

Chapter 4

Build the Devastator to be an unmanned tank

The PIR module that we introduced in chapter 3 enables our robot to detect people nearby. However, the PIR module can neither detect general objects nor send back exact position parameters to the micro controller. Therefore, an ultrasonic sensor module will be used for more complicated positioning functions. In addition to that, a mechanical mount (the pan and tilt kit) is also needed for adjusting the ultrasonic sensor module with the camera module facing towards required orientations. Therefore, in this chapter, we will briefly explain the mechanism of the ultrasonic sensor module and introduce how to make our robot capable of detecting random obstacles by setting up the pan&tilt kit with the ultrasonic sensor module. By the end of this chapter, we should be getting an unmanned tank which is able to detect and avoid obstacles automatically.

Ultrasonic sensor module

1. What is ultrasound

To understand the mechanism of the Ultrasonic sensor, we need to look into a few physical properties of ultrasounds. Ultrasounds are mechanical waves with frequencies higher than the upper audible limit of human hearing. Its frequency ranges from 20 kHz up to several gigahertz. Same as other mechanical waves, Ultrasounds travel through materials with some certain velocity and can be deflected or reflected when it hits obstacles.

2. Mechanism of the ultrasonic sensor module

The ultrasonic sensor module is composed by ultrasound transceiver and ultrasound receiver. Ultrasound transceiver generates sound waves range from 40 kHz to 45 kHz, once the sound waves hit an obstacle and get reflected back, ultrasound receiver captures the echo. By measuring the time interval between sending and receiving action, the distance can be determined based on the velocity of the sound waves (344 m/s in the air).

Connection

Make sure all components are connected to the board as following:

Vcc(Left_Motor) >> VCC(Arduino M2)

(Left_Motor) >> GND(Arduino M2)

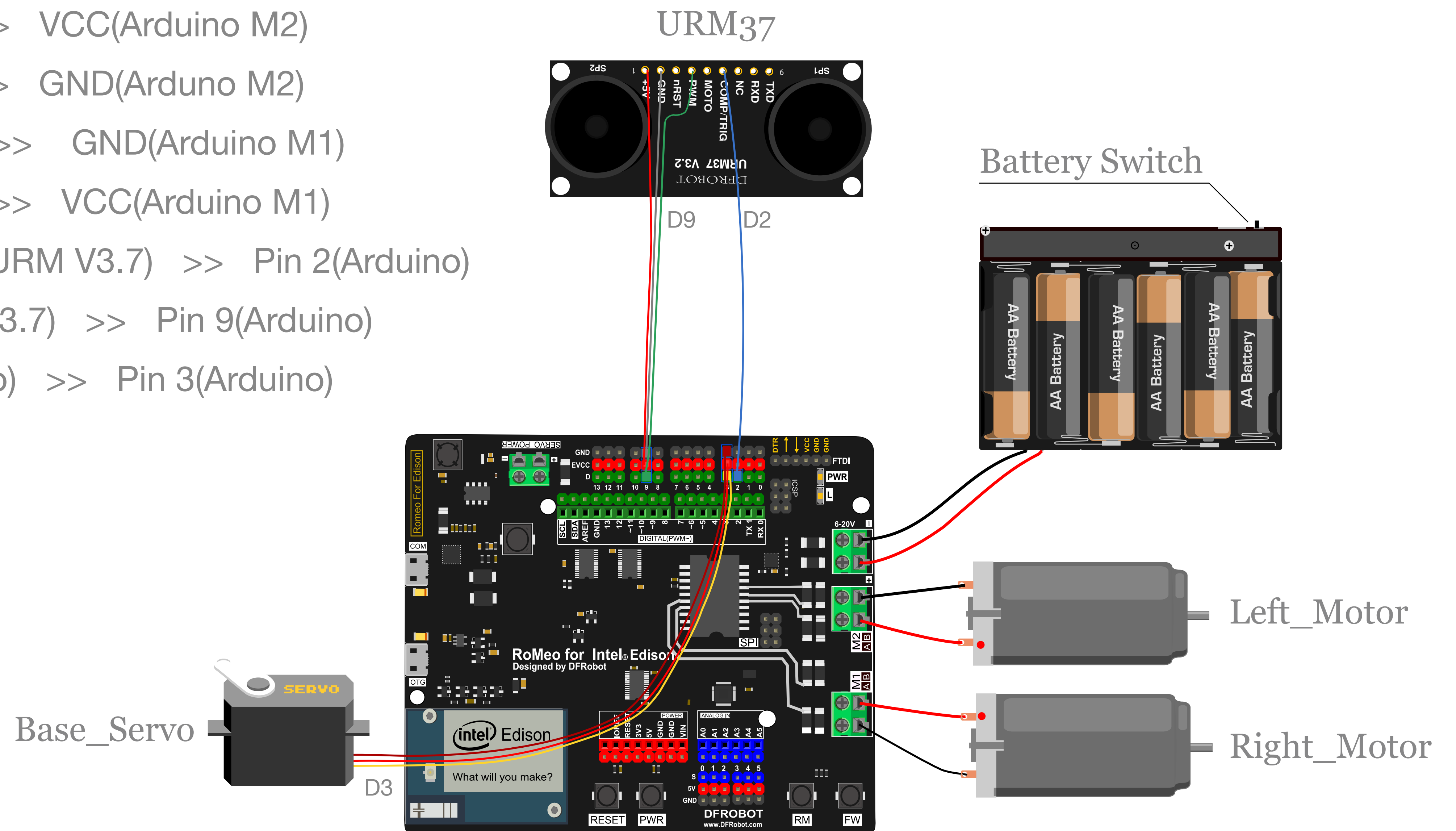
Vcc(Right_Motor) >> GND(Arduino M1)

