else if (analogRead(A0) > l

void m1(int power) {
  if (power > 0) {
    digitalWrite(8, HIGH);
    digitalWrite(7, LOW);
    analogWrite(9, power);
  }
  else if (power > mmax) {
    digitalWrite(8, HIGH);
    digitalWrite(7, LOW);
    analogWrite(9, mmax);
  }
  else if (power < 0) {
    digitalWrite(8, LOW);
    digitalWrite(7, HIGH);
    analogWrite(9, -power);
  }
  else if (power < -mmin) {
    digitalWrite(8, LOW);
    digitalWrite(7, HIGH);
    analogWrite(9, mmin);
  }
  else {
    digitalWrite(8, LOW);
    digitalWrite(7, LOW);
    analogWrite(9, 0);
  }
}
void m2(int power) {
  if (power > 0) {
    digitalWrite(4, HIGH);
    digitalWrite(5, LOW);
    analogWrite(6, power);
  }
  else if (power > mmax) {
    digitalWrite(4, HIGH);
    digitalWrite(5, LOW);
    analogWrite(6, mmax);
  }
  else if (power < 0) {
    digitalWrite(4, LOW);
    digitalWrite(5, HIGH);
    analogWrite(6, -power);
  }
  else if (power < -mmin) {
    digitalWrite(4, LOW);
    digitalWrite(5, HIGH);
    analogWrite(6, mmin);
  }
  else {
    digitalWrite(4, LOW);
    digitalWrite(5, LOW);
    analogWrite(6, 0);
  }
}
void right(int s, int t) {
  m1(s);
  m2(s);
  delay(250);
  m1(s);
  m2(-s);
  delay(t);
  while (analogRead(A3) < light[3]);
}
void left(int s, int t) {
  m1(s);
  m2(s);
  delay(250);
  m1(-s);
  m2(s);
  delay(t);
  while (analogRead(A2) < light[2]);
}
```

- **Topic that uses in Coding**
  - **Array**: used to input the value into ‘light’ variable

    Ex.
    ```
    int light[] = {300, 300, 300, 300, 300};
    ```
  - **Structure:** used to group several related variables into one place.

    Ex.
    ```
    struct st_c {
        int sensor0;
        int sensor1;
        int sensor2;
        int sensor3;
        int sensor4;
      };
    ```
  - **Pointer** : used to store the address of variables or a memory location.

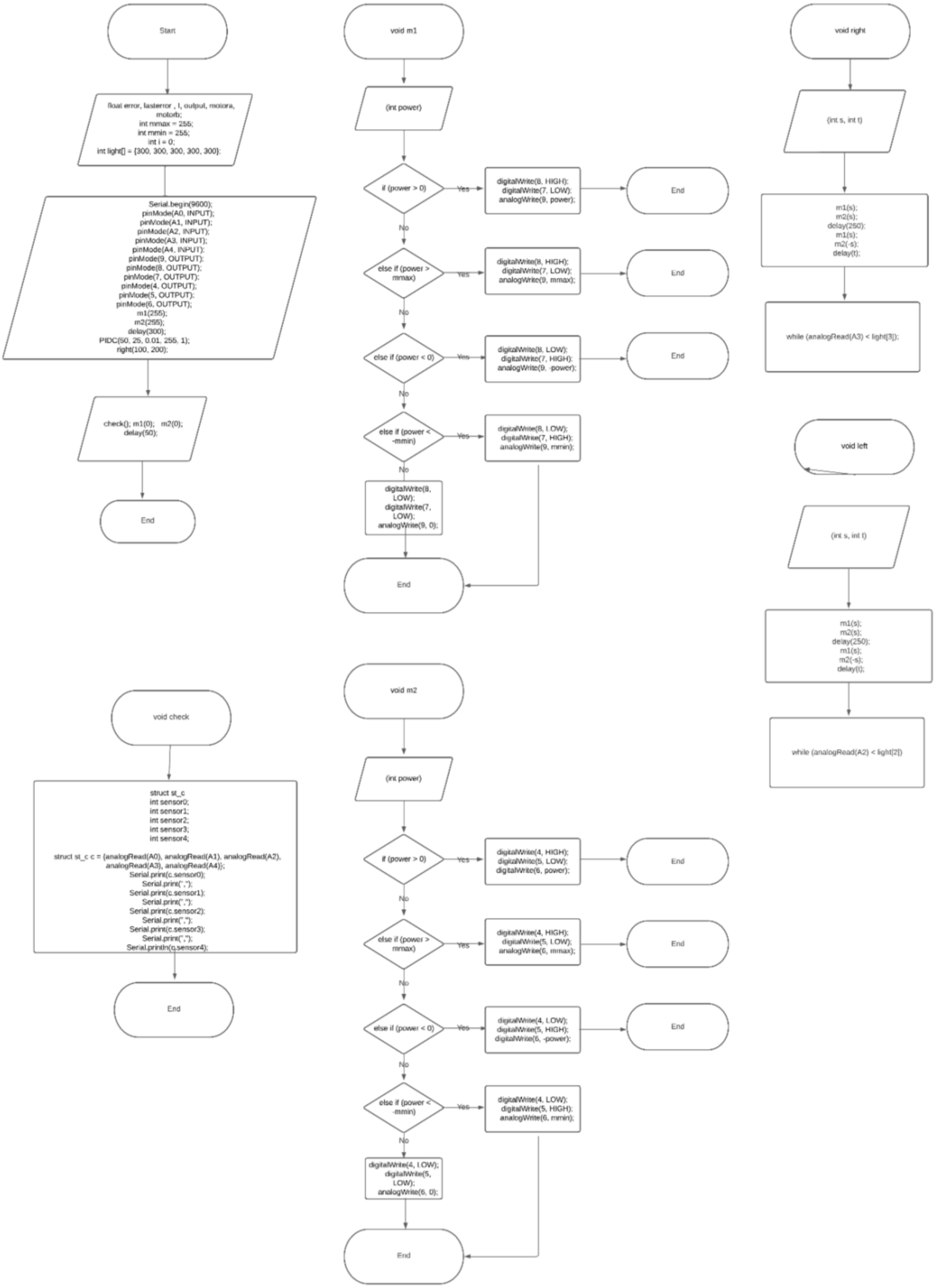
    Ex.
