Lab1 Report

> Course: 2021 HCI Lab1, School of Software Engineering, Tongji Univ.
>
> Automatic Speech Recognition
>
> Name：沈益立
>
> Student Number: 1851009

1. Introduction

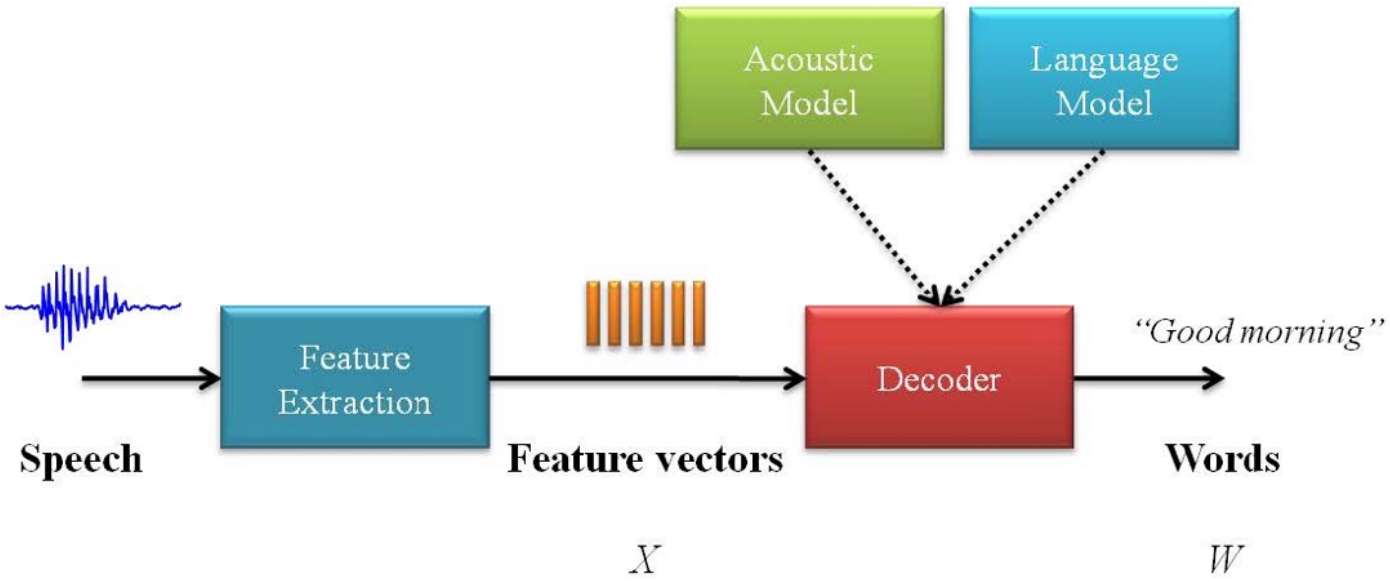
This project is a voice assistant software running in the Mac system, based on Speech Recognition System, CMU Sphinx Speech Recognition Engine and Baidu Speech Recognition API, using Qt to make the UI graphic.

This project accepts the users' command through speech and recognize their requirements and call several operating system APIs to implement several functions.

The project is developed and tested on MacOS. The system commands aren't compatible with windows or other OS, so do not try run it on them.

2. Architecture

2.1 Speech Recognition



As shown above, speech recognition is basically assembled like this. The model behind the API extracts several features, decode them and then convert them into words through a series of algorithms.

3. GUI

The project's user interface is based on PyQt, which is a great open source graphic UI engine. And this program is organized as a multi-thread project. The main thread is allocated by GUI and the recording function allocates another and the speech recognition allocates the third sub-thread.

Thus, two labels are added to represent the two other threads. One represents the program is currently listening to the user, another shows the result of recognition of what user said.

Modifications:

1. Translated the whole project to make it show Chinese words.
2. Created the label: `label6` to show the interaction tips and current status of listening.
3. Eliminated extra labels and merged them into one label `label4` to give tips to user.
4. Created a label: `label5` to show the result of the recognition.
5. Changed the colors.
6. Empower the voicefig gif with the button functions, and user is able to click the gif and speak to the system.