(Right_Motor) >> VCC(Arduino M1)

Pin 6 COMP/TRIG(URM V3.7) >> Pin 2(Arduino)

Pin 4 ECHO(URM V3.7) >> Pin 9(Arduino)

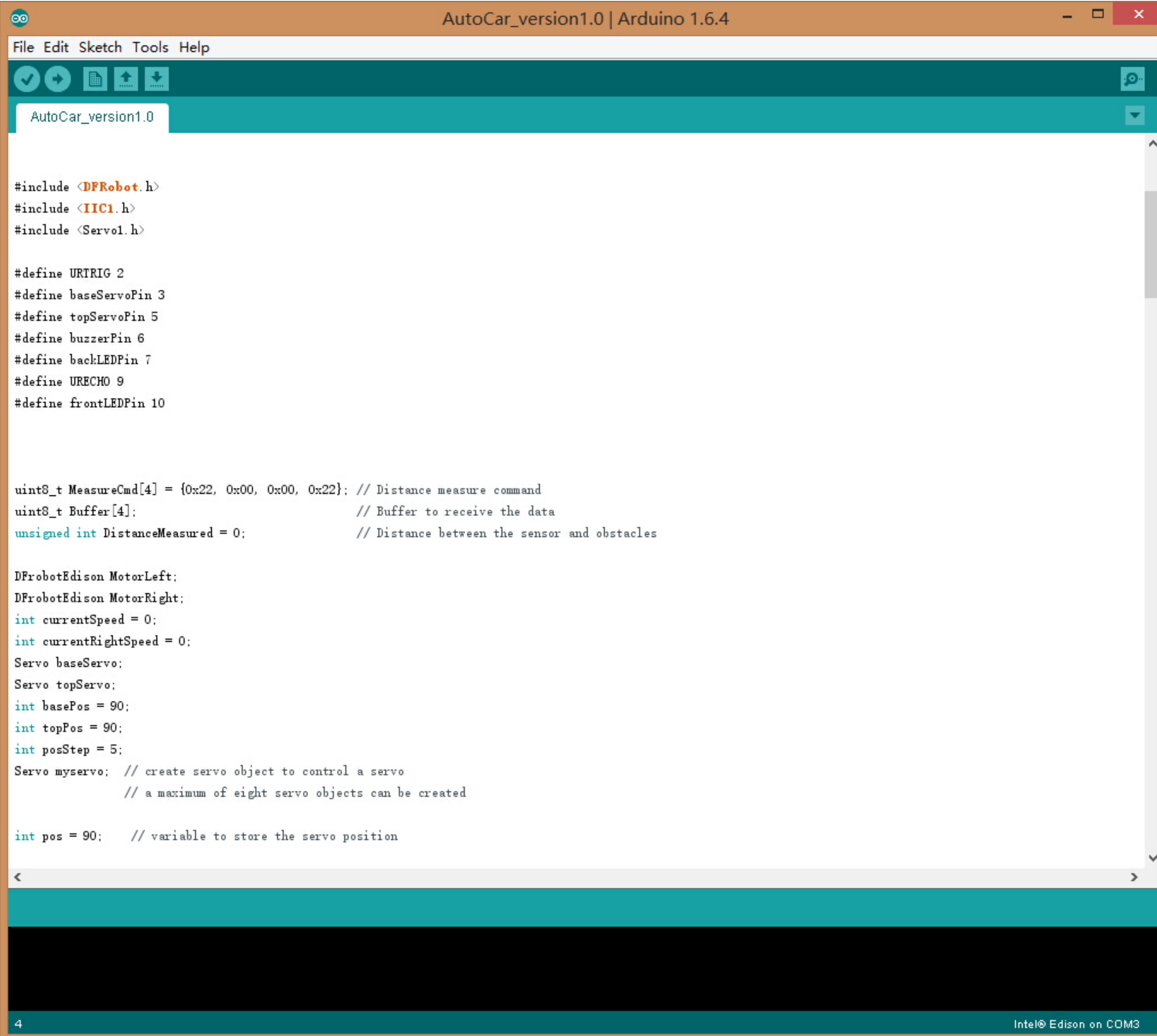
baseServoPin(Servo) >> Pin 3(Arduino)



