対話型オーディオ機器の製作

上西康晴　大石祥太郎
大橋敬斗　森定祥汰

１.研究概要

Google社が開発したVoice Kitと呼ばれるAIスピーカの製作を通じてpythonやIoTに関する知識を身につける為に研究を行った。

２.AI スピーカとは

Amazon AlexaやGoogleアシスタントなどに代表されるAIアシスタント機能を搭載したスピーカである。私たちの製作したAIスピーカではRaspberry Piを利用して入力された音声がクラウドサーバ上のGoogle Assistantというサービスで処理、変換される。例えば、「犬の鳴き声」と話しかけると犬の鳴き声が再生されるというものである。図１に仕組みを示す

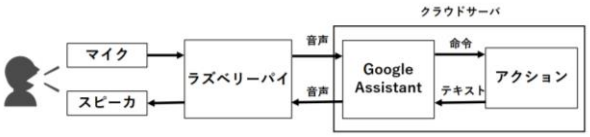

図１　AIスピーカの仕組み

３.Voice Kit の製作

ゼロからAIスピーカを製作するとなると時間的にも技術的にも厳しいため、今回はGoogle社のAIスピーカ作成キット「Voice Kit」を製作し、それを独自に改造して研究を進めることにした。

（１）Voice Kit について

Voice KitとはGoogle Assistant APIというサービスをRaspberry Piに組み込み、AIスピーカを製作することができるものである。今回私たちが使用した、Voice Kit V２の内容物は図２のようになっている。

図２　Voice Kitの内容物

左からMicro USBケーブル、内部フレーム用ボール紙、ボタンハーネスケーブル、スピーカ、Voice Bonnet、Raspberry Pi Zero WH、ボタン、外装用段ボール紙、OS書込済microSDカードとなっている。説明書通りに組み立てを行い、完成したものが図３である。

図３　完成イメージ

キットではフレームがボール紙であるため耐久性が低く、付属しているRaspberry Pi ZeroWHでは処理速度が非常に遅く作業効率が低下した。そのためフレームを３Dプリンタで製作し、Raspberry Pi ZeroWHをより高速なRaspberry Pi ３ MODEL Bに変更した。図４はRaspberry Piにスピーカと Voice Bonnetを取り付けたものである。

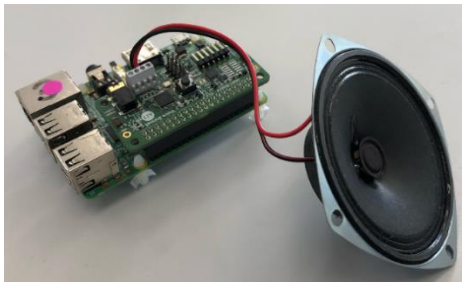

図４　本体

４.フレームの製作

強度を高めるためと自分達の技術向上のためにフリーの３DCAD ソフトである Fusion 360 を利用し、フレームを製作した。その手順を以下に示す。

（１）設計

フレームを製作する準備としてRaspberry Pi ３の各部分の寸法を測り、それを元に大まかな設計をした。次に、作成した設計図を元に Fusion360 を使って図面を書いた。
完成した設計図はUltimaker Cure を利用し、GCODE ファイルに変換した。

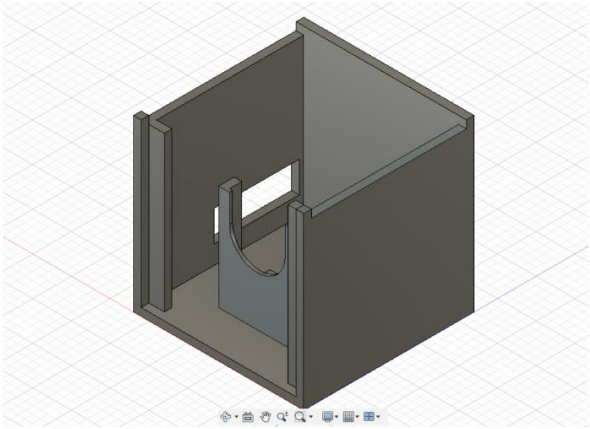

図５　フレームの設計図

（２） 3D プリンタで印刷

作成した GCODE ファイルを３Dプリンタに送りフレームの印刷を行った。フレームは２つのパーツに分け、２つのの印刷に約 20 時間必要となった。
完成したフレーム及び本体を搭載した写真を図６，７に示す。

図６　完成品の画像

図７　搭載した画像

５.システムセットアップ

製作した AI スピーカのアプリケーション等のセットアップを行った。以下に手順を示す。それぞれの工程は様々なサイトを参考にしながら行ったが、Google 等の UI アップデートや仕様変更等が毎月のように行われ、セットアップは予想よりもかなり難航した。

（１） Raspberry Pi の OS インストール

Raspberry Pi は Linux を使用した、シングルコンピュータである。使用するには OS のインストールから行わなければならない。
https://www.raspberrypi.org/software/
上記のサイトから Raspbian をインストールした。

（ａ）Google Assistant API の準備と認証

Google のサービスを受ける為、Google Assisant API の認証を行った。この作業は想像以上に困難で、１か月以上かかってしまった。その要点を以下に示す。Google cloud platform で、サービスを利用するための専用プロジェクトを新規作成し、「API とサービス」と「Google Assistant API」を有効化する。

