doi: 10.7690/bgzdh.2015.12.009

用于模拟飞行指挥系统的语音识别模块

王江南，张福转，孔庆波，张　原，谷树山
（海军航空兵学院飞行模拟训练系，辽宁 葫芦岛 125001）

摘要：针对模拟飞行指挥系统中人机交互的问题，开发一种语音识别模块将语音指令翻译成计算机指令。基于微软语音开发包 5.1 版开发语音识别模块，降低了软件的开发难度；通过规范指令、拆分识别指令、配置识别语法、精简识别词库等方法来提高语音识别率。应用结果表明：该方法简单易用、开发难度低、识别精度能达到实用要求。

关键词：语音识别；模拟飞行指挥；人机交互；微软语音开发包
中图分类号：TP391.9　　**文献标志码：**A

Speech Recognition Module Used in Simulated Flight Command System

Wang Jiangnan, Zhang Fuzhuan, Kong Qingbo, Zhang Yuan, Gu Shushan
(*Department of Flight Simulation Training*, *Naval Aviation Academy*, *Huludao* 125001, *China*)

Abstract: Aiming at human-computer interaction in the simulated flight command system, develop a speech recognition module for translating speech order into computer code. Develop a speech recognition module based on Microsoft Speech SDK 5.1 to reduce development difficulty; For increasing rate of speech recognition, use methods containing regulating order word, order-separation, configuring recognition-grammar, simplifying recognition words library is used. The application method shows that: the method is simple and easy to use, with low development difficulty, and its recognition precision reaches application requirement.

Keywords: speech recognition; simulated flight command; human-computer interaction; Microsoft Speech SDK

0　引言

在培养塔台飞行指挥人员的过程中，越来越多的院校用飞行指挥模拟器来代替外场实习[1]，这是因为模拟器在费效比及特情模拟、训练评估上具有不可比拟的优势。在模拟飞行指挥演练中，首先将指挥员的声音指令传递给计算机，由计算机识别后解算出计算机指令，传递给主控调度模块；主控调度模块按照指令指引飞机按不同的轨迹运动；数据送与其他模块，系统完成流程运行。语音识别在系统中占据了重要的作用，是指挥员与计算机系统进行人机交互的必要通道；因此，必须要开发一种识别率及识别速度能达到要求的语音识别模块，这是整个指挥模拟器运行的关键。

1　语音识别简介

模拟飞行指挥系统的程序流程如图 1 所示。语音识别模块将语音指令转化为计算机指令，指挥系统才能得以运行，语音识别是系统中人机交互的重要通道。随着计算机技术的发展，计算机语音识别已成为现实。

语音识别技术研究涉及到多个学科领域[2-3]，对于不同领域的应用程序开发人员来说，不可能研究掌握语音技术所涉及的每个学科，只需要有合适的语音开发工具。现在，在不同的系统下语音识别技术已获得广泛的应用：比如苹果 iOS 系统下的 Siri 语音助手、Android 系统下的语音助手服务，还有 Windows 系统下的 Cortana（个人智能助理）。这些基于语音识别的应用服务开拓了另一种人机交互方式，使计算机变得更加智能。苹果系统下的 Siri 语音助手及 Android 系统下的语音助手都基于网络服务才可以运行。而在 Windows 系统下为程序开发人员提供了实用的开发工具和接口，即微软的 Speech 语音开发包[4]，利用这样的开放平台可以比较容易开发出具有语音功能的应用程序[5]。Cortana 及 Win7 操作系统中所内置的语音识别功能都是以这一系列语音开发包为基础研制的。文中采用的是 Microsoft Speech SDK 5.1，这是微软发布的免费开发包，可供程序开发人员免费使用。

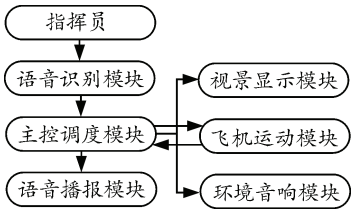

图 1　模拟塔台飞行指挥系统结构

收稿日期：2015-07-03；**修回日期：**2015-08-15
作者简介：王江南（1981—），男，河南人，硕士，工程师，从事飞机运动方程建模、计算机模拟仿真研究。

