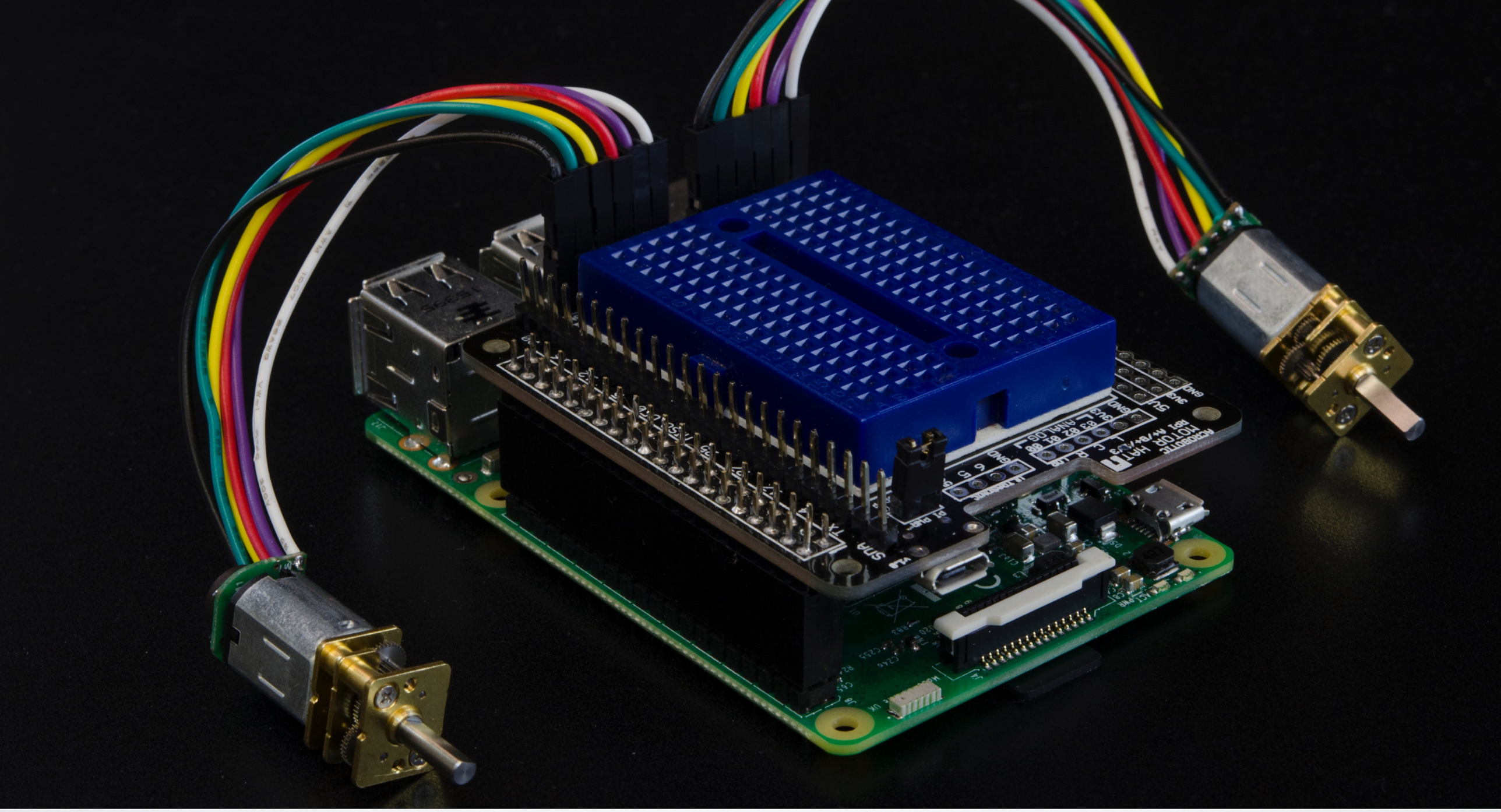


Intro to Robotics with Raspberry Pi!

Section 2. Physical Computing – Motion Control

Outline

Generating and Controlling Motion with Raspberry Pi

Motor Operation

Different Types of Motors

How DC Motors Work

Controlling DC Motors

Powering Motors

Controlling Motor Speed with PWM

Measuring Motor Speed

Using Wheel Encoders

Measuring Rotations

Calculating Speed (Rotation and Translation)

Closed-Loop Speed Control

Maintaining a Constant Speed

Intro to Navigation

Controlling Direction

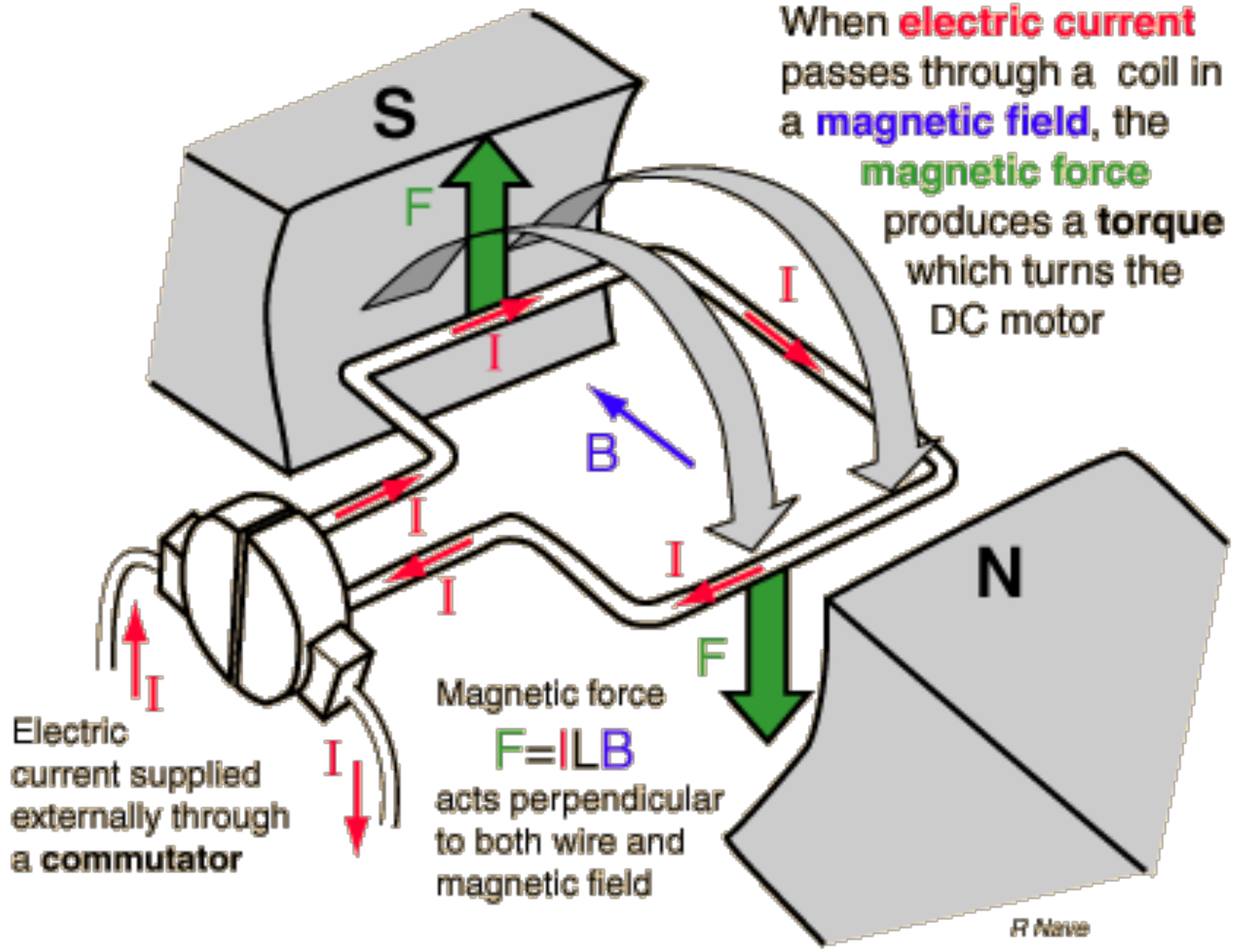
Differential Steering and Turning

Motor Operation

What is a Motor?

- Machine powered by electricity or internal combustion that supplies motive power

- A motor converts **electrical** power → **mechanical** power.



