音声によりロボットを制御する方法について述べる

仕組み：

1）Julius を用いて音声認識を行い、
2）その結果をプロセス間通信により Robot の制御プログラムに渡し、
3）Robot の制御プログラムは受信した命令により、Robot を動かす

なお、以下では、Robot は Raspberry Pi で制御されていると仮定しているが、ARDrone などでも同様である。もっともロボットに通信する制御コマンドと、ロボットにある通信・制御プログラムが異なるので、「それに合わせた改変」は必要である(後述する).

段取り：

仕組みの項で書いたように、3 つのプロセスが協調して一つの仕事が行われる。そのため一つ一つのプロセスがちゃんと動くことを十分テストしないと、動かないことが多い、また動かない時にはどれがトラブルになっているがわからず困ってしまう。

そこで、以下の手順を踏む

(1) Julius を単独に動かし、認識できることを確認、
(2) PC のプログラムと Julius を連動させ、プログラムから Julius の認識結果を受け取り、プログラムか音声によって適切な出力ができる（画面に『ロボットに送る制御コマンド相当のもの』を出力する）ことを確認、
(3) Robot の制御プログラムを単独で動かし、Robot が指定通り動くことを確認、
(4) PC の通信プログラム(入力はキー操作など容易なもの)と Robot 間で通信が行えることを確認、
(5) PC の通信プログラムと Robot の制御プログラムを連動させ、PC からの入力により、適切に Robot が動くことを確認、
(6) Julius、通信プログラム、Robot の制御プログラムを連動させ、音声によって適切に Robot が動くことを確認

準備:

(A) Julius

1) http://julius.sourceforge.jp/
から、「文法認識キット」をダウンロードし、インストールしてください
なお、これには Linux 版と Windows 版の両方があります。
2) PC で Rapiro フォルダをゼミページからダウンロードして
(Julius の)grammar-kit-v4.2-win フォルダの SampleGrammars の下に置く
注意: Rapiro フォルダは、ゼミページでは rapiro.zip となっている。この

ままでは使えないので必ず『展開』(ダブルクリック、もしくはマウスで右ボタン、「自動解凍する」を選ぶ)

これは Rapiro を動かすための『単語』と「文パターン」を設定してある。用途に応じて改変すること。なお、単語や文パターンを変えるには(文字コードに注意する, Windows の場合は Shift-JIS、Linux の場合は Unicode がよいだろう)

(a) 単語帳 --- 拡張子が voca のファイルを編集. 文法カテゴリごとに分けて書く。単語表記と音素とを以下のように書く

```
%HOUKOU                ←  文法カテゴリ名
前    m a e            ←  単語  と  その音素列
前    m a e n i        ←  この後の処理を考慮して単語を登録
```

(b) 文法 --- 文(S)は B から始まり E で終わる、システムが受理する文や文法カテゴリのパタンを書く。単語帳で指定した文法カテゴリを使用する。例:

```
S  : B SONOTA NOISE HOUKOU NOISE HUKUSI NOISE KYORI NOISE UGOKI E
S  : B SONOTA NOISE HOUKOU NOISE KYORI NOISE UGOKI E
KYORI      : KAZU ZYOSUSI          ←  上で使用した KYORI の定義
KYORI      : CHOTTO
```

(c) 拡張子 dfa, term, dict のファイルを作成

手で作ることもできるが、Julius では mkfda.pl という Perl 言語によるプログラムが用意されているので、 perl mkfda.pl rapiro のようにすれば (そして、voca と grammar ファイルの名前がみな rapiro.voca, rapiro.grammar となっていれば) 、これらのファイルを作ってくれる

(d) これらのファイルをみな grammar-kit-v4.2-win/SampleGrammars の下に適当なフォルダを作って(例えば Rapiro)、そこに置く

(e) 以下の内容の rapiro.conf ファイルを作っておく (意味を簡単にいえば、これらは julius を動かすための環境を指定するもの、文法として rapiro.dfa を、その他の環境の設定には hmm_ptm.jconf の内容を使う、入力はマイクから、など)

```
-gram rapiro
-C ../../hmm_ptm.jconf
-input mic
-rejectshort 800
-lv 1800
```

3) マイクを PC に接続し、Windows の場合、Rapiro フォルダの Test.bat をダブルクリックし、

「右曲がれ」「左曲がれ』「前行け」「後下がれ」「とまれ」と発声して、

これらの言葉が認識されることを確認する

注意: マイクの調整を必ずすること。認識率が低い場合は、マイクの調整が

ちゃんとなされていないことが原因であることが多い。

参考: **マイクの準備**

Windows:　「コントロールパネル」で

1 サウンドの「録音」でマイクがあるかチェック

2 「音声認識」でマイク録音のレベルをチェック

Linux

Rapiro 資料を参照(日経 Linux 2014 年 6 月号 pp.69-75)

arecord -l　　で録音デバイスの確認

arecord -D plughw:,0 -f S6_LE -r 16000 test.wav　　で録音テスト

amixer -c 1 sset 'Mic' 10%　　でマイク音量セット

play　test.wav　　(sox 必要)録音ファイルの確認

Julius 起動の前に

export ALSADEV=plughw:1,0　　が必要

julius -C main.jconf -C am-gmm.jconf -demo　を実行してみる

(B) PC のプログラムと Julius の連動

これは同じ PC 上で動く二つのプロセスの間のプロセス間通信である。Julius は

julius -C rapiro.jconf -module

というように、-module パラメタを指定すると、モジュールモード、つまり他のプロセスと通信して動くモードになる。

この目的のためのプログラムとしては次のようなものがある:

```
#!/usr/bin/env python
import socket, re

def catchSent(clientsock):
  flag=False
  answer=[]
  while True:
    recv_data = clientsock.recv(512)
    # print ">>",recv_data
    if re.search('<RECOGOUT>',recv_data):
       flag=True
    elif (flag==True):
       tmp = re.findall('<WHYPO WORD="[^"]+"',recv_data)
       # print ">>> ",tmp
       for one in tmp:
```

```
            answer.append(one.split('"')[1])
        if re.search('</RECOGOUT>',recv_data):
            return(answer)

Host = 'localhost'
Port = 10500                        # Julius のポート番号

clientsock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
clientsock.connect((Host, Port))

while True:
     sent = catchSent(clientsock)
     print(sent)
```