    ```
    struct st_c c = {analogRead(A0), analogRead(A1), analogRead(A2),
     analogRead(A3), analogRead(A4)};
     Serial.print(c.sensor0);
     Serial.print(",");
     Serial.print(c.sensor1);
     Serial.print(",");
     Serial.print(c.sensor2);
     Serial.print(",");
     Serial.print(c.sensor3);
     Serial.print(",");
     Serial.println(c.sensor4);
    }
    ```
  - **Loop:** used to execute the block of code several times according to the condition given in the loop.

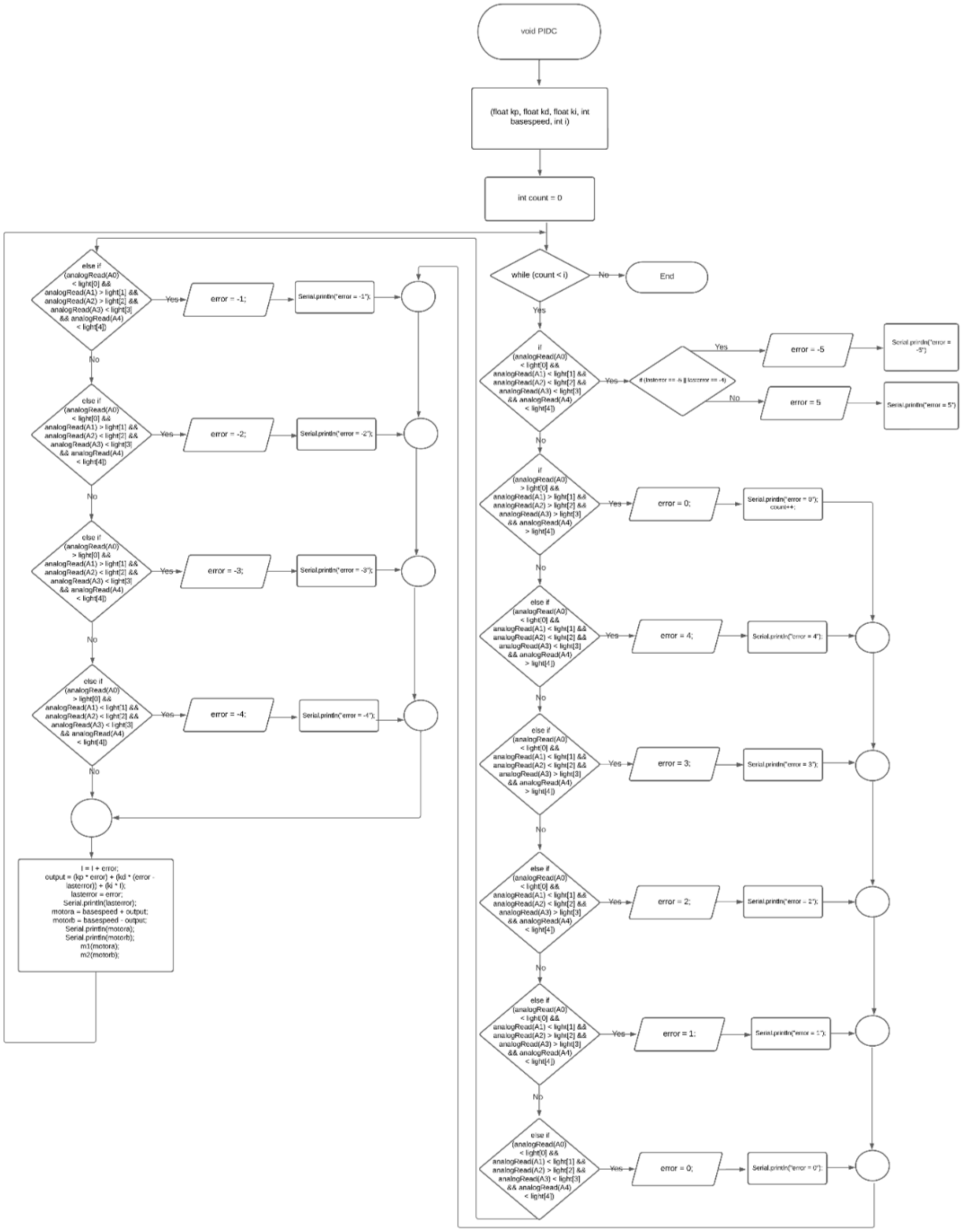
    Ex.
    ```
    while (count < i) {
        if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
        analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
        analogRead(A4) < light[4]) { // white
          if (lasterror == -5 || lasterror == -4) {
            error = -5;
            Serial.println("error = -5");
          }
          else if (lasterror == 5 || lasterror == 4) {
            error = 5;
            Serial.println("error = 5");
          }
          else {
            error = 0;
            Serial.println("error = 0");
          }
    ```
  - **Function:** used to perform certain actions

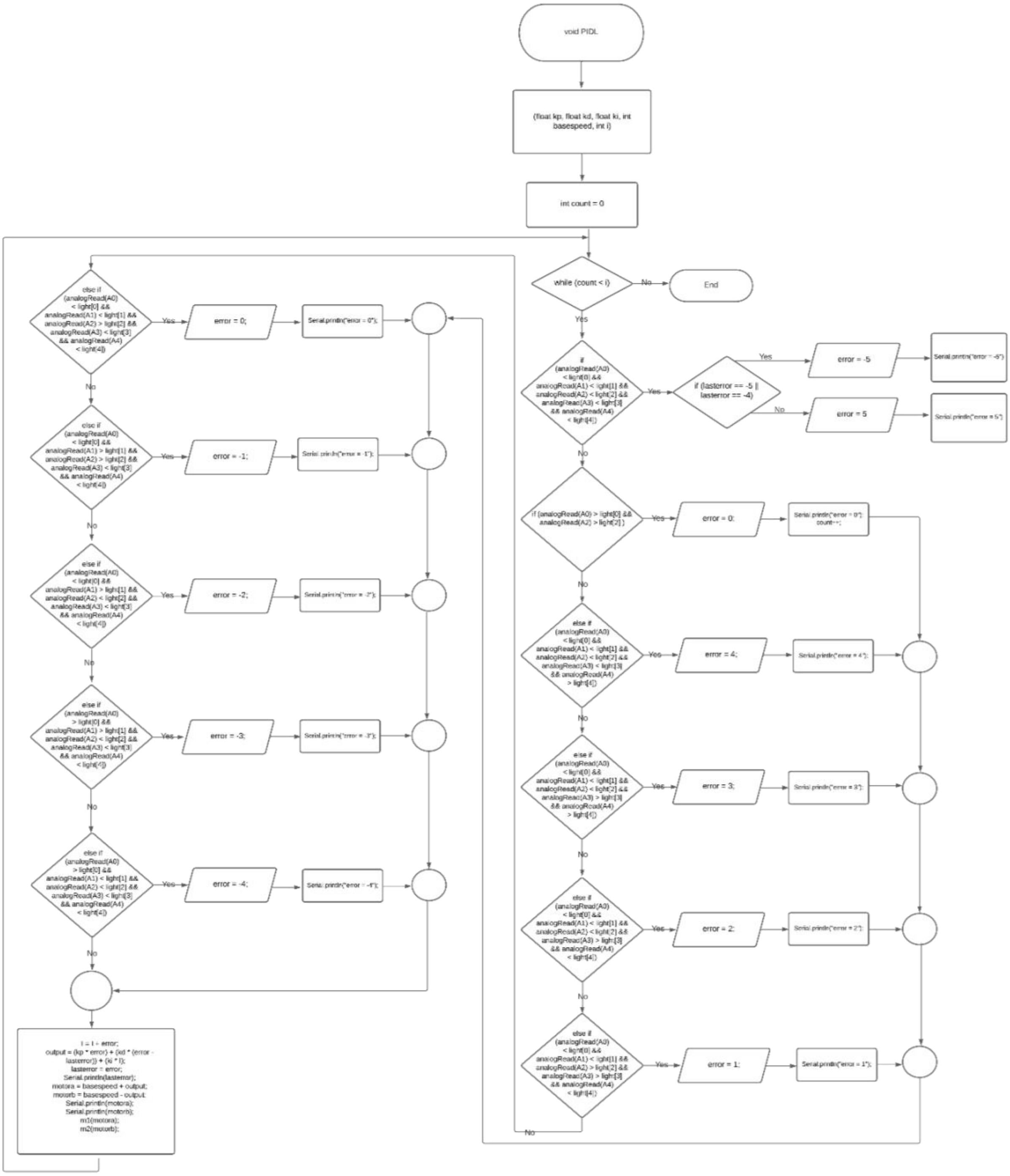
    Ex.
    ```
    void PIDC (float kp, float kd, float ki, int basespeed, int i) {
      int count = 0;
      while (count < i) {
        if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
        analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
        analogRead(A4) < light[4]) { // white
          if (lasterror == -5 || lasterror == -4) {
            error = -5;
            Serial.println("error = -5");
    ```

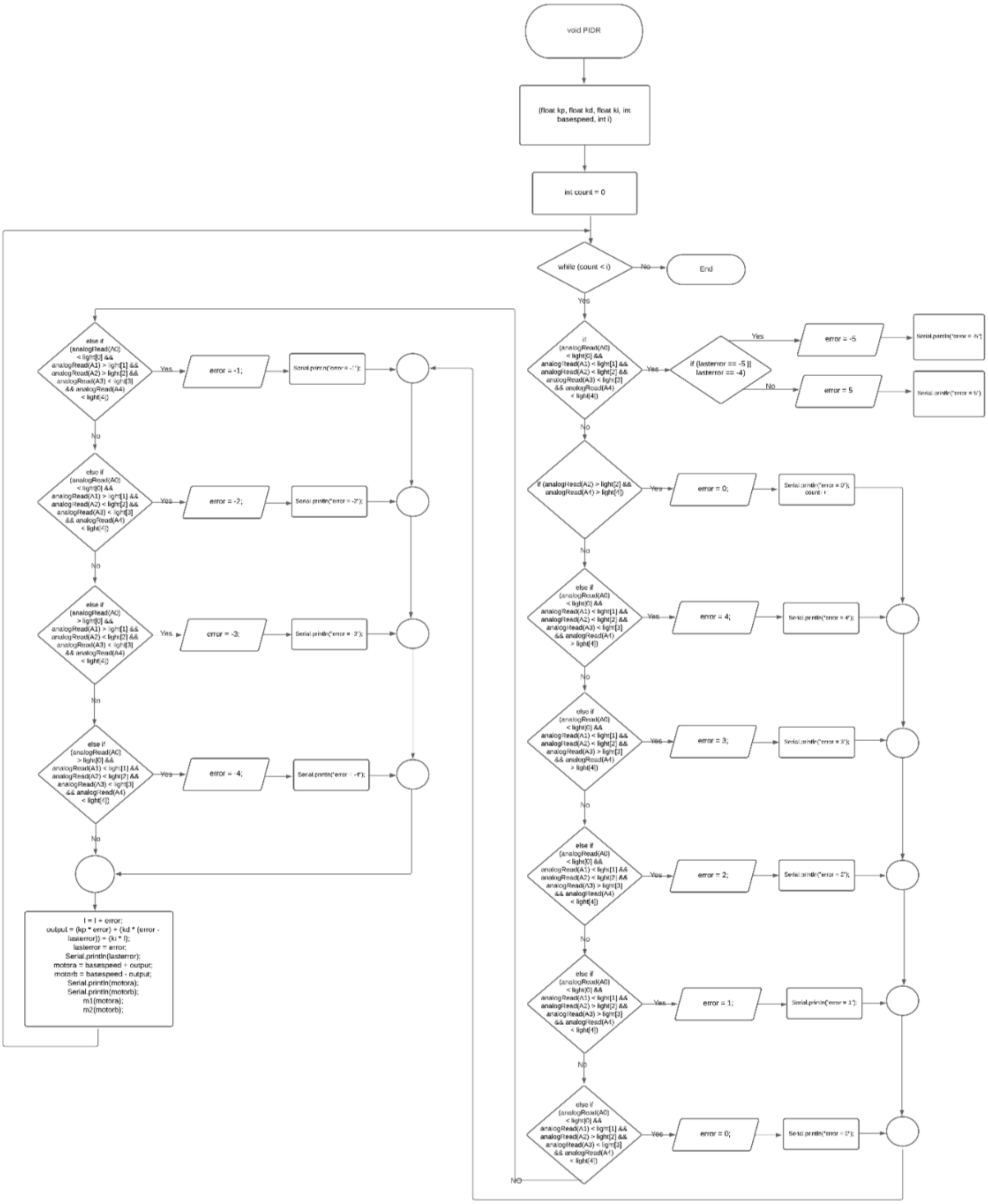
```
     }
     else if (lasterror == 5 || lasterror == 4) {
       error = 5;
       Serial.println("error = 5");
     }
     else {
       error = 0;
       Serial.println("error = 0");
     }
   }
   if (analogRead(A0) > light[0] && analogRead(A1) > light[1] &&
   analogRead(A2) > light[2] && analogRead(A3) > light[3] &&
   analogRead(A4) > light[4]) { // black
     error = 0;
     Serial.println("error = 0");
     count++;
   }
   else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
   analogRead(A2) < light[2] && analogRead(A3) < light[3] &&
   analogRead(A4) > light[4]) { //4
     error = 4;
     Serial.println("error = 4");
   }
   else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] &&
   analogRead(A2) < light[2] && analogRead(A3) > light[3] &&
   analogRead(A4) > light[4]) { // 3,4
     error = 3;
     Serial.println("error = 3");
   }
   I = I + error;
   output = (kp * error) + (kd * (error - lasterror)) + (ki * I);
   //how quick it reach the set point + slow down the speed before reach
   the set point + quickness of response of the system to the change from
   the set point
   lasterror = error;
   Serial.println(lasterror);
   motora = basespeed + output;
   motorb = basespeed - output;
   Serial.println(motora);
   Serial.println(motorb);
   m1(motora);
   m2(motorb);
}
```

5. Flowchart


Start
float error, lasterror , I, output, motora, motorb;
int mmax = 255;
int mmin = 255;
int i = 0;
int light[] = {300, 300, 300, 300, 300};
Serial.begin(9600);
pinMode(A0, INPUT);
pinMode(A1, INPUT);
pinMode(A2, INPUT);
pinMode(A3, INPUT);
pinMode(A4, INPUT);
pinMode(9, OUTPUT);
pinMode(8, OUTPUT);
pinMode(7, OUTPUT);
pinMode(4, OUTPUT);
pinMode(5, OUTPUT);
pinMode(6, OUTPUT);
m1(255);
m2(255);
delay(300);
PIDC(50, 25, 0.01, 255, 1);
right(100, 200);
check(); m1(0); m2(0);
delay(50);
End
void m1
(int power)
if (power > 0)
Yes
digitalWrite(8, HIGH);
digitalWrite(7, LOW);
analogWrite(9, power);
End
No
else if (power > mmax)
Yes
digitalWrite(8, HIGH);
digitalWrite(7, LOW);