Different Types of Motors

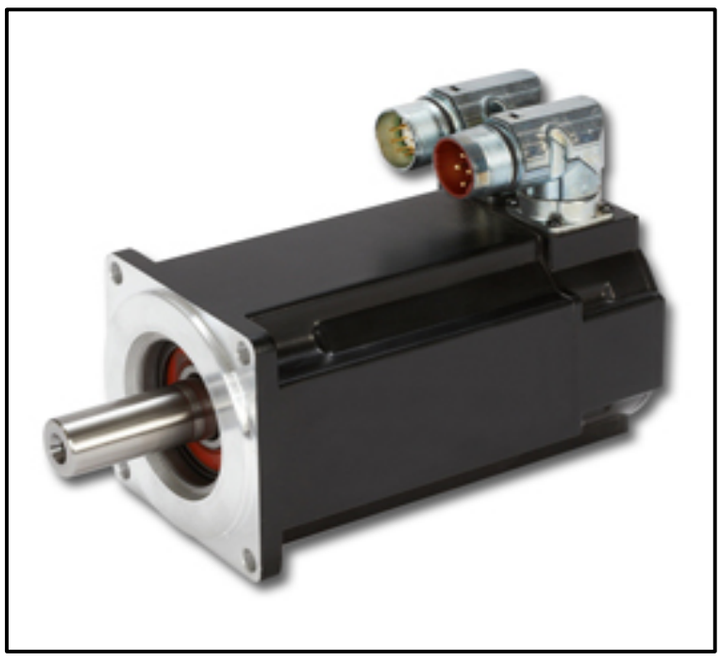
- **DC Motors**

Electric motor that runs on direct current electricity.

Two main types: Brush DC & Brushless DC motor

Brush DC motor uses and internal powers supply with stationary magnets.

Brushless DC motors use rotation permanent magnet.



Different Types of Motors

- **Stepper Motors**

Brushless DC motor that divides full rotation into multiple steps.

The motor's position can move and hold at one of these steps without any feedback sensor.

Used in Floppy disk drives, plotters, CD ROMs, scanners and many other appliances.

Different Types of Motors

- **Servo Motors**

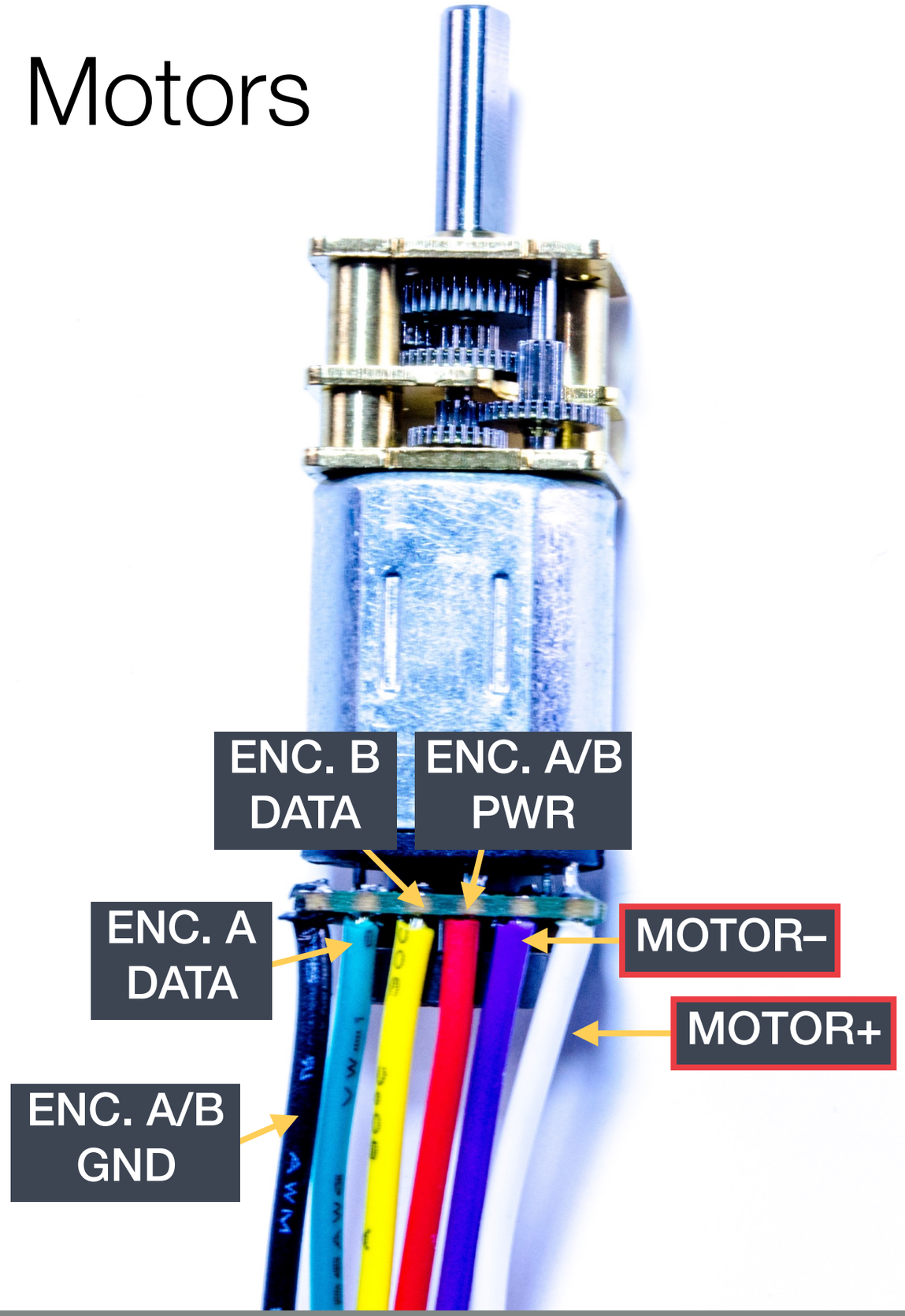
Not a class of DC motor *per se*,
but an assembly of:
DC motor + gear set + control
circuit + position sensor.

Always built including position
feedback device.

Controlled by changing the
duration of control signal's positive
pulse (determines shaft position).