3.1 Code

The modified parts are shown as follows.

```
class MyQLabel(QtWidgets.QLabel):
    # make the gif clickable
    button_clicked_signal = QtCore.pyqtSignal()

    def __init__(self, parent=None):
        super(MyQLabel, self).__init__(parent)

    def mouseReleaseEvent(self, QMouseEvent):
        self.button_clicked_signal.emit()

    # 可在外部与槽函数连接
    def connect_customized_slot(self, func):
        self.button_clicked_signal.connect(func)
class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
       # =============
        # Other code...
        # =============
        self.label_4 = QtWidgets.QLabel(self.centralwidget)
        self.label_4.setGeometry(QtCore.QRect(60, 220, 201, 150))
        font = QtGui.QFont()
        font.setFamily("Calibri")
        font.setPointSize(14)
        self.label_4.setFont(font)
```

```
        self.label_4.setStyleSheet("color: rgb(0, 117, 210);")
        self.label_4.setWordWrap(True)
        self.label_4.setObjectName("label_4")



        # status label created to show the current status
        self.label_5 = QtWidgets.QLabel(self.centralwidget)
        font = QtGui.QFont()
        font.setFamily("Calibri")
        font.setPointSize(14)
        self.label_5.setFont(font)
        self.label_5.setStyleSheet("color: #ffffff;")
        self.label_5.setWordWrap(True)
        self.label_5.setObjectName("label_5")
        self.label_5.setGeometry(QtCore.QRect(60, 380, 201, 200))
        self.label_5.setAlignment(QtCore.Qt.AlignTop)
        font = QtGui.QFont()
        font.setFamily("Calibri")
        font.setPointSize(14)
        self.label_6 = QtWidgets.QLabel(self.centralwidget)
        self.label_6.setGeometry(QtCore.QRect(60, 160, 201, 51))
        self.label_6.setFont(font)
        self.label_6.setStyleSheet("color: #948d8d;")
        self.label_6.setWordWrap(True)
        self.label_6.setObjectName("label_6")

        # =============
        # Other code...
        # =============


    def update_label_5(self, text):
        _translate = QtCore.QCoreApplication.translate
        self.label_5.setText(_translate("MainWindow", text))

    def update_label_6(self, text):
        _translate = QtCore.QCoreApplication.translate
        self.label_6.setText(_translate("MainWindow", text))
    def retranslateUi(self, MainWindow):
        # =============
        # Other code...
        # =============
        self.label_4.setText(_translate("MainWindow", "你好，这是一个中文语音助手。\n\n你可以这样问我：\n1. 说\"播放\"以播放音乐\n" + "2. 说\"打开文档\"以查看记事本文档\n" +
                                                      "3. 说\"看图\"以查看示例图片\n" +
                                                        "4. 说\"看视频\"以查看抖音小视频\n"))
        # self.label_5.setText(_translate("MainWindow", "I\'m hearing..."))
```

```
        self.label_6.setText(_translate("MainWindow", "点击按钮来和我对话。"))
```

