Hull PixelBot Construction Details

Version 2.0 September 2016

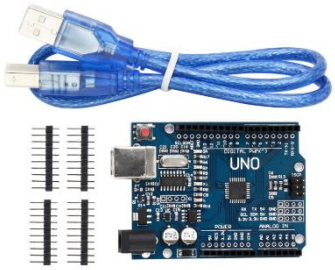
Rob Miles

Welcome to the instructions for the Hull PixelBot. Find out more at **hullpixelbot.com**.

The basic platform is how you get started. You build this first to get a feel for the construction and get it moving around. Then you can add the pixels and the distance sensors.

Parts

Arduino



The Arduino is the "brains" of your robot. A program running in the Arduino provides the drive signals for the motors to make them turn, and we will also use this to run the programs that will make the robot react to its environment.

There are mounting pillars on the top panel for a standard Arduino Uno device. Later we might replace this with different devices to add things like WiFi networking.

Search for "Arduino Uno" on eBay.

Stepper Motors

You'll need a couple of these motors. You can find them on eBay for very low prices, particularly if you buy a lot of them from china. Make sure you get ones that have the tiny driver board included (the greenish thing in the picture). You mount that onto the motor base and connect it to the motors and the Arduino.

Search for "Arduino stepper" on eBay.

Cables

Use 20cm male to female jumper cables. You'll need six per motor, making twelve in all.

Search for "Arduino jumper" on eBay

Bolts

I've made the fixing holes ever so slightly smaller than M3 bolts, so they should just tighten into the holes with no need for nuts. M3 bolts with a screw length of 10mm work fine for the motors and the top panel, M3 bolds with a screw length of 6mm work well for the stepper driver boards. It's worth buying a few of each as then you'll have some left for construction projects later

Search for "M3 bolt" on eBay

Battery Holder

You'll need 4xAA batteries to power the HullPixelBot. If you can find a holder with a 2.1mm power plug already soldered on you might save yourself a bit of work. I've pre-made some holes for the battery holders that I used, but yours might have holes in a different place. Alternatively, stick the battery box using glue or double sided tape. If you want a bit more voltage use a 5xAA holder. It will fit just fine if you rotate it so that the box overhangs the sides slightly.

Search for "AA battery holder" on eBay

Power Connector

You need a 2.1mm centre hole power plug to go into the Arduino.

Search for "power plug 2.1mm" on eBay

Skid Bearing

You only actually need two of these. They go into the skids at the front and the back of the robot for it to roll on.

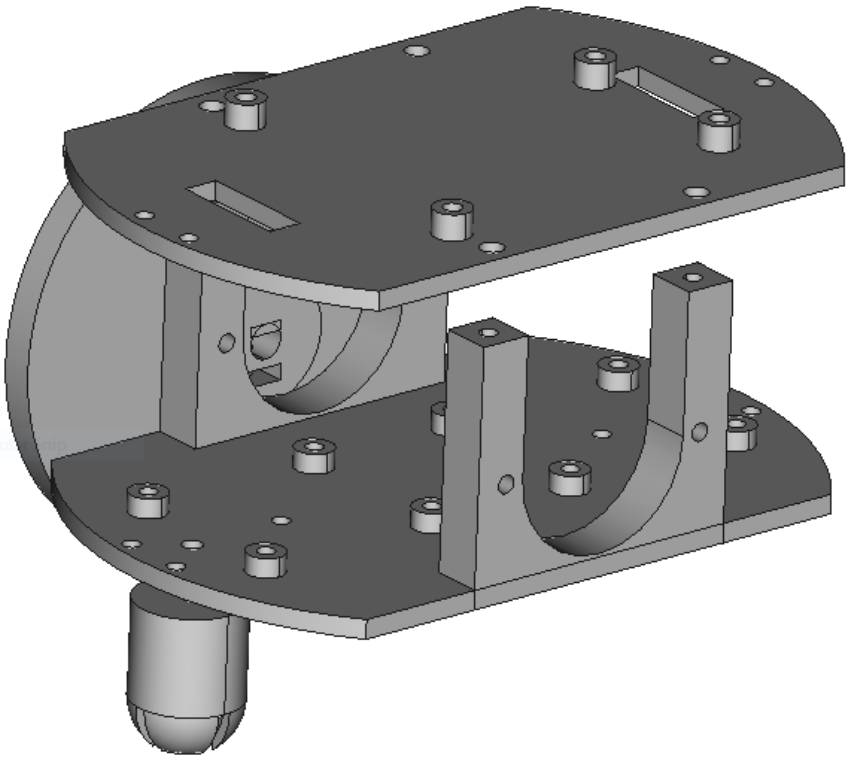
Search for "10mm ball bearings" on eBay. You might find some items labelled "Catapult ammunition". These work fine.

