DF Robot – Turtle – 2 Motor Robot with HC-05 Bluetooth Module

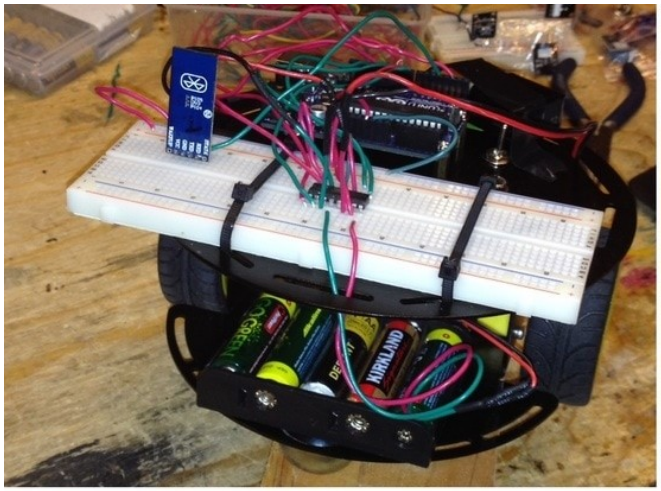
(G Payne – 2017)

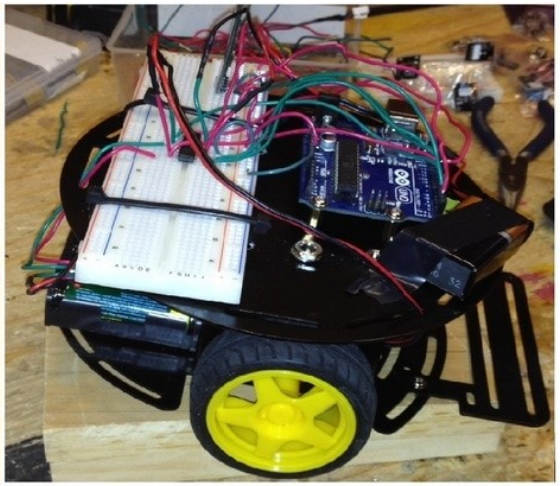
This document details the wiring pattern to connect the robot chassis to an Arduino UNO. Included is the L293D H-Bridge IC and the HC-05 Bluetooth module for controlling the robot via an Android RC Car app.

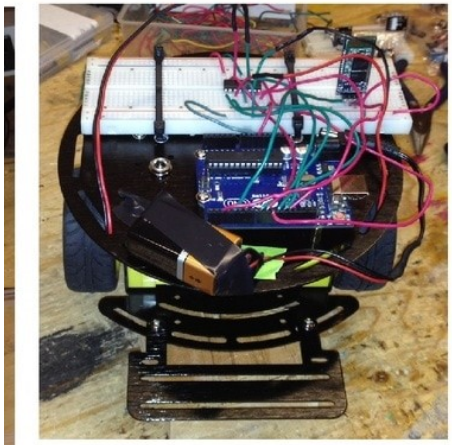
The Arduino code included here gives basic functionality for driving the robot. Forward, Reverse, Left and Right differential turning and Left and Right rotate-in-place.

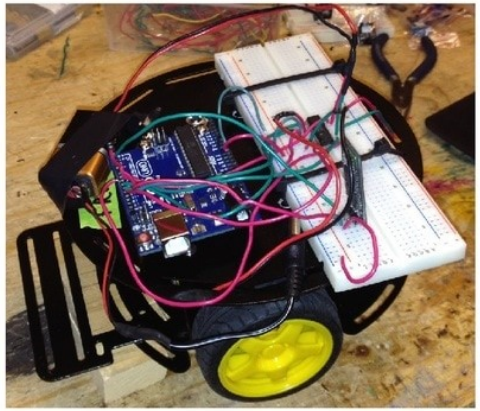
The orientation of the Arduino Uno reflects the position of the pre-drilled Arduino mounting holes on the DF Robot chassis. It is recommended that some insulating material be placed between the underside of the Arduino and the mounting posts on the robot chassis.