1.1 Microsoft Speech SDK 5.1 简介

Microsoft Speech SDK 5.1 是微软公司发布的免费语音应用开发工具包，是基于 COM 标准开发的。应用程序开发人员可免费使用此工具包中的内容，用来开发具有语音识别或语音合成的各种应用程序，同时又不必掌握复杂的语音技术。而且 Microsoft Speech SDK 5.1 语音开发包支持 3 种语言的识别(汉语、英语、日语)以及 2 种语言的合成(汉语、英语)，对于语音开发来说，此语音开发包是一个理想的应用程序开发工具。其中，语音合成由语音合成引擎(text to speech，TTS)负责，语音识别由识别引擎(speech recognition，SR)负责，程序开发人员只需要专注于自己的程序开发，用到语音功能时，只要调用相应的语音应用程序接口(SAPI)就能实现语音功能。SAPI 中包括了直接语音管理、训练向导、事件、语法编译、语音识别(SR)管理等强大的设计接口。这些设计接口还担负着底层控制的作用，实现应用程序的调用功能。整个语音开发包的结构如图 2 所示。

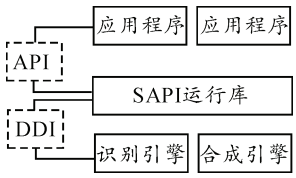

图 2 SAPI 结构

语音引擎和 SAPI 通过 DDI 层(设备驱动接口)进行交互，应用程序和 SAPI 通过 API(应用程序接口)进行通信。通过这些 API，用户可以快速开发具有语音识别功能和文字转换为语音的应用程序。

1.2 Microsoft Speech SDK SR 的开发流程

笔者采用 Microsoft Speech SDK 5.1 开发包，编程语言为 Visual C++，利用微软基础类库(microsoft foundation class，MFC)进行语音识别开发。因为采用了微软封装的工具包，在流程上要按照微软程序开发的流程进行。语音识别程序的开发流程如图 3 所示，步骤如下：

1) 初始化 COM 端口。

在程序初始化时，同时需要对语音识别函数也进行初始化，可调用 CoInitializeEx 函数来进行初始化。

2) 创建识别引擎，创建识别上下文接口。

初始化时调用函数 ISpRecognizer:: CoCreateInstance 来创建识别引擎。

Microsoft Speech SDK 5.1 支持 2 种模式的识别引擎：共享(Share)模式和独享(InProc)模式。对应的参数设置为 CLSID_SpSharedRecognizer、CLSID_SpInprocRecognizer。一般情况下可以使用共享模式，而独享模式适用于大型服务程序。

如果要使用共享型，可以直接进入下一步的设置；如果是使用独享型，要使用函数 ISpRecognizer:: SetInput 设置语音输入，让系统知道采用了哪个语音输入设备。

创建识别引擎后，即可调用接口函数 ISpRecognizer::CreateRecoContext 创建识别上下文接口(ISpRecoContext)。

3) 设置识别消息，设置我们感兴趣的事件。

调用 ISpNotifySource::SetNotifyWindowMess-age 告诉 Windows 哪个是我们的识别消息，需要进行处理。

调用 ISpRecoContext::SetInterest 设置程序感兴趣的事件。我们要知道什么时候开始语音识别(SPEI_SOUND_START)，什么时候结束(SPEI_SOUND_END)，是识别出最终结果(SPEI_RECOGNITION)，还是无法识别(SPEI_ FALSE_RECOGNITION)。各事件对应的参数可参照 SPEVENTENUM 的说明文档。

4) 创建语法规则并激活。

语法规则是进行语音识别的关键，必须要进行设置。语法规则分为 2 种：一种是听说式(dictation)，一种是命令式(C&C)。听说式检索范围大，识别率低，且后续操作处理也不方便，适用于随机听写的文字录入操作；命令式需要自己定义命令语句，包含的语句数量有限，在用户自己定义的语句中进行检索对比，识别率高，同时方便后续的操作。

采用听说式规则时，先用函数 ISpRecoContext:: CreateGrammar 创建语法对象，然后再用函数 ISpRecoGrammar::LoadDictation 加载字典。

采用命令式规则不但可以提高语音识别的精度，而且也方便于命令语句的拆分及命令语句的规范化。创建命令语法规范时，首先，利用 ISpRecoContext:: CreateGrammar 创建语法对象；然后利用 ISpRecoGrammar::LoadCmdxxx 加载命令式语法规则。命令式语法规则存储在 XML 格式的文

件中，包括语法规则及命令条目语句。笔者将在后面举例说明语法规则的配置。

在开始识别时，激活语法进行识别。听说式调用函数 ISpRecoGrammar::SetDictationState 进行激法；命令式规则需要调用 ISpRecoGrammar::SetRuleState 语句进行激活语法。