Circuit connection

Upload testing code

As always, before we get started, make sure all the libraries that we need is imported, correct COM port and board type is selected. Due to the size of the sample code, you can download the INO format file in link below and open the file with Arduino IDE.



```
AutoCar_version1.0 | Arduino 1.6.4
File Edit Sketch Tools Help

#include <DFRobot.h>
#include <IIC1.h>
#include <Servo.h>

#define URTRIG 2
#define baseServoPin 3
#define topServoPin 5
#define buzzerPin 6
#define backLEDPin 7
#define URECHO 9
#define frontLEDPin 10

uint8_t MeasureCmd[4] = {0x22, 0x00, 0x00, 0x22}; // Distance measure command
uint8_t Buffer[4]; // Buffer to receive the data
unsigned int DistanceMeasured = 0; // Distance between the sensor and obstacles

DFRobotEdison MotorLeft;
DFRobotEdison MotorRight;
int currentSpeed = 0;
int currentRightSpeed = 0;
Servo baseServo;
Servo topServo;
int basePos = 90;
int topPos = 90;
int posStep = 5;
Servo myservo; // create servo object to control a servo
// a maximum of eight servo objects can be created

int pos = 90; // variable to store the servo position
```

4 Intel® Edison on COM3

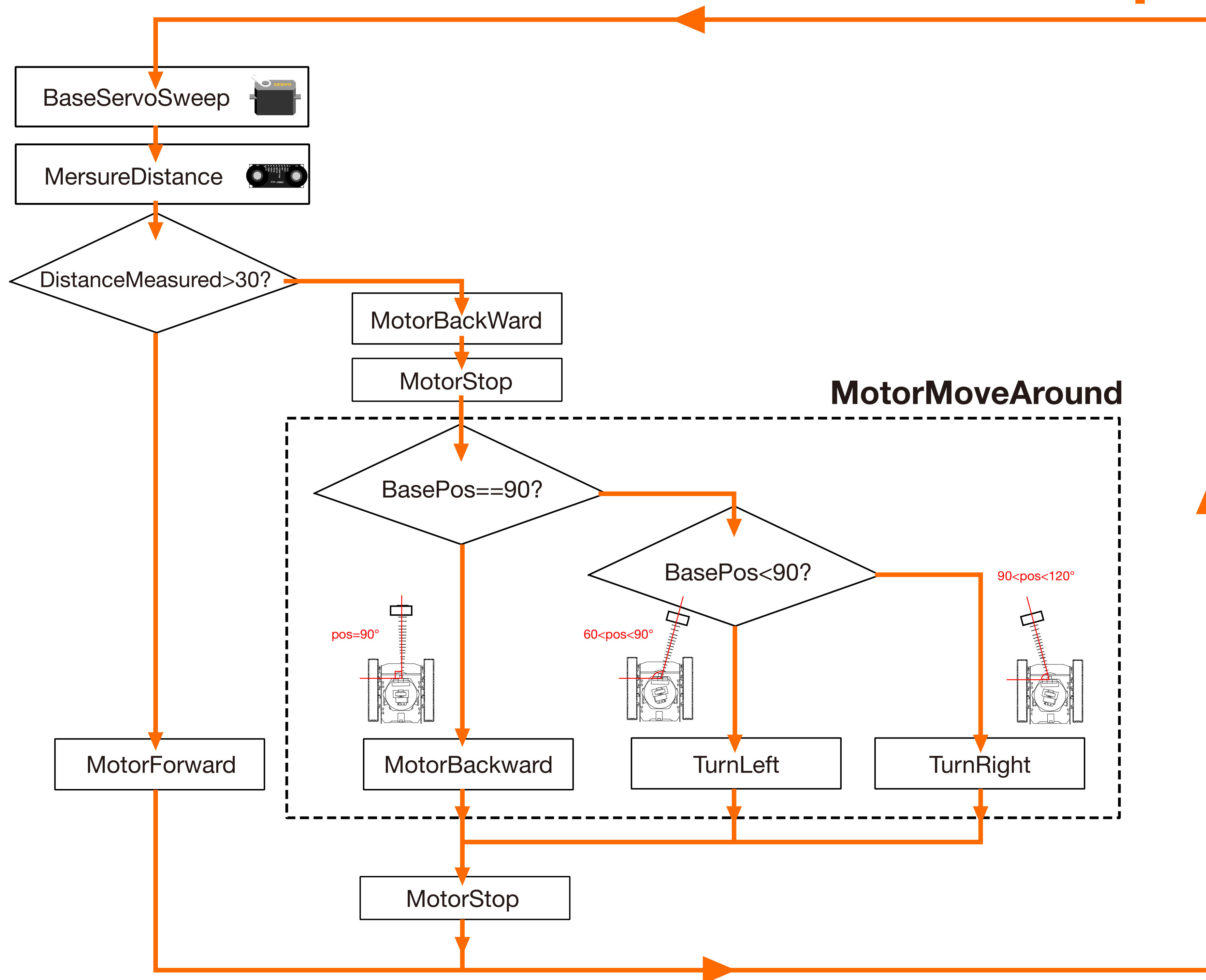
Obstacle detecting and avoiding mechanism

The obstacle detecting mechanism of Devastator is rather simple. As the orientation of ultrasonic sensor module rotates within a forward horizontal range of 60° , we can set the sensor send back distance measurement in various orientations. In our code, the orientation parameter (basePos) is set to 90 when the sensor faces straight to the front. Accordingly, when $\text{basePos} < 90$, sensor faces front right, vice versa.

Based on it, the Obstacle avoiding mechanism that we adapt can be interpreted as following:

Whenever the ultrasonic sensor module detects obstacle locates less than 30cm from the sensor, depends on the orientation of the mount, following action will be taken to avoid the obstacle: When value of basePos is exactly 90, which implies the obstacle is located right ahead, the tank moves straight backward. When $\text{basePos} < 90$, which implies the obstacle is located on the right side, so the tank turns to left, vice versa.