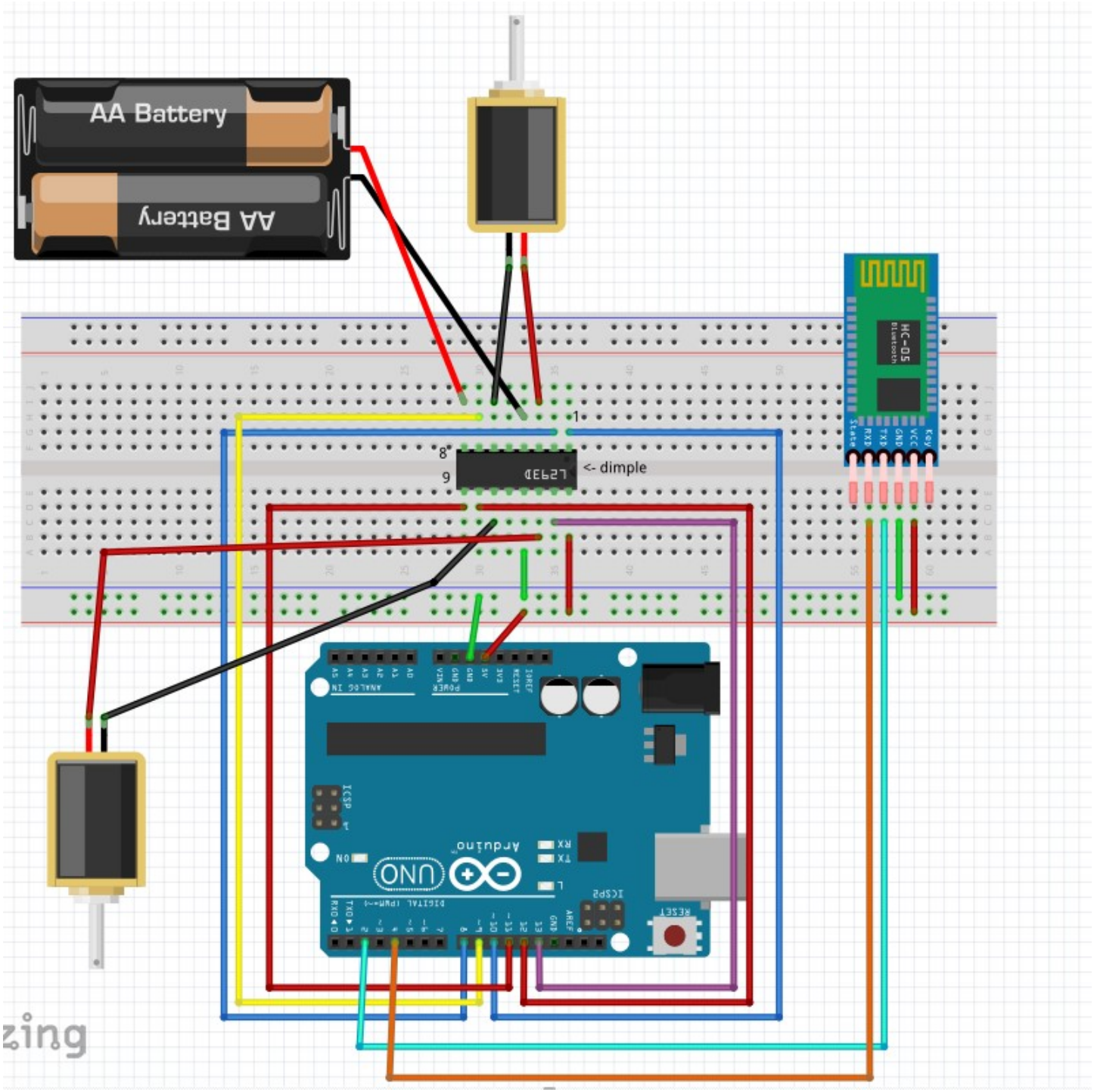
The L293D H-Bridge IC has its 'dimple' on the right hand side, meaning that Pin 1 is on the top-right and pin 16 is on the bottom right.
Have fun!











Wiring Diagram

L293D Pin	Arduino Digital Pin
1	10
2	8
7	9
9	11
10	12
15	13
3	Motor 1 red wire
6	Motor 1 black wire
11	Motor 2 black wire
14	Motor 2 red wire
16	+5V rail
8	+ side Robot battery
4	- side Robot battery
13	GND rail

HC-05 Bluetooth

VCC	+5V rail
GND	GND rail
TX	2
RX	4

For more details on the L293D IC and pin functions, please see "Activity 16 – Control Motors with an H-Bridge "