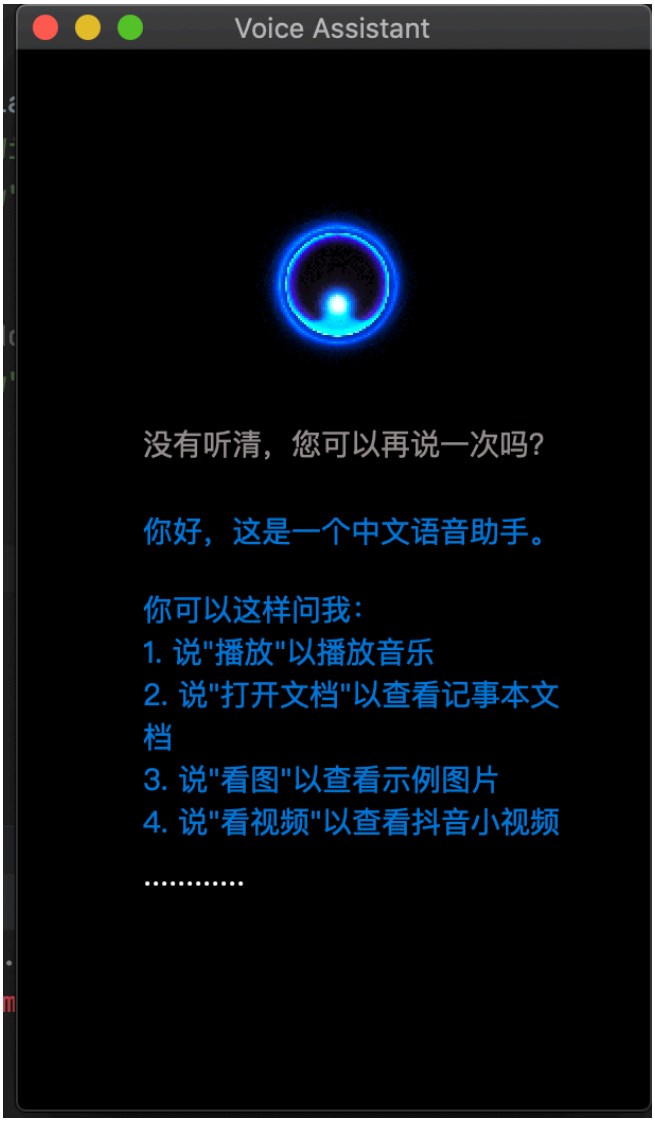
3.2 Appearence

3.2.1 Launch

3.2.2 Listening to user

3.2.3 Got What User Said

3.2.4 Heard nothing from user



4 Functions and Code

4.1 Record Speech

I used the `Speech_Recognition` and `pyaudio` to call the device to record through microphone. The api is equipped with the ablity to reduce the noise and convert it into numpy array and then save it locally. So I used these functions to record the user's voice and save it in `tmp.wav`.

4.1.1 Code

```
def speech_interaction():
    global t
    if t.is_alive():

        return

    print(threading.current_thread())
    r = sr.Recognizer()
    try:
        mic = sr.Microphone()
```

```
        t = threading.Thread(target=get_speech_from_mic, args=[r, mic])
        t.start()
        application.ui.update_label_6('你说，我在听')
        # t.join()
        i = 0
        t2 = threading.Thread(target=update_dots)
        t2.start()

    except:
        application.ui.update_label_6('请检查麦克风是否工作')
def get_speech_from_mic(r, mic):
    # getting wav file from mic
    with mic as source:
        r.adjust_for_ambient_noise(source)
        print('start')

        try:
            audio = r.listen(source, timeout=5)
            wav_data = audio.get_wav_data()
            wf = wave.open('tmp.wav', 'wb')  # type: wave
            wf.setnchannels(1)

            wf.setsampwidth(2)
            wf.setframerate(8000)
            wf.writeframes(np.array(wav_data).tostring())
            wf.close()
            get_text_from_api()
            application.ui.update_label_6('点击按钮来和我对话。')
        except:
            application.ui.update_label_6('没有听清，您可以再说一次吗？')
```

4.2 Recognize Speech

The recognization of speech is implemented by Baidu API, which provides a stronger and more corrective recognition of user's speech. It requires the API key and Secure Key, which was provided when applying for a Baidu developer account and authorization of using such APIs.

4.2.1 Code

```
def get_text_from_api():
    try:
        print('start')
        j = baidu_api.get_json()
        application.ui.update_label_5(j['result'][0])
        keyword = j['result'][0]
        if keyword[:2] == '播放':
            c.play_music()
        elif keyword[:4] == '打开文档':
```

```
            c.open_text()
        elif keyword[:2] == '看图':
            c.open_img()

        elif keyword[:3] == '看视频':
            c.play_video()
    except:
        application.ui.update_label_5('Something wrong happens.')
def get_json():
    token = fetch_token()

    speech_data = []
    with open(AUDIO_FILE, 'rb') as speech_file:
        speech_data = speech_file.read()

    length = len(speech_data)
    if length == 0:
        raise DemoError('file %s length read 0 bytes' % AUDIO_FILE)
    speech = base64.b64encode(speech_data)
    if (IS_PY3):
        speech = str(speech, 'utf-8')
    params = {'dev_pid': DEV_PID,
              #"lm_id" : LM_ID,    #测试自训练平台开启此项
              'format': FORMAT,
              'rate': RATE,
              'token': token,
              'cuid': CUID,
              'channel': 1,
              'speech': speech,
              'len': length
              }
    post_data = json.dumps(params, sort_keys=False)
    # print post_data
    req = Request(ASR_URL, post_data.encode('utf-8'))
    req.add_header('Content-Type', 'application/json')
    try:
        begin = timer()
        f = urlopen(req)
        result_str = f.read()
        f = urlopen(req)
        j = json.load(f)
        # print(j['result'][0])
    except URLError as err:
        print('asr http response http code : ' + str(err.code))
        result_str = err.read()

    if (IS_PY3):
        result_str = str(result_str, 'utf-8')
```

```
    with open("result.txt","w") as of:
        of.write(result_str)
    return j
```

