CamJam EduKit Robotics - Driving and Turning

Project Driving and Turning

Description You will learn how to make your robot move in the direction you want it to.

Equipment Required

For this worksheet, you will require:

- Your robot

Going Forwards and Backwards

In the last worksheet, you saw the basics required to turn the wheels on your robot.

In this section of the worksheet, you are going to make the Robot go forwards and backwards and make the beginnings of a robot control program.

If you recall from the last worksheet, the EduKit Motor Controller Board uses two GPIO pins to make each motor go forwards or backwards. You have to turn the correct ones on and off to make the motors, and therefore your robot, move in the direction you want it to.

To make a wheel go forward or backwards, you have to turn one pin on and the other pin off. If you turn them both off, the motor will not move. Remember the following:

- For the right-hand motor, the controller board uses pin 10 to turn the motor in the forwards direction, and pin 9 to turn it backwards.
- For the left-hand motor, pin 8 is used to turn the motor forwards, and pin 7 for backwards.

Therefore, to make your robot move forwards, you have to turn pins 10 and 8 on, and 9 and 7 off. To go backwards, you have to turn pins 9 and 7 on, and pins 10 and 8 off.

Code

The code below uses variables to hold the pin numbers; this makes it much easier to remember which pins do what. These worksheets will use pinMotorAForwards, pinMotorABackwards, pinMotorBForwards, and pinMotorBBackwards.

It also uses Python 'functions' to group commands together to achieve a single task. Using functions, you are able to say only one simple thing in the code to do many things on the Robot. The three functions in this section are:

- stopmotors() – turns all motors off
- forwards() – makes both motors turn forwards
- backwards() – makes both motors turn backwards

Note: In Python, the spaces at the start of lines are important. They are Python's way of recognising code that is grouped together into functions or within other command structures, like `if`, `for` and `while`.

Create a new Python script and type in the following code.

```
# CamJam EduKit 3 - Robotics
# Worksheet 4 - Driving and Turning

import RPi.GPIO as GPIO  # Import the GPIO Library
import time  # Import the Time library
```

```
# Set the GPIO modes
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

# Set variables for the GPIO motor pins
pinMotorAForwards = 10
pinMotorABackwards = 9
pinMotorBForwards = 8
pinMotorBBackwards = 7

# Set the GPIO Pin mode
GPIO.setup(pinMotorAForwards, GPIO.OUT)
GPIO.setup(pinMotorABackwards, GPIO.OUT)
GPIO.setup(pinMotorBForwards, GPIO.OUT)
GPIO.setup(pinMotorBBackwards, GPIO.OUT)


# Turn all motors off
def stopmotors():
    GPIO.output(pinMotorAForwards, 0)
    GPIO.output(pinMotorABackwards, 0)
    GPIO.output(pinMotorBForwards, 0)
    GPIO.output(pinMotorBBackwards, 0)


# Turn both motors forwards
def forwards():
    GPIO.output(pinMotorAForwards, 1)
    GPIO.output(pinMotorABackwards, 0)
    GPIO.output(pinMotorBForwards, 1)
    GPIO.output(pinMotorBBackwards, 0)


# Turn both motors backwards
def backwards():
    GPIO.output(pinMotorAForwards, 0)
    GPIO.output(pinMotorABackwards, 1)
    GPIO.output(pinMotorBForwards, 0)
    GPIO.output(pinMotorBBackwards, 1)


GPIO.cleanup()
```

Save the file as `4-driving.py` in the EduKitRobotics directory.

Running the Code

When you run the code, it will not actually do anything at the moment, but if it displays any errors, you can re-edit the code by using your editor.

Moving the Robot

To make the robot actually move, you need to edit the code again, and insert the following lines just before the last line (`GPIO.cleanup()`):

```
forwards()
time.sleep(1)
```

```
backwards()
time.sleep(1)
stopmotors()
```

Now run the code again. This time your robot will move forward for one second, and backwards for one second, then stop.

Left and Right

The next step is to add another two functions that will turn the robot left or right. Edit the code again and type in the following code after the current three functions but before the code you added in 'Moving the Robot'.

