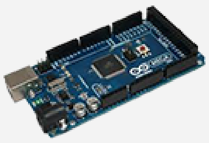
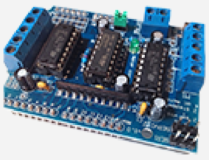
CONTROLLING A 4×4 ROVER ROBOT

Home ● DC Motors ● Controlling A 4×4 Rover Robot

OBJECTIVES

- ★ We keep on playing with DC motors and the Adafruit **Motor Shield** V1.
- ★ We will see the foundations of making a rover turn without using a steering wheel.
- ★ We will program the basic movements of the robot.

BILL OF MATERIALS

	**Arduino MEGA** or equivalent.
	A **Motor Shield V1.**
	A robot chassis with 4 motors.
	A **battery holder**.

CONTROLLING THE TURN

have not talked about a basic question that some people often find difficult to understand: the turn.

As automobiles are part of our daily life, we easily understand the principle they use to turn via the steering wheel, and we have an intuitive idea of how they deflect the tracing direction of the wheels with respect to the direction of movement of the car. This way, the front wheels always trace the tangent to the curve of movement.

- But perhaps it is not so obvious that the two front wheels of modern cars can not turn at the same speed in the curves, if we want to maintain the stability and governance of the vehicle.

It is quite easy, and if you think about it, it is the same principle used by tractors in the field, that can make unthinkable turns for a car, or tanks and excavators that are moved by metal tracks (without steering wheel).

As cars have a single combustion engine, we have to assemble a complicated mechanical system, which distributes traction between the wheels, but in electric cars in general, and in our 4×4 Rover in particular, we have one motor per wheel, so we can control the speed of each one independently.

- This is true even in commercial hybrid vehicles, which use a combustion engine to recharge the central batteries, but they have an electric motor per wheel which allows us, apart from 4×4 traction, to do some unusual things in a traditional car.

By controlling the speed of each wheel we can avoid the need for a mechanical steering and we can make movements that would surprise a car driver quite a lot, including turning without touching the steering wheel.

The trick is precisely to spin each wheel at a different speed, so as to force the vehicle to rotate in the desired direction and with the desired radius.

If the wheels on the right side turn twice as fast as those on the left side, our Rover must turn to the left and vice versa, making a continuous turn. The smoother the turn, the less speed difference between the turns of the two sides.

The turning radius will be more or less closed, depending on the relationship between the speed of both sides. Not only we can turn the Rover but also control how close is the turn. If we stop one side completely and hold the other, the turn will be very sharp.

With four independently driven wheels we can even turn the wheels on one side in the opposite direction to the other side, and in this case the Rover will turn around its center of gravity without moving.

To govern our four-wheel drive Rover it will suffice to control the speed of rotation of the wheels, and we already know how to do this from previous chapters. So let's put our hands to work with the sketch.

THE ROVER'S CONTROL PROGRAM

Let's start with some declarations, as always:

Prog_94_1

```
#include <AFMotor.h>

AF_DCMotor Motor1(1);
AF_DCMotor Motor2(2);
AF_DCMotor Motor3(3);
AF_DCMotor Motor4(4);

int Speed = 255 ;      // Defines the base speed of the Rover
float P = 0.25 ;       // Proportion of the turn giro
```

We include the **Adafruit Motor library** and instantiate four motor objects corresponding to the 4 motors. Then we create the **Speed** variable, that will contain the speed at which we want to move the Rover. This way, by varying the value of Speed, we will vary the overall speed of the vehicle. Finally we define the **P** variable, that sets the speed ratio between one side and the other when turning.

As we are going to make the rover turn by modifying the speed ratio of the motors, we should start by defining a function to program the speed of each wheel independently:

```
   }
```

Bear in mind that these commands only set the speed of each motors, they don’t make them spin yet. In order for the motors to turn we will have to define some functions corresponding to basic movements, such as Forward, Reverse and Stop:

```
void Forward()
    {  SetVel(Speed,Speed,Speed,Speed);       // Same speed to the 4 wheels
       Motor1.run(FORWARD) ;
       Motor2.run(FORWARD);
       Motor3.run(FORWARD);
       Motor4.run(FORWARD);
   }

void Reverse()
   {  SetVel(Speed,Speed,Speed,Speed);        // Same speed to the 4 wheels
      Motor1.run(BACKWARD) ;
      Motor2.run(BACKWARD);
      Motor3.run(BACKWARD);
      Motor4.run(BACKWARD);
   }

void Stop()
   {  Motor1.run(RELEASE);
      Motor2.run(RELEASE);
      Motor3.run(RELEASE);
      Motor4.run(RELEASE);
   }
```

We have already discussed that in order to make the rover turn, it will suffice to move the wheels of each side of the Rover at different speed. We can test it by making one of the sides turn, for example, at 1/3 of the speed on the other side, using the variable P defined at the beginning.