analogWrite(9, mmax);
End
No
else if (power < 0)
Yes
digitalWrite(8, LOW);
digitalWrite(7, HIGH);
analogWrite(9, -power);
End
No
else if (power < -mmin)
Yes
digitalWrite(8, LOW);
digitalWrite(7, HIGH);
analogWrite(9, mmin);
No
digitalWrite(8, LOW);
digitalWrite(7, LOW);
analogWrite(9, 0);
End
void right
(int s, int t)
m1(s);
m2(s);
delay(250);
m1(s);
m2(-s);
delay(t);
while (analogRead(A3) < light[3]);
void left
(int s, int t)
m1(s);
m2(s);
delay(250);
m1(s);
m2(-s);
delay(t);
while (analogRead(A2) < light[2])
void check
struct st_c
int sensor0;
int sensor1;
int sensor2;
int sensor3;
int sensor4;
struct st_c c = {analogRead(A0), analogRead(A1), analogRead(A2), analogRead(A3), analogRead(A4)};
Serial.print(c.sensor0);
Serial.print(",");
Serial.print(c.sensor1);
Serial.print(",");
Serial.print(c.sensor2);
Serial.print(",");
Serial.print(c.sensor3);
Serial.print(",");
Serial.println(c.sensor4);
End
void m2
(int power)
if (power > 0)
Yes
digitalWrite(4, HIGH);
digitalWrite(5, LOW);
digitalWrite(6, power);
End
No
else if (power > mmax)
Yes
digitalWrite(4, HIGH);
digitalWrite(5, LOW);
analogWrite(6, mmax);
End
No
else if (power < 0)
Yes
digitalWrite(4, LOW);
digitalWrite(5, HIGH);
digitalWrite(6, -power);
End
No
else if (power < -mmin)
Yes
digitalWrite(4, LOW);
digitalWrite(5, HIGH);
analogWrite(6, mmin);
No
digitalWrite(4, LOW);
digitalWrite(5, LOW);
analogWrite(6, 0);
End


void PIDC
(float kp, float kd, float ki, int basespeed, int i)
int count = 0
while (count < i)
No
End
Yes
if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
if (lasterror == -5 || lasterror == -4)
Yes
error = -5
Serial.println("error = -5")
No
error = 5
Serial.println("error = 5")
No
if (analogRead(A0) > light[0] && analogRead(A1) > light[1] && analogRead(A2) > light[2] && analogRead(A3) > light[3] && analogRead(A4) > light[4])
Yes
error = 0;
Serial.println("error = 0"); count++;
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) > light[4])
Yes
error = 4;
Serial.println("error = 4");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) > light[3] && analogRead(A4) > light[4])
Yes
error = 3;
Serial.println("error = 3");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) > light[3] && analogRead(A4) < light[4])
Yes
error = 2;
Serial.println("error = 2");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) > light[2] && analogRead(A3) > light[3] && analogRead(A4) < light[4])
Yes
error = 1;
Serial.println("error = 1");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) > light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = 0;
Serial.println("error = 0");
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] && analogRead(A2) > light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -1;
Serial.println("error = -1");
No
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -2;
Serial.println("error = -2");
No
else if (analogRead(A0) > light[0] && analogRead(A1) > light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -3;
Serial.println("error = -3");
No
else if (analogRead(A0) > light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -4;
Serial.println("error = -4");
No
I = I + error; output = (kp * error) + (kd * (error - lasterror)) + (ki * I); lasterror = error; Serial.println(lasterror); motora = basespeed + output; motorb = basespeed - output; Serial.println(motora); Serial.println(motorb); m1(motora); m2(motorb);


void PIDL
(float kp, float kd, float ki, int basespeed, int i)
int count = 0
while (count < i)
No
End
Yes
if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
if (lasterror == -5 || lasterror == -4)
Yes
error = -5
Serial.println("error = -5")
No
error = 5
Serial.println("error = 5")
No
