PYTHON Code used to connect Visual Recognition Service to Virtual Capture for Obstacle detection (IBM CLOUD)

import numpy as np

import cv2

import time

from datetime import datetime

import os

#using Google Text-to-Speech

from gtts import gTTS

from pygame import mixer

import cv2

import json

from watson\_developer\_cloud import VisualRecognitionV3

import RPi.GPIO as GPIO

#Detecting the object using visual recognition service{IBM CLOUD)

visual\_recognition = VisualRecognitionV3(

'2018-03-19',

iam\_apikey='RLYVVSSFLFSjrr9pueeCaL4pTjEqs-7cLuCaG4njjqym')

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

#connect ultrasonic sensor to the following pins of the raspberrypi

TRIG = 23

ECHO = 24

print "Distance Mesurement In Progress"

GPIO.setup(TRIG, GPIO.OUT)

GPIO.setup(ECHO, GPIO.IN)

print "Waiting for Sensor Data"

def detect1():

with open('./image.jpg', 'rb') as images\_file:

classes = visual\_recognition.classify(

images\_file,

threshold='0.6').get\_result()

print(json.dumps(classes, indent=2))

r=classes['images'][0]['classifiers'][0]['classes'][0]['class']

r2=classes['images'][0]['classifiers'][0]['classes'][1]['class']

print r

print r2

tts = gTTS('hello'+ r+r2+'ahead')

tts.save('hello.mp3')

mixer.init()

mixer.music.load("hello.mp3")

mixer.music.play()

def detect():

model = app.public\_models.general\_model

response = model.predict\_by\_filename('/home/pi/clarify/image.jpg')

concepts = response['outputs'][0]['data']['concepts']

for concept in concepts:

if (concept['value']>0.97):

print(concept['name'], concept['value'])

tts = gTTS('hello'+ r+'ahead')

tts.save('hello.mp3')

mixer.init()

mixer.music.load("hello.mp3")

mixer.music.play()

return

cap = cv2.VideoCapture(0)

print 'camera is initialized'

while(True):

# Capture frame-by-frame

ret, frame = cap.read()

# do what you want with frame

# and then save to file

cv2.imwrite('/home/pi/clarify/image.jpg', frame)

#object distance measurement

'''GPIO.output(TRIG, False)

time.sleep(2)

GPIO.output(TRIG, True)

time.sleep(0.00001)

GPIO.output(TRIG, False)

while GPIO.input(ECHO)==0:

pulse\_start = time.time()

while GPIO.input(ECHO)==1:

pulse\_end = time.time()

pulse\_duration = pulse\_end - pulse\_start

# 34300=Distance/Time/2, 17150=Distance/Time, 17150 X Time = Distance (cm)

distance = pulse\_duration \* 17150

distance = round(distance,0)'''

distance=10 #Static distance value

if(distance<15):

detect1()

if cv2.waitKey(30) & 0xFF == ord('q'): # you can increase delay to 2 seconds here

break

# When everything done, release the capture

cap.release()

cv2.destroyAllWindows()