Tyres

I used elastic bands from a ball of bands that I got in Staples. Bands with a 6cm diameter seem to work best. They can be a bit hard to

Search through Staples for a ball of bands.

Assembly



It's fairly obvious where everything goes. One fiddly bit is the skids. The secret here is to put a bolt inside the skid, screwing upwards. This provides the bearing surface that the ball will roll against. It's lower friction than the PLA. Then you screw down through the hole in the robot into the back of the skid. There are some pictures at the end of these notes that should help. The top plate has slots you can use to route the cables from the stepper driver boards onto the Arduino.

Fit the skids before you fit the stepper motor driver boards.

Wiring

We use 8 pins on the Arduino to control the stepper motors, four for each motor. In addition, we need two power connections for each motor. The row of four pins on the motor driver board are the stepper control signals, these need to go in the same order into pins on the Arduino.

Motor 1: the four stepper signals go to digital pins 4,5,6,7 on the Arduino

Motor 2: the four stepper signals go to digital pins 8,9,10,11 on the Arduino

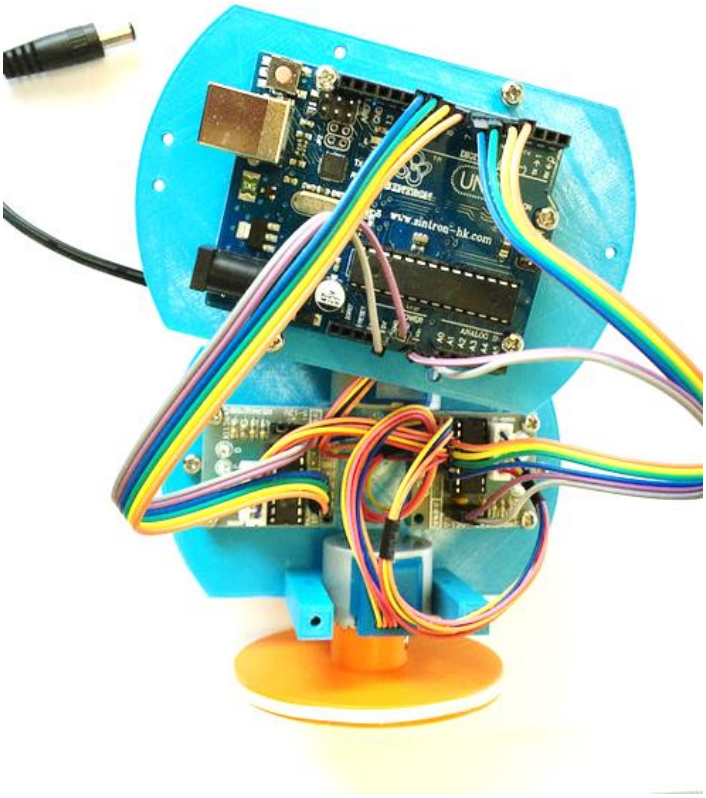
You also need to connect the two power pins to the Arduino to power the motors. Each motor needs a ground connection and a power connection. There are two ground connections on the Arduino, you can use the Vin connection for one motor power and the 5V connection for the other. If you are good at soldering you could connect the motor power connections directly to the 6 volt supply from the battery. That would free up some power connections on the Arduino for other devices. You could even add a power switch (at the moment you will turn your Hull Pixelbot off by unplugging the power connector from the Arduino).

