Nama : Mitsal Fabian Nadhiem (NIM 6702193063)
: Dika Achmad Putra (NIM 6702191072)
Tim : Beskarr

Praktikum Sistem Kendali Kasus P

Maksud dan Tujuan

1. Dapat memahami fungsi dan cara kerja PID pada motor DC
2. dapat membuat program sistem kendali berbasis PID dengan error yang dihubungkan dengan konstanta proporsional dan derivative

Peralatan dan Bahan

Pada kegiatan praktikum kali ini kita membangun sebuah system kendali PID DC motor encoder dengan input *Photodiode* dengan menggunakan Arduino Uno R3. Berikut adalah rincian komponen dan rangkaian yang digunakan.

Komponen	Fungsi	Jumlah
Arduino Uno R3	Mikrokontroller sebagai pusat pemroses input sinyal elektronik menjadi sinyal elektronik yang dibutuhkan.	1
DC Motor	Output rangkaian, mengubah sinyal menjadi energi gerak.	1
Photodiode	Sensor untuk mendeteksi cahaya, Photodiode ini akan mengubah cahaya menjadi arus listrik.	6
H-bridge Motor Driver	Rangkaian untuk mengubah arah arus listrik di motor. Perubahan arah arus tersebut digunakan untuk mengubah putaran motor *ClockWise(CW)* atau CounterClockWise (CCW).	1
33kΩ Resistor	Menghambat serta mengatur arus listrik dalam suatu rangkaian.	6
Breadboard mini	Board yang digunakan untuk membuat rangkaian elektronik sementara tanpa harus menyolder.	1
Breadboard	Board yang digunakan untuk membuat rangkaian elektronik sementara tanpa harus menyolder.	1
Power Supply	Berfungsi untuk menyuplai tegangan langsung ke komponen-komponen	1

Push Button	Berfungsi sebagai input, untuk menambah dan mengurangi nilai Kp	2
DC Motor Encoder	Output rangkaian, mengubah sinyal menjadi energi gerak.	1

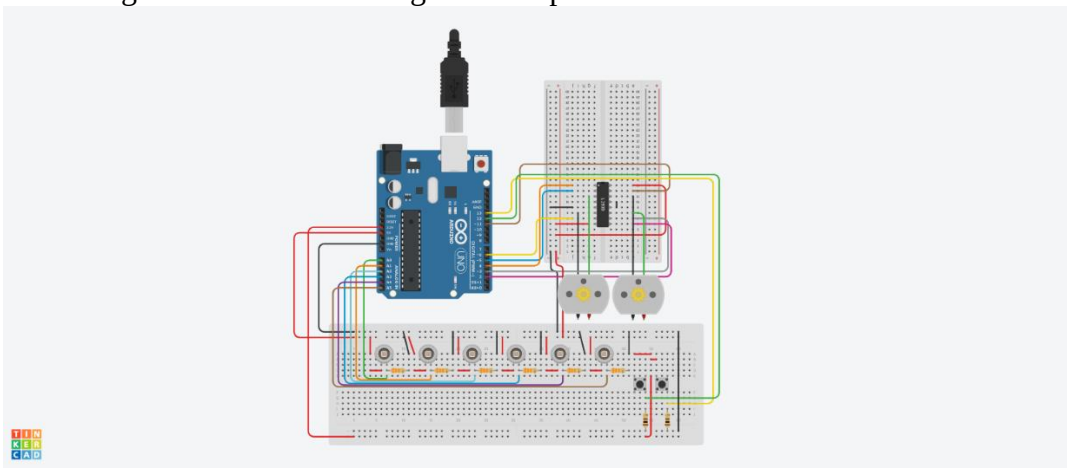
Dasar Teori

Aksi kendali proporsional (P) adalah aksi kendali yang memiliki karakter dapat mengurangi waktu naik (rise time), tetapi tidak menghilangkan kesalahan keadaan tunak (steady satate error).