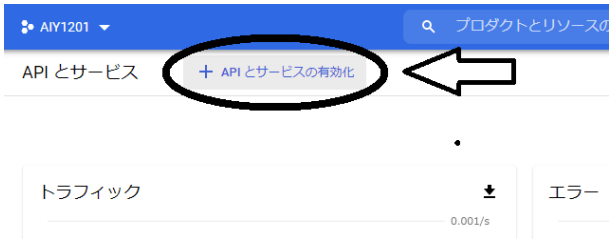

図８　GCP の API とサービス

APIをVoice Kitから呼び出すために必要な認証に使用するファイルを、「認証情報を作成」から「OAuthクライアントID」を選択し、任意のクライアント名「Voicekit1234」をつけて保存し、同意画面を作成する必要がある。この後ダウンロードアイコンをクリックしてファイルを保存する。

図９　GCP の認証画面

（ｂ）ファイルを raspberry Pi に複製

（ａ）で保存したファイル「client_secret_[略].apps.googleusercontent.com」をリネーム「assistant.json」して、「/home/pi」ディレクトリーにコピーしておく。作業用の PC に保存した場合は、USB フラッシュメモリーやネットワークを通じて Raspberry Pi にコピーする。

（ｃ）Google アカウントの設定

Google アカウントで外部から利用可能な履歴情報（アクティビティ）を制限している場合は、動作に必要なアクティビティが不足する。「アクティビティ管理」画面で Google アカウントによる履歴管理を有効にし、Raspberry Pi で使用しているアカウントにログインする。

（ｄ）バグファイルの修正

作業中に発生するエラーを解決するために Python ファイルを解析して、エラーの原因となるファイルの修正を行った。

「/home/pi/AIY-projects-python/src/aiy/assistant」内にある「auth_helpers.py」の 75 行目「**webbrowser.register(‘chomiumu-broeser’,None,webbrowser.Chrome(‘chromium-browser’),-1)**」の「**,-1**」を Text Editor などで削除する。

（３）デモプログラムの実行

この時点で AI スピーカを喋らせる準備はできたため、デモプログラムを実行して動作を確認した。コマンド実行画面で打ち込むコマンドを以下に示す。

「**sudo cp/home/pi/assistant.json /root**」
「**sudo su**」（「ls」コマンドで assistant.json があることを確認）
「**exit**」
「**cd/home/pi/AIY-projects-python/src/examples/voice**」（「ls」で確認）
「**sudo python3 assistant_grpc_demo.py**」
音声アシスタントを動作させ完成を確認。
※通信環境が悪いとエラーが発生する。

６.応用

ここまでで AI スピーカとしては十分な物となったが、ただ喋るだけでは物足りないと思ったため、他の機能の追加に挑戦した。

（１）IoT の搭載

IoT 搭載の前に、GPIO ピンの動作確認の為 LED を点滅させるデモプログラムを作成した。

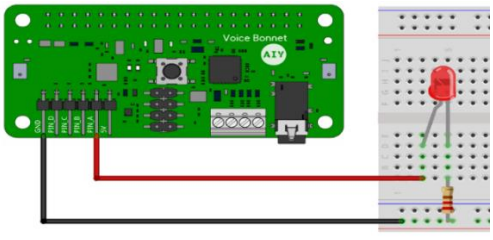

図 10　LEDを接続した図

図 10 のようにVoice Bonnetと LEDに配線して、デモスクリプトの中のled_chaser.pyを以下のように書き換えた。

```
from time import sleep
from gpiozero import LED
from aiy.pins import (PIN_A, PIN_B,
PIN_C, PIN_D)
leds = (LED(PIN_A), LED(PIN_B),
LED(PIN_C), LED(PIN_D))
while True:
    for led in leds:
        led.on()
        sleep(0.5)
        led.off()
```

本来ならばこれで LED が点滅するはずだったが、LED を変えても点滅しなかった。様々な方法を試した結果、GPIO は使用不可と考え、IoT 実装は諦めることになった。

（２）自動起動

私たちは、Raspberry Pi の電源を入れると自動でプログラムを動かすための自動起動機能を実装した。自動起動を行うために Raspberry pi の「/etc/rc.local」にシェルスクリプトファイル（例: test.sh）を作成した。

内容は、「cd/home/pi/AIY-projects-python/src/examples/voice」

「sudo python3 assistant_grpc_demo.py」に手を加えて、

#!/bin/sh

「sudo python3 /home/pi/AIY-projects-python/src/examples/voice assistant_grpc_demo.py」の形でファイルを作成した。

次に、ネットワークの設定を変えるためにコマンド「sudo raspi-config」を打ち込むと図 11 のような環境設定の UI が起動する。

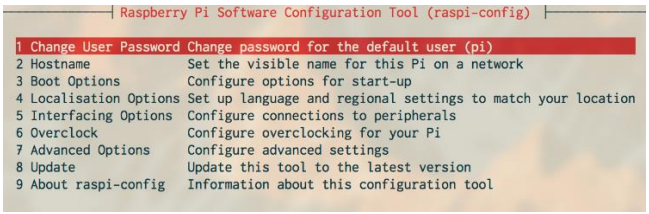

図 11　環境設定の UI

図中５の Interfacing option でネットワークに繋がったとき起動するという項目を選択することで、自動起動の実装に成功した。

７．研究のまとめ

今回の課題研究を通して、私たちはモノづくりの難しさを実感することができた。必要なサービスやファイルの欠損など、当初は想定してなかった問題が序盤から発生し予定の進行具合との差も大きかったが、その分初めて使う python や Raspberry Pi などについての知識や使用法などをより多く学ぶことができた。また、インターネット上にある多くの情報の中から今必要なものを探すということを繰り返していくことで打開策を見つけ進めていく力も身に付けることができた。

８．参考文献

・Voice Kit ユーザーガイド
https://aiyprojects.withgoogle.com/voice/