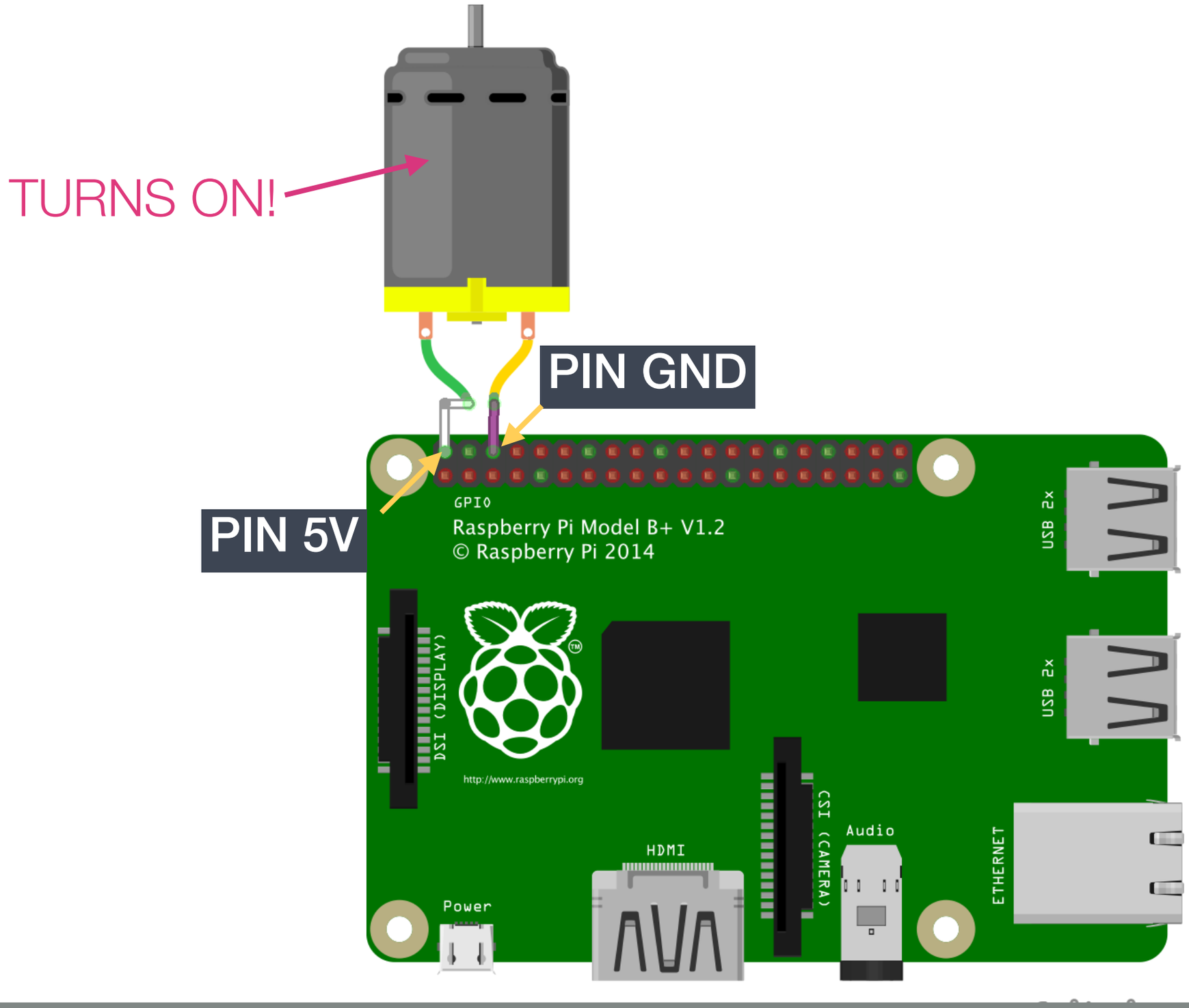
Controlling DC Motors

Denbot's DC Motors



- The DC motors used in this class use 6 connections for Power, Motion, and Speed Data.

Controlling DC Motors



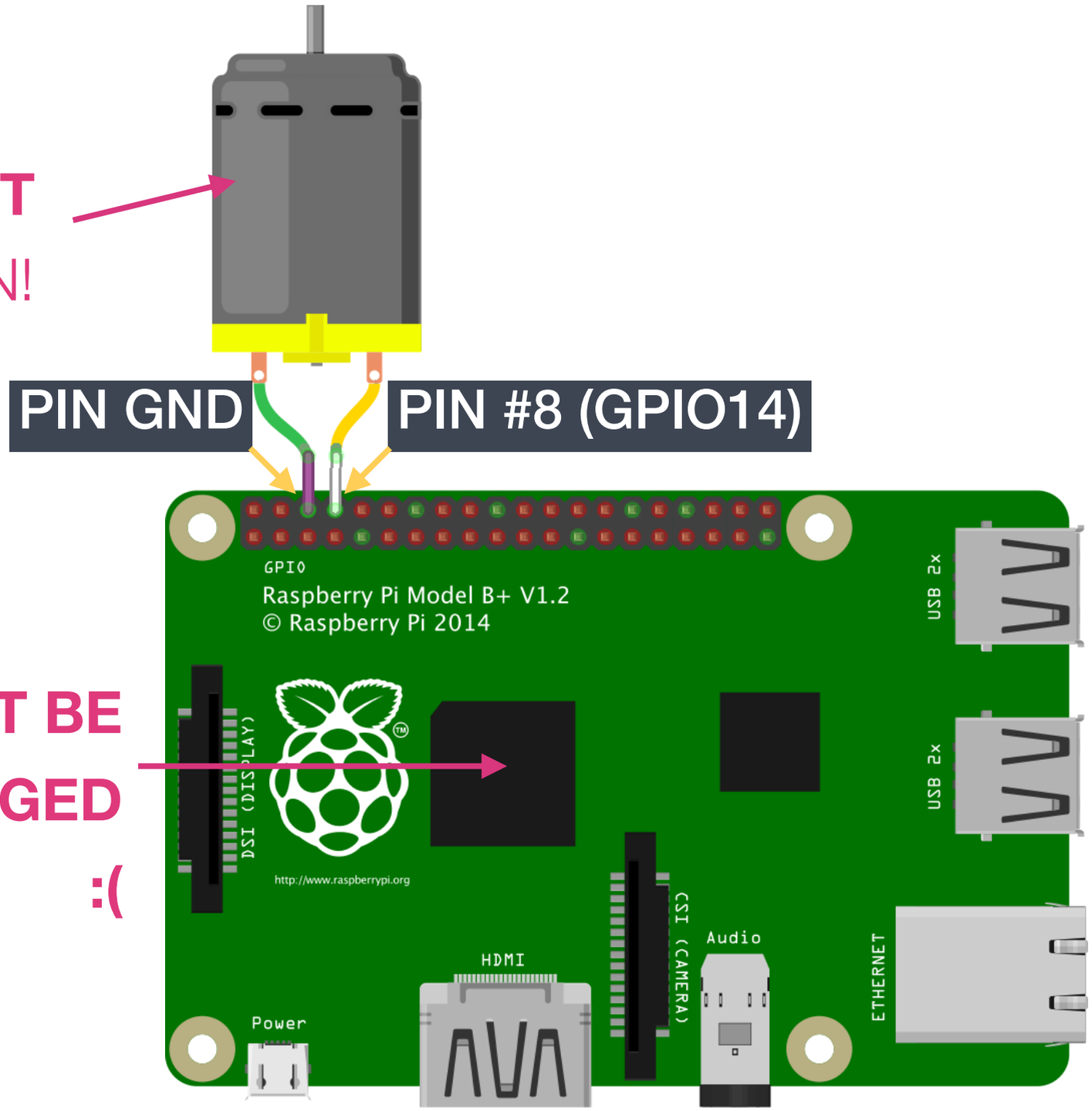
- The DC motors used in this class can be powered directly from the 5V supply of the Raspberry Pi!

Controlling DC Motors (Do Not Attempt)

DOES NOT

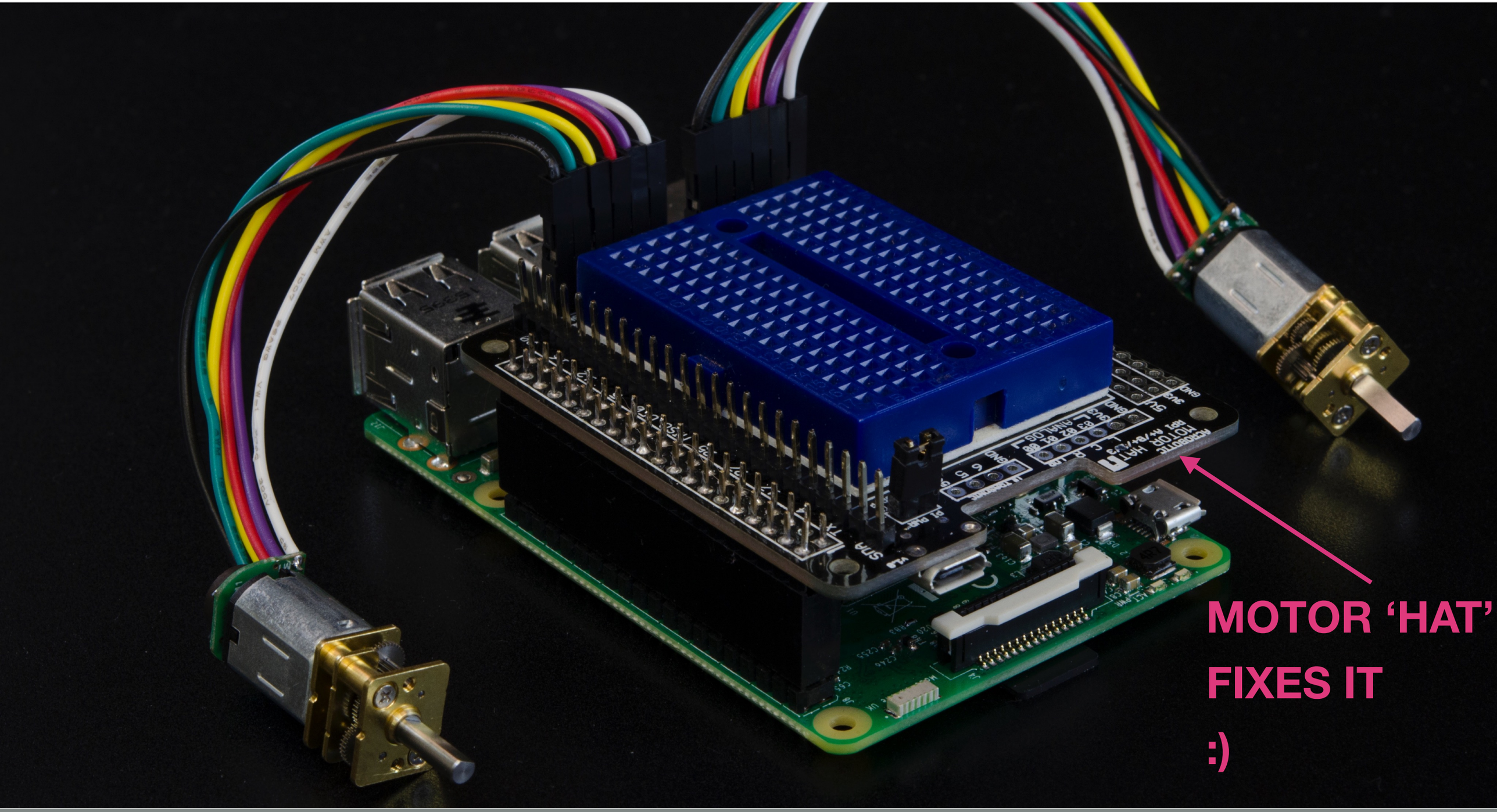
TURNS ON!