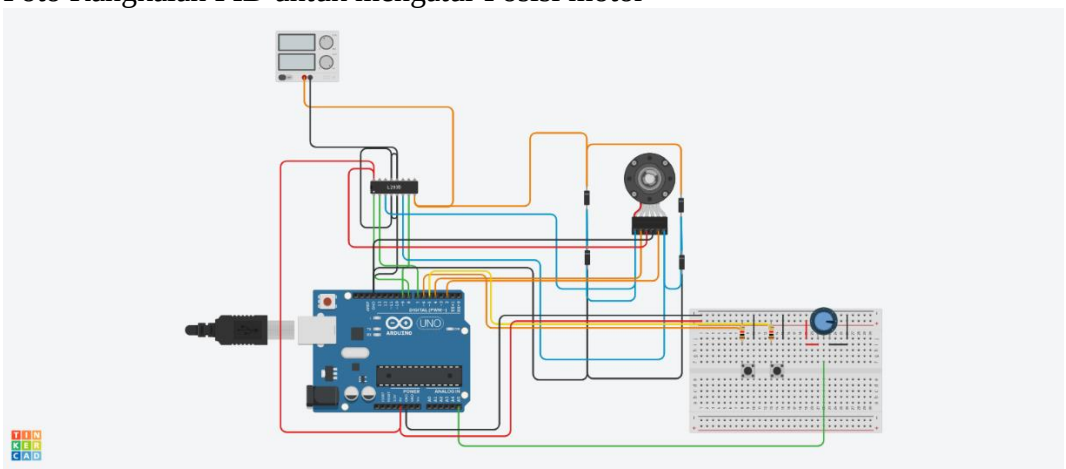
Pengontrol proposional memiliki keluaran yang sebanding atau proposional dengan besarnya sinyal kesalahan (selisih antara besaran yang di inginkan dengan harga aktualnya). Secara lebih sederhana dapat dikatakan bahwa keluaran pengontrol proporsional merupakan perkalian antara konstanta 6 proposional dengan masukannya. Perubahan pada sinyal masukan akan segera menyebabkan sistem secara langsung mengeluarkan output sinyal sebesar konstanta pengalinya.

Foto Rangkaian

a. Foto Rangkaian PID untuk mengatur Kecepatan motor



b. Foto Rangkaian PID untuk mengatur Posisi motor



Hasil Praktikum

Tabel 1Tabel Pengaturan kecepatan berdasarkan nilai sensor

Sensor								Error	Nilai Setpoint	Analog Value	
0	1	2	3	4	5					Motor Kiri	Motor Kanan
0	1	1	1	1	1			-5	100	125	175
0	0	1	1	1	1			-4	100	130	170
1	0	1	1	1	1			-3	100	135	165
1	0	0	1	1	1			-2	100	140	160
1	1	0	1	1	1			-1	100	145	155
1	1	0	0	1	1			0	100	150	150
1	1	1	0	1	1			1	100	155	145
1	1	1	0	0	1			2	100	160	140
1	1	1	1	0	1			3	100	165	135
1	1	1	1	0	0			4	100	170	130
1	1	1	1	1	0			5	100	175	125