Loop



Flow chart of the Obstacle detecting and avoiding mechanism

Code Analysis

1.Servo control

sample code

```
int basePos = 90;
```

```
int posStep = 5;
```

```
Servo baseServo;
```

```
void baseServoSweep()      //baseServo sweep from 60 to 120
```

```
{
```

```
    if(basePos >= 120 || basePos <= 60)
```

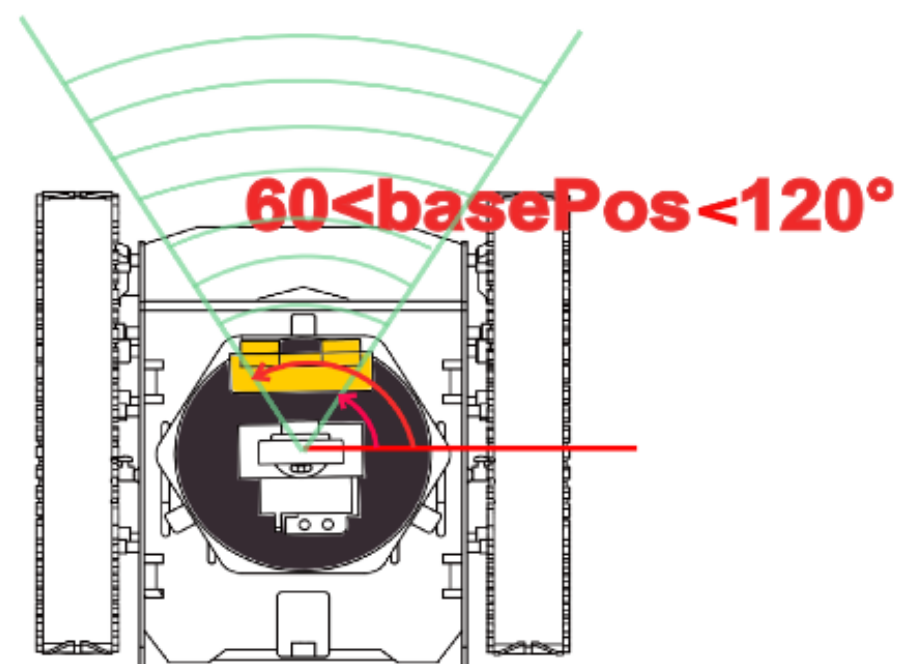
```
        posStep = -posStep;
```

```
    basePos += posStep;      //position increases or decreases by step
```

```
    baseServo.write(basePos);
```

```
}
```

The program above will set the mount to be rotating back and forth within an allowed range ($60 < \text{basePos} < 120$), the variable `posStep` is the angle by which the mount rotates each time. Once the position variable `basePos` reaches either 60 or 120, the sign of `posStep` switches so that the mount rotates in another direction. The allowed range can be adjusted by modifying maximum and minimum values of `basePos` in the statement below.



Pan and tile kit

```
if(basePos >= 120 || basePos <= 60)
```

```
posStep = -posStep;
```