MIGHT BE DAMAGED :(



- The Raspberry Pi I/O pins **cannot** supply enough current to operate this motor directly!

Controlling DC Motors



- The Motor HAT I/O pins **<u>can</u>** supply enough current to operate the motor directly!

Controlling Speed Using PWM

- A PWM signal is a **digital signal** whose average value of voltage (and current) is controlled by turning the switch between supply and load on and off at a fast rate.

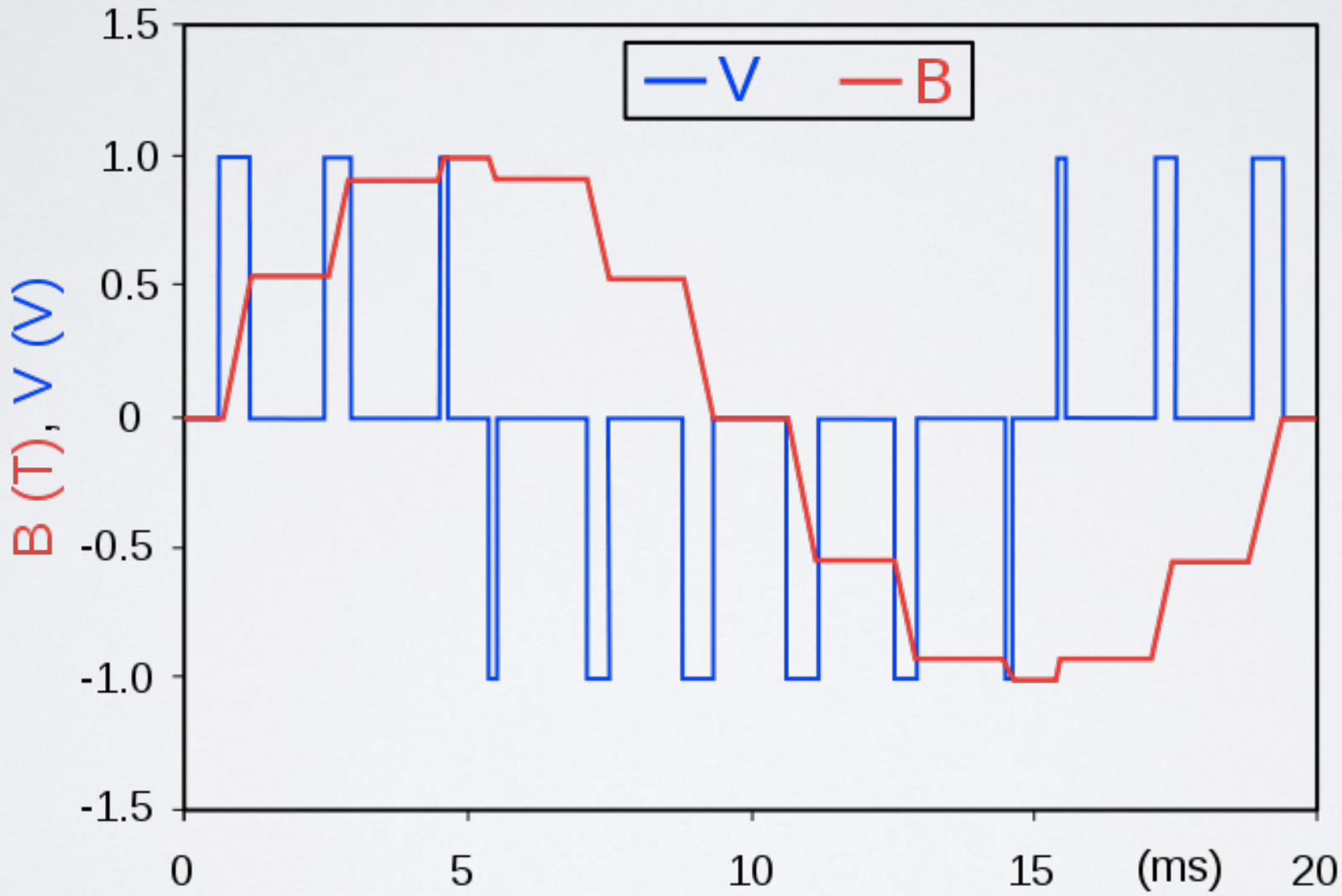


Image credit: Zureks - Own work, CC BY-SA 3.0.

- The longer the switch is **on** compared to the **off** periods, the higher the total power supplied.

Controlling DC Motors With PWM

IPy led_control/ | IPy led_dimmer | Makerbro

denbot16.local:8888/notebooks/led_control/led_dimmer.ipynb

IP[y]: Notebook led_dimmer (unsaved changes) Logout

File Edit View Insert Cell Kernel Help

Code Cell Toolbar: None

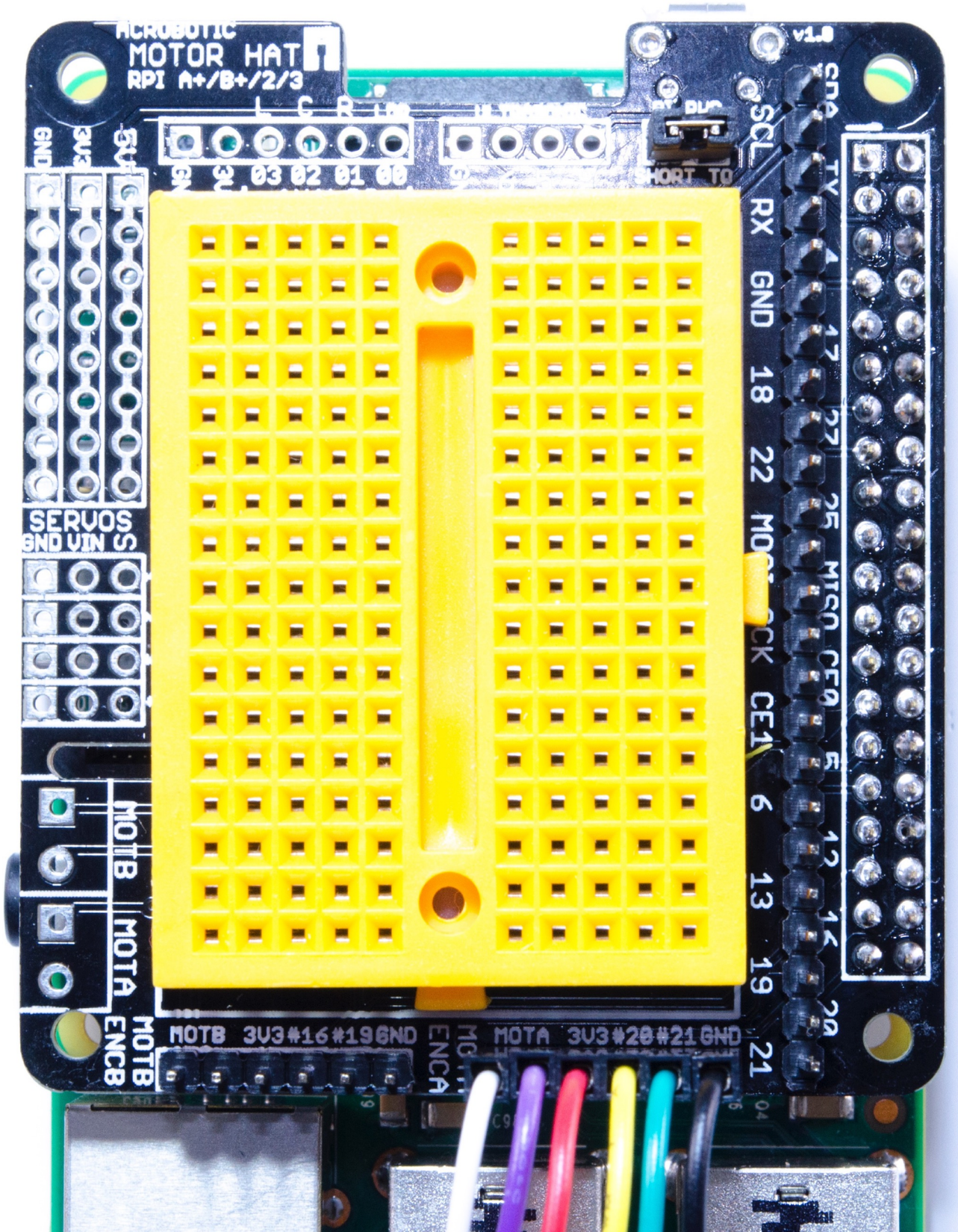
In [ ]:
```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)
GPIO.setup(12, GPIO.OUT)

# Initialize PWM on pin 12 at 1KHz
pwm_obj = GPIO.PWM(12, 1000)

# Start PWM with duty cycle of 0
pwm_obj.start(0)
time.sleep(1)
# Change PWM to a 25% duty cycle
pwm_obj.ChangeDutyCycle(25)
time.sleep(1)
# Change PWM to a 50% duty cycle
pwm_obj.ChangeDutyCycle(50)
time.sleep(1)
# Change PWM to a 100% duty cycle
pwm_obj.ChangeDutyCycle(100)
time.sleep(1)

GPIO.cleanup()
```