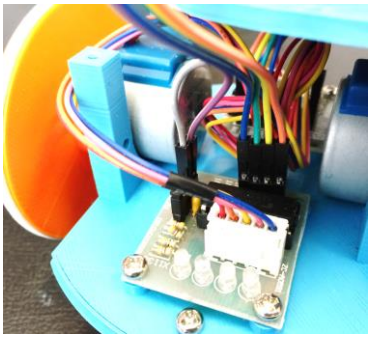
Hacking

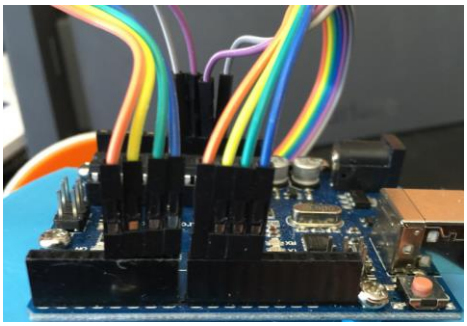
There are mounting holes 10mm apart on the front and back of the robot which you can use to mount sensors. Alternatively, you can stick a small mini-breadboard on the top to hold whatever sensors you fancy.

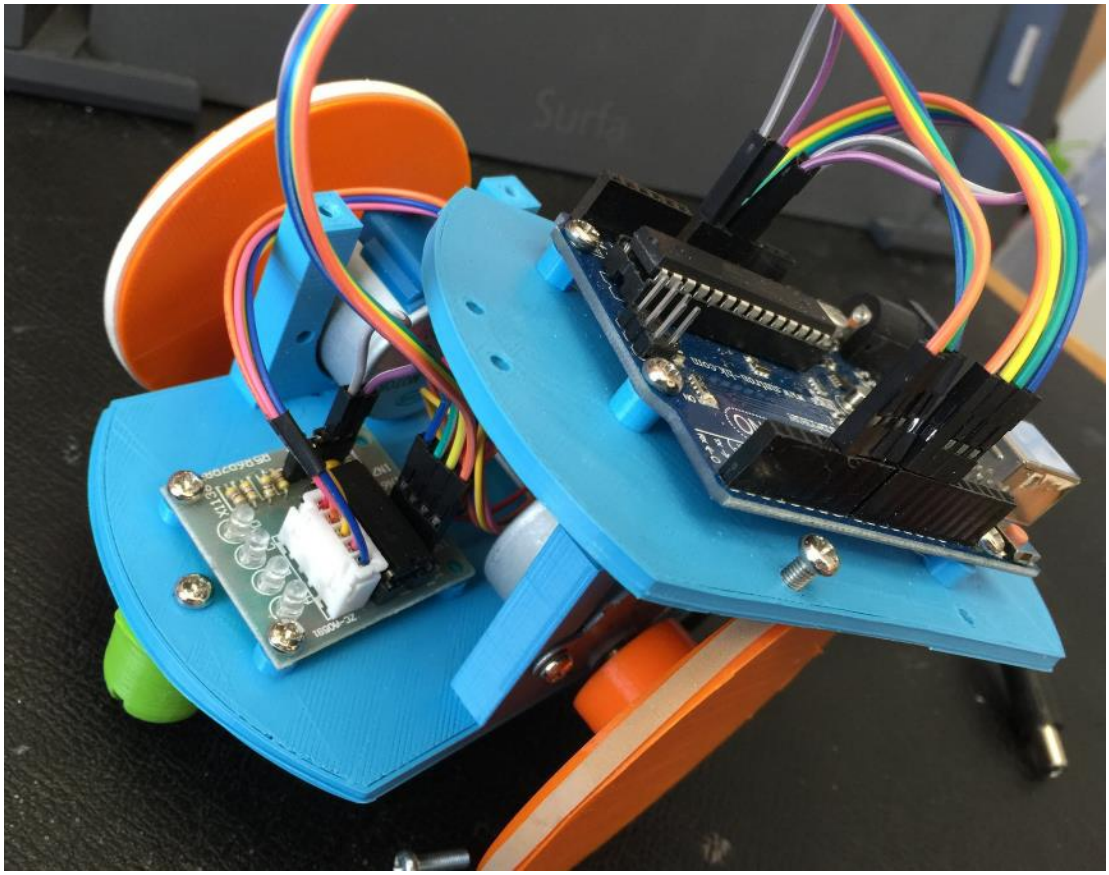
Search eBay for “mini breadboard”