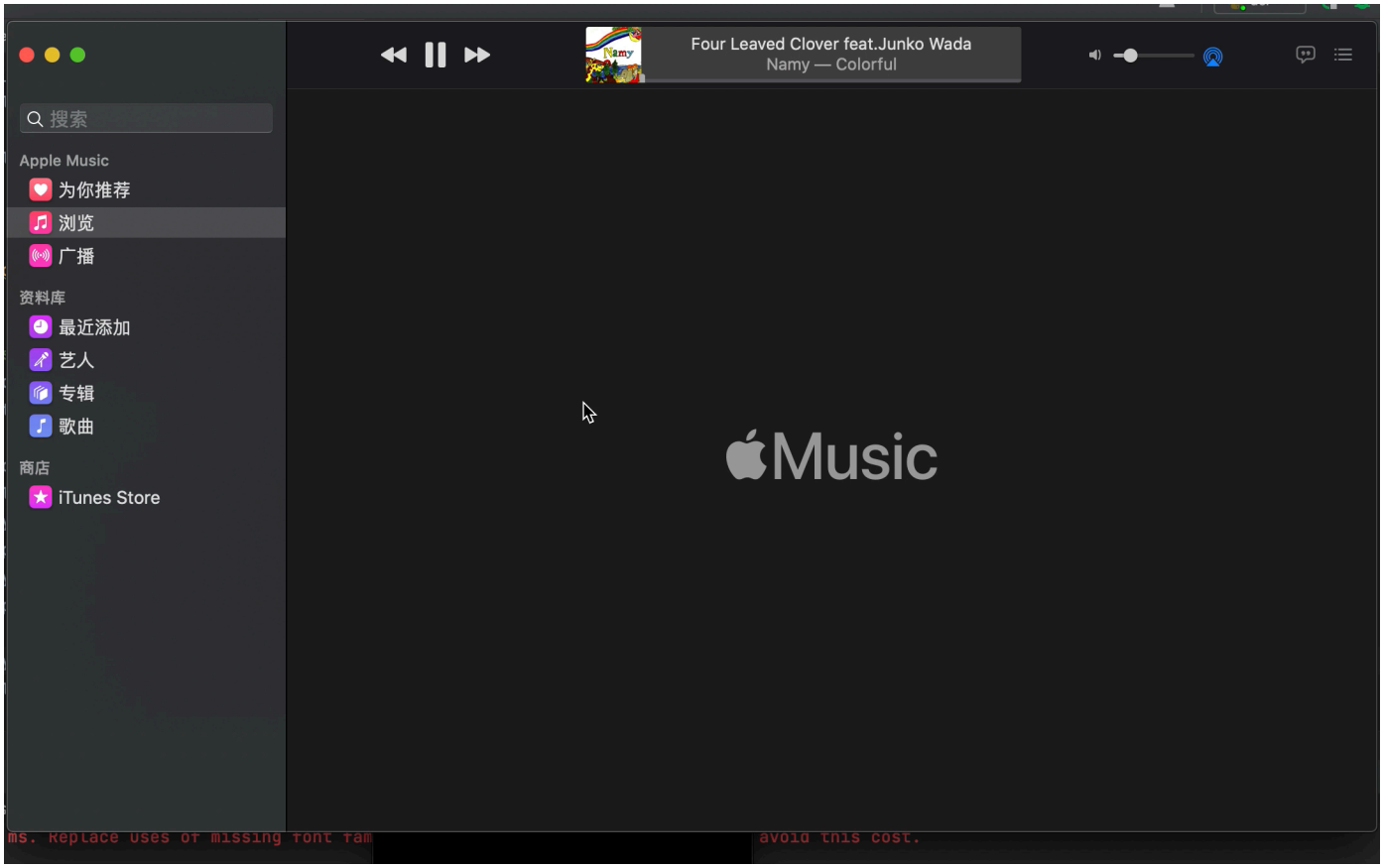
4.3 Play Music

Launch the default music player of MacOS and play the instanced song `music.mp3` for user.

4.3.1 Code

```
def play_music(self):
    if sys.platform == 'darwin':