- The Motor HAT has its own PWM generator chip, so we won't use RPi.GPIO's PWM functionality!

Controlling DC Motors With PWM

- Let's go ahead and wire our first motor:

  MOTA: White, Purple (interchangeable)
  3V3: Red
  #20: Yellow
  #21: Green
  GND: Black



Controlling DC Motors With PWM

- Let's use the **PCA9685 module** inside the **motor_control** directory to generate a PWM signal (similar to dimming the LED)!

```
>>> from PCA9685 import Driver
```

```
>>> Driver().setFrequency(1000)
```

```
>>> Driver().setOn(Driver().CHANNEL_0)
```

```
>>> Driver().setPWM(Driver().CHANNEL_1, 2047)
```

MOTOR A	MOTOR B
Direction: CHANNEL_0	**Direction**: CHANNEL_2
Speed: CHANNEL_1	**Speed**: CHANNEL_3

- Direction can be either **On** or **Off**.
- Speed ranges between **0** and **4095**.

Controlling DC Motors With PWM

- As always, let's use **Python** on our Raspberry Pi's Operating System to control the DC motors.

- Connect to your Raspberry Pi via SSH or VNC.

- From a Terminal window, navigate to the **motor_control** directory.

```
cd ~/Makerden/motor_control
```

- Fire up an interactive Python session.

```
python
```

Controlling DC Motors With PWM

- Let's use the **PCA9685 module** to generate a PWM signal (similar to dimming the LED)!

x1sc0 — pi@denbot16: ~/Makerden/motor_control — ssh pi@denbot16.local

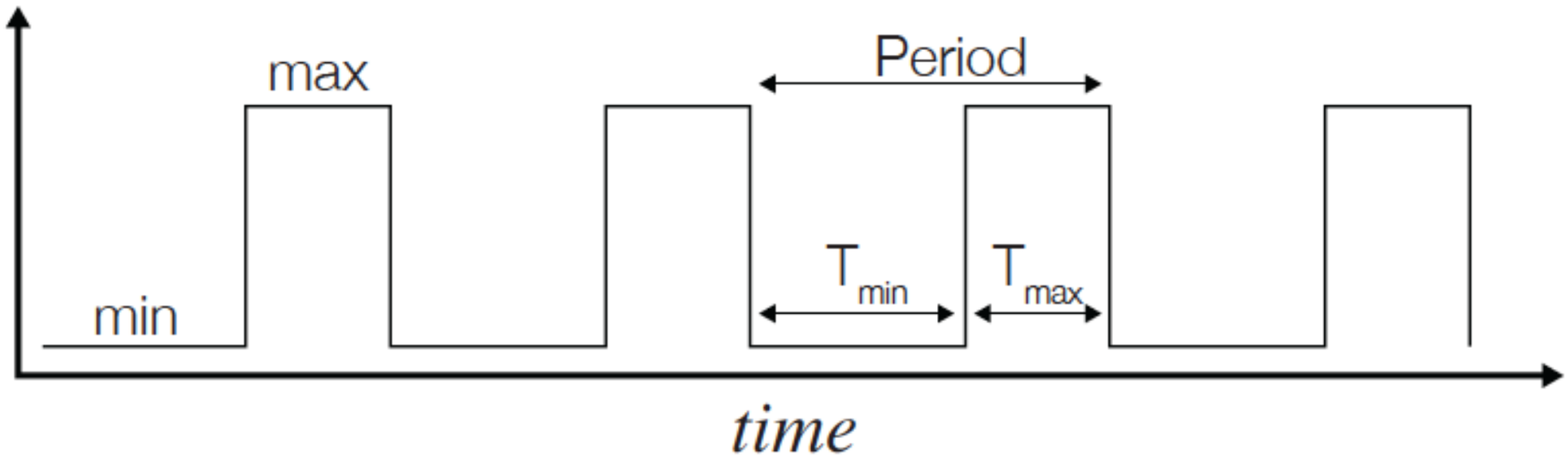
pi@denbot16: ~/Makerden/motor_control — ssh pi... | ...ntro_Robotics/Level_III/04. Motion Control — -bash

```
# Create an instance of PWM Driver class
driver = Driver()
driver.setFrequency(1000)

pwm_dir = driver.CHANNEL_0
pwm_spd = driver.CHANNEL_1

try:
    driver.setOn(pwm_dir)
    driver.setPWM(pwm_spd,4095)
    time.sleep(5)
    driver.setOff(pwm_dir)
    driver.setPWM(pwm_spd,512)
    time.sleep(5)
except KeyboardInterrupt:
    print "Program interrupted...!"
finally:
    print "Turning Off All Channels...!"
    driver.setAllOff()
~
~
~
```

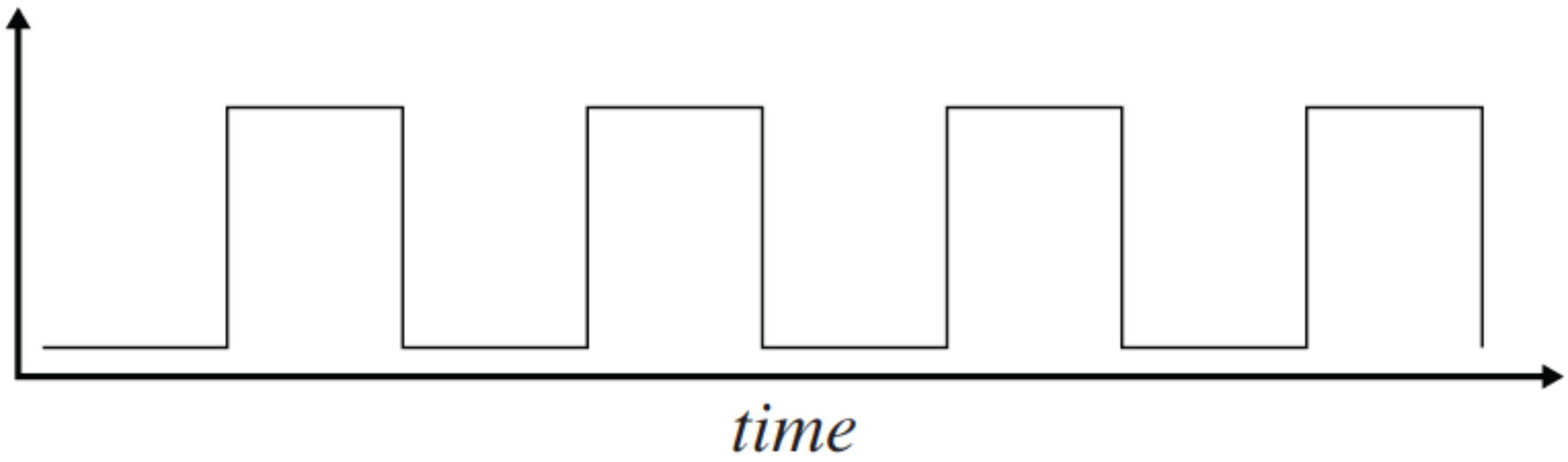
Create a new notebook "motor_control" inside the motor_control folder.

