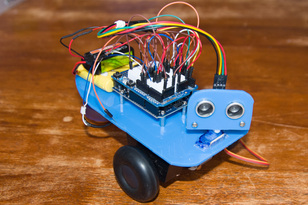
**James - Obstacle Avoiding Robot**

[](http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4988564_orig.jpg?308)

James is your stock standard obstacle avoiding robot.   
The goal of it is to drive around aimlessly not bumping into objects. He 'sees' these obstacles using his ultrasonic 'eyes', doing some basic math we can determine how far away these objects are. If there is an object too close in front of him he will maneuver around trying to find a better route.  
The brain of James consists of an Arduino Uno that is mounted onto the base. Using a prototyping shield we can easily wire up the circuitry using breadboarding jumper cables. The drivetrain can consist of either continuous rotation servos or large DC motors with gearboxes. A special chip is required to drive the DC motors.

**Step 1: Gather your Components and Tools**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/6331974_orig.jpg | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/6331974.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/9893106.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/6225160.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1858375.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/8539916.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/8181673.jpg | | | For this build we will be using the Arduino Uno with a Prototyping Shield. You could use whichever Arduino you want with its own Prototyping Shield or Breadboard.  Components:   * [Arduino Uno](https://core-electronics.com.au/store/index.php/arduino-uno-r3.html/?acc=eccbc87e4b5ce2fe28308fd9f2a7baf3) * [Prototyping Shield](https://core-electronics.com.au/store/index.php/protoshield-mini-breadboard-for-arduino.html/?acc=eccbc87e4b5ce2fe28308fd9f2a7baf3) * [HC-SR04](https://core-electronics.com.au/store/index.php/hc-sr04-ultrasonic-module-distance-measuring-sensor.html/?acc=eccbc87e4b5ce2fe28308fd9f2a7baf3) * Continuous Rotation Servos * Micro Servo * L293D * [Breadboard Jumper Cables](https://core-electronics.com.au/store/index.php/breadboard-jumper-cable-wires-65-pack.html/?acc=eccbc87e4b5ce2fe28308fd9f2a7baf3) * [Screws, Spacers and Nuts](https://core-electronics.com.au/store/index.php/15mm-20mm-40mm-m3-standoff-pack-116pcs.html/?acc=eccbc87e4b5ce2fe28308fd9f2a7baf3) |

**Step 2: Planning**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| On a piece of paper sketch out a rough sketch of the shape you want your robot body to have. It is always best to have an idea of what you want your robot to look like before you begin its construction.  Next you will want to sketch the shape of the robot body onto the protective sheet of the acrylic, this can be done easily with a pencil. If you do not have acrylic you could use; balsa wood, ply-board, foam-core, cardboard plastic containers or even food containers! You will also need to mark out where you want your servo to go. To cut out the hole for the servo drill four holes on each corner where you want to put the servo, this is to allow the blade of the saw to be easily maneuverered, preventing finger loss. | |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/8355721_orig.jpg | | |  |  | | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/8355721.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/3672886.jpg | | |

**Step 3: Arduino Mounting Holes**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1979345_orig.jpg | | |  |  | | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1979345.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4797125.jpg | | | The Arduino will be attached to the robot body using M3 screws and spacers so you will also mark where the mounting holes are on the Arduino so they can be drilled out. Once all of this has been done cut out the robot shape on a bandsaw, if you don’t have a bandsaw you can use a coping saw. |

**Step 4: Ultrasonic Mounting Bracket**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Next you need to make the bracket to hold the ultrasonic sensor. Mark and cut the aluminium square bracket to an appropriate size. Hot glue the servo horn onto the aluminium. Drill some holes in the bracket, these are what we are going to use to attach the acrylic front plate to the bracket. Trace around the sensor on a piece of acrylic, cutting out the general shape and drilling holes for the 'eyes' to poke through. Make sure that the width of the acrylic piece is the same as that as your bracket so when joining them the edges will be flush. Using screws and nuts attach the acrylic plate onto the square bracket. | |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1438498_orig.jpg | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1438498.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/5232633.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4100696.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/8928400.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4873685.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/54550.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/2578225.jpg | | |

**Step 5: Drivetrain**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/6585414_orig.jpg | | |  |  |  | | --- | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/6585414.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/252103.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/9641601.jpg | | | Now you will want to attach your drivetrain. In my case I am using 2 DC Motor Gearboxes that are from an old robot kit called ‘Cybot’, you could just as well use a Tamiya DC Motor Gearbox or some Continuous Rotation Servos. If you are using Continuous Rotation Servos you will not need the L293D Motor Driver IC. You can attach the drivetrain using either Hot glue or Double Sided Foam Tape, I have used the latter in this case. You will also need to attach the Micro Servo to the robot body, I did this using Hot glue. Now because the robot won't balance on two wheels we will need to add what is known as a castor or support wheel. To do this I took a lid off a roll on deodorant stick, brought it to the appropriate height using some scrap timber and squished a third of a table tennis ball. Inserting the table tennis ball creates a pretty nifty castor wheel. |