なお、上記のプログラムは(見て分かるように)無限ループするので、ctl-C で終了させる。また再開させるには Julius も停止→起動させたほうがよいだろう。

(C) PC の通信プログラムとロボットとの通信

Julius とは異なり、異なるホストで走るプロセス間の通信である。Julius との違いは、ホスト間で通信ができていなければならない、ということである。その条件さえクリアされていれば、Windows、Linux, Arduino, Android などいろいろなホストで動くプロセス同士を協調させられる。

ここでは、サーバークライアント方式での通信を考える。ロボットの場合、PC がロボットに注文を出す方なのでクライアント(お客)、ロボットはその注文にしたがって仕事をするのでサーバーとなる(問題: Julius と PC のプログラムの場合も、サーバークライアント方式であった。さて、どちらがサーバーでどちらがクライアントであったか？)

サーバーとなるプロセスでは、一見ややこしいプログラムを書く必要がある。少なくとも、そのプロセスが走るホストの IP アドレスと、ポート番号(プロセス同士が通信するための「連絡番号」、1000 よりも大きな値を指定)が必要である。また予定するお客の数や、お客の注文を待ち受けるためのプログラムを書く必要がある。ここらへんは、既に動いているプログラムを見て真似して欲しい(ちゃんと勉強することは歓迎する)。ゼミのページに roboTest.py と roboServer.py があるので、Raspberry Pi にダウンロードしたのち、IP アドレスを修正して動かすことができる(もしくは PC でダウンロードして修正を行い、その後で winscp により(IP アドレスを修正した)roboTest.py と roboServer.py を Raspberry Pi にファイル転送する)。なお、roboTest.py は送られて

きたメッセージを単に表示するもの、roboServer.py はraspirobotboardモジュールと連動し、送られてきたメッセージに従ってロボットを制御するプログラムである(つまり、それが実装されていないと動かない)。

クライアントになるプログラムは、サーバープログラムと比べるととても簡単で、Juliusとの通信プログラムとほぼ同様(ホストとポートが異なる、Juliusからの入力に対しロボットに出力する、という違いがあるが)。

(D) 3つのプロセスの連動(以降は書きかけ)

作業: これからは、ロボット(Raspberry Pi)にはキーボード、マウス、モニタを接続せずに使う。一旦Raspberry Piをシャットダウンし、モニタなどを外す。そして立ち上げ直すこと。このロボットにアクセスするには Wifiを経由してPCからTeraTermでログインすること. この順番にやること (その理由を考えよ)

(1) PCの作業: Juliusをモジュールモードで動かす (この意味がわからない人はJuliusのマニュアルを読むこと) 。ここで使うシステムはgrammar-kit-v4.2-winフォルダのSampleGrammarsに置いた Rapiro ファイル一式---この Rapiro.bat をダブルクリックすれば、Juliusがモジュールモードで動くはず

(2) ソケット通信の参考プログラムを使用: サーバーとクライアントプログラムのIPのアドレスを修正し、それぞれを実行する。それにより、ソケット通信が可能であることを確認する(時間がなければ省略してもよい)

(3) TeraTermにより、ロボット(Raspberry Pi)にログインしておく。以降で「ロボット側の作業」と書いてあるのは、このTeraTermでの作業を意味する

(4) ロボット側の作業: sudo python roboTest.py を実行(roboTest.pyが異常終了しないことを確認する---異常終了した場合は、IPアドレスの修正ができていない可能性がある)

(5) PCの作業: Test.py を実行(予めIPアドレスを修正しておくこと)
Test.pyの実行画面で『前』「後ろ」「右」「左」『止まれ』などを文字入力する (入力したらEnterキーを押すこと) 。ソケット通信によりPCからデータが送られる。その結果、Raspberry Pi (にログインしたTeraTerm上) にＦＢＲＬＳが表示されることを確認。PCで balse と文字入力すれば、Raspberry PiのRoboTest.pyが終了するはず.

PCのTest.pyを停止させておくこと。

(6) ロボット側の作業: Raspberry Piのroboテスト.pyを停止させる (balse入力で停止しなかったら、Ctrl-C (コントロールキーを押しつつCを押す)

で停止させる。次に、roboServer.py を次のように実行する: sudo python roboServer.py

(7) PC の作業: Test.py を立ち上げ直し、「前』「後ろ」「右」「左」『止まれ』などを文字入力し、ロボットがそれに反応して動くことを確認

注意: モーターが動くので、あらかじめロボットを持ち上げておき、モーターの動きの方向を確認すること。

(8)以上ができたら、PC の Test.py とロボットの roboServer.py を終了させる。

（ここまでが準備）

まずロボット側の作業: ロボットの robotServer.py を立ち上げ直し、その後で PC の作業: PC 上の Trial2.py を実行する。
そしてマイクから「右曲がれ」『左曲がれ』「前行け」『後下がれ』「とまれ」などを吹き込み、それに応じてロボットが動くことを確認する
(Julius が動いていることが条件。止まっている場合は、Rapiro.bat を再度クリックする)