        os.system('open ' + 'music.mp3')
```

4.3.2 Screenshot



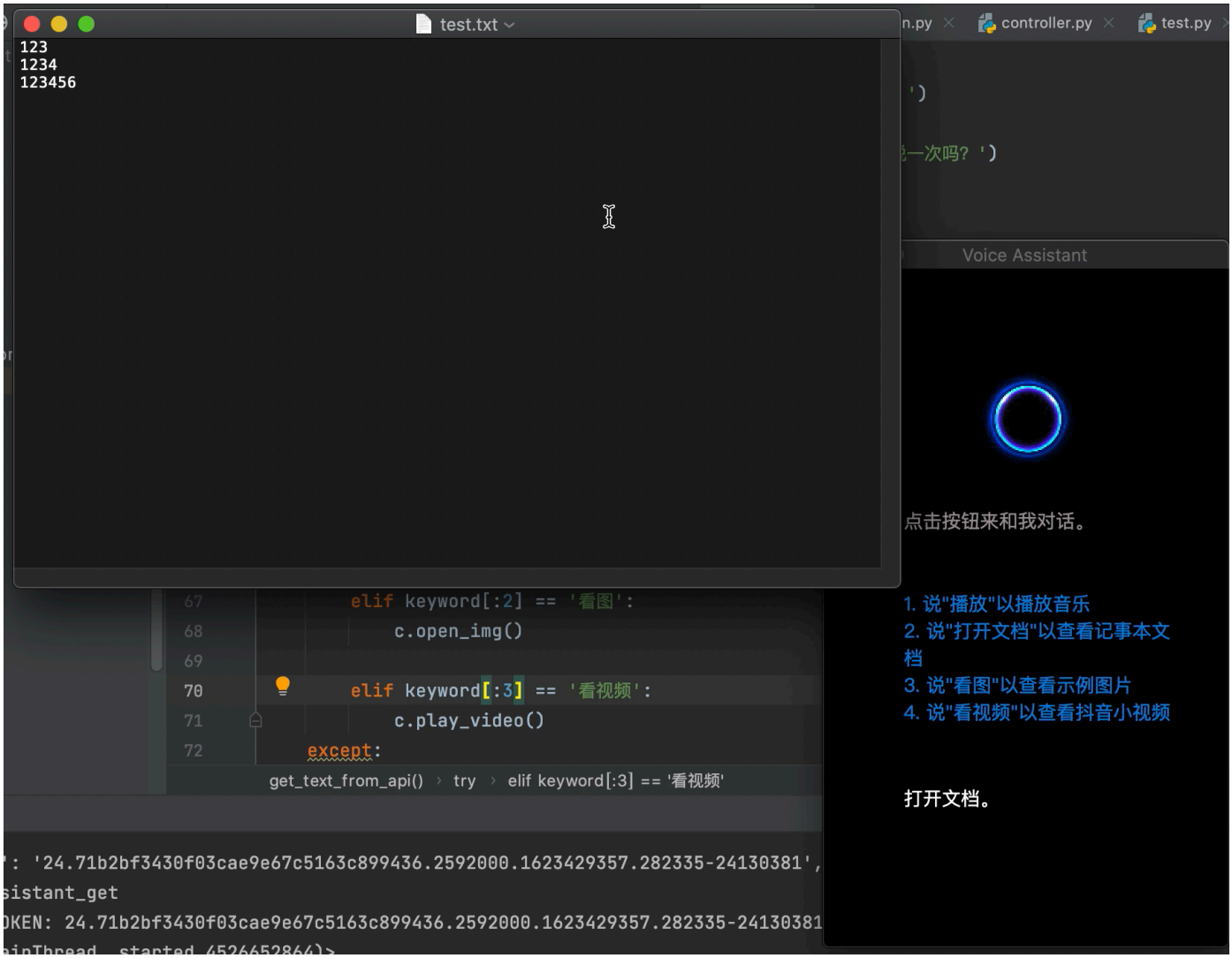
4.4 Open Files

Launch the default text editor of MacOS and show file `text.txt` for user.

