Public Transport Optimization

Sensor and its Features

- GPS
  - Visualization of the GPS data can be challenging when using a programming language to automate it. There are some interesting Python packages that can be used for such purposes. But, in some cases, it can be hard to install and use them, especially if you have only a simple task to do.
- RADAR
  - An Arduino-based radar project was implemented in this tutorial using an Arduino, HC-SR04 ultrasonic distance sensor, MG90S micro servo motor, and Python code run on a Raspberry Pi. The goal of this project was to introduce a novel concept related to real-world technology, but implemented through inexpensive tools available to the maker and aspiring engineer. The HC-SR04 uses sound waves to approximate the distance between its receiver and an object in the distance, while the MG90S servo rotates in a prescribed fashion according to pulse-width modulation signals controlled by the Arduino board. In order to visualize the outputted angular position and approximate ranging of the HC-SR04 – Python code was implemented on a Raspberry Pi to create a plan position indicator on a polar plot. This PPI gives the user a way of visualizing the objects that surround the motor and ultrasonic sensor, much like a radar approximates the objects surrounding its base station. Several skills used in this tutorial can be applied to real-world applications, whether through obstacle detection, motor control, distancing and ranging, or even a new tool for visualizing data.

Main Sensor in intelligent Transport system

- Positon sensor
  - The term position sensor is used for a sensor that gives a measure of the distance between a reference point and the current location of the target, while a displacement sensor gives a measure of the distance between the present position of the target and the previously recorded position
- Chemical sensor
  - A chemical sensor is a device that transforms chemical data into a valuable output signal. Generally, it converts physicochemical properties or chemical interactions (e.g., concentration or total composition of specific species into an output signal).
- Air Bag Sensor
  - Airbag sensors are small pieces of electronics are designed to tell whe the vehicle has been damaged in an accident. These sensors respond t... several different sets of stimuli, including sudden stopping, increased pressure as pieces of the car are moved due to the force of the collision, and more.
- Gas Composition Sensor
  - Gas sensors are devices that can detect the presence and concentration of various hazardous gases and vapors, such as toxic or explosive gases, volatile organic compounds (VOCs), humidity, and odors.

Python Code For Sensors used in Transport.

- Traffic Light .
  - Code:
  - #Function to simulate a traffic light
  - #It is required to make 2 user defined functions trafficLight() and light().
  - Def trafficLight():
  - signal = input("Enter the colour of the traffic light: ")
  - if (signal not in ("RED","YELLOW","GREEN")):
  - print("Please enter a valid Traffic Light colour in CAPITALS")
  - else:
  - value = light(signal) #function call to light()
  - if (value == 0):
  - print("STOP, Your Life is Precious.")
  - elif (value == 1):
  - print ("PLEASE GO SLOW.")
  - else:
  - print("GO!,Thank you for being patient.")
  - #function ends here
  -
  - def light(colour):
  - if (colour == "RED"):
  - return(0);
  - elif (colour == "YELLOW"):
  - return (1)
  - else:
  - return(2)
  - #function ends here
  -
  - trafficLight()
  - print("SPEED THRILLS BUT KILLS")

Output:

>> Enter the colour of the traffic light: RED

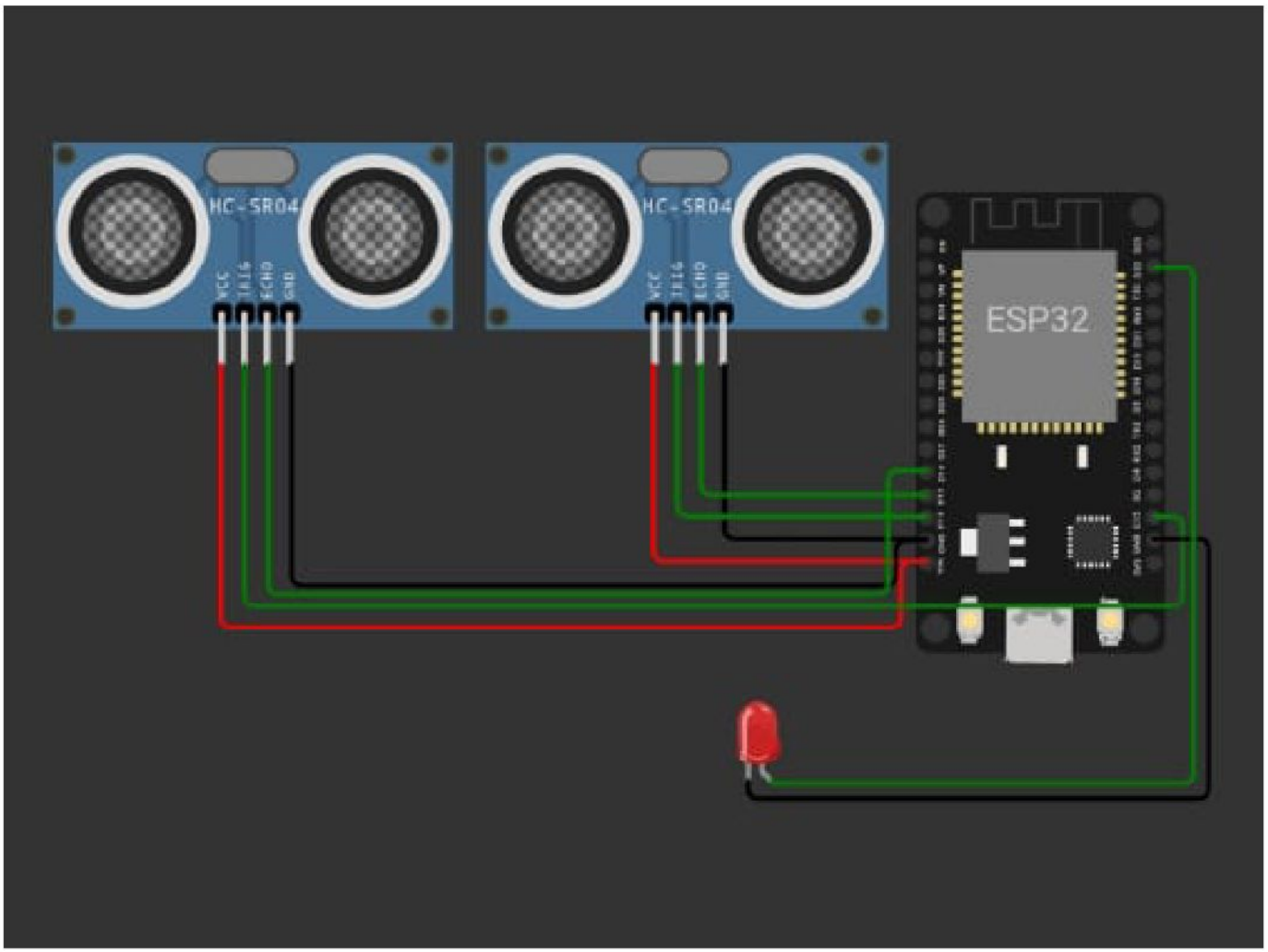
>> STOP, Your Life is Precious.

>> SPEED THRILLS BUT KILLS

Gas Sensor

```
Code:
Import time
import board
import busio
import adafruit_sgp30

i2c = busio.I2C(board.SCL, board.SDA, frequency=100000)

# Create library object on our I2C port
sgp30 = adafruit_sgp30.Adafruit_SGP30(i2c)

print("SGP30 serial #", [hex(i) for i in sgp30.serial])

sgp30.set_iaq_baseline(0x8973, 0x8AAE)
sgp30.set_iaq_relative_humidity(celsius=22.1, relative_humidity=44)

elapsed_sec = 0

while True:
    print("eCO2 = %d ppm \t TVOC = %d ppb" % (sgp30.eCO2, sgp30.TVOC))
    time.sleep(1)
    elapsed_sec += 1
    if elapsed_sec > 10:
        elapsed_sec = 0
        print(
            "**** Baseline values: eCO2 = 0x%x, TVOC = 0x%x"
            % (sgp30.baseline_eCO2, sgp30.baseline_TVOC)
```

OUTPUT:



ets Jul 29 2019 12:21:46

rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:1156
load:0x40078000,len:11456
ho 0 tail 12 room 4
load:0x40080400,len:2972
entry 0x400805dc
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200
100
200