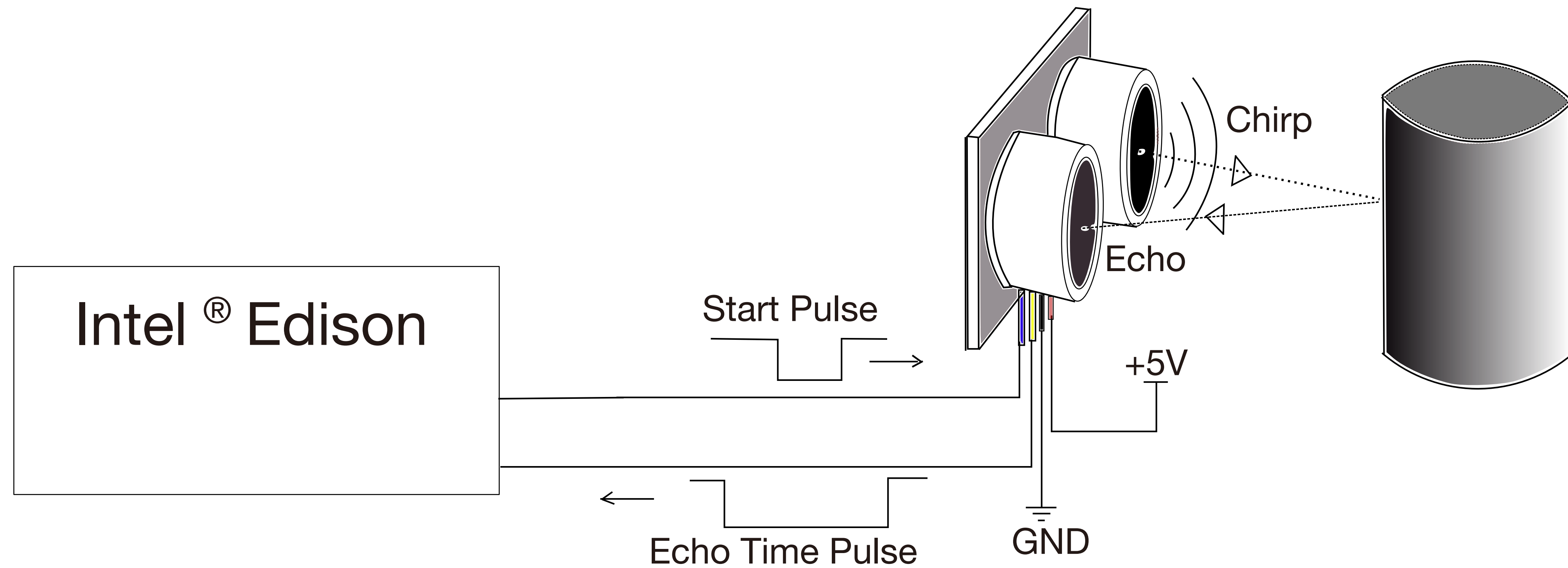

2.URM37 Ultrasonic Sensor

sample code

```
void MeasureDistance()
{
    unsigned long LowLevelTime = pulseIn(URECHO, LOW) ;
    Serial.print("Distance Measured=");
    digitalWrite(URTRIG, LOW);
    delay(10);
    digitalWrite(URTRIG, HIGH);           // reading Pin PWM will output pulses
    if (LowLevelTime >= 145000)           // the reading is invalid.
    {
        Serial.print("Invalid");
    }
    else
    {
        DistanceMeasured = LowLevelTime / 50; // every 50us low level stands for 1cm
        Serial.print(DistanceMeasured);
        Serial.println("cm");
    }
}
```

In our case, URM37 Ultrasonic Sensor module will be operating under PWM triggering control mode. When the COMP/TRIG port is triggered by a LOW signal from the microcontroller, URM37 module sends out an ultrasounds chirp to target. Once its echo is captured, the module sends a PWM wave as feedback to the microcontroller. The time interval can be interpreted as the time length of the duty cycle (the length of high voltage signal in each period) of the PWM wave. Here, based on the speed of sound, every 50us in time length corresponds to 1cm in distance. Moreover, if no echo is captured, it sends a PWM wave with 50000us High voltage time length in each period.

If the detection is invalid, 50000US is shown on Echo.