if (analogRead(A0) > light[0] && analogRead(A2) > light[2])
Yes
error = 0;
Serial.println("error = 0"); count++;
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) > light[4])
Yes
error = 4;
Serial.println("error = 4");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) > light[3] && analogRead(A4) > light[4])
Yes
error = 3;
Serial.println("error = 3");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) > light[3] && analogRead(A4) < light[4])
Yes
error = 2;
Serial.println("error = 2");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) > light[2] && analogRead(A3) > light[3] && analogRead(A4) < light[4])
Yes
error = 1;
Serial.println("error = 1");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) > light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = 0;
Serial.println("error = 0");
No
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] && analogRead(A2) > light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -1;
Serial.println("error = -1");
No
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -2;
Serial.println("error = -2");
No
else if (analogRead(A0) > light[0] && analogRead(A1) > light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -3;
Serial.println("error = -3");
No
else if (analogRead(A0) > light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -4;
Serial.println("error = -4");
No
I = I + error;
output = (kp * error) + (kd * (error - lasterror)) + (ki * I);
lasterror = error;
Serial.println(lasterror);
motora = basespeed + output;
motorb = basespeed - output;
Serial.println(motora);
Serial.println(motorb);
m1(motora);
m2(motorb);


void PIDR
(float kp, float kd, float ki, int basespeed, int i)
int count = 0
while (count < i)
No
End
Yes
if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
if (lasterror == -5 || lasterror == -4)
Yes
error = -5
Serial.println("error = -5");
No
error = 5
Serial.println("error = 5");
No
if (analogRead(A2) > light[2] && analogRead(A1) > light[1])
Yes
error = 0;
Serial.println("error = 0"); count++;
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) > light[4])
Yes
error = 4;
Serial.println("error = 4");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) > light[3] && analogRead(A4) > light[4])
Yes
error = 3;
Serial.println("error = 3");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) > light[3] && analogRead(A4) < light[4])
Yes
error = 2;
Serial.println("error = 2");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) > light[2] && analogRead(A3) > light[3] && analogRead(A4) < light[4])
Yes
error = 1;
Serial.println("error = 1");
No
else if (analogRead(A0) < light[0] && analogRead(A1) < light[1] && analogRead(A2) > light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = 0;
Serial.println("error = 0");
NO
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] && analogRead(A2) > light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -1;
Serial.println("error = -1");
No
else if (analogRead(A0) < light[0] && analogRead(A1) > light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -2;
Serial.println("error = -2");
No
else if (analogRead(A0) > light[0] && analogRead(A1) > light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -3;
Serial.println("error = -3");
No
else if (analogRead(A0) > light[0] && analogRead(A1) < light[1] && analogRead(A2) < light[2] && analogRead(A3) < light[3] && analogRead(A4) < light[4])
Yes
error = -4;
Serial.println("error = -4");
No
I = I + error;
output = (kp * error) + (kd * (error - lasterror)) + (ki * I);
lasterror = error;
Serial.println(lasterror);
motora = basespeed + output;
motorb = basespeed - output;
Serial.println(motora);
Serial.println(motorb);
m1(motora);
m2(motorb);

6. Discussion