a. Kode Rangkaian PID untuk Mengatur Kecepatan Motor

```
//deklarasi pin sensor
int sensor1 = A0;
int sensor2 = A1;
int sensor3 = A2;
int sensor4 = A3;
int sensor5 = A4;
int sensor6 = A5;
int dataSensor[6];

int pushp = 13;
int pushm = 12;

//deklarasi pin enable
int leftEN = 4;
int rightEN = 2;

//deklarasi pin motor kiri
int leftMotor1 = 5;
int leftMotor2 = 6; //default selalu 0

//deklarasi pin motor kanan
int rightMotor1 = 3;
int rightMotor2 = 11; //default selalu 0

//
int rightMotorSpeed;
int leftMotorSpeed;

//deklarasi variabel untuk menyimpan nilai error
int lastError = 0;
int error = 0;
```

```
//PID
int moveControl;
int motorSpeed = 150;
int kp = 5;
int ki = 0;
int kd = 0;
int sensorBit;



void setup()
{
  //inisialisasi pin sensor
  pinMode(sensor1, INPUT);
  pinMode(sensor2, INPUT);
  pinMode(sensor3, INPUT);
  pinMode(sensor4, INPUT);
  pinMode(sensor5, INPUT);
  pinMode(sensor6, INPUT);

  //inisialisasi pin enable
  pinMode(leftEN, OUTPUT);
  pinMode(rightEN, OUTPUT);

  //inisialisasi pin motor kiri
  pinMode(leftMotor1, OUTPUT);
  pinMode(leftMotor2, OUTPUT);

  //inisialisasi pin motor kanan
  pinMode(rightMotor1, OUTPUT);
  pinMode(rightMotor2, OUTPUT);

  //inisialisasi pin button
  pinMode(pushp, INPUT);
  pinMode(pushm, INPUT);
  Serial.begin(9600);
}
void readSensor(){
  dataSensor[0] = analogRead(sensor1);
  dataSensor[1] = analogRead(sensor2);
  dataSensor[2] = analogRead(sensor3);
  dataSensor[3] = analogRead(sensor4);
  dataSensor[4] = analogRead(sensor5);
  dataSensor[5] = analogRead(sensor6);
  for(int i = 0; i <=5; i++){
    if (dataSensor[i] > 35){
      dataSensor[i] = 1;
    }
    else {
      dataSensor[i] = 0;
```

```
    }
  }

  sensorBit = 0;
  for(int i = 0; i <=5; i++){
    sensorBit += dataSensor[i] * (1 << i);
  }

}

void loop()
{
  if(digitalRead(pushp)==HIGH){
    kp = kp + 1;
    while(digitalRead(pushp)==HIGH){}
  }
  if(digitalRead(pushm)==HIGH){
    kp = kp - 1;
    while(digitalRead(pushm)==HIGH){}
  }

  while (digitalRead(pushp)==LOW && digitalRead(pushm)==LOW){

    readSensor();

          //menentukan pembacaan sensor
    switch(sensorBit){
      case 62: error = -5; break;
      case 60: error = -4; break;
      case 61: error = -3; break;
      case 57: error = -2; break;
      case 59: error = -1; break;
      case 51: error = 0; break;
      case 55: error = 1; break;
      case 39: error = 2; break;
      case 47: error = 3; break;
      case 15: error = 4; break;
      case 31: error = 5; break;
    }


    moveControl = setMotor(error, lastError,kp,ki,kd);
    //rumus motor kanan motor kiri
    rightMotorSpeed = motorSpeed - moveControl;
    leftMotorSpeed = motorSpeed + moveControl;

    digitalWrite(rightEN, HIGH);
    analogWrite(rightMotor1, rightMotorSpeed);
    analogWrite(rightMotor2, 0);
```

```
    digitalWrite(leftEN, HIGH);
    analogWrite(leftMotor1,  leftMotorSpeed);
    analogWrite(leftMotor2, 0);
    Serial.print(rightMotorSpeed);
    Serial.print("\n");
    lastError = error;
  }
}

int setMotor(int error, int lastError, int kp, int ki, int kd){
    int rate_d = error - lastError;
    int rate_i = error + lastError;
    int moveControl = (kp * error) + (kd * rate_d) + (ki * rate_i); //(-25) + 0 + 0

    return moveControl;
}
```

b. Kode Rangkaian PID untuk Mengatur Posisi Motor

```
//motor directory
#define CW  0
#define CCW 1

//define button
#define pushsub 6
#define pushplus 5

//motor control pin
#define motorDirPin 7
#define motorPWMPin 9
#define enablePin 8

//encoder pin
#define encoderPinA 2
#define encoderPinB 4

//encoder var
int encoderPos = 0;

float Kp          = 10;
int   targetPos;
int   error;
int   control;
int   velocity;

//external interrupt encoder
void doEncoderA()
{
  digitalRead(encoderPinB)?encoderPos--:encoderPos++;
}
```

```
void setup()
{
  //setup interrupt
    pinMode(encoderPinA, INPUT_PULLUP);
    pinMode(encoderPinB, INPUT_PULLUP);
    attachInterrupt(digitalPinToInterrupt(encoderPinA), doEncoderA,RISING);

    //setup motor driver
    pinMode(motorDirPin, OUTPUT);
    pinMode(enablePin, OUTPUT);
    digitalWrite(enablePin, HIGH);

      pinMode(pushplus, INPUT);
      pinMode(pushsub, INPUT);


    Serial.begin(9600);
}


void loop()
{
  //Kondisi untuk menaikan atau menurunkan KP menggunakan push button
      if(digitalRead(pushplus)==LOW){
        Kp = Kp + 1;
        while(digitalRead(pushplus)==LOW){}
      }
      if(digitalRead(pushsub)==LOW){
        Kp = Kp - 1;
        while(digitalRead(pushsub)==LOW){}
      }

    targetPos = analogRead(A5)/10; //potentiometer sebagai penentu targetpos
    error   = targetPos - encoderPos;
    control = Kp * error;

    velocity = min(max(control, -255), 255);

    if(velocity >= 0)
    {
       digitalWrite(motorDirPin, CW); //output
       analogWrite(motorPWMPin, velocity); //output duty
    }
    else
    {
       digitalWrite(motorDirPin, CCW);
       analogWrite(motorPWMPin, 255+velocity);
    }
    Serial.println(encoderPos);
```

}

Kesimpulan

Berdasarkan percobaan praktikum yang telah dilakukan, dapat disimpulkan bahwa:

Aksi kendali proporsional (P) adalah aksi kendali yang memiliki karakter dapat mengurangi waktu naik (rise time), tetapi tidak menghilangkan kesalahan keadaan tunak (steady satate error).