To turn left, the right-hand motor is turned forward, and the left hand one turns backwards. To turn right, the left-hand motor is turned forward, and the right hand one turns backwards.

```
# Turn left
def left():
    GPIO.output(pinMotorAForwards, 0)
    GPIO.output(pinMotorABackwards, 1)
    GPIO.output(pinMotorBForwards, 1)
    GPIO.output(pinMotorBBackwards, 0)

# Turn Right
def right():
    GPIO.output(pinMotorAForwards, 1)
    GPIO.output(pinMotorABackwards, 0)
    GPIO.output(pinMotorBForwards, 0)
    GPIO.output(pinMotorBBackwards, 1)
```

Then, replace the code you added in 'Moving the Robot' with:

```
forwards()
time.sleep(1) # Pause for 1 second

left()
time.sleep(0.5) # Pause for half a second

forwards()
time.sleep(1)

right()
time.sleep(0.5)

backwards()
time.sleep(0.5)

stopmotors()
```

The full listing should look similar to this:

```
# CamJam EduKit 3 - Robotics
# Worksheet 4 - Driving and Turning

import RPi.GPIO as GPIO  # Import the GPIO Library
import time  # Import the Time library

# Set the GPIO modes
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
```

```
# Set variables for the GPIO motor pins
pinMotorAForwards = 10
pinMotorABackwards = 9
pinMotorBForwards = 8
pinMotorBBackwards = 7

# Set the GPIO Pin mode
GPIO.setup(pinMotorAForwards, GPIO.OUT)
GPIO.setup(pinMotorABackwards, GPIO.OUT)
GPIO.setup(pinMotorBForwards, GPIO.OUT)
GPIO.setup(pinMotorBBackwards, GPIO.OUT)


# Turn all motors off
def stopmotors():
    GPIO.output(pinMotorAForwards, 0)
    GPIO.output(pinMotorABackwards, 0)
    GPIO.output(pinMotorBForwards, 0)
    GPIO.output(pinMotorBBackwards, 0)


# Turn both motors forwards
def forwards():
    GPIO.output(pinMotorAForwards, 1)
    GPIO.output(pinMotorABackwards, 0)
    GPIO.output(pinMotorBForwards, 1)
    GPIO.output(pinMotorBBackwards, 0)


# Turn both motors backwards
def backwards():
    GPIO.output(pinMotorAForwards, 0)
    GPIO.output(pinMotorABackwards, 1)
    GPIO.output(pinMotorBForwards, 0)
    GPIO.output(pinMotorBBackwards, 1)


# Turn left
def left():
    GPIO.output(pinMotorAForwards, 0)
    GPIO.output(pinMotorABackwards, 1)
    GPIO.output(pinMotorBForwards, 1)
    GPIO.output(pinMotorBBackwards, 0)


# Turn Right
def right():
    GPIO.output(pinMotorAForwards, 1)
    GPIO.output(pinMotorABackwards, 0)
    GPIO.output(pinMotorBForwards, 0)
    GPIO.output(pinMotorBBackwards, 1)


forwards()
time.sleep(1)  # Pause for 1 second

left()
```

```
time.sleep(0.5)  # Pause for half a second

forwards()
time.sleep(1)

right()
time.sleep(0.5)

backwards()
time.sleep(0.5)

stopmotors()

GPIO.cleanup()
```

Once again, run the code.

Summary

You now have a robot that will move around to your command. Experiment by writing your own code using `forwards()`, `backwards()`, `left()`, `right()` and `stopmotors()` to move the robot around, and changing how long they run for by changing the numbers of seconds the code pauses for using `time.sleep(X)`, where X is the number of seconds to pause. You can use decimal numbers, like 0.5 for half a second.

Challenge

There are different ways of making your robot turn. The code above drives one motor forwards and the other backwards to turn the robot. As an alternative, you can turn just one motor forwards or one motor backwards. Expand the code to add functions to do the following:

- Turn left by turning the right motor forwards.
- Turn left by turning the left motor backwards.
- Turn right by turning the left motor forwards.
- Turn right by turning the right motor backwards.

Experiment with moving your robot around using these new functions as well as the original ones. You may prefer to use these new functions to turn your robot, depending on the surface you are driving on.