Controlling Motor Speed With PWM

Rectangular or Pulse Wave



Special case: Square Wave ($T_{min} = T_{max}$)



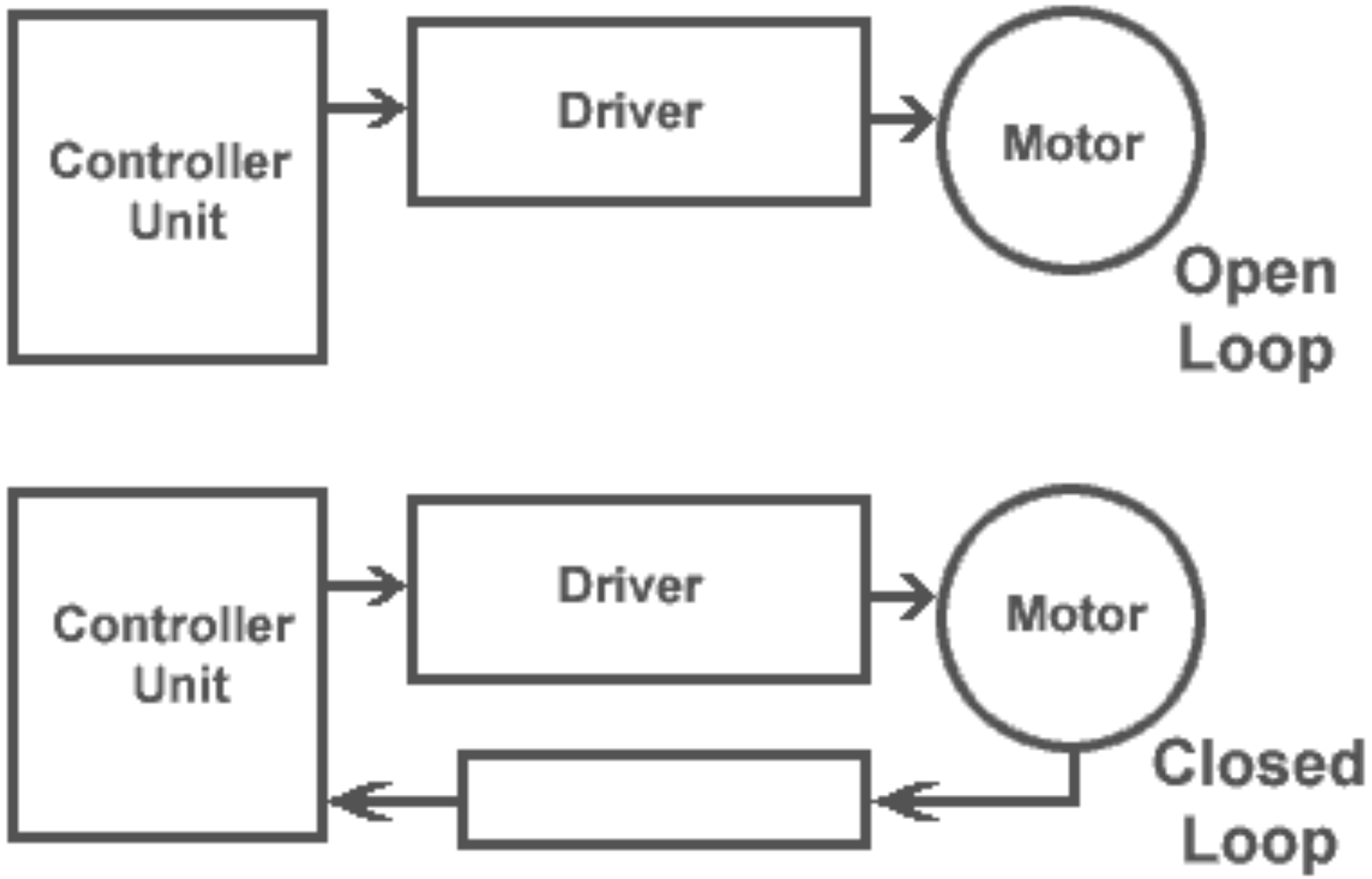
Frequency = 1/Period

Duty Cycle = 100% x $T_{max}/(T_{max}+T_{min})$

Measuring Motor Speed

Controlling Motor Speed: Open-Loop

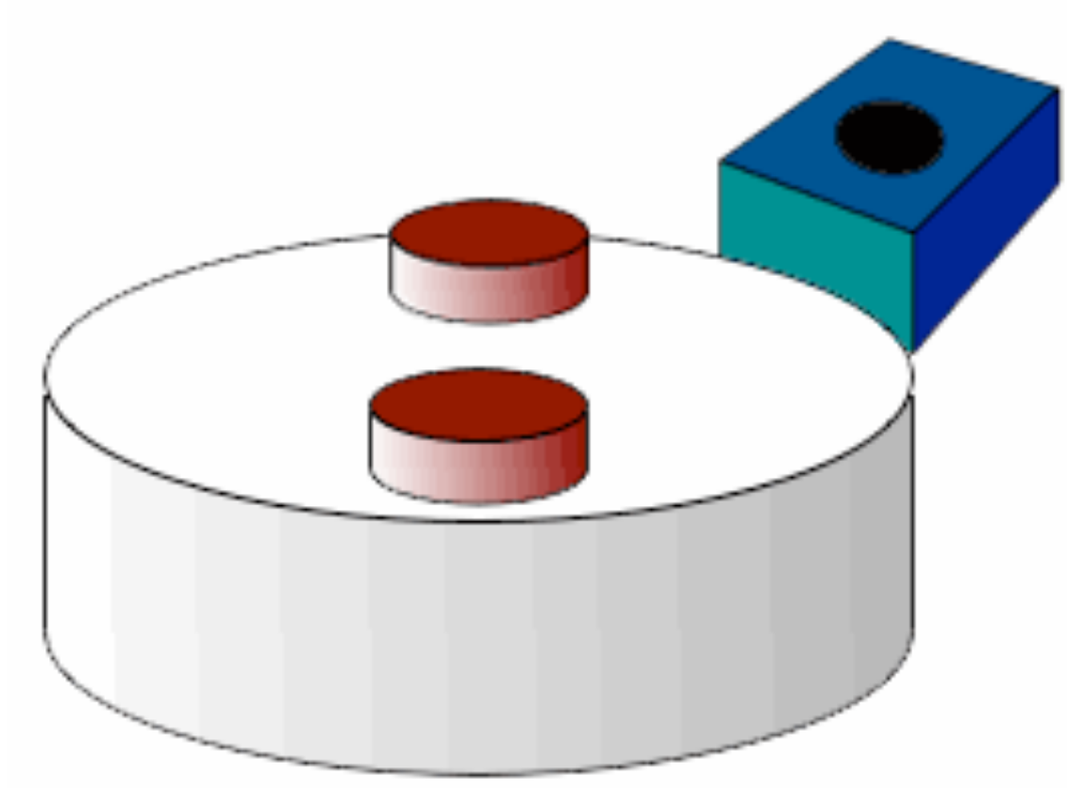
- Insofar, we've applied individual control efforts to the motor speed and assumed the desired results (speed) were achieved.



Measuring Motor Speed

- Controlling motor speed accurately requires that we measure actual speed.