at

hausofpayne.weebly.com/arduino

Arduino Sketch for Controlling the Robot and Communicating via Bluetooth

This code could be optimized to give better response to Android app command inputs. You will probably want to adjust some of the 'delay()' statements.

```
// Control TurtleBot via Bluetooth HC-05 module
#include <SoftwareSerial.h>// import the serial library to allow Bluetooth communications

int E1 = 10; // Enable Pin for motor 1
int E2 = 11; // Enable Pin for motor 2

int I1 = 8; // Control pin 1 for motor 1
int I2 = 9;  // Control pin 2 for motor 1
int I3 = 12; // Control pin 1 for motor 2
int I4 = 13; // Control pin 2 for motor 2

SoftwareSerial Genotronex(2, 4); // TX, RX // create serial object connected to pins on the Arduino

char btData ; // the data given from the Android RC Car app
char turn = ' '; // variable to manage current directional function

void setup() {

  pinMode(E1, OUTPUT); // enable motor 1
  pinMode(E2, OUTPUT); // enable motor 2
  digitalWrite(E1, LOW); // make sure motors are OFF
  digitalWrite(E2, LOW);
  pinMode(I1, OUTPUT); // these 4 pins control the direction of each motor
  pinMode(I2, OUTPUT);
  pinMode(I3, OUTPUT);
  pinMode(I4, OUTPUT);
  Genotronex.begin(9600);// enable communication with the Bluetooth module

  Serial.begin(9600); // enable serial communication between Arduino and Serial Monitor for debugging as necessary

}
```

```
void loop() {

  if (Genotronex.available()) { // if there is data to read from the Android app
    btData = Genotronex.read(); // read the character
  }
  switch (btData) { // based on what character is sent to Arduino from Android app

    case '1': // forward
      turn = ' ';
      modFwd(); // call 'forward' method
      break;
    case '2': // reverse
      turn = ' ';
      modRev(); // call 'reverse' method
      break;
    case '5': // stop
      turn = ' ';
      digitalWrite(E1, LOW); // turn off both motor channels
      digitalWrite(E2, LOW);
      break;
    case '4': // stationary LEFT - Rotate in place
      turn = 'l'; //
      break;
    case '6': // stationary RIGHT - Rotate in place
      turn = 'r';
      break;
    case '7': // differential motor Left turn
      turn = 'L';
      break;
    case '9': // differential motor Right turn
      turn = 'R';
      break;
    default:
      break;
  }
  Serial.println(turn); // diagnostic only
          // call appropriate control method based on current directional function
  if (turn == 'L')gradLeft();
  if (turn == 'R')gradRight();
  if (turn == 'l')statLeft();
  if (turn == 'r')statRight();
```

```
  delay(100);
}


void modFwd() {
  digitalWrite(E1, LOW);
  digitalWrite(E2, LOW);
  digitalWrite(E1, HIGH);
  digitalWrite(E2, HIGH);
  digitalWrite(I1, HIGH);
  digitalWrite(I2, LOW);
  digitalWrite(I3, HIGH);
  digitalWrite(I4, LOW);
}

void modRev() {
  digitalWrite(E1, LOW);
  digitalWrite(E2, LOW);
  digitalWrite(E1, HIGH);
  digitalWrite(E2, HIGH);
  digitalWrite(I1, LOW);
  digitalWrite(I2, HIGH);
  digitalWrite(I3, LOW);
  digitalWrite(I4, HIGH);
}

void statRight() {
  if (turn == 'r') {
    digitalWrite(E1, LOW);
    digitalWrite(E2, LOW);
    digitalWrite(E1, HIGH);
    digitalWrite(E2, HIGH);
    digitalWrite(I1, LOW);
    digitalWrite(I2, HIGH);
    digitalWrite(I3, HIGH);
    digitalWrite(I4, LOW);
    delay(80); // delays are to slow down the rate of rotation
    digitalWrite(E1, LOW);
    digitalWrite(E2, LOW);
    delay(80);
  }
```

```
}

void statLeft() {
  if (turn == 'l') {
    digitalWrite(E1, LOW);
    digitalWrite(E2, LOW);
    digitalWrite(E1, HIGH);
    digitalWrite(E2, HIGH);
    digitalWrite(I1, HIGH);
    digitalWrite(I2, LOW);
    digitalWrite(I3, LOW);
    digitalWrite(I4, HIGH);
    delay(80);
    digitalWrite(E1, LOW);
    digitalWrite(E2, LOW);
    delay(80);
  }
}
void gradLeft() {
  if (turn == 'L') {
    digitalWrite(E1, HIGH);
    digitalWrite(E2, HIGH);
    digitalWrite(I1, HIGH);
    digitalWrite(I2, LOW);
    digitalWrite(I3, HIGH);
    digitalWrite(I4, LOW);
    delay(80);
    digitalWrite(E2, LOW);
    delay(200);
  }
}
void gradRight() {
  if (turn == 'R') {
    digitalWrite(E1, HIGH);
    digitalWrite(E2, HIGH);
    digitalWrite(I1, HIGH);
    digitalWrite(I2, LOW);
    digitalWrite(I3, HIGH);
    digitalWrite(I4, LOW);
    delay(80);
    digitalWrite(E1, LOW);
```

```
    delay(200);
  }
}
```

NOW GO MAKE SOMETHING WONDERFUL!!!