4.4.1 Code

```
def open_text(self):
    if sys.platform == 'darwin':
        os.system('open ' + 'test.txt')
```

4.4.2 Screenshot



4.5 Open Photos

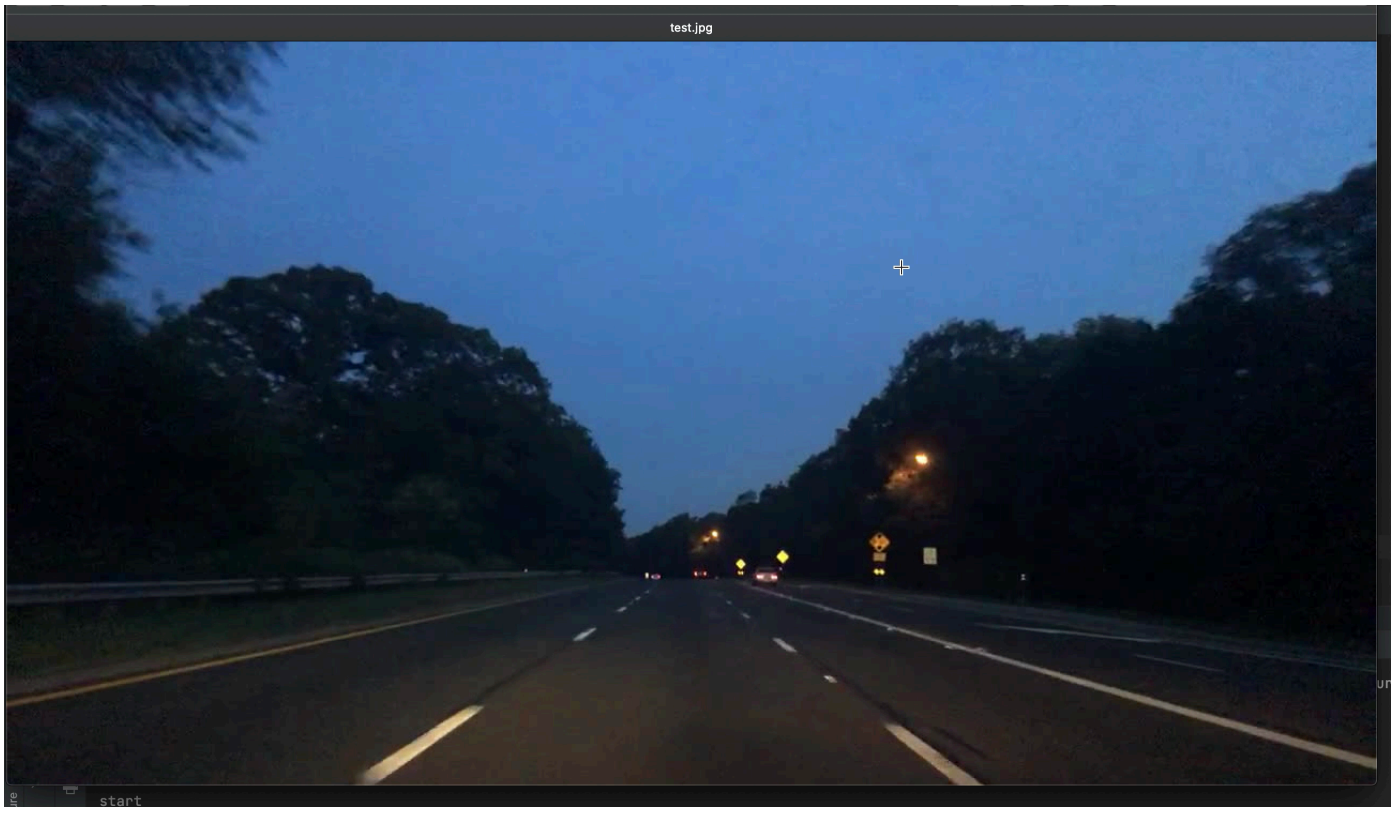
Launch the default photo previewer of MacOS and show file `test.jpg` for user.

