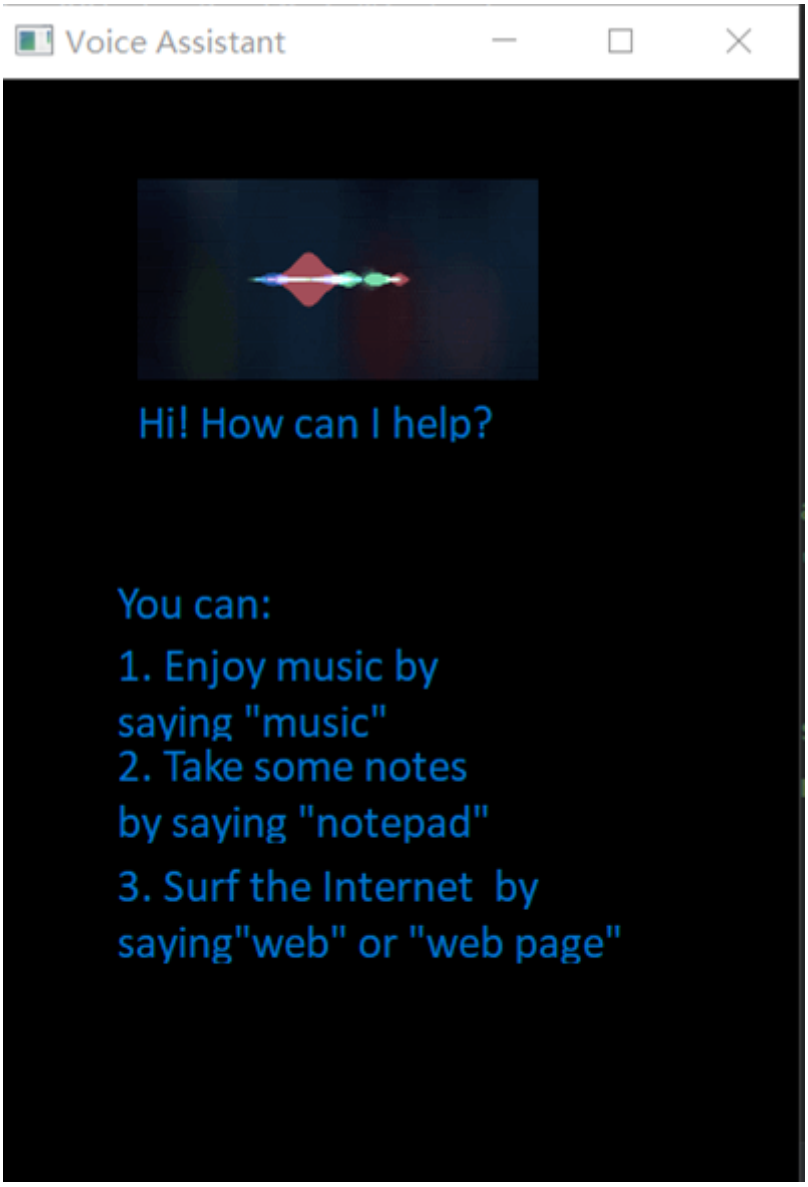
Report

1. The modifications to GUI

首先，我更改了背景的图标，使其更美观，然后，我修改并添加了一些提示，以帮助用户更轻松地使用该程序。



2. The modifications to Codes

2.1 Add functions to make the program stronger

- **Play music**

```
win32api.ShellExecute(0, 'open', 'music.mp3', '', '', 1)
```

- **Open a text file**

```
win32api.ShellExecute(0, 'open', 'notepad.exe', '', '', 1)
```

- **Open web browser**

```
webbrowser.open("https://www.baidu.com")
```

2.2 Use thread to control the recgonizition

```
class MyThread(threading.Thread):
    def __init__(self):
        super(MyThread, self).__init__()
        self._running=True

    def recognize_speech_from_mic(self,recognizer, microphone):
        """Transcribe speech from recorded from `microphone`.

        Returns a dictionary with three keys:
        "success": a boolean indicating whether or not the API request was
                   successful
        "error":   `None` if no error occured, otherwise a string containing
                   an error message if the API could not be reached or
                   speech was unrecognizable
        "transcription": `None` if speech could not be transcribed,
                   otherwise a string containing the transcribed text
        """
        # check that recognizer and microphone arguments are appropriate type
        if not isinstance(recognizer, sr.Recognizer):
            raise TypeError("`recognizer` must be `Recognizer` instance")

        if not isinstance(microphone, sr.Microphone):
            raise TypeError("`microphone` must be `Microphone` instance")

        # adjust the recognizer sensitivity to ambient noise and record audio
        # from the microphone
        with microphone as source:
            recognizer.adjust_for_ambient_noise(source)
            audio = recognizer.listen(source)

        # set up the response object
        response = {
            "success": True,
            "error": None,
            "transcription": None
        }

        # try recognizing the speech in the recording
        # if a RequestError or UnknownValueError exception is caught,
        #     update the response object accordingly
        try:
            response["transcription"] = recognizer.recognize_sphinx(audio)
        except sr.RequestError:
            # API was unreachable or unresponsive
            response["success"] = False
```

```
            response["error"] = "API unavailable"
        except sr.UnknownValueError:
            # speech was unintelligible
            response["error"] = "Unable to recognize speech"

        return response

    def stop(self):
        self._running=False
        print(self._running)

    def run(self):
        recognizer = sr.Recognizer()
        microphone = sr.Microphone()
        while self._running:
            res = self.recognize_speech_from_mic(recognizer, microphone)
            if res["error"]:
                print("ERROR: {}".format(res["error"]))
                continue
            words = res["transcription"]
            if self._running:
                print(words)
                if words.lower() in ["music","you think"]:
                    win32api.ShellExecute(0, 'open', 'f1lcapae.wav', '', '', 1)
                elif words.lower() in ["notepad","note pad","goat's head","goat
head"]:
                    win32api.ShellExecute(0, 'open', 'notepad.exe', '', '', 1)
                elif words.lower() in ["web page","webpage","web"]:
                    webbrowser.open("https://www.baidu.com")
            else:
                break
```

3.The accuracy of speech recognition

由于使用了外部语音识别api，识别准确率相对较低，主要体现在：

1.将一个单词识别为另一个单词

2.识别不清楚的单词

该程序有时会从背景噪音中识别出模棱两可的单词。

4.How to improve the accuracy

1. **Speak in quiet environment**

   在安静的环境中，程序可以避免环境中噪声的干扰，从而提高准确性。

2. **Use words that are easier to recognize as commands**

   一开始我用播放音乐作为播放音乐的命令，但很快我发现程序对清音辅音/p/不敏感，所以这个命令很难被准确识别。然后我简单地用音乐代替播放音乐，识别准确率得到了显著提高。

3. **Associate the command to be recognized with words that sound similar to it.**

   还有另一种方法可以提高识别精度。那就是将命令与听起来相似的单词相关联。