Pictures









Software

This sample program will get your robot moving. Open a new Arduino project and paste the entire text into it. Alternatively you can download the script file at www.robmiles.com/hullpixelbot

```
// This example drives the stepper motors directly
// It makes the robot perform the "HullPixelBot Dance"
// See if you can change the moves.
// www.robmiles.com/hullpixelbot
////////////////////////////////////////////////

//declare variables for the motor pins
 int rmotorPin1 = 8;    // Blue   - 28BYJ48 pin 1
 int rmotorPin2 = 9;    // Pink   - 28BYJ48 pin 2
 int rmotorPin3 = 10;   // Yellow - 28BYJ48 pin 3
 int rmotorPin4 = 11;   // Orange - 28BYJ48 pin 4
                        // Red    - 28BYJ48 pin 5 (VCC)

 int lmotorPin1 = 4;    // Blue   - 28BYJ48 pin 1
 int lmotorPin2 = 5;    // Pink   - 28BYJ48 pin 2
 int lmotorPin3 = 6;   // Yellow - 28BYJ48 pin 3
 int lmotorPin4 = 7;   // Orange - 28BYJ48 pin 4
                        // Red    - 28BYJ48 pin 5 (VCC)


int motorSpeed = 1200;  //variable to set stepper speed
int count = 0;          // count of steps made
int countsperrev = 512; // number of steps per full revolution
int lookup[8] = {B01000, B01100, B00100, B00110, B00010, B00011, B00001, B01001};


////////////////////////////////////////////////////////////////////////////
 void setup() {
   //declare the motor pins as outputs
   pinMode(lmotorPin1, OUTPUT);
   pinMode(lmotorPin2, OUTPUT);
   pinMode(lmotorPin3, OUTPUT);
   pinMode(lmotorPin4, OUTPUT);
   pinMode(rmotorPin1, OUTPUT);
   pinMode(rmotorPin2, OUTPUT);
   pinMode(rmotorPin3, OUTPUT);
   pinMode(rmotorPin4, OUTPUT);
   Serial.begin(9600);
 }

const int STOP = 0;
const int FORWARD = 1;
const int BACK = 2;
const int LEFT = 3;
const int RIGHT = 4;

int moveState = FORWARD;

void loop(){
   if(count < countsperrev ) {
     moveStep();
   }
   else
   {
    moveState++;
    if(moveState > RIGHT)
      moveState = FORWARD;
    count=0;
   }
   count++;
}
```

```
void moveStep()
{
   for(int i = 0; i < 8; i++)
   {
    switch(moveState)
    {
      case STOP:
        return;
      case FORWARD:
        setOutputDir(i,7-i);
        break;
      case BACK:
        setOutputDir(7-i,i);
        break;
      case LEFT:
        setOutputDir(7-i,7-i);
        break;
      case RIGHT:
        setOutputDir(i,i);
        break;
    }
     delayMicroseconds(motorSpeed);
   }
}

void setOutputDir(int leftOut, int rightOut)
{

   digitalWrite(lmotorPin1, bitRead(lookup[leftOut], 0));
   digitalWrite(lmotorPin2, bitRead(lookup[leftOut], 1));
   digitalWrite(lmotorPin3, bitRead(lookup[leftOut], 2));
   digitalWrite(lmotorPin4, bitRead(lookup[leftOut], 3));

   digitalWrite(rmotorPin1, bitRead(lookup[rightOut], 0));
   digitalWrite(rmotorPin2, bitRead(lookup[rightOut], 1));
   digitalWrite(rmotorPin3, bitRead(lookup[rightOut], 2));
   digitalWrite(rmotorPin4, bitRead(lookup[rightOut], 3));
}
```