**­­­­­­A Smart Cane POC system**

**James Huang**

***james.huang@student.isb.bj.edu.cn***

1. **Introduction**

**1.1 Mission**

Nowadays, many smart devices are designed to be capable of many advanced abilities, such as Google navigation, Bluetooth connection to cell phones, etc. But few of them have tried applying computer vision to create an object identification and warning effect for the needy people. The two pictures below illustrates the smart cane:

**穿白色衣服的男人在路上

描述已自动生成**

**图表, 折线图

描述已自动生成**

**1.2 Conceptual Design**

**A picture containing tool, broom

Description automatically generated**

**1.3 Proof of Concept**

This project will focus on designing a smart cane POC prototype that can identify obstacles and provide warnings such as audio and humming warnings, as well text output for technical debugging/logging purpose.

**1.4 Success Criteria**

The POC product has to be able to

* + Real time video capture surrounding environment
  + Identify and recognize at least 2~3 types of objects within a certain range (let’s say 5 meters) in a few seconds; such recognized objects are labelled in the video window
  + Speak out what the object is with a speaker
  + Ultrasonic sensor can detect and measure objects that are within 4~5 meters; output the measured distance in terminal for debugging/technical purpose

1. **Design**

|  |  |
| --- | --- |
| **Object Recognition workflow** | **Object Distance Detection** |
| **图示  描述已自动生成** | **图示  描述已自动生成** |

**3. Implementation**

**3.1 Possible Hardware: (highlighted ones are chosen)**

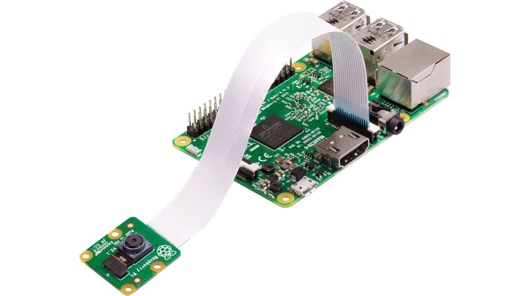
* *Sensor*
  + Environment/3D
    - The environment sensor can sense the surrounding and draw out a 3-dimensional map (usually used in auto-driving). However, the smart cane only needs to look at one direction within a few meters range.
    - Way too advance and expensive in this case.
  + Ultrasonic
    - Need this for object distance detection
  + Radar
    - Way too expensive
    - Too good for sensing a few meters distance
  + Lidar
    - Way too expensive
    - Too good for sensing a few meters distance
  + Raspberry Pi Camera
    - Camera is the most commonly used sensor for video capture
    - And this sensor goes with the controller well.
  + Human Presence
    - It uses heat to detect the presence of human, which means it cannot detect normal objects.
    - The school doesn’t provide
* *Actuator*
  + Motor for oscillation
    - The type of actuator I am most familiar with.
    - School provides
  + Actuators designed for oscillation purpose
    - If I can use a normal motor the school provides, why would I choose to buy this kind of actuator.
  + Raspberry Pi Speaker / Earphones
    - For sound effects/voice warning
* *Controls*
  + Candidate A: Microbit
    - Sometimes the area for coding is too small, thus it has limited flexibility, -- that’s why I decided to not use it as my controller.
    - Super-weak computing power, can not support tensforflow + image recognition
  + Candidate B: Arduino
    - I have not found much resource on this one
    - Computing power not sufficient for tensorflow + image recognition
  + Candidate C: Raspberry Pi 4B
    - This one has a rather powerful OS, supports various programming models, and possesses strongest computing power among the pack
    - For tensorflow lite + image recognition to work, it is advisable to have at least 4GB memory, so Pi 4B/8GB model is chosen