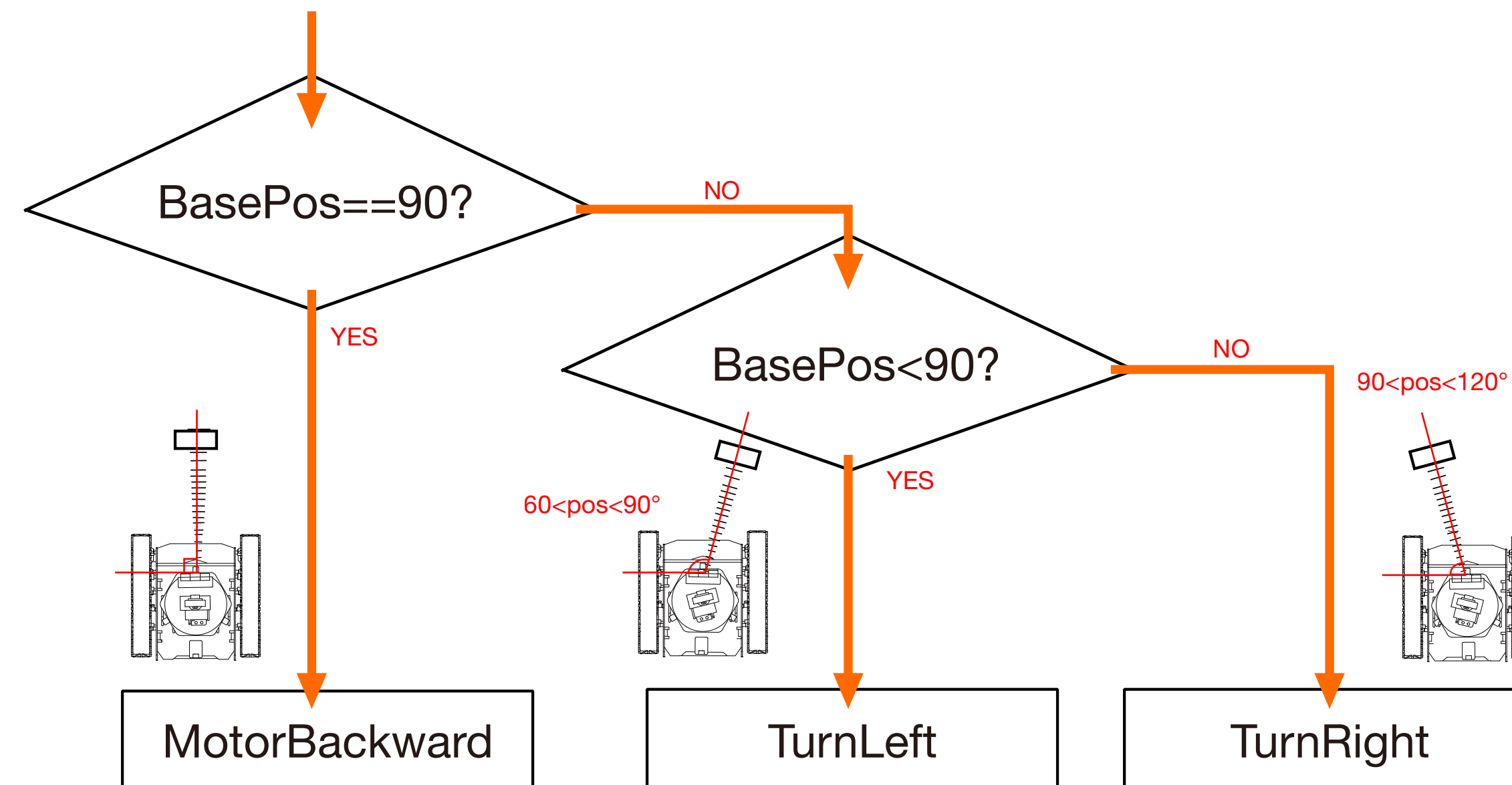
For more information about the URM37 module, please visit the wiki page on DFrobot website.

Please refer to the link below:

[https://www.dfrobot.com/wiki/index.php?title=URM37_V4.0_Ultrasonic_Sensor_\(SKU:SEN0001\)#Introduction](https://www.dfrobot.com/wiki/index.php?title=URM37_V4.0_Ultrasonic_Sensor_(SKU:SEN0001)#Introduction)

3. Obstacle avoiding function

In our case, we add the sub function motorMoveAround() to our program to control the motions of the motors based on the direction of the obstacle.



Sample code:

```
else if (basePos < 90)
{
    //turn left if sensor detected obstacles at right of the car
    directionMotorLeft = ANTICLOCKWISE ;
    directionMotorRight = CLOCKWISE;
    speedMotorLeft = 250 - (90-basePos)/30*50;
    speedMotorRight = 250 - (90-basePos)/30*50;
    delayTime = 300 - (90-basePos)/30*80;
}
```

Function:

By applying the obstacle avoiding mechanism that we mentioned above, the sample code steers the tank based on the difference between the value of the basePos and the preset value corresponds to the orientation towards forward (90 in our case). To be more specific, the larger difference is, the bigger its steering angle will be.