Capstone Design

**Assignment #2**

Raspberry Pi Smart Car Kit

**Tesla Team members:**

|  |  |
| --- | --- |
| Muslimjon Kholjuraev | U1510394 |
| Jakhongir Bakhodirov | U1510369 |
| Oybek Beknazarov | U1510379 |
| Sardorkhuja Ibrokhimov | U1510397 |

**Motor application**

**DC motor class**

|  |
| --- |
| from gpiozero import PWMOutputDevice  from time import sleep  class Tesla\_motor:  def \_\_init\_\_(self, pin1, pin2, pin3, pin4):  # Motor A, Left Side GPIO CONSTANTS  self.PWM\_FORWARD\_LEFT\_PIN = pin1 # IN1 - Forward Drive 18  self.PWM\_REVERSE\_LEFT\_PIN = pin2 # IN2 - Reverse Drive 23  # Motor B, Right Side GPIO CONSTANTS  self.PWM\_FORWARD\_RIGHT\_PIN = pin3 # IN1 - Forward Drive 24  self.PWM\_REVERSE\_RIGHT\_PIN = pin4 # IN2 - Reverse Drive 25  self.set\_pwm()  def set\_pwm(self):  # Initialise objects for H-Bridge PWM pins  # Set initial duty cycle to 0 and frequency to 1000  self.forwardLeft = PWMOutputDevice(self.PWM\_FORWARD\_LEFT\_PIN, True, 0, 100)  self.reverseLeft = PWMOutputDevice(self.PWM\_REVERSE\_LEFT\_PIN, True, 0, 100)  self.forwardRight = PWMOutputDevice(self.PWM\_FORWARD\_RIGHT\_PIN, True, 0, 100)  self.reverseRight = PWMOutputDevice(self.PWM\_REVERSE\_RIGHT\_PIN, True, 0, 100)  def allStop(self):  self.forwardLeft.value = 0  self.reverseLeft.value = 0  self.forwardRight.value = 0  self.reverseRight.value = 0  def forwardDrive(self):  self.forwardLeft.value = 1.0  self.reverseLeft.value = 0  self.forwardRight.value = 1.0  self.reverseRight.value = 0  def reverseDrive(self):  self.forwardLeft.value = 0  self.reverseLeft.value = 1.0  self.forwardRight.value = 0  self.reverseRight.value = 1.0  def spinLeft(self):  self.forwardLeft.value = 0  self.reverseLeft.value = 1.0  self.forwardRight.value = 1.0  self.reverseRight.value = 0  def SpinRight(self):  self.forwardLeft.value = 1.0  self.reverseLeft.value = 0  self.forwardRight.value = 0  self.reverseRight.value = 1.0  def forwardTurnLeft(self):  self.forwardLeft.value = 0.2  self.reverseLeft.value = 0  self.forwardRight.value = 0.8  self.reverseRight.value = 0  def forwardTurnRight(self):  self.forwardLeft.value = 0.8  self.reverseLeft.value = 0  self.forwardRight.value = 0.2  self.reverseRight.value = 0  def reverseTurnLeft(self):  self.forwardLeft.value = 0  self.reverseLeft.value = 0.2  self.forwardRight.value = 0  self.reverseRight.value = 0.8  def reverseTurnRight(self):  self.forwardLeft.value = 0  self.reverseLeft.value = 0.8  self.forwardRight.value = 0  self.reverseRight.value = 0.2  def run\_very\_slow(self):  self.forwardLeft.value = 0.1  self.reverseLeft.value = 0.1  self.forwardRight.value = 0.1  self.reverseRight.value = 0.1 |

**DC motor control**

|  |
| --- |
| from motor import Tesla\_motor  import wiringpi as wp  from time import sleep  def my():  wp.wiringPiSetupGpio()  wp.wiringPiSetup()  tesla = Tesla\_motor(23,18,25,24)  tesla.allStop()  #1  tesla.forwardDrive()  sleep(5)  tesla.allStop()  #2  tesla.reverseDrive()  sleep(5)  tesla.allStop()  #3  tesla.forwardTurnLeft()  sleep(1)  tesla.allStop()  tesla.forwardDrive()  sleep(3)  tesla.allStop()  #4  tesla.reverseTurnRight()  sleep(1)  tesla.allStop()  tesla.reverseDrive()  sleep(3)  tesla.allStop()  #5  tesla.forwardDrive()  sleep(2)  tesla.allStop()  tesla.spinLeft()  sleep(1)  tesla.allStop()  #6  tesla.reverseDrive()  sleep(2)  tesla.allStop()  tesla.SpinRight()  sleep(1)  tesla.allStop()    my() |

**Sensor applications**

**Ultrasonic class**

|  |
| --- |
| import wiringpi as wp  import time  LOW = 0  HIGH =1  OUTPUT = wp.OUTPUT  INPUT = wp.INPUT  MAX\_SENSOR\_DISTANCE = 500  NO\_ECHO = 0  class Ultrasonic(object):  def \_\_init\_\_(self, trig=3,echo=23):  self.trig = trig  self.echo = echo    wp.pinMode(self.trig,OUTPUT)  wp.pinMode(self.echo,INPUT)    time.sleep(0.5)  def distance(self):  wp.digitalWrite(self.trig, LOW)    time.sleep(0.1)  wp.digitalWrite(self.trig, HIGH)  time.sleep(0.00001)  wp.digitalWrite(self.trig, LOW)    now\_time = time.time()  pulse\_start=0  pulse\_end=0  while wp.digitalRead(self.echo)== LOW:  #and time.time()-now\_time<timeout):  pulse\_start = time.time()    while wp.digitalRead(self.echo)== HIGH:  pulse\_end=time.time()    pulse\_dur = pulse\_end-pulse\_start    distance=round(pulse\_dur\*17000.0,2)  return distance  def record\_pulse\_length(self):  self.start\_time=time.time()  while ( wp.digitalRead(self.echo) == HIGH):  pass  self.end\_time=time.time() |

**Ultrasonic application 1**

|  |
| --- |
| from motor import Tesla\_motor  from ultrasonic import Ultrasonic  import wiringpi as wp  def my():  wp.wiringPiSetupGpio()  wp.wiringPiSetup()  uls = Ultrasonic(20, 21)  tesla = Tesla\_motor(23,18,25,24)  tesla.allStop()  while 1:  distance = uls.distance()  print(distance)  if distance > 30:  tesla.forwardDrive()  #sleep(5)  else:  tesla.allStop()    my() |

**Ultrasonic application 2**

|  |
| --- |
| from motor import Tesla\_motor  from ultrasonic import Ultrasonic  import wiringpi as wp  def my():  wp.wiringPiSetupGpio()  wp.wiringPiSetup()  uls = Ultrasonic(20, 21)  tesla = Tesla\_motor(23, 18, 25, 24)  tesla.allStop()  while 1:  distance = uls.distance()  print(distance)  if distance > 50:  tesla.forwardDrive()  # sleep(5)  else:  tesla.reverseDrive()  my() |