音声（ジェスチャ）によるスライド、ビデオの操作

概要

本システムは、音声による指示でアプリケーションを操作するシステムです。

「スライド」「ビデオ」等の発話で与えた指示に合わせて、Microsoft Office PowerPoint や Mediaplayer (産総研研究者：原功)を操作します。

本システムは、JSTプロジェクト「先端計測分析機器用共通ソフトウエアプラットフォームの開発」(研究代表者：大阪大学大学院·佐藤了平教授) プロジェクトで行われた講習会でデモストレーションを行いました。

システムのハードウェア・ソフトウェア構成

ハードウェア

- Windows OSの動作するパソコン
- パソコン対応マイク

ソフトウェア

- Microsoft Office PowerPoint
- Java SDK

使用するコンポーネント

この例では、以下のコンポーネントを利用します。

- PortAudioInputコンポーネント: 音声データの取得
- Juliusコンポーネント: 日本語、英語音声認識
- SEATコンポーネント: 音声対話制御

ダウンロード

本システムを構築するために必要なデータを、ここでダウンロードすることが出来ます。以下のデータにはミドルソフトウェア、アプリケーション(Microsoft PowerPoint、 Java SDKは除く)、文法モデル、状態遷移モデル、バッチファイルなどが含まれます。

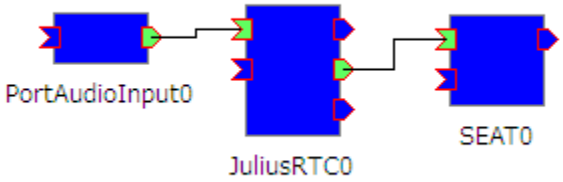
- Seminar-OpenHRI.zip

コンポーネントの接続

すべてのコンポーネントをエディタ画面に配置し、各コンポーネント間に以下のリンクを作成します。:

```
PulseAudioInput -> Julius
Julius(result) -> SEAT
```

リンクを作成すると、下図のようになります。



音声認識文法モデル

Juliusコンポーネントが起動時に読み込む操作パターンの文法モデルを定義します。

操作パターンは「ビデオ」「ビデオプレーヤ」「スライド」「リスト」「次へ」「前へ」「オープン」「実行」「終了」「再生」「ストップ」「停止」「一時停止」「早送り」「巻き戻し」「全画面」「次のスライド」「前のスライド」「最初のスライド」「最後のスライド」「いち番目」から「じゅう番目」のいずれかにします。このような命令を認識する音声認識文法は、下のように定義することができます。

kinect_demo2.grxml

```
<?xml version="1.0" encoding="UTF-8" ?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://www.w3.org/2001/06/grammar
                             http://www.w3.org/TR/speech-grammar/grammar.xsd"
         xml:lang="jp"
         version="1.0" mode="voice" root="command">

  <rule id="command">
    <one-of>
      <item><ruleref uri="#command_simple" /></item>
      <item><ruleref uri="#command2" /></item>
      <item><ruleref uri="#command_select" /></item>
    </one-of>
  </rule>
  <rule id="command_simple">
    <one-of>
      <item>ビデオ</item>
      <item>ビデオプレーヤ</item>
      <item>スライド</item>
      <item>リスト</item>
      <item>次へ</item>
      <item>前へ</item>
      <item>オープン</item>
      <item>実行</item>
      <item>終了</item>
      <item>再生</item>
      <item>ストップ</item>
      <item>停止</item>
      <item>一時停止</item>
      <item>早送り</item>
      <item>巻き戻し</item>
      <item>全画面</item>
    </one-of>
  </rule>
  <rule id="command2">
    <one-of>
      <item>次の</item>
      <item>前の</item>
      <item>最初の</item>
      <item>最後の</item>
    </one-of>
    <one-of>
      <item>スライド</item>
    </one-of>
  </rule>
   <rule id="command_select">
    <one-of>
      <item>いち</item>
      <item>に</item>
      <item>さん</item>
      <item>よん</item>
      <item>ご</item>
      <item>ろく</item>
      <item>なな</item>
      <item>はち</item>
      <item>きゅう</item>
      <item>じゅう</item>
    </one-of>
    <one-of>
```

```
      <item>番目</item>
    </one-of>
  </rule>
</grammar>
```