- In a real Rover, we can modify the turning radius by reading a potentiometer, like a lever, so that as we move the lever to the right, for example, it reduces proportionally the speed of the wheels on the right.
- I leave it as an exercise as we do not think you will find it difficult. Later on we will talk about it when we see how to use a remote control to move the Rover.

```
void TurnLeft()
   {
     int s = Speed * P ;
     SetSpeed( s, Speed, s, Speed) ;

     Motor1.run(FORWARD) ;
     Motor2.run(FORWARD);
     Motor3.run(FORWARD);
     Motor4.run(FORWARD);
   }

void TurnRight()
   {
      int s = Speed * P ;
      SetSpeed( Speed, s, Speed, s) ;

      Motor1.run(FORWARD) ;
      Motor2.run(FORWARD);
      Motor3.run(FORWARD);
      Motor4.run(FORWARD);
   }
```

And we just need to do a test in real motion. Let’s program the movement inside the sketch, (because we still do not have a remote control).

```
void loop()
   {
       Forward();
       delay(2000);
```

```
        delay(3000);
    }
```

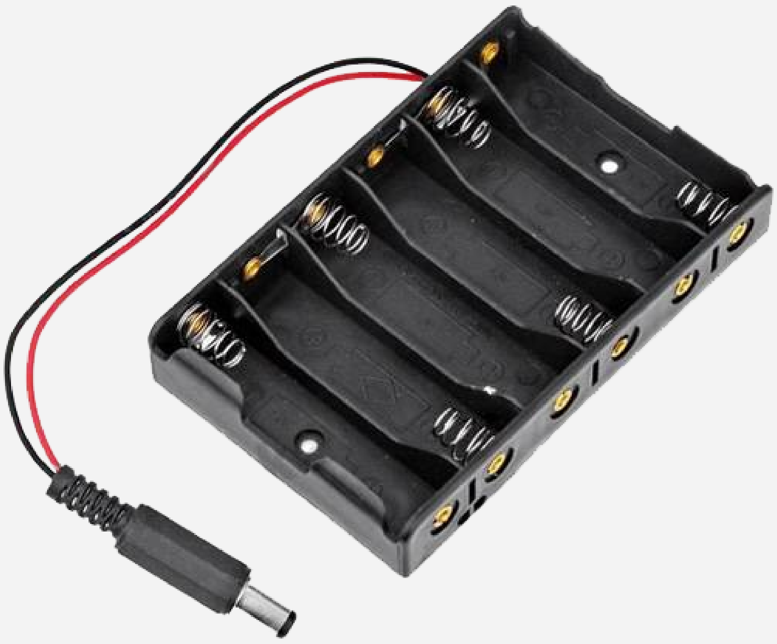
Easy, isn't it? It moves fordward, backwards, turns left and then right. Upload the sketch to the board and test how it moves.

TESTING THE MOVEMENT OF THE 4×4 ROVER

To move the Rover the first intention is, usually, to use the USB cable and leave it plugged to the Rover when moving, but although it seems to be a good idea, we regret to say it will not work.

- The USB connection can drive the motors without load, but you can barely move the Rover on a carpet, as the motors have the bad habit of consuming a lot of intensity and a typical USB connection can not supply more than half an Ampere, maximum.
- You will have to use batteries. You'd better use 6 batteries instead of 4, because we need at least 7V to supply 5V guaranteed (and 4 batteries x 1,5V = 6V, which is insufficient)
- In addition and to make things worse, cheap motors are usually very inefficient in terms of energy and these, believe me, are very cheap.

To test the movement we will need an battery holder and some bateries. Something like this:



The connector on the image above, fits perfectly in your Arduino (that connector that nobody uses) and avoid us to carry a hanging cable, which always ends up getting entangled due to the movement.

I can help showing you what happens if we turn the wheels on each side in opposite directions. Let's see the sketch:

Prog_94_3:

```
void TurnLeft()
   {
       SetSpeed( Speed, Speed, Speed, Speed) ;

       Motor1.run(BACKWARD) ;
       Motor2.run(FORWARD);
       Motor3.run(BACKWARD);
       Motor4.run(FORWARD);
   }

void TurnRight()
   {
       SetVel( Speed, Speed, Speed, Speed) ;

       Motor1.run(FORWARD) ;
```

This kind of turn is a bit counter intuitive, because cars can not do it, but it works very well and causes the rover to turn around its own geometric center.

And there is only one thing left, to define a way to control the Rover 4×4 by means of a remote control system. In the next chapter we will see which options we have and which to choose.

SUMMARY

- We keep on using the Adafruit MotorShield v1, and there is not much left to do with it to control DC motors.
- We have seen the foundations of how to make a four-wheel drive rover turn by applying different speeds to the wheels of each side.
- We have seen some basic sketches to control the turn of the robot, including turning around its center.

PREVIOUS

NEXT

NO COMMENTS

GIVE A REPLY

Message

Name (required)

Email (required)

URL

Post comment