- The IR sensor is prone to errors because of its limit, which has an impact on the robot's movement. The sensor may be changed, or the sensor's height could be altered, to resolve this issue.
- The motor's low revolutions per minute (rpm) cause the robot to move more slowly. It can be resolved by using a different motor.

7. Improvement

- Try adding more sensors to know will it improve the performance of the robot
- Try changing the motor to increase it speed

Manual robot (Gripper and Projectile launcher)

1. Background Research

- **Gripper**

  A gripper is a tool that makes it possible to manipulate the holding of an object. Consider the human hand as an analogy to describe a gripper more easily. A gripper allows for the holding, tightening, handling, and releasing of an object, just like a hand. One element of an automated system is a gripper. A gripper can be a fixed automation device, or it can be coupled to a robot. Grippers come in a variety of shapes and sizes so that the ideal model can be chosen for the purpose.

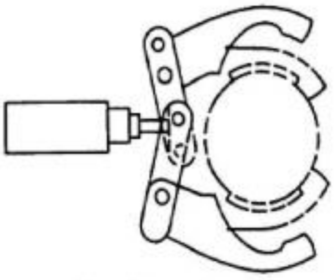


Figure 1 External gripper.

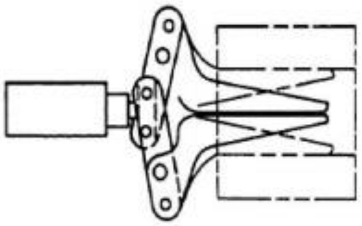


Figure 2 Internal gripper.

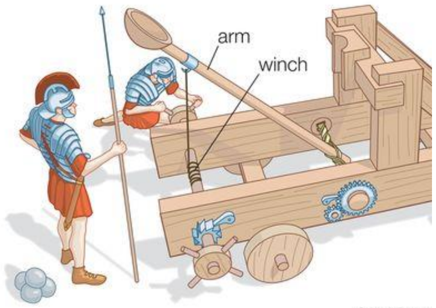
- **Selection and design considerations in robot gripper**

  Only with the right gripper selection and design can a robot successfully grasp objects. These are some of the several elements that must be considered while choosing and designing grippers.

  - The gripper needs to be able to access the work part's surface.
  - The gripper must not distort or scrape the delicate work parts, and the change in work part size must be considered to ensure correct positioning.
  - The gripper must hold the broader region of a work part if it has diverse dimensions, which will undoubtedly boost stability and control in placing.
  - If a work item has different dimensions, the gripper must hold the larger portion of it, which will undoubtedly boost positioning stability and control.
  - For additional gripping interactions in the work component, the gripper may be constructed with robust pads. Due to their interchangeability, the interchangeable fingers can also be used to grip various work part sizes.

- **Projectile Launcher**

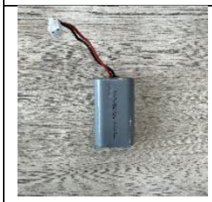
  To make it able to throw the ball over 16 cm we researched many kind of launcher and found that catapult is the best way for other projects as it is not very complex on designing and does not have a lot of resistance force do to it.

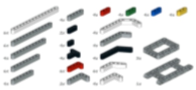


© 2010 Encyclc

We chose to use an outdated catapult for our project launcher. Greeks and Romans invented the catapult in the middle centuries. They discovered that a catapult's projectile motion can cause significant harm. To do the most harm, missiles were launched into the castle walls using catapults. They were among the earliest biological weapons that were known to exist. The Greeks would toss diseased bodies over the opposing line, infecting the populace and weakening the opposition. One of the most effective weapons in battle was catapults. The words "Catapult" and "Pluto’s," which together translate to "downward" and "small round combat shield," respectively, in Greek, originally meant "Shield Piercer." Today, catapults are employed in a variety of other settings, such as on Navy aircraft carriers. Because the runway on Navy battle ships is so short, takeoff presents a significant issue. The military developed a catapult device that enables the planes to accelerate to a speed at which they can take off. Today, most science and math theories are shown and demonstrated using catapults.

