Python Script – trligt.py

#!usr/bin/python

#

# IOT Based Traffic Management System Project :

# By Srujan Deshpande, 19183291

# M.E. (Information and Network Security FT)

# Academic Year : 2019-2020

#

# This file contains python script to control GPIO pins

# that turns RED, YELLOW and GREEN LED on and off

# simulating real life traffic signal lights.

#

import RPi.GPIO as GPIO

import os

import time

import signal

import sys

from datetime import datetime

from collections import namedtuple

sigStatus = "RED"

prevSigStatus = sigStatus

ledStatus = [ {"Status":"RED"}, {"Status":"YELLOW"}, {"Status": "GREEN"} ]

sigStatusFile = "./log/sigop.txt" # This file is used to record traffic ligh status

50 | P a g e51 | P a g e

vehTypeFile = "./log/veh.txt" # This file is used to detect presence of emergency vehicle.

Detection is done by another script "ocr.py" that performs DSP on vehicle image to extract

vehicle registration number

scriptName = ""

Month = ["JAN", "FEB", "MAR", "APR", "MAY", "JUN", "JUL", "AUG", "SEP", "OCT",

"NOV", "DEC"]

dateStr = ""

def getDateTime():

#

# This funcion retruns a string composed of script name,

# current date and time. This general purpose function

# is useful to keep track of simulaeted events of traffic

# on the road which faces real time signal light

# implementation using raspberry pi #4 which is handled by

# another script <trlight.py>

#

scrName = str(sys.argv[0])

dateTimeObj = datetime.now()

year = str(dateTimeObj.year)

month = dateTimeObj.month

day = str(dateTimeObj.day)

hour = str(dateTimeObj.hour)

minute = str(dateTimeObj.minute)

second = dateTimeObj.second

usecond = dateTimeObj.microsecond

float\_sec = round((second\*1000000 + usecond)/1000000,3)

y = str(float\_sec)

nfsecStr = y[3:len(y)]52 | P a g e

scrName = scrName.rjust(10)

dateStr = '[' + scrName + '/' + day + '-' + Month[month-1] + '-' + year + ' :: ' + hour + ':'

+ minute + ':' + str(second).zfill(2) + '.' + nfsecStr.zfill(2) + '' + '] ->'

return (dateStr)

def signal\_handler(signal,frame):

#

# This is a Linux signal handler for handling abrupt

# termination of scriprt i.e. CTRL+C while script is

# doing processing. It is important to handle to secure

# raspberry pi GPIO setup to its initial state.

#

print("%s Abrupt termination :: Performing clean up .." % (getDateTime()))

GPIO.output (11, GPIO.LOW) # Turn RED light OFF

GPIO.output (15, GPIO.LOW) # Turn RED light OFF

GPIO.output (37, GPIO.LOW) # Turn RED light OFF

destroySetup()

print("%s Abrupt termination :: Clean up completed .. Exiting" % (getDateTime()))

sys.exit(0)

signal.signal(signal.SIGINT, signal\_handler)

#

# GPIO to Bread Board connections

# Red, Yellow, Green Light Control

# Pin #11 (GPIO #17) = Red Light

# Pin #15 (GPIO #22)= Yellow Light

# Pin #37 (GPIO #26) = Green Light

#53 | P a g e

def turnSignal(sigStatus):

#

# This function turns RED and GREEN LED ON for 5-sec and

# YELLOW LED for 2-sec. It turns LEDs OFF for 0.1 sec

#

myFile = open(sigStatusFile,"w")

myFile.write(str(sigStatus))

myFile.close()

if(sigStatus == "RED"):

GPIO.output (11, GPIO.HIGH) # Turn RED light ON

time.sleep(5)

GPIO.output (11, GPIO.LOW) # Turn RED light OFF

time.sleep(0.1)

if(sigStatus == "YELLOW"):

GPIO.output (15, GPIO.HIGH) # Turn YELLOW light ON

time.sleep(2)

GPIO.output (15, GPIO.LOW) # Turn YELLOW light OFF

time.sleep(0.1)

if(sigStatus == "GREEN"):

GPIO.output (37, GPIO.HIGH) # Turn GREEN light ON

time.sleep(5)

GPIO.output (37, GPIO.LOW) # Turn GREEN light OFF

time.sleep(0.1)

def chkVehType():

#

# This function checks vehicle type arrving at traffic

# signal. The vehicle type is detected and stored by

# another script ocr.py that performes Optical Character

# Recognition (OCR) to detect vehicle number plate and

# extract vehicle resgistration number. The vehicle type

# is stored by ocr.py in a file "./log/veh.txt" based on status.

#

myVehFile = open(vehTypeFile,"r")

return(myVehFile.read())

def makeSetup():

#

# This function makes GPIO setup for BOARD layout

#

#

# Define usage of Raspberry PI board pins

#

GPIO.setmode (GPIO.BOARD)

GPIO.setup(11, GPIO.OUT) # Red LED

GPIO.setup(15, GPIO.OUT) # Yellow LED

GPIO.setup(37, GPIO.OUT) # Green LED

def runTrafficSignals():

#

# This function runs a loop to control traffic light

# sequentially in RED-YELLOW-GREEN order. On detection of

# emergency vehicle presence, it automatically turns

# signal light GREEN for the easy passage of emergency

# vehicle.

#

global trl\_dur

sigLight = "RED"

for i in range (0,trl\_dur): # run it for 3-min

if(chkVehType() == "EMERGENCY VEHICLE"):

print("%s Emergency Vehicle Detected by Traffic Control Room :

Overriding traffic signal to Green" % (getDateTime()))

sigLight = "GREEN"

else:

sigLight = ledStatus[i%3]["Status"]

turnSignal(sigLight)

print("%s Signal Light = %s" % (getDateTime(),sigLight))

def destroySetup():

GPIO.cleanup() # clean up Raspberry for next setup

# main program

def main():

global scriptName, trl\_dur

# Usage python trlight.py

scriptName = str(sys.argv[0])

if(len(sys.argv) < 2):

print("Error :: Usage trlight.py <no. of sec to run traffic light>")

exit(0)56 | P a g e

trl\_dur = int(sys.argv[1])

makeSetup()

runTrafficSignals()

destroySetup()

main() # call main script to run