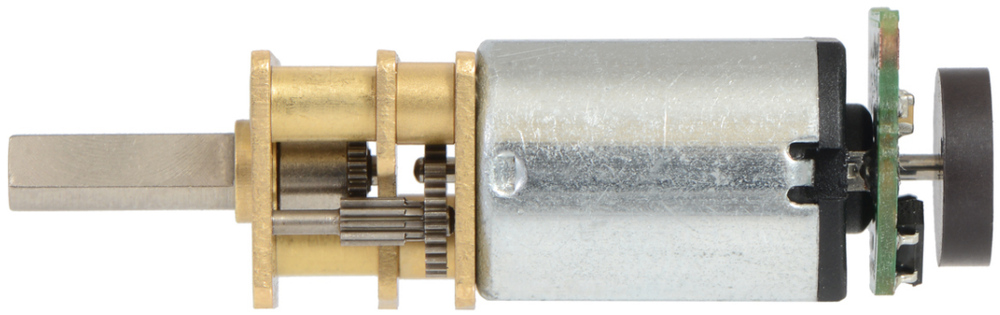
- Easiest way to measure motor speed is to measure shaft rotations

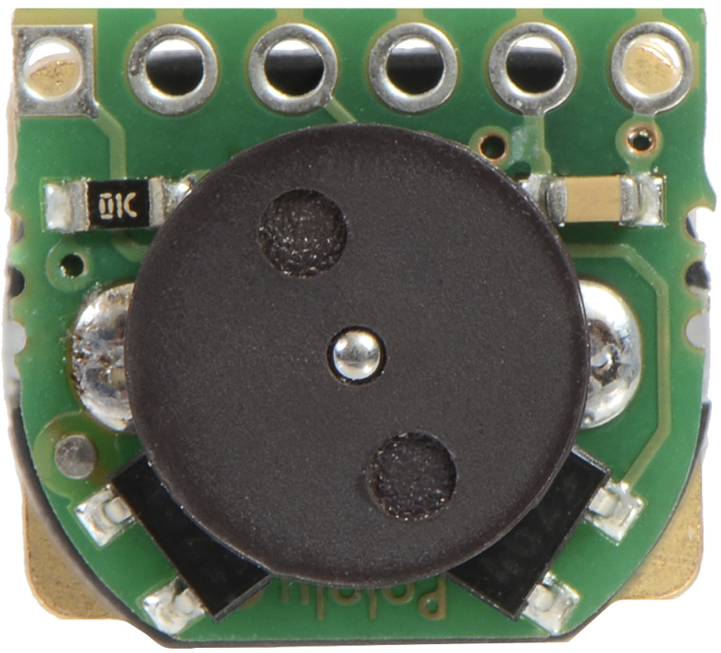


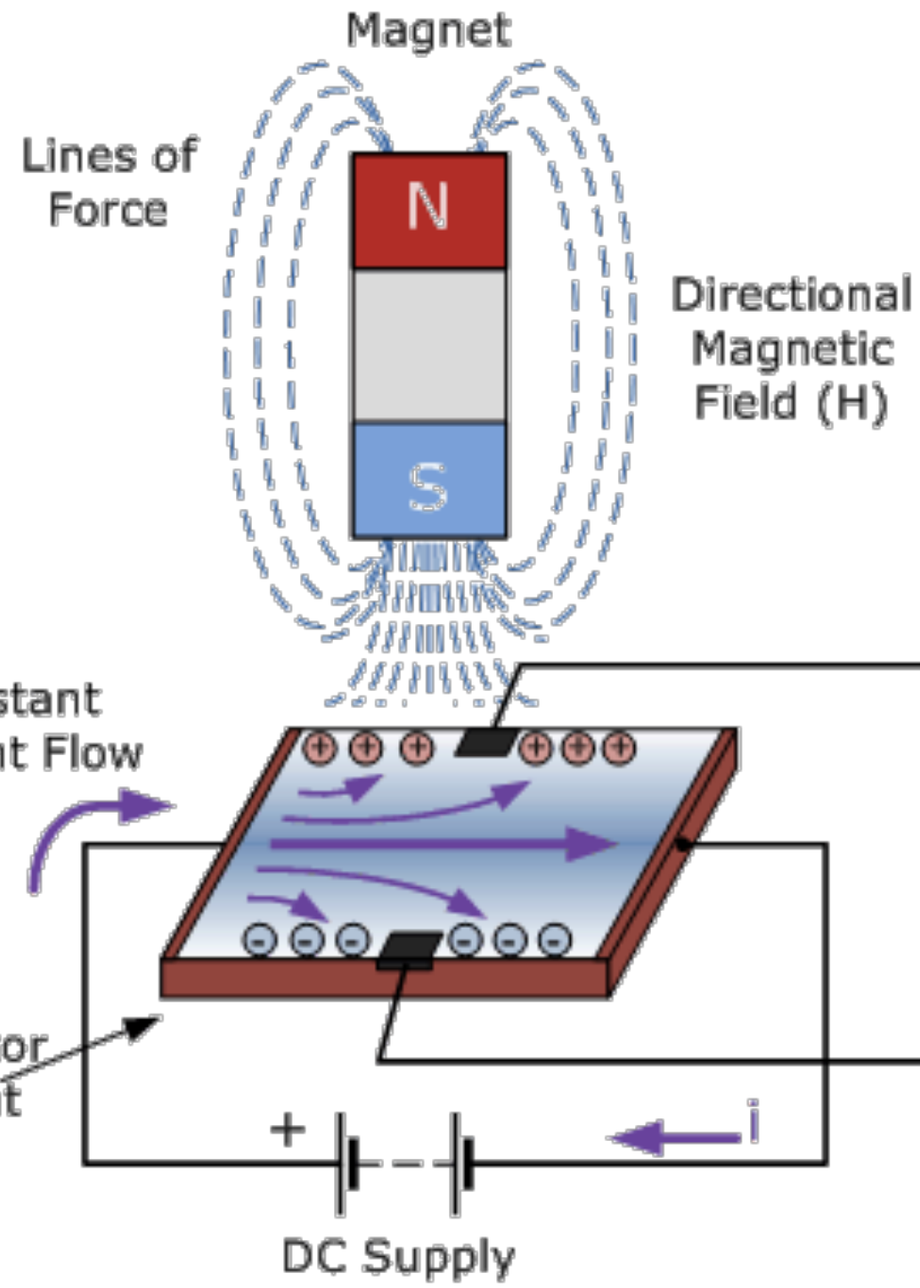
- Rotary encoders can be used to convert the angular position of motor shaft to a digital code.

- Most popular rotary encoders are either optical or magnetic

Measuring Motor Speed (Wheel Encoders)







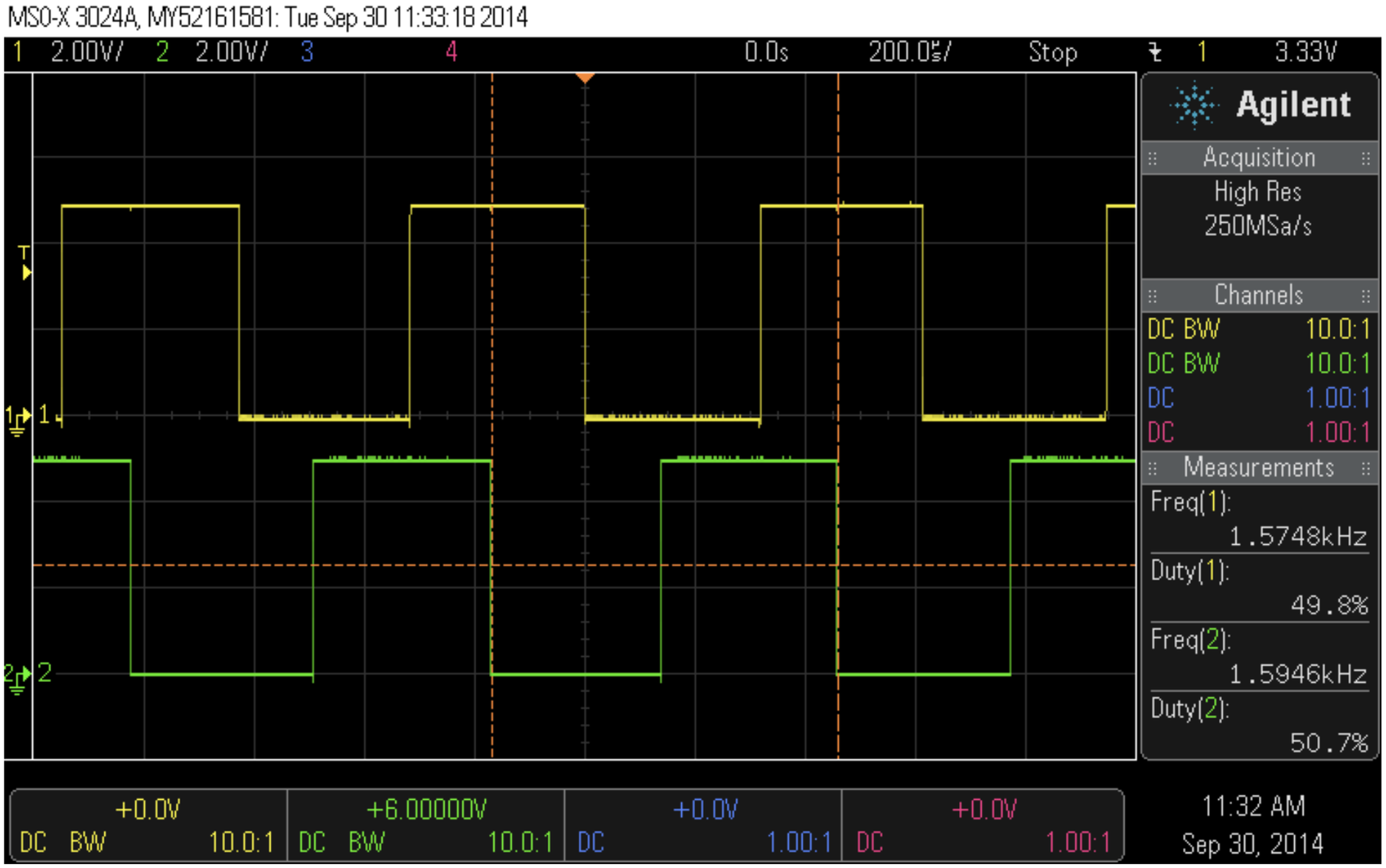
- Denbot's rotary encoder uses 2 Hall Effect sensors to detect the rotation of two magnets mounted on motor shaft

Measuring Motor Speed (Wheel Encoders)

- Let's use the **<u>Encoder module</u>** to read a digital (gray code) signal!