2. List of the component

Component	Name	Specification	Reference
	Makeblock Bluetooth Controller		https://www.makeblock.com/accessories/bluetooth-controller
	Mbot2 shield	Dimensions 170 x 90 x 130 mm and 1072g	https://education.makeblock.com/help/mbot2-shield/
	CyberPi	Professor using cyberOS, dimensions 84 x 35 x 13 mm and 36g	https://education.makeblock.com/help/cyberpi-series-cyberpi/
	Makeblock 180 Encoder Motor	7.4v 350 RPM±5 (without load) , torque 800 g·cm	http://docs.makeblock.com/diy-platform/en/electronic-modules/motors/180-optical-encoder-motor.html
	DC gearbox motor	6.0v 55 - 110 rpm(without load) torque 2 kg.cm Gear Ratio 1:120	https://store.makeblock.com/products/tt-geared-motor-dc6v-200rpm
	MS-1.5A Servo motor	6.0v with torque 1.8 kg ± 0.30 kg·cm	https://education.makeblock.com/help/cyberpi-series-ms-1-5a-servo/
	Rechargeable li-ion battery pack	8000mah 3.7v	https://www.imagineering.co.th

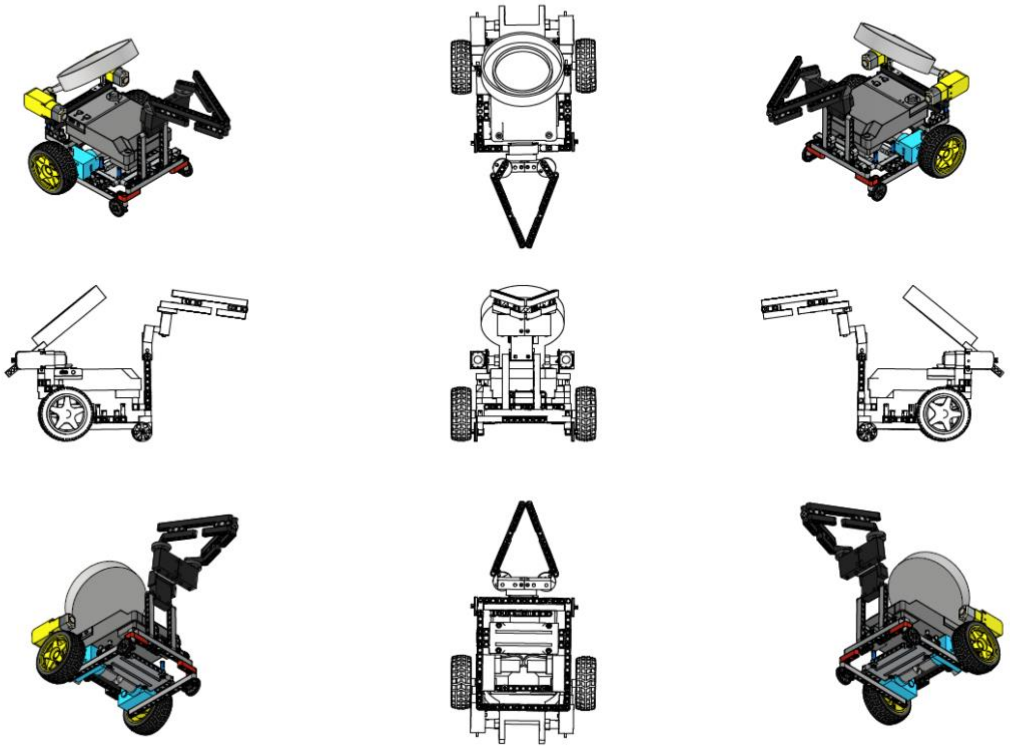
	TT motor wheel	68mm diameter	https://www.addicore.com/Wheel-for-TT-Motor-p/ad480.htm
	Pan screw	M3x15, M3x25, M4x15, M4x20, M4x25, M4x45	https://www.amazon.com/340pcs-Phillips-Stainless-Assortment-Thread/dp/B075RCVVYN
	Lego Mindstorms components		https://antonsmindstorms.com/2020/10/10/lego-mindstorms-robot-inventor-set-51515-part-list/
	Angle Strut Bracket	3mm hold	https://www.monotaro.co.th/p/68568884.html?utm_id=google_pla_th&gclid=CjwKCAiAy_CcBhBeEiwAcoMRHLnx95jEyIvZFIQjP5_J-dKsRq7EsHaTvBesfNITbM6RAjA7A4lTqxoCB10QAvD_BwE

3. Design

- **Design Concept**

Because we did not yet know or understand the area when we started developing the robot, utilizing Lego components made it easier to change or modify the design than using materials like acrylic or PLA plastic. However, as we design, we are aware that the 7 cm and 10 cm diameter balls must be grabbed and used to load the ball into the launcher. Therefore, we begin by putting the Lego piece together with the electric component that we have. We can modify the robot we've previously assembled to make it fit for the competition after we've seen the field.

