Raspberry Pi Voice Control

Tutorial by Andrew Oakley
Public Domain 23 Sept 2017
www.cotswoldjam.org

Speech Synthesis

Start the Terminal by clicking the Raspberry menu in the top-right corner, select Accessories, then click on Terminal. A black box will appear. Type the following then press Enter:

```
espeak "Hello world"
```

You should hear a retro-sounding voice, like that used by Professor Stephen Hawking. Try making it say different phrases.

Next, let's try using a better voice, used by Android smartphones:

```
pico2wave -w output.wav "Pico sounds much better than E speak"
aplay output.wav
```

Notice how `pico2wave` creates a `.wav` sound file, which we have to play separately with the `aplay` command.

Shall We Play A Game?

We can use speech synthesis to speak instead of printing text to the screen. Let's start by looking at a simple guess-the-number game that doesn't yet have speech.

From the Raspberry menu, select Programming, Python 3 (IDLE) . Click File menu, Open and pick the `python` directory, `voice` subdirectory and the file: `guess1.py`

Run the game program by clicking Run menu, Run module and play the game yourself.

Have a look at the program. See how it uses the `print` command to show text on the screen.

Close the program with File menu, Close. Now open the program `guess2.py` and run that.

You will hear instructions spoken instead of the text being shown on the screen.

Have a look at the `guess2.py` program. See how it creates a `say` function, that is used instead of the `print` command.

```
def say(input):
  os.system("pico2wave -w output.wav \"{}\" && aplay -q
output.wav".format(input))

say ("I am thinking of a number from 1 to 99.")
```

The `say` function looks complicated, but we don't need to worry about that, because we can now just use `say ("whatever we like")` in the program. If you prefer, you could change the `say` function to use the older `espeak` speech synthesiser instead:

```
def say(input):
  os.system("espeak \"{}\"".format(input))

say ("I am thinking of a number from 1 to 99.")
```

Speech Recognition

It's easy to synthesise (make up) speech, but it's much harder to recognise it - as in, have a computer listen to a human voice and understand the words being spoken.

The Raspberry Pi doesn't really have quite enough power to perform good speech recognition, especially not with hundreds of different languages and accents in the world. We could train it to recognise our own voice, but that could take days or weeks.

Instead, we can use the Raspberry Pi to record some speech, and send that recording to more powerful computers somewhere else on the Internet.

In our examples, we will use Google's Voice Application Programming Interface (API). This is quite complex to set up, but if you're attending one of our workshops, we have done this setup for you, so you should be able to use it straight away.

Recording Audio

The Raspberry Pi does not have a built-in microphone, nor does it have a microphone socket. We are using a cheap USB microphone to record audio, and we have already set it up for our workshops.

From the Terminal (Raspberry menu, Accessories, Terminal), enter the following:

```
cd /home/pi/python/voice
arecord -f cd -f S16_LE -c 1 -D hw:1,0 -d 10 test.wav
```

Press Enter then talk into the microphone for ten seconds (try a nursery rhyme). Next, play back the sound using:

```
aplay test.wav
```

You can change it to 5 seconds recording time by changing `-d 10` to `-d 5` .

Transcribing Audio

"Transcribing" means to write down in words, something that has been said out loud. We will send our recording to Google's computers and they'll send back the words.

```
python3 transcribe.py test.wav
```

It will take a few seconds, then it should show the words you spoke.

If it didn't understand your speech, try talking louder and nearer to the microphone. You might also need to ask other people in the room to be a bit quieter.

The Guessing Game in Full

Going back to the IDLE window, close the program with File menu, Close. Now open the program `guess3.py` and run that with Run menu, Run module.

This time, the game will not only talk to you, but also listen to your answers. It will only wait 3 seconds after each question, though, so be prompt.

Have a look at the program. Notice how it will say back anything that isn't a number:

```
  guess=transcribe.transcribe_file("recording.wav")
…
  if not guess.isdigit() :
    say ("I think you said: {}".format(guess))
    say ("That's not a number. Try again.")
    continue
```

You can have fun making it say things. Don’t be rude!

If 2 seconds is too short for you, you can change this timeout by changing this line:

```
def listen(filename,seconds=3):
```

Change `seconds=3` to `seconds=5` or whatever you want. Making it very long would make the game quite boring, though.

Notice how we say "I'm thinking…" when we've finished recording, but before we send the recording off to be transcribed. The transcribing can take a little while, so we need to let the player know that the computer has stopped recording but is still working.

```
  listen ("recording.wav")
  say ("I'm thinking...")
  guess=transcribe.transcribe_file("recording.wav")
```

Workshop setup

This section is for teachers who want to use this workshop independently of Cotswold Jam. As well as an internet connection, you will need a microphone. We used this cheap USB microphone: https://goo.gl/tj6mRX

In the .zip file that contained this document, you should also find other files which will need to be placed into the /home/pi/python/voice directory. You may need to make this folder if it doesn't exist.

You can install the speech synthesiser programs and configure the USB microphone from the Terminal with:

```
sudo apt update && sudo apt install -y espeak pico2wave
sudo cat <<! /etc/asound.conf
pcm.!default {
    type asym
    playback.pcm "hw:0,0"
    capture.pcm "hw:1,0"
}
```

Now run `alsamixer` and press F6 to select the USB mic, press F4 to select capture. Make sure it says "LR Capture" below the bar chart; use Space to toggle capture/mute on/off. Press the Esc key to exit.

Getting Google Cloud Speech API working is really a job for a Linux and API expert. You can find instructions at https://cloud.google.com/sdk/docs/quickstart-debian-ubuntu but be warned, it is not trivial. In particular, you will need to apply for a Google Cloud account and enable voice services, which although comes with US$300 free credit (at the time of writing), requires a bank card to set up.

It may be easier to just download our ready-made Raspbian image; look under http://www.cotswoldjam.org/tutorials although it is likely that the credit on the included Cotswold Jam API key will have long since expired. You will need to put your own Google Voice API key into the ~/python/voice directory to replace our `voice-workshop-key.json` file (which is called from `transcribe.py`). Note that our Raspbian image uses the headphone socket to play audio; you may need to right-click the desktop volume control to switch to HDMI output.