**3.2 Materials – key hardware**

* Raspberry Pi 4B/8GB

电子零件

中度可信度描述已自动生成

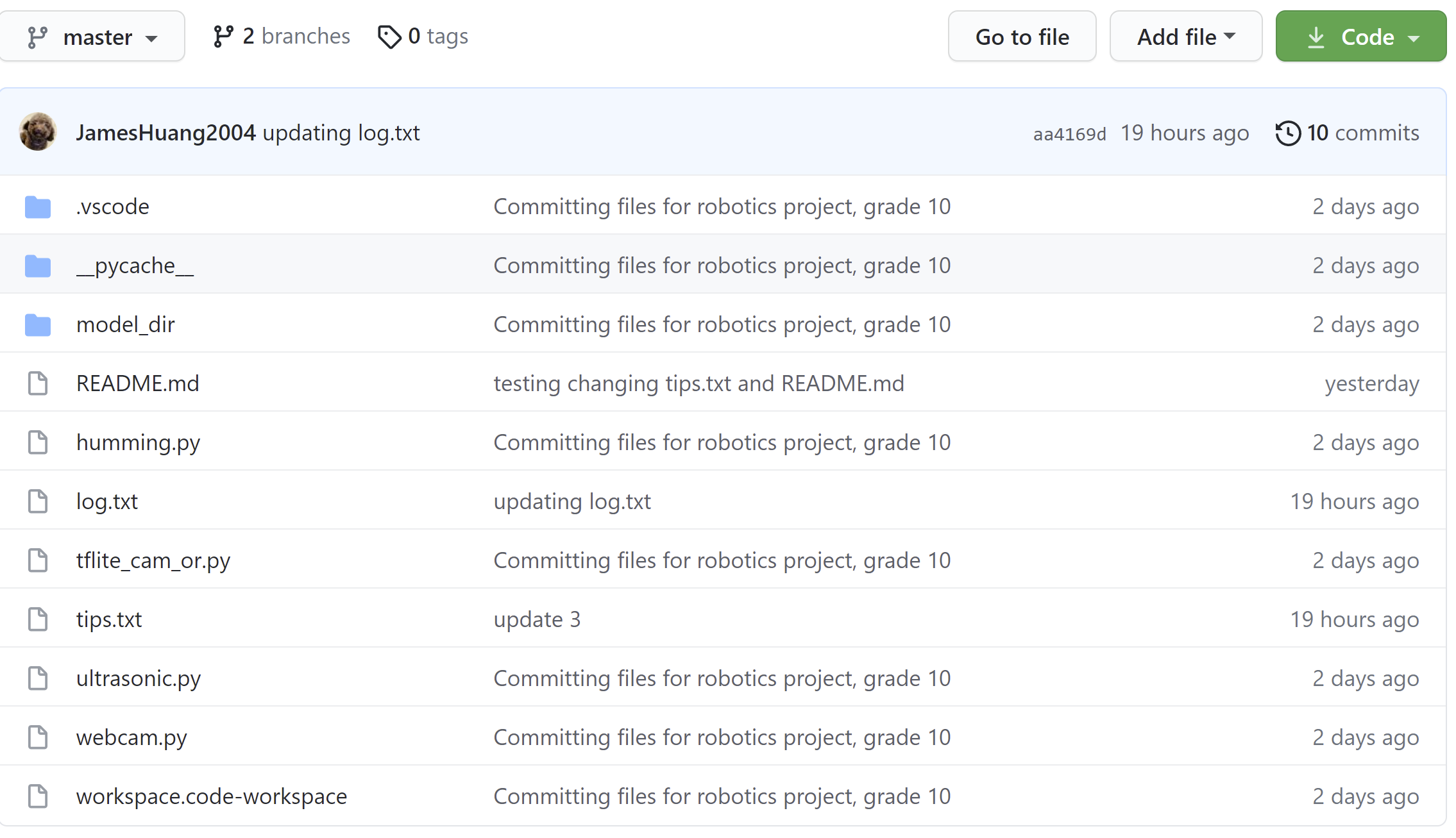
* Raspberry Pi Camera (8MP)
  + 
* Speaker
* Ultrasonic Sensor
* Working Computer (Lenovo Thinkpad + windows 10)
* Breadboard
* Female to Male wire \* 6
* 1 k resistor \* 3
* 32 GB TF card and USB-TFcard writer

**3.3 Software**

* Raspberry pi OS Imager (to create the latest full 32bit OS with recommended desktop software onto the TF card) <https://www.raspberrypi.org/software/>
* Git for source version control （built in pi OS）
* Tensorflow lite （v1.0.1/tflite\_runtime-2.5.0-cp37-cp37m-linux\_armv7l.whl）<https://github.com/google-coral/pycoral/releases/download/v1.0.1/tflite_runtime-2.5.0-cp37-cp37m-linux_armv7l.whl>
* OpenCV (version: 4.1.0.25)
* Winscp (for sFTP between windows machine and Pi) <https://winscp.net/eng/index.php>
* VNCViewer for windows (for remote desktop into Pi) <https://www.realvnc.com/en/connect/download/viewer/>
* Putty for windows (for SSH access from windows to Pi) <https://www.putty.org/>
* Visual Studio Code for Pi (enable in Pi); python extension for vs code
* Google object recognition model: <https://storage.googleapis.com/download.tensorflow.org/models/tflite/coco_ssd_mobilenet_v1_1.0_quant_2018_06_29.zip>
* espeakng module (for text-to-speech)