- **Drawing**



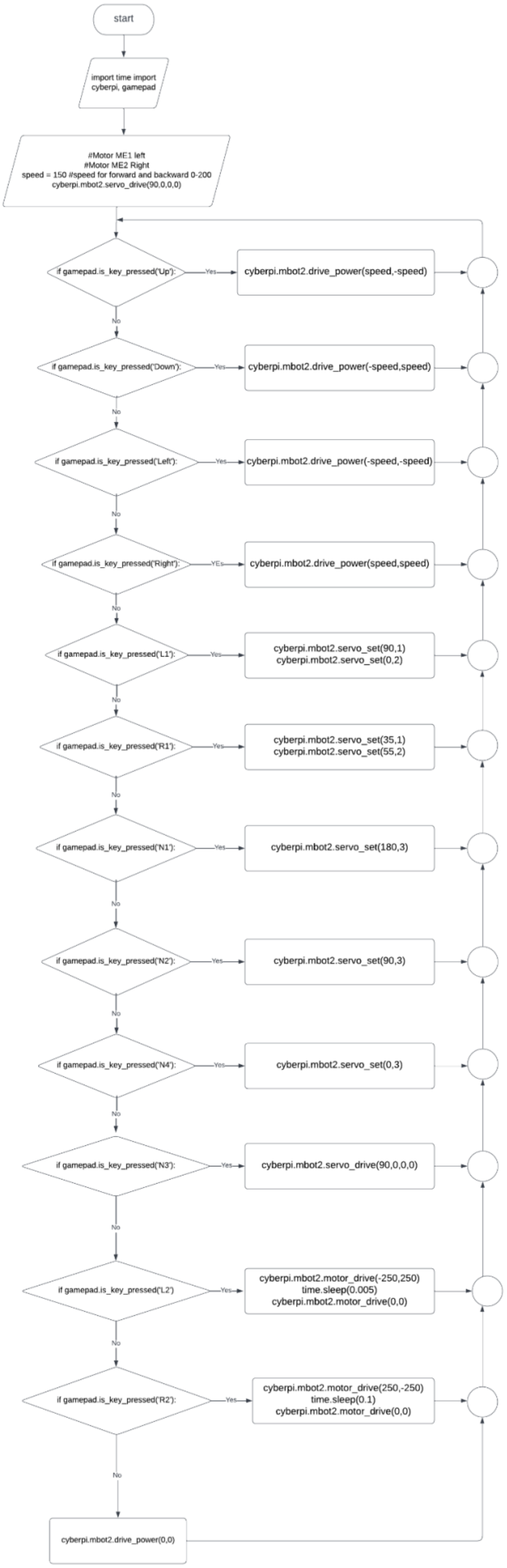
Third-angular-orthographic projection for all side view of the Manual Robot

4. Programming

- **Coding with concept**

```
import time
import cyberpi, gamepad

#Motor ME1 left
#Motor ME2 Right
speed = 150 #speed for forward and backward 0-200
cyberpi.mbot2.servo_drive(90,0,0,0)


while True:
    #Movement
    if gamepad.is_key_pressed('Up'):
        cyberpi.mbot2.drive_power(speed,-speed) #forward
    if gamepad.is_key_pressed('Down'):
        cyberpi.mbot2.drive_power(-speed,speed) #backward
    if gamepad.is_key_pressed('Left'):
        cyberpi.mbot2.drive_power(-speed,-speed)#left
    if gamepad.is_key_pressed('Right'):
        cyberpi.mbot2.drive_power(speed,speed) #right

    #Gripper
    if gamepad.is_key_pressed('L1'):    #outstretched
        cyberpi.mbot2.servo_set(90,1)
        cyberpi.mbot2.servo_set(0,2)
    if gamepad.is_key_pressed('R1'):    #fold
        cyberpi.mbot2.servo_set(35,1)
        cyberpi.mbot2.servo_set(55,2)

    #Gripper height
    if gamepad.is_key_pressed('N1'):    #ground
        cyberpi.mbot2.servo_set(180,3)
    if gamepad.is_key_pressed('N2'):    #parallel16cm
        cyberpi.mbot2.servo_set(90,3)
    if gamepad.is_key_pressed('N4'):    #aboveshotter
        cyberpi.mbot2.servo_set(0,3)
    if gamepad.is_key_pressed('N3'):    #resetgripper
        cyberpi.mbot2.servo_drive(90,0,0,0)

    #Shootter
    if gamepad.is_key_pressed('L2'):   #normal
        cyberpi.mbot2.motor_drive(-250,250)
        time.sleep(0.005)
        cyberpi.mbot2.motor_drive(0,0)
    if gamepad.is_key_pressed('R2'):   #shooterpoint
       cyberpi.mbot2.motor_drive(250,-250)
       time.sleep(0.1)
       cyberpi.mbot2.motor_drive(0,0)
    else:
        cyberpi.mbot2.drive_power(0,0)
```

5. Flowchart


start
import time import cyberpi, gamepad
#Motor ME1 left
#Motor ME2 Right
speed = 150 #speed for forward and backward 0-200
cyberpi.mbot2.servo_drive(90,0,0,0)
if gamepad.is_key_pressed('Up'):
Yes
cyberpi.mbot2.drive_power(speed,-speed)
No
if gamepad.is_key_pressed('Down'):
Yes
cyberpi.mbot2.drive_power(-speed,speed)
No
if gamepad.is_key_pressed('Left'):
Yes
cyberpi.mbot2.drive_power(-speed,-speed)
No
if gamepad.is_key_pressed('Right'):
YEs
cyberpi.mbot2.drive_power(speed,speed)
No
if gamepad.is_key_pressed('L1'):
Yes
cyberpi.mbot2.servo_set(90,1)
cyberpi.mbot2.servo_set(0,2)
No
if gamepad.is_key_pressed('R1'):
Yes
cyberpi.mbot2.servo_set(35,1)
cyberpi.mbot2.servo_set(55,2)
No
if gamepad.is_key_pressed('N1'):
Yes
cyberpi.mbot2.servo_set(180,3)
No
if gamepad.is_key_pressed('N2'):
Yes
cyberpi.mbot2.servo_set(90,3)
No
if gamepad.is_key_pressed('N4'):
Yes
cyberpi.mbot2.servo_set(0,3)
No
if gamepad.is_key_pressed('N3'):
Yes
cyberpi.mbot2.servo_drive(90,0,0,0)
No
if gamepad.is_key_pressed('L2')
Yes
cyberpi.mbot2.motor_drive(-250,250)
time.sleep(0.005)
cyberpi.mbot2.motor_drive(0,0)
No
if gamepad.is_key_pressed('R2'):
Yes
cyberpi.mbot2.motor_drive(250,-250)
time.sleep(0.1)
cyberpi.mbot2.motor_drive(0,0)
No
cyberpi.mbot2.drive_power(0,0)

6. Calculation

- **Calculation 1**
  - **Find the highest point of the ball**

$$t = 2u\,sin\theta/g$$
$$t = (2u \sin 8\ 0°)/9.8$$
$$t = \big(2(4.2)(0.98)\big)/9.8$$
$$t = 0.84\ s$$

- **Calculation 2**
  - **Find the Displacement of x-axis**

$$S_x = U_x \times t$$
$$S_x = U \cos(80°)\,.\,t$$
$$S_x = 4.2 \cos(80°)\,.\,0.84$$
$$S_x = 0.60\ m$$

- **Calculation 3**
  - **Find the velocity of both ball**

$$Displacement = velocity \times time$$
$$S = vt$$
$$v = s/t$$
$$v = 0.75\ m/0.48\ s$$
$$v = 1.56m/s$$

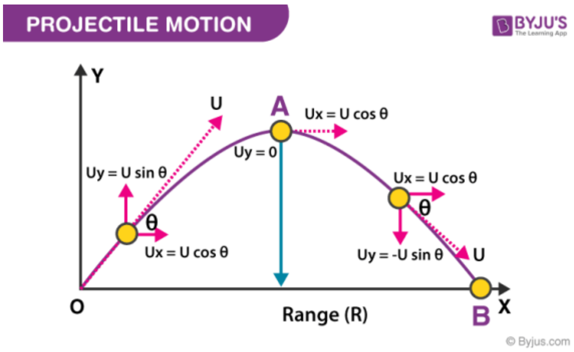
- **Calculation 4**
  - **Find the Relative Error**

$$\left|\frac{Measured\ Value\ -\ Expected\ Value}{Expected\ Value}\right| \times 100$$

$$\left|\frac{0.60 - 0.74}{0.60}\right| \times 100$$

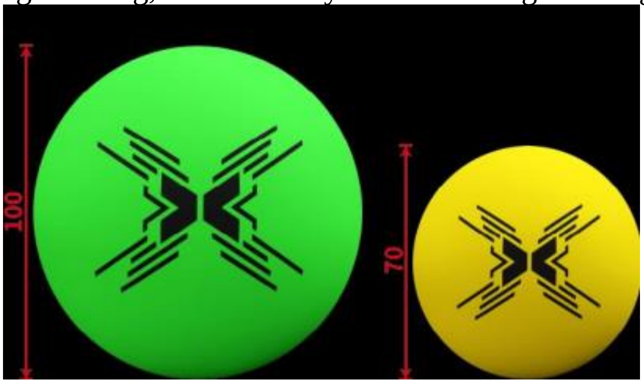
$$= 23.3\%$$

7. Result