**Step 6: Checkpoint**

So far this is what the robot looks like.

|  |  |
| --- | --- |
| Picture |  |

**Step 7: Attaching the Arduino**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Using the M3 screws attach the Arduino to the robot’s body. And place the Prototyping shield onto the Arduino. Next insert the L293D Motor Driver IC into the mini breadboard with the small indent facing the front of the robot. | |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/3186413_orig.jpg | | |  |  |  | | --- | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/3186413.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1341458.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1596306.jpg | | |

**Step 8: Wiring the Motor Driver**

Connect the DC Motor leads up to the chip as shown, the left motor goes on the left side and vice versa.  
Connect the enables of the motors and the voltage in for chip operation up to the 5V. To save space on the rail on the Prototyping Shield I have connected the lines that need to be connected to 5V together so that only one cable needs to go the rail.  
Next connect all the ground together and then connect them up the Ground rail.  
Now connect the Motor Control pins up to the Arduino pins. Left motor pins are connected to pins 6 and 5. Right motor pins are connected to pins 8 and 9.  
I didn't have a power switch handy so I connected the power lines for the battery to go to the Arduino into the breadboard.  
Connect the ground of the battery into the ground of the Arduino. To turn my robot on and off I will have to connect and disconnect the V of the battery into the Vin pin on the Arduino.  
If you want to learn more about the L293D motor driver IC and how it should be wired up, be sure to check out the[Arduino Tutorial](http://www.duino-robotics.com/arduino-tutorials.html) on it.

|  |
| --- |
| http://www.duino-robotics.com/uploads/2/2/9/8/22988902/7891711_orig.jpg |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/7891711.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/45711.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/1004722.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/9410534.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/9026515.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/585802.jpg | |

**Step 9: Connecting the Ultrasonic Sensor**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4779627_orig.jpg | | |  |  | | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4779627.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/3792912.jpg | | | Next connect some female Breadboard cables up to the Ultrasonic sensor. Then using some male cables connect the Trigger pin of the sensor up to Arduino pin 3 and the Echo to pin 2. You will also need to connect the servo to the Arduino, do this by connecting the voltage in of the Servo up to the 5V rail, the ground to the ground rail and the signal wire to Arduino pin 4. |

**Step 10: Power**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| To power my DC Motors I am using a 4.8V Ni-Cd rechargeable battery pack. The V gets connected to the Motor V on the L293D chip and the ground up to the Arduino ground rail. Next connect a 9V battery up to the Arduino to power it. | |  | | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4541603_orig.jpg | | |  |  | | --- | --- | | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/4541603.jpg | http://www.duino-robotics.com/uploads/2/2/9/8/22988902/8195598.jpg | | |

**Step 11: Code**

Here is some basic code to get you started. I strongly encourage you to edit it and make it better once you have learned how it works.  
**To make the use of the ultrasonic sensor easier I have used the [new ping](http://playground.arduino.cc/Code/NewPing" \o ") library.**  
Download link for the sketch can be found at the bottom.

//Since we are using servos and ultrasonic sensors in the robot we will include some libraries written to make their use easier

#include <Servo.h>

#include <NewPing.h>

//Below are the symbolic constants. Instead of having to type in a non-sensical pin number each time we want to do something we can write an easy to understand name which represents the pin, the compiler will then replace the names with the numbers

#define LeftMotorForward 2

#define LeftMotorBackward 3

#define RightMotorForward 5

#define RightMotorBackward 4

#define USTrigger 8

#define USEcho 9

#define MaxDistance 100

#define LED 13

//Here we have created two 'objects', one for the servo and one for the ultrasonic sensor

Servo servo;

NewPing sonar(USTrigger, USEcho, MaxDistance);

//Below we are creating unsigned integer variables which we will use later on in the code. They are unsigned as they will only have postive values

unsigned int duration;

unsigned int distance;

unsigned int FrontDistance;

unsigned int LeftDistance;

unsigned int RightDistance;

unsigned int Time;

unsigned int CollisionCounter;

void **setup**() //This block happens once on startup

{

**Serial**.begin(9600); //I have included the serial initialize but commented it out, if you want to debug and print information to the serial monitor just uncomment

