

LESSON 6

Programming Motion Control

Using the motor controller and motor gearbox we constructed earlier, your Pi-Bot is able to move around on its own. But first, we need to understand programming motor control.

Connect the USB cable from your STEM Board microcontroller to the computer. Make sure the motor controller power switch is turned **OFF**.

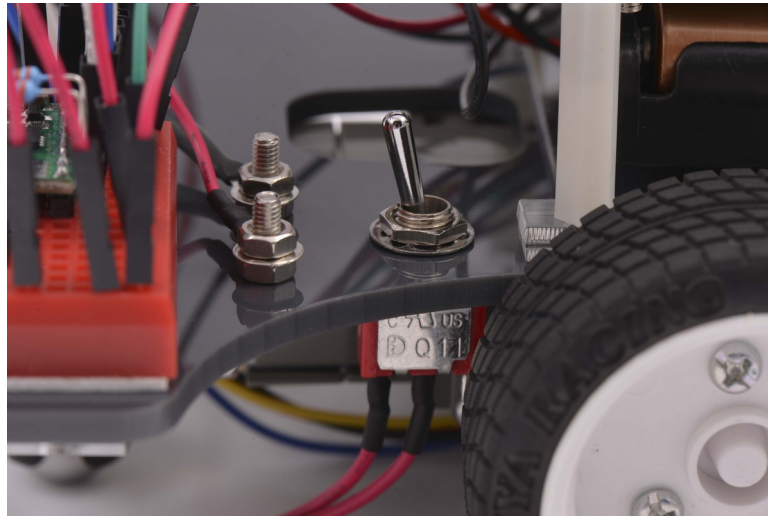


Figure 6.1

If you haven't done so already, connect the following wires:

- IN1 of the motor controller to pin 3
- IN2 of the motor controller to pin 5
- IN3 of the motor controller to pin 6
- IN4 of the motor controller to pin 11

Refer to [Lesson 2](#) for the wiring overview. Figure 6.2 depicts white wires connecting the motor controller to the STEM Board microprocessor.