The diagram clarifies Displacement X for the upcoming table

The green ball weighs 43.5 g, whereas the yellow ball weighs 14.3 g.



The displacement in x, time, and velocity of both balls are displayed in the table.

Round	Green Ball			Yellow Ball		
	Displacement X (m.)	time(s.)	Velocity(m/s)	Displacement X (m.)	time(s.)	Velocity(m/s)
1	0.75	0.48	1.56	0.93	0.54	1.72
2	0.78	0.47	1.65	0.90	0.60	1.50
3	0.70	0.50	1.40	0.88	0.61	1.44
4	0.72	0.51	1.41	0.91	0.58	1.56
5	0.74	0.50	1.48	0.90	0.57	1.57
Average	0.74	0.50	1.50	0.90	0.58	1.36
Relative error	23.30%	-	-	50%	-	-

The average displacement in the X-axis for the green ball is 0.74 m, and the average time is 0.50 s. Additionally, the yellow ball has an average displacement in the X-axis of 0.90 m and an average time of 0.58 s. For the green ball and the yellow ball, respectively, the relative error of displacement in the X-axis is 23.30% and 50%.

8. Discussion

- We need to modify the screw so that it is higher than the ball holder since the gripper is too low to grasp the ball.
- Since we must measure from the ball's center to the point at which it lands on the ground, there will be a human error when measuring the displacement of the x axis.
- To grab the green ball, we first grabbed it, dropped it because it was too huge, and then we grabbed it again.

9. Improvement

- Add a projectile launcher so that the angle may be changed to increase and decrease the length; try a different speed and kind of motor for the gripper and projectile launcher.

10.Reference

- https://www.elprocus.com/the-working-of-a-pid-controller/
- https://www.britannica.com/technology/catapult-military-weaponry
- https://www.automation.com/en-us/articles/2011-1/gripper-101-what-is-a-gripper#:~:text=A%20gripper%20is%20a%20device,component%20of%20an%20automated%20system.
- https://www.brainkart.com/article/Selection-and-design-considerations-in-robot-gripper_5141/#:~:text=The%20gripper%20must%20have%20the,surface%20of%20a%20work%20part.&text=The%20change%20in%20work%20part,accounted%20for%20providing%20accurate%20positioning.&text=During%20machining%20operations%2C%20there%20will,when%20the%20size%20is%20varied.
- https://robotsiam.blogspot.com/2016/10/ir-infrared-obstacle-avoidance-sensor.html
- https://einstronic.com/product/tt-motor-yellow-geared-dc-motor/
- https://th.rs-online.com/web/c/batteries-chargers/battery-charger-accessories/battery-holders/
- https://thai.alibaba.com/product-detail/60mm-plastic-push-button-switch-electric-1600070601327.html
- https://www.arduino4.com/product/325/arduino-uno-r3-sensor-shield-v5-expansion-board
- https://www.tido.tech/index.php/product/l298n-motor-driver-module/
- http://docs.makeblock.com/diy-platform/en/mechanical-parts/wheels/slick-tyre-64-16mm.html
- https://th.misumi-ec.com/en/vona2/detail/221000547315/
- https://www.makeblock.com/accessories/bluetooth-controller

- https://www.mcmaster.com/strut-channel-angle-brackets/
- https://education.makeblock.com/help/mbot2-shield/
- https://education.makeblock.com/help/cyberpi-series-cyberpi/
- http://docs.makeblock.com/diy-platform/en/electronic-modules/motors/180-optical-encoder-motor.html
- https://education.makeblock.com/help/cyberpi-series-ms-1-5a-servo/https://store.makeblock.com/products/tt-geared-motor-dc6v-200rpm
- https://www.imagineering.co.th
- https://www.addicore.com/Wheel-for-TT-Motor-p/ad480.htm
- https://antonsmindstorms.com/2020/10/10/lego-mindstorms-robot-inventor-set-51515-part-list/
- https://www.amazon.com/340pcs-Phillips-Stainless-Assortment-Thread/dp/B075RCVVYN
- https://byjus.com/physics/projectile-motion/
- https://www.monotaro.co.th/p/68568884.html?utm_id=google_pla_th&gclid=CjwKCAiAy_CcBhBeEiwAcoMRHLnx95jEyIvZFIQjP5_JdKsRq7EsHaTvBesfNITbM6RAjA7A4lTqxoCB10QAvD_BwE
- https://www.makeblock.in.th/wp-content/uploads/2022/03/2022-season-makex-explorer-eco-pioneer-rules-guide-v1.01.pdf