```
// Pins for trigger and echo on SR04
int triggerPin = 4;
int echoPin = 5;
double duration, distance;

// Initialize all variables for PIDs
int prevMillis = 0;
int currentMillis = 0;
double dt = 0;
int setPoint = 0;
int currentDistance = 0;
int error = 0;
int lastError = 0;
int cumError = 0;
double P = 0;
double Kp = 4;
double I = 0;
double Ki = 0.01;
double D = 0;
double Kd = 5;
int c = 0;

int throttle;
int setThrottle;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600); //Serial communication
```

```
//  Setup sonar
//  HCSR04.begin(triggerPin, echoPin);
  pinMode(triggerPin, OUTPUT);
  pinMode(echoPin, INPUT);
  setThrottle = 1000;
  Serial.print("Initial Throttle value: ");
  Serial.println(throttle);
  Serial.println();
  delay(1000);
}

void loop() {
// put your main code here, to run repeatedly:
//set PIDs here with data from sonar
  if (Serial.available()){
    int setThrottle = Serial.parseInt();
  }
  if (c==0) {
    throttle = 1000;
    prevMillis = millis();
    c += 1;
    throttle = throttle;
//    Serial.println("Drone started and minimum
throttle given\n\n");
  } else {
//    Serial.println("Entered main body");
    currentMillis = millis();
    dt = currentMillis - prevMillis;
```

```
    setPoint = map(setThrottle, 1000, 2000, 8, 100);

    double distance = computeDistance();
    while(distance<0 || distance>200){
      distance = computeDistance();
    }

    currentDistance = distance;
    error = setPoint - currentDistance;
    cumError += error;
    P = Kp*error;
    Serial.print("P: ");
    Serial.println(P);
    I = Ki*cumError*dt;
    Serial.print("I: ");
    Serial.println(I);
    D = Kd*(error-lastError)/dt;
    Serial.print("D: ");
    Serial.println(D);
    throttle = P+I+D;
    Serial.print("PID in distance: ");
    Serial.println(throttle);
    throttle = map(throttle, 8, 100, 1000, 2000);
    Serial.print("PID in throttle: ");
    Serial.println(throttle);
    lastError = error;
    prevMillis = currentMillis;
//    Remove anomalies in throttle values
```

```
    if(throttle<1000){
      throttle = 1000;
    } else if (throttle > 2000){
      throttle = 2000;
    } else {
      throttle = throttle;
    }
//    ppm[0] = throttle;

    Serial.print("Setpoint: ");
    Serial.println(setPoint);
    Serial.print("Current Distance: ");
    Serial.println(currentDistance);
    Serial.print("Error: ");
    Serial.println(error);
//    Serial.print("Previous Throttle: ");
//    Serial.print(map(setPoint, 8, 100, 1000,
2000));
    Serial.print("Current Throttle: ");
    Serial.println(throttle);
    Serial.println(""); //For Testing Purpose
  }
  delay(1000);
}

double computeDistance(){
  double distance = 0;
  for (int i = 0; i< 10; i++){
```

```
    digitalWrite(triggerPin, LOW);
    delayMicroseconds(2);
//      Sets the trigPin on HIGH state for 10 micro
seconds
    digitalWrite(triggerPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(triggerPin, LOW);
//      Reads the echoPin, returns the sound wave
travel time in microseconds
    duration = pulseIn(echoPin, HIGH);
//      Calculating the distance
    distance += duration*0.034/2;
  }
  return distance /= 10;
}
```