4.4.1 Code

```
def open_img(self):
  if sys.platform == 'darwin':
    os.system('open ' + 'test.jpg')
```

4.4.2 Screenshot


Voice Assistant
你说，我在听
1. 说"播放"以播放音乐
2. 说"打开文档"以查看记事本文档
3. 说"看图"以查看示例图片
4. 说"看视频"以查看抖音小视频
看图。


test.jpg

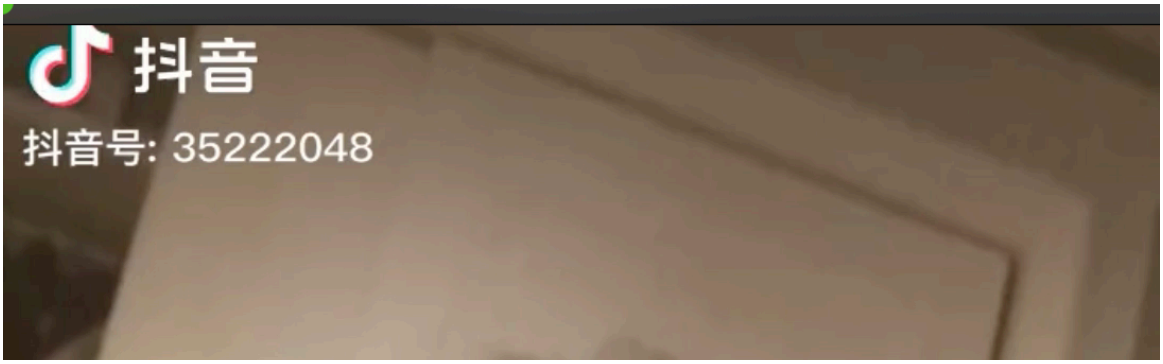
4.6 Open Videos

Launch the default video previewer of MacOS and show file `video.mp4` for user.

4.4.1 Code

```
def play_video(self):
  if sys.platform == 'darwin':
    os.system('open ' + 'video.MP4')
```

4.4.2 Screenshot



5. Analysis

5.1 Recognition Accuracy Analysis

The recognition accuracy greatly depends on the ability of the recognition engine. At the first time I tested the recommended PyPI API CMU Sphinx for many times, its delay is really high and it seldom identified what I said. However, there are a lot of other commercial APIs provided by AI buisnesses in Internet.

So I modified my strategy. This project is actually based on Baidu API to do speech recognition. As it is well-trained and used in various domains, acturally, given a well-performed device, the real speech recognition accuracy is fairly great. The accuracy matches my expectation.

However, if we want to raise the accuracy, we can

1. Use a better microphone rather than the microphone embedded in the laptop or earphone.
2. Preprocess the audio before uploading to the server .

5.2 Existing Disadvantages and Reasons

1. Not able to switch between Chinese and English
2. The recognition speed is not fast enough due to the delay of updating and model inferring.

   These two problems are caused by the speech engine.
3. The code only supports MacOS.

5.3 Solutions

1. Considering change a more intelligent model which can easily recognize whether user is speaking English or Chinese.
2. Considering change a lighter model whose parameters are tiny and infers faster.
3. Considering maintain and upgrade the code to support more commands of other OS like win32 and win64.

6. How To Run My Code?

1. 开发环境为macOS Catalina + Python3.9, 暂时不支持MacOS以外的其它操作系统调用指令。
2. 在终端输入`pip install -r requirements.txt` 或者手动安装`SpeechRecognition pocketsphinx PyQt5==5.11.3`以及其依赖库。
3. 在根目录下放入你想看的视频、文件等，并以`music.mp3` `test.txt` `test.jpg` `video.MP4`命名。
4. Run `./asr.py`