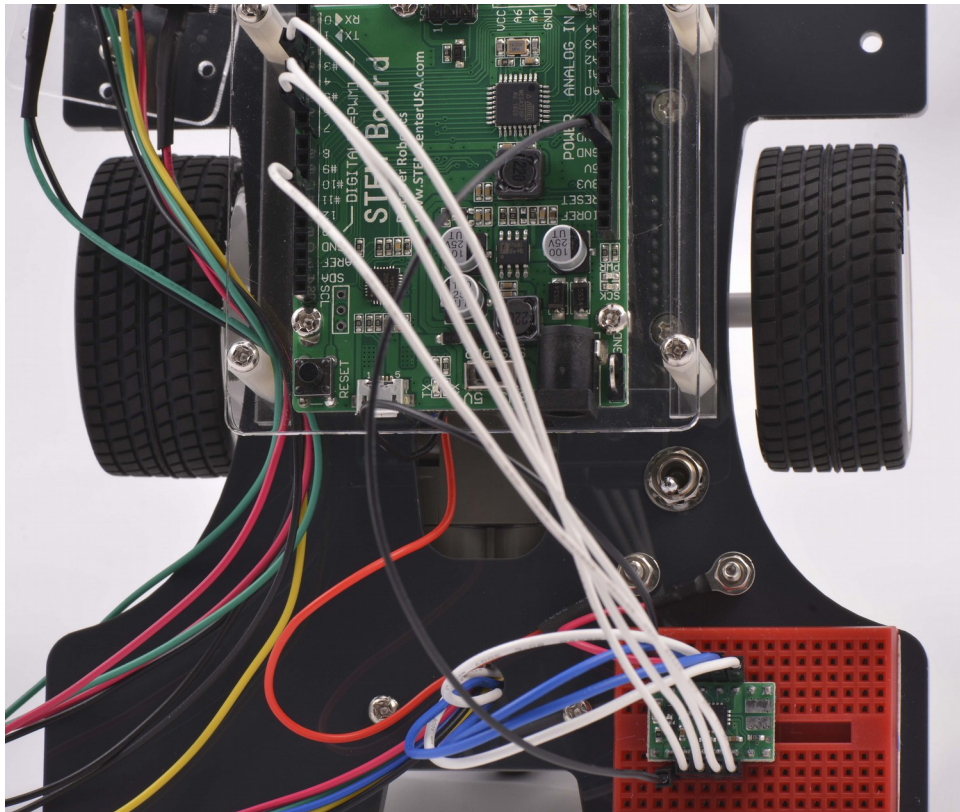


Figure 6.2

Using other pins for this example may not work as you must use an `analogWrite` function call if speed control is wanted.

Our first motor control program: [motion_control_01.ino](#) is shown in figure 6.3.

Test the program.

If everything is hooked up correctly, the wheels will turn forward for 3 second, stop, and turn in reversal for 1 second at about half-speed.

If the motors turn but in the wrong direct, you may need to switch the motor connection leads on the motor controller.

```

// motion_control_01.ino

const int In4 = 11;    // In4
const int In3 = 6;     // In3
const int In2 = 5;     // In2
const int In1 = 3;     // In1

void setup()
{
    // initialize the pins an output:
    pinMode(In1, OUTPUT);
    pinMode(In2, OUTPUT);
    pinMode(In3, OUTPUT);
    pinMode(In4, OUTPUT);
}

void loop()
{
    int speed1;
    int speed2;
    speed1 = 150;
    speed2 = 150;
    while(1==1)
    {
        analogWrite(In4, speed1);
        analogWrite(In3, 0);
        analogWrite(In2, 0);
        analogWrite(In1, speed2);
        delay(3000);
        analogWrite(In4, 0);
        analogWrite(In3, 0);
        analogWrite(In2, 0);
        analogWrite(In1, 0);
        delay(100);
        analogWrite(In4, 0);
        analogWrite(In3, speed1);
        analogWrite(In2, speed2);
        analogWrite(In1, 0);
        delay(1000);
        analogWrite(In4, 0);
        analogWrite(In3, 0);
        analogWrite(In2, 0);
        analogWrite(In1, 0);
        delay(100);
    }
}

```

Figure 6.3: Program – motion_control_01.ino

Did your Pi-Bot perform as expected?

Our next program: motion_control_02.ino, as shown in figure 6.4, is the similar to the previous program except now it uses function calls.

Keep the same STEM board to motor controller wiring setup.

Test out the new program using function calls. The program: motion_control_01.ino is not an efficient method for programming motor controls. The improved method in program: motion_control_02.ino uses function calls.

```

// motion_control_02.ino

const int In4 = 11;    // In4
const int In3 = 6;     // In3
const int In2 = 5;     // In2
const int In1 = 3;     // In1

void setup()
{
    // initialize the pins an output:
    pinMode(In1, OUTPUT);
    pinMode(In2, OUTPUT);
    pinMode(In3, OUTPUT);
    pinMode(In4, OUTPUT);
}

void loop()
{
    int speed1, speed2;
    int delayTime;
    speed1 = 150;
    speed2 = 150;
    while(1==1)
    {
        forward(speed1, speed2);
        delay(3000);
        stopNow();
        delay(100);
        reverse(speed1, speed2);
        delay(1000);
        stopNow();
        delay(100);
    }
}

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

```

Figure 6.4: Program – motion_control_02.ino

Exercise 1

Using the program: [motion_control_02.ino](#), make modifications to control the motors to move the wheels forward for 5 seconds, stop for 2 seconds, reverse for 3 seconds, and stop indefinitely.

Exercise 2

Add another function called `TURN`. Have it turn the wheels in opposite directions. This will cause your Pi-Bot to spin in circles.

You have successfully tested the motion control capabilities of your Pi-Bot!

In [Lesson 7](#), we will discuss collision detection.