**3.4 Code**

[**http://github.com/JamesHuang2004/object\_detection**](http://github.com/JamesHuang2004/object_detection) **(private repo)**

****

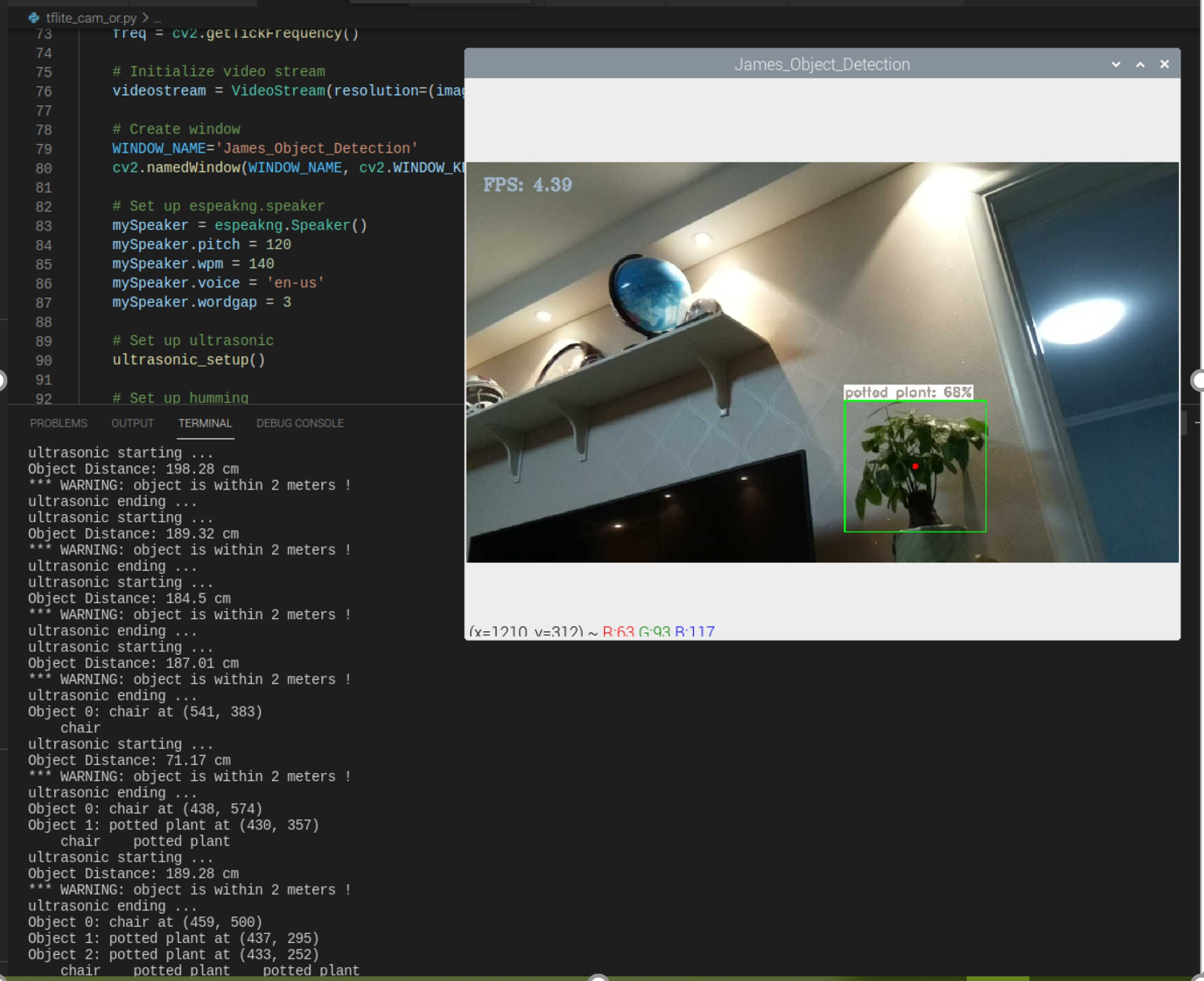
**3.5 Results:**

**桌子上放了游戏机

中度可信度描述已自动生成**

**The POC system has been able to achieve all the goals:**

* + Real time video capture surrounding environment
  + Identify and recognize at least 2~3 types of objects within a certain range in a real-time fashion; such recognized objects are labelled in the video window
  + Speak out what the object is
  + Ultrasonic sensor can detect and measure objects that are within 4~5 meters; output the measured distance in terminal for debugging/technical purpose