ENC. A DATA

ENC. B DATA




```
>>> from Encoder import Encoder
```

```
>>> cycles, dir_flag = Encoder(38,40).get_cycles()
```

Controlling DC Motors With PWM

- Let's use the **Encoder module** to read a digital (gray code) signal!

x1sc0 — pi@denbot16: ~/Makerden/motor_speed — ssh pi@denbot16.local

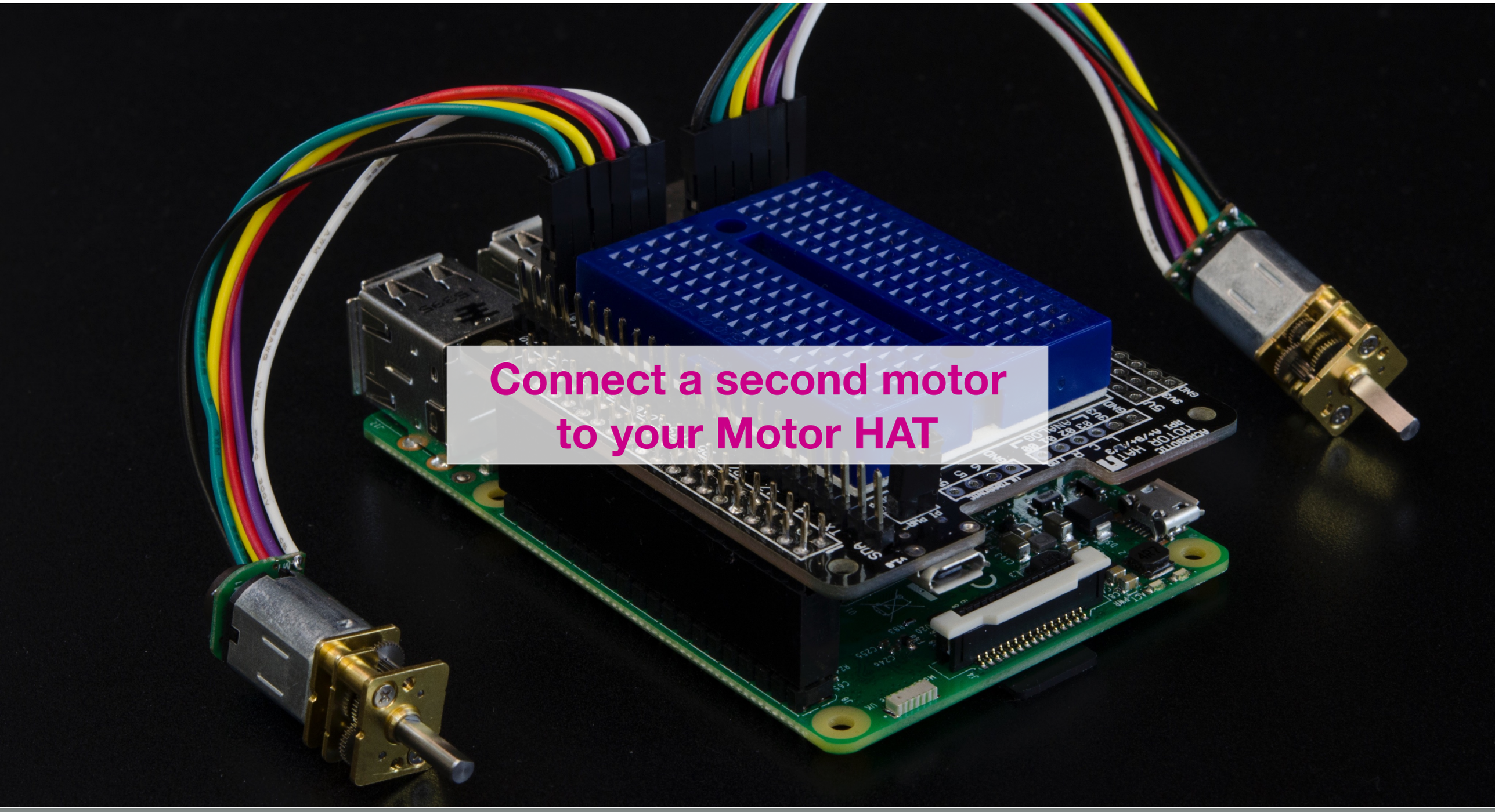
pi@denbot16: ~/Makerden/motor_speed — ssh pi... | ~/Work/Intro_Robotics — git • git push origin master

```
1  #!/usr/bin/env python
2  from Encoder import Encoder
3  import time
4
5  encoder = Encoder(38, 40)
6  t = time.time()
7  while True:
8      cycles, dir_flag = encoder.get_cycles()
9      if dir_flag:
10         t = time.time()
11         print "Wheel changed directions!"
12         continue
13     if cycles != 0:
14         dt = time.time()-t
15         t = time.time()
16         print "Completed cycle in %f seconds" % dt
~
~
~
~
~
~
```

Create a new notebook "motor_speed" inside the motor_speed folder.

Intro to Navigation

Programming Simple Movements Commands



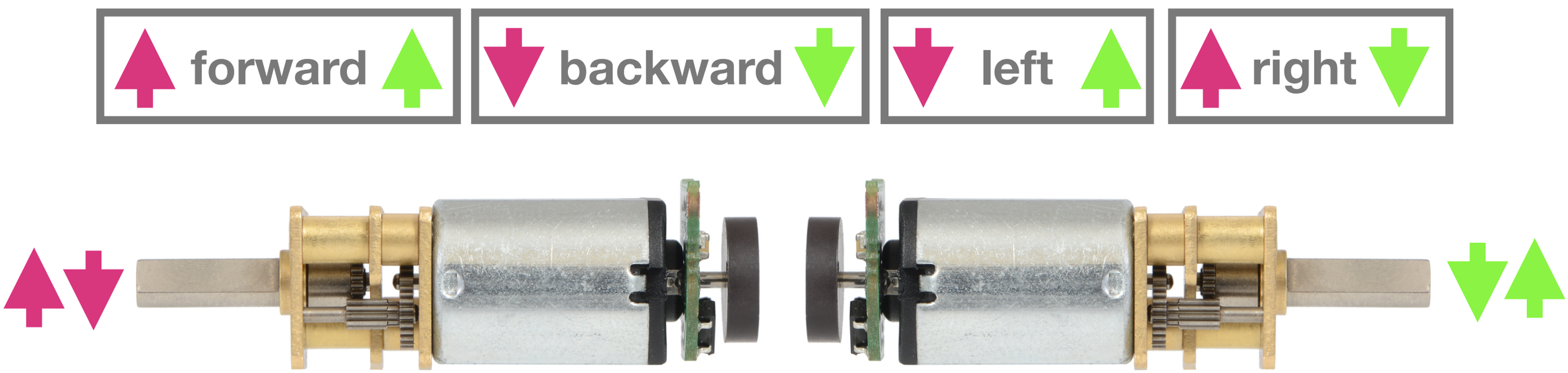
- Using the PCA9685 module we can control 2 motors simultaneously and prescribe the movement

Programming Simple Movements Commands

- We'd like to have a custom module that allows us to set the movement of our robot so that we can call:

```
>>> motion.forward(2)
```

```
>>> motion.left(5)
```



- We need to specify the time for how long these commands are run!