  //Here we are setting the pin modes. As we will sending out signals from the pins we set them as outputs

  pinMode(LeftMotorForward, OUTPUT);

  pinMode(LeftMotorBackward, OUTPUT);

  pinMode(RightMotorForward, OUTPUT);

  pinMode(RightMotorBackward, OUTPUT);

  pinMode(LED, OUTPUT);

  servo.attach(6); //The servo is attached to pin 4

}

void **loop**() //This block repeats itself while the Arduino is turned on

{

  servo.write(90); //Rotate the servo to face the front

  scan();                                             //Go to the scan function

  FrontDistance = distance;                           //Set the variable FrontDistance to the value of the distance returned from the scan function

**Serial**.println("Front distance = ");

**Serial**.print(distance);

  if(FrontDistance > 40 || FrontDistance == 0) //If there is nothing infront of the robot within 40cm or the distance value is 0 (which for the newping libary means no ping was returned) then...

  {

   moveForward();                                     //Go to the moveForward function

  }

  else //Else (if there is something infront of the robot within 40cm) then...

  {

    CollisionCounter = CollisionCounter + 1;

    moveStop();                                       //Go to the moveStop function

    navigate();

  }

}

void moveForward() //This function tells the robot to go forward

{

**Serial**.println("");

**Serial**.println("Moving forward");

  digitalWrite(LeftMotorBackward, LOW);

  digitalWrite(LeftMotorForward, HIGH);

  digitalWrite(RightMotorBackward, LOW);

  digitalWrite(RightMotorForward, HIGH);

}

void moveBackward() //This function tells the robot to move backward

{

**Serial**.println("");

**Serial**.println("Moving backward");

  digitalWrite(LeftMotorForward, LOW);

  digitalWrite(LeftMotorBackward, HIGH);

  digitalWrite(RightMotorForward, LOW);

  digitalWrite(RightMotorBackward, HIGH);

}

void moveLeft() //This function tells the robot to turn left

{

**Serial**.println("");

**Serial**.println("Moving left");

  digitalWrite(LeftMotorForward, LOW);

  digitalWrite(LeftMotorBackward, HIGH);

  digitalWrite(RightMotorBackward, LOW);

  digitalWrite(RightMotorForward, HIGH);

}

void moveRight() //This function tells the robot to turn right

{

**Serial**.println("");

**Serial**.println("Moving right");

  digitalWrite(LeftMotorBackward, LOW);

  digitalWrite(LeftMotorForward, HIGH);

  digitalWrite(RightMotorForward, LOW);

  digitalWrite(RightMotorBackward, HIGH);

}

void moveStop() //This function tells the robot to stop moving

{

**Serial**.println("");

**Serial**.println("Stopping");

  digitalWrite(LeftMotorBackward, LOW);

  digitalWrite(LeftMotorForward, LOW);

  digitalWrite(RightMotorForward, LOW);

  digitalWrite(RightMotorBackward, LOW);

}

void scan() //This function determines the distance things are away from the ultrasonic sensor

{

**Serial**.println("");

**Serial**.println("Scanning");

  Time = sonar.ping();

  distance = Time / US\_ROUNDTRIP\_CM;

  delay(500);

}

void navigate()

{

**Serial**.println("There's an obstacle!");

    servo.write(167); //Move the servo to the left (my little servos didn't like going to 180 so I played around with the value until it worked nicely)

    delay(1000); //Wait half a second for the servo to get there

    scan();                                           //Go to the scan function

    LeftDistance = distance;                          //Set the variable LeftDistance to the distance on the left

**Serial**.println("Left distance = ");

**Serial**.print(distance);

    servo.write(0); //Move the servo to the right

    delay(1000); //Wait half a second for the servo to get there

    scan();                                           //Go to the scan function

    RightDistance = distance;                         //Set the variable RightDistance to the distance on the right

**Serial**.println("Right distance = ");

**Serial**.print(distance);

    if(abs(RightDistance - LeftDistance) < 5)

    {

      moveBackward();                                  //Go to the moveBackward function

      delay(200); //Pause the program for 200 milliseconds to let the robot reverse

      moveRight();                                     //Go to the moveRight function

      delay(100); //Pause the program for 200 milliseconds to let the robot turn right

    }

    else if(RightDistance < LeftDistance) //If the distance on the right is less than that on the left then...

    {

     moveLeft();                                      //Go to the moveLeft function

     delay(100); //Pause the program for half a second to let the robot turn

    }

    else if(LeftDistance < RightDistance) //Else if the distance on the left is less than that on the right then...

    {

     moveRight();                                     //Go to the moveRight function

     delay(100); //Pause the program for half a second to let the robot turn

    }

}

**Downloads:**