**3.6 Main Problems encountered:**

* **Setting up Raspberry Pi for programming and Connection to Computer**

Raspberry Pi 4B needed to be set up for programming, and this wasn’t in my planning (I changed raspberry pi zero to 4B for more computing power). So, I had to do some extensive research on how to do it from a Pi-board state. A lot of and research and learning along the way:

* + On Windows, Download the app “Imager”, use a USB-TF card writer to “BURN” “Raspberry Pi Full OS” into the TF/micro-SD card.
  + Insert the TF card into the PI-board tf card-slot, and boot up; with a monitor, a keyboard and a mouse, set up the timezone, language, wifi, etc
  + On Raspberry Pi, Click on the berry symbol on the top-left ---> go to preference ---> go to Raspberry Pi config ---> go to System ---> enable SSH, VNC, and Camera.
  + On Windows, Install “PUTTY” client from (<https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.74-installer.msi>)
  + On Windows, Install VNC Viewer from (<https://www.realvnc.com/en/connect/download/viewer/>)
  + On Raspberry Pi, Go to menu > preferences > recommended software > programming > install visual studio code
  + Set up python and pip：on Raspberry Pi, Go to terminal, Set up the right version of Python and Pip.
    - python3 --version
    - echo "alias python=/usr/bin/python3.7" >> ~/.bashrc
    - echo "alias pip=/usr/bin/pip3" >> ~/.bashrc
    - source ~/.bashrc
    - python –version
    - pip –version
  + On windows, install “winscp” from (<https://winscp.net/eng/index.php>)
  + On Raspberry Pi\, Go to terminal, enter:
    - ifconfig
    - And then try finding the internet ip.
    - These Ip can then be used when you are trying to connect your raspberry pi to your computer using VNC Viewer or Putty (SSH), as well as Winscp (file uploader for Raspberry Pi)
* **Setting up the environment & tools I need for video capturing, object detection and text-to-speech: tflite, opencv2, espeakng, etc.**

I had done a lot of research on google, stackoverflow and youtube – quite time-consuming. I found the deep conv2d net of google’s object model to be amazing, and the tensforflow framework to be very powerful.

* + - mkdir -p Src/Python/tflite
    - cd Src/Python/tflite
    - python -m pip install virtualenv
    - python -m venv tflite-env
    - source tflite-env/bin/activate
    - sudo apt -y install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev libavcodec-dev libavformat-dev libswscale-dev libv4l-dev libxvidcore-dev libx264-dev
    - sudo apt -y install qt4-dev-tools libatlas-base-dev libhdf5-103
    - python -m pip install opencv-contrib-python==4.1.0.25
    - uname -m ===> armv7l
    - python --version
    - python -m pip install <https://github.com/google-coral/pycoral/releases/download/v1.0.1/tflite_runtime-2.5.0-cp37-cp37m-linux_armv7l.whl>
    - Google object recognition model: <https://storage.googleapis.com/download.tensorflow.org/models/tflite/coco_ssd_mobilenet_v1_1.0_quant_2018_06_29.zip>
* **Ultrasonic sensor wiring up and programming using GPIO-**

I had a webpage that teaches me how to connect an Ultrasonic sensor to Raspberry Pi, but I wanted to learn more about the logic behind it. When wiring it using the breadboard, I have thought over and over to understand why 3 resistors are needed to reduce the voltage from 5v (ECHO) down to 3.3 v to the GPIO pin. It is a good learning experience 😊 I also had fun learning how it measures objects’ distance using the speed of sound and time lapse.

图片包含 游戏机, 电脑

描述已自动生成

Graphical user interface

Description automatically generatedIllustration of the ultrasonic sensor and Raspberry Pi 4b:

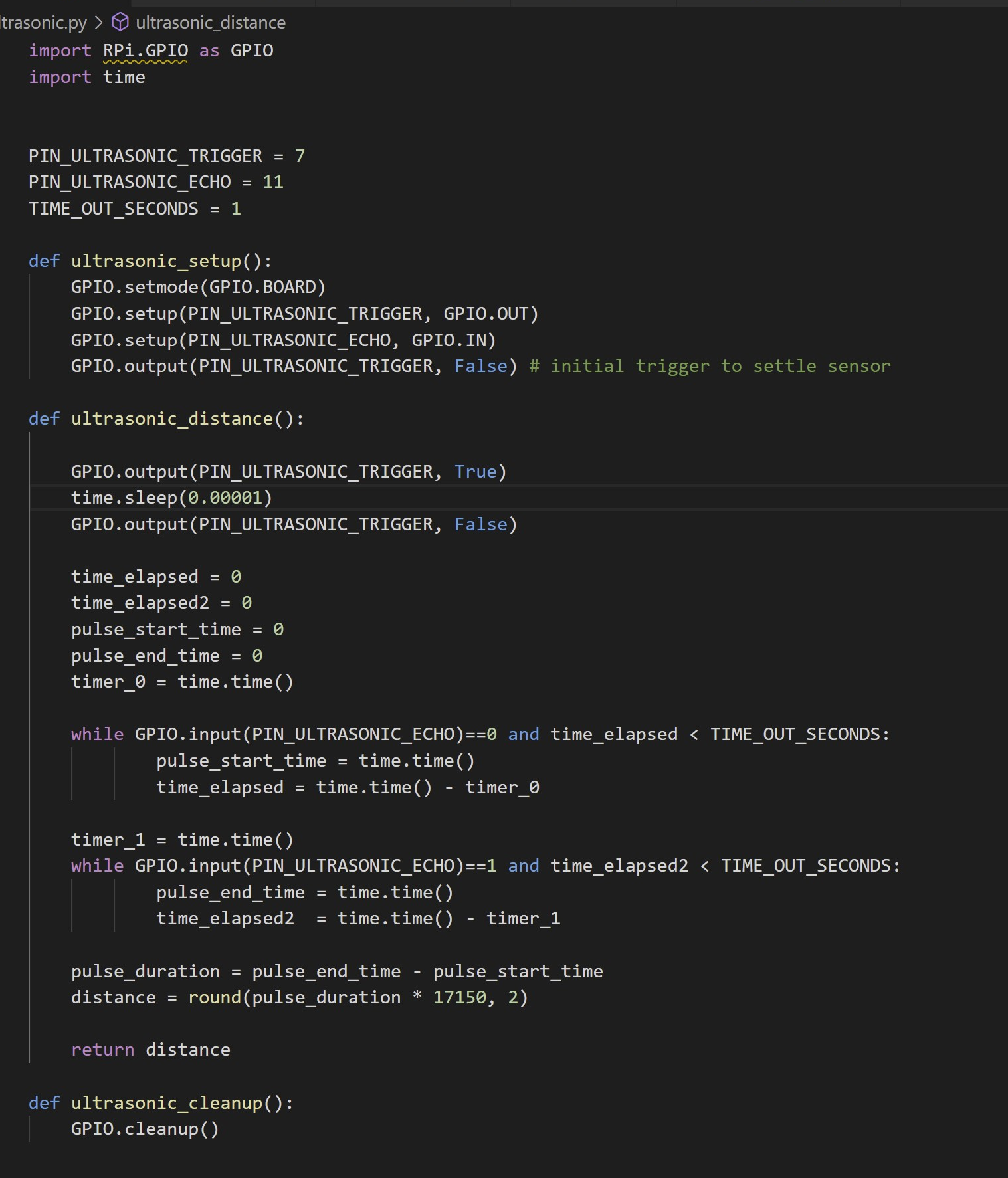
A screenshot of a computer

Description automatically generated with low confidence

These are the PIN layout for the ultrasonic sensor.

* 2 - VCC (5 V)
* 7 - TRG
* 11 - ECH
* 14, 20 - GND

And my code module for it:



**4. Future Planning**

Though the POC has achieved its initial goals, to make the smart cane a reality, there are still work to be done. I list a few key ones here:

* Build everything into an integrated and sleek smart cane, just like the sleek one in the mission section
* Currently the ultrasonic and video capture object recognition are two different pathways, it would be good to integrate them
* The code will need to run automatically when the cane is powered on
* The smart cane will need to be a battery-powered system
* Additional capabilities like GPS, radio, phone, SOS, etc are worth adding

**APPENDIX:**

**Milestones:**

* Research (End of April)
* Set up the Raspberry Pi (May 5th ~ May 10th)
  + Account
  + Connect to my computer using VNC or SSH
  + Import the code from an object detection project into the raspberry pi using Winscp
* Test and make adjustments for object detection (May 10 ~ May 12)
* Include a speaker (May 11th ~ May 14th)
  + Connect a speaker to Raspberry Pi using espeaker
  + Program the speaker to shout out detected objects
* Ultrasonic sensor (May 15 ~ May 17)
  + Connect an ultrasonic sensor to Raspberry Pi
    - Resistors
    - Female to Male Wires
    - Breadboard
* Connect a motor to Raspberry Pi (N/A)
  + Program the motor to vibrate when the ultrasonic sensor senses objects.
* Smart cane design (May 24 ~ May ~26)
* Finishing up my documentation (May 26 ~ May 28)