5) 获取识别消息，进行处理。

截获识别消息(WM_RECOEVENT)，然后进行处理。识别结果可以显示到计算机屏幕上，也可发送给网络上其他计算机，完成整个系统的有序运行。

6) 释放创建的引擎等。

释放创建的引擎、识别上下文对象、语法等。调用相应的 Release 函数即可。

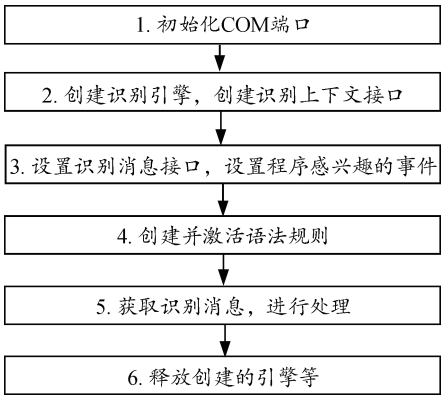

图 3　微软语音识别流程

2　构建语音识别模块

在构建语音识别模块时，需要对软硬件进行配置。硬件方面需要 1 台具备声卡输入的高性能计算机，并且配备具有良好抗噪声干扰性能的声音采集设备。软件方面需要对计算机进行相应配置，包括对计算机的语音训练，及语音识别程序中用到的语法规则的 XML 的配置。硬件方面的配置在这里就不再一一阐述，笔者主要介绍下计算机的软件运行环境及语音识别的语法规则配置。

1) 计算机环境的配置。

文中所用到计算机安装有 Windows XP 系统；用微软 Microsoft Visual 2003 编译器进行编程，采用 C++语言；安装 Microsoft Speech SDK 开发工具包，同时安装中文语言包，使之可识别汉语普通话。

2) 对计算机进行语音训练。

在安装好必要的软件后，在控制面板会出现“语音”项，进入语音项选择“微软简体中文识别器(microsoft simplified chinese recognizer)V5.1”然后可以进行新建语音用户，配置麦克风，训练配置文件等操作，语音属性面板如图 4 所示；语音训练使计算机了解指挥员的说话方式，也使指挥员了解计算机的识别习性，使计算机与指挥员相互适应。可以在控制面板里进入“语音”项，也可以在程序中调用 Windows 控制面板项，进行此步操作。

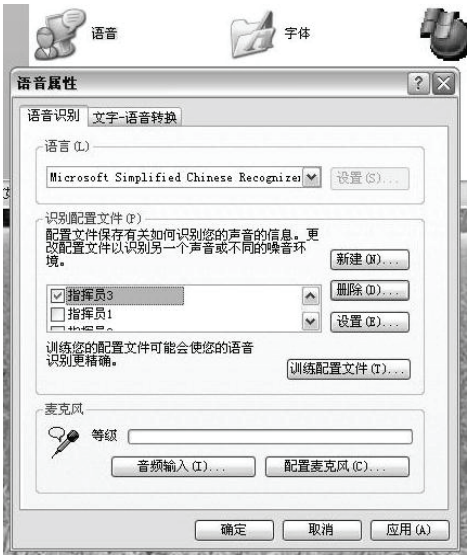

图 4　语音属性面板

3) 建立语法规则。

根据飞行指挥特点，需要建立一套合适的语法规则，这可以规范训练时的用语习惯，同时也可以提高语音识别的精度。

2.1　指挥口令特点

在飞行指挥中，指挥员需要将指令清晰准确地传递给飞行员。这要求指挥员用规范的口令准确地下达命令。所以指挥口令用词固定，言简意赅，口令规范，具有标准化、词汇集有限、指令有限、指令重复率高的特点。这些特点为计算机语音识别提供了很好的先天条件。在收集整理命令语句时可将语句进行整合，按标准化、规范化的要求对命令语句进行梳理。固定指挥口令用语用词，制定一套指挥口令。其后基于指挥口令的特点再对指挥口令进行拆分。

2.2　拆分识别指令

对识别指令进行拆分可以大大精简指令的数目，减少语音识别时计算机所要检索的词汇量，这对于识别效率及反应速度都有很大帮助。

在指挥调度中，指挥口令的一般格式是：飞行员呼号+指令(例如：拐洞幺起飞)、飞行员呼号+指令+空域号(例如：拐洞幺起飞去 5 号空域)等。对于指挥员来说，飞行员呼号为必备的，这将明确指向，

