

## LESSON 7

### **Collision Detection**

In this lesson, you will program your Pi-Bot to prevent accidents.

Combining with what we have learned about the ultrasonic sensor, together using interrupt control and motion control, we shall now investigate obstacle avoidance.

Download and run the program: collision\_detection.ino, as shown in figure 7.1. Your Pi-Bot will continue forward at a medium speed until an obstacle is detected within 10cm of the sensor. Upon detecting the object in front of the sensor, your Pi-Bot will reverse in a backward turn until the obstacle is no longer in front of the sensor plus 1 second after all obstacles have left the field of view.

#### Exercise 1

1. Modify the forward speed and the backward speed of your Pi-Bot. Try different speeds on the two wheels when traveling in reverse.

#### Exercise 2

1. Upon detecting an obstacle, have your Pi-Bot reverse in different directions.
2. For example, upon detecting the first obstacle, have your Pi-Bot reverse while turning to the right. The next time it detects an obstacle, have your Pi-Bot turn to the left.

The program in figure 7.1 contains additional functions which are not used. These are included for your own modification and learning. Experiment with the various functions to see what they do!

```

// collision_detection.ino
// Collision avoidance program

const int trigPin = 2;    // trigger pin
const int echoPin = 4;    // echo pin
const int Led = 13;       // led pin
const int In4 = 11;       // In4
const int In3 = 6;        // In3
const int In2 = 5;        // In2
const int In1 = 3;        // In1

void setup()
{
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(In1, OUTPUT);
  pinMode(In2, OUTPUT);
  pinMode(In3, OUTPUT);
  pinMode(In4, OUTPUT);
}

void loop()
{
  int speed1, speed2;
  float distance;
  speed1 = 200;
  speed2 = 200;
  forward(speed1, speed2);
  while (1==1)
  {
    distance = trigPulse();
    if(distance < 10)
    {
      stopNow();
      delay(10);
      speed1 = 200;
      speed2 = 200;
      reverse(speed1, speed2);
      while(distance < 25)
      {
        distance = trigPulse();
        delay(10);
      }
      stopNow();
      delay(10);
      speed1 = 200;
      speed2 = 200;
      spinLeft(speed1, speed2);
      delay(500);
      stopNow();
      speed1 = 200;
      speed2 = 200;
      forward(speed1, speed2);
    }
    delay(20);
  }
}

```

```

float trigPulse()
{
    long duration;
    float distance;
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    duration = pulseIn(echoPin, HIGH);
    distance = (duration/2) / 29.1;
    return distance;
}

void forward(int speed1, int speed2)
{
    analogWrite(In4, speed1);
    analogWrite(In3, 0);
    analogWrite(In2, 0);
    analogWrite(In1, speed2);
}

void reverse (int speed1, int speed2)
{
    analogWrite(In4, 0);
    analogWrite(In3, speed1);
    analogWrite(In2, speed2);
    analogWrite(In1, 0);
    return;
}

void spinLeft (int speed1, int speed2)
{
    analogWrite(In4, speed1);
    analogWrite(In3, 0);
    analogWrite(In2, speed2);
    analogWrite(In1, 0);
    return;
}

void spinRight (int speed1, int speed2)
{
    analogWrite(In4, 0);
    analogWrite(In3, speed1);
    analogWrite(In2, 0);
    analogWrite(In1, speed2);
    return;
}

void stopNow()
{
    analogWrite(In4, 0);
    analogWrite(In3, 0);
    analogWrite(In2, 0);
    analogWrite(In1, 0);
    return;
}

```

*Figure 7.1: Program – collision\_detection.ino*

Next lesson, we will cover line following.