便于调度。拿飞行员+指令的组合来说，如果有 20 名飞行员，100 条不同指令，未对指挥口令进行拆分前，需要识别的指令词汇的数量为 2 000 个，而拆分的语句命令只需要 120 个。拆分后指令词汇数目急剧下降，同时在编辑预置指挥口令时节省了大量的精力；对于计算机来说，词汇检索量也将大幅度地缩减。将指令拆分后，可以在 XML 文件中设置指令组合方式恢复成一般指令。

2.3 编辑语法规则文件

整理拆分后的指挥口令，编辑语法规则，存为 XML 格式的文件，以供语音识别引擎调用。在编辑语法的规则时，对指挥口令添加属性值。口令语句用于语音识别，属性值用于计算机内部计算所用到的数字化指挥调度。对每条语句添加属性值的好处是：在语音识别过程中，可以直接识别出数字化的指令，同时将飞机呼号与调度命令分离；不再需要对每条语句进行解析，减少了对识别出的语句的比对解析环节，节省计算机运算能力，减少延迟时间。数字化的指令可以直接发送至主控调度系统，以完成飞机的调度。以下为简单例子：

```
<GRAMMAR LANGID="804">
  <DEFINE>
    <ID NAME="COMMAND" VAL="10"/>
    <ID NAME="HUHAO" VAL="101"/>
    <ID NAME="ZHILING" VAL="102"/>
  </DEFINE>
  <RULE NAME="Command" ID="COMMAND"
TOPLEVEL="ACTIVE">
    <p>
    <RULEREF NAME = "huhao"/>
    <RULEREF NAME = "zhiling"/>
    </p>
  </RULE>
  <RULE NAME="zhiling" >
    <L PROPNAME = "zhiling"
PROPID="ZHILING">
      <p VAL="1001">开车</p>
      <p VAL="1002">滑出停机坪</P>
      <p VAL="1003">起飞</p>
      <p VAL="1004">着陆</p>
      </L>
  </RULE>
  <RULE NAME="huhao" >
    <L PROPNAME = "huhao"
PROPID="HUHAO">
      <p VAL="701">拐洞幺</p>
      <p VAL="702">拐洞俩</P>
      <p VAL="703">拐洞叁</p>
      <p VAL="707">拐洞拐</p>
      </L>
    </RULE>
</GRAMMAR>
```

其中 LANGID="804"代表简体中文，设置为对中文进行语音识；在本例中，共有 4 个飞机呼号("huhao")，4 个指挥指令("zhiling")。整个指挥口令是“呼号”+“指令”的组合方式。对应每一个条目，都有相应的属性值。比如指令("zhiling")"滑出停机坪"对应的属性值就是“1002”。在语音识别处理过程中，如果识别的结果为“拐洞幺滑出停机坪”，在语音识别端可显示同样的文字。同时，其属性值“701”及“1002”也被提取出来，送给主控调度系统；调度系统即可以指定“701”号飞机按“1002”命令进行运行。

2.4 实时加载识别词库

不同的指挥训练场次所用到的指令是不一样的，系统根据不同的指挥科目实时生成对应的词库。最基本的本场起落指令大概 20 条，就能满足基本的流程训练。其他科目的训练也只需要极少量的指令即可。根据不同的训练内容加载不同的指令库，从而进一步减少识别指令条目数量。

同样，对于飞机呼号的数量也可实时加载，这也可以大幅缩减飞机呼号的识别量。一是固定有限的飞机呼号，只要数量上能够达到可以区分飞行训练的不同架次飞机即可；二是筛选飞机呼号，剔除掉拗口的、不清晰的、计算机不易识别的飞机呼号；三是提取制定的飞行计划表中的飞机呼号来加载相应的飞机呼号；从而提高识别率。

其中文献[5]提供了一种在 XML 文件中写入占位符(placehold)的方法，可以在程序中实现动态词库，随时增减指令条目。以实现实时加载识别词库的目地。通过实时加载词库，使指挥指令保持在能满足相应训练的水平，可以大幅提高语音识别效率，使语音识别的精度更好。

3 程序实现

在设计界面的过程中，借鉴 Microsoft Speech SDK 5.1 中例子程序，使程序实现过程清晰。从引擎的创建到载入命令语法，再到激活命令(C&C)模式实现，都一一体现在程序界面上。

（下转第 41 页）