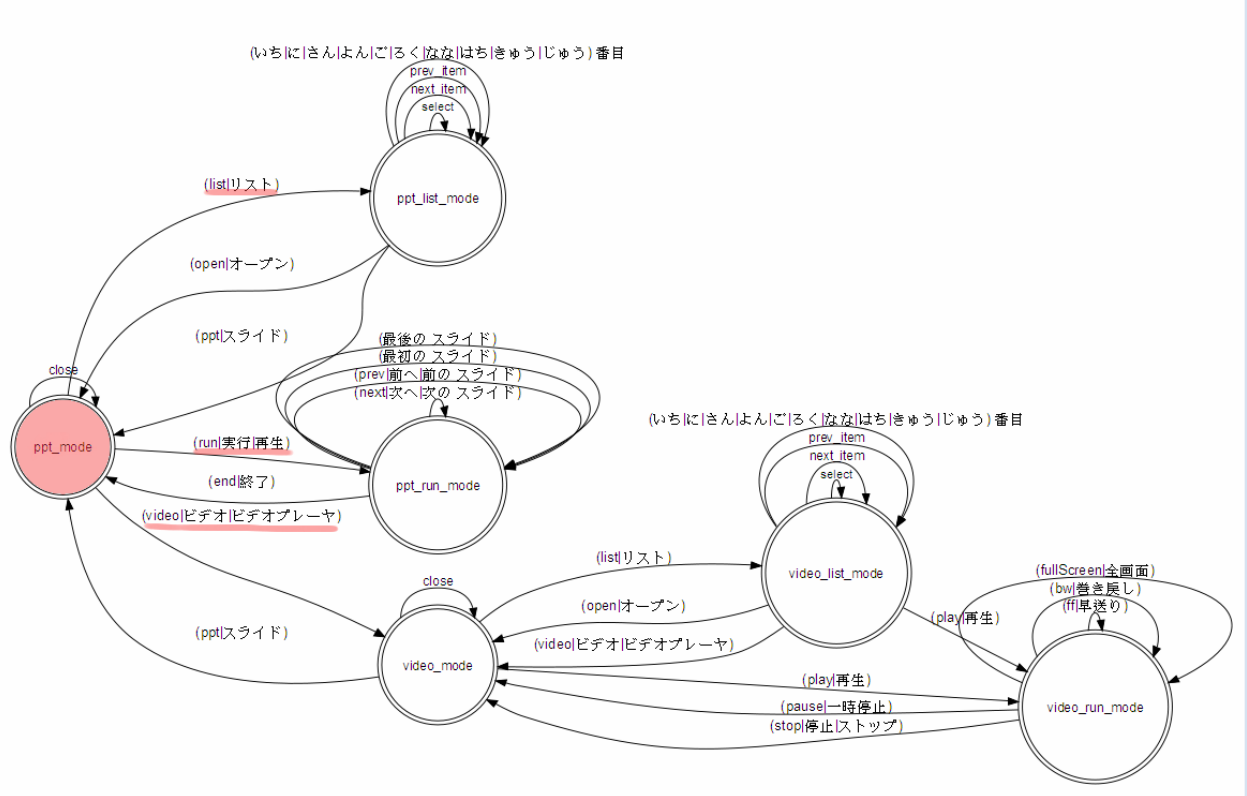
上記の文法モデルは、「次の」「前の」「最初の」「最後の」の次に、「スライド」と続く文法と、「いち」から「じゅう」の次に「番目」と続く文法を定義しています。

対話スクリプト

ここではSEATコンポーネントが起動時に読み込む状態遷移モデルを定義します。

本システムには、PowerPointを操作する状態のPPT操作状態 [ppt_mode] 、PPTリスト選択状態 [ppt_list_mode] 、PPT起動状態 [ppt_run_mode] と、動画操作する状態のビデオ操作状態 [video_mode] 、ビデオリスト選択状態 [video_list_mode] 、ビデオ起動状態 [video_run_mode] があり、システム起動後はPPT操作状態になります。

システム起動後の状態、PPT操作状態 [ppt_mode] は「リスト」、「実行」又は「再生」、「ビデオ」又は「ビデオプレーヤ」などの命令をすると、下の図のように状態が遷移し、指示も変わります。



上記の状態遷移モデルは次のように記述します。

SEATコンポーネントの seateditor を使うと、上記の状態遷移図を確認しながら記述することが出来ます。

kinect_demo2.seatml

```
<?xml version="1.0" encoding="UTF-8"?>
<seatml>
  <general name="flaggame">
    <agent name="gesture" type="rtcin" datatype="TimedString" />
    <agent name="longvalue" type="rtcin" datatype="TimedLong" />
    <agent name="command" type="rtcout" datatype="TimedString" />
    <agent name="ppt" type="socket" host="localhost" port="10030" />
    <agent name="mplayer" type="socket" host="localhost" port="10020" />
  </general>

  <state name="ppt_mode">
    <rule>
      <key>(list|リスト)</key>
      <command host="ppt">(openList)</command>
      <script host="ppt">
global current_item
current_item=0
rtc_result="(select %d)" % current_item
</script>
      <statetransition>ppt_list_mode</statetransition>
    </rule>

    <rule>
      <key>close</key>
      <command host="ppt">(close)</command>
    </rule>
    <rule>
      <key>(run|実行|再生)</key>
      <command host="ppt">(run)</command>
      <statetransition>ppt_run_mode</statetransition>
    </rule>

    <rule>
      <key>(video|ビデオ|ビデオプレーヤ)</key>
      <statetransition>video_mode</statetransition>
    </rule>
  </state>

  <state name="ppt_list_mode">
    <rule>
      <key>(open|オープン)</key>
      <command host="ppt">(open)</command>
      <statetransition>ppt_mode</statetransition>
    </rule>
        <rule>
      <key>select</key>
      <script host="ppt">
global current_item
current_item=0
```

```
rtc_result="(select %d)" % current_item
</script>
    </rule>

    <rule>
      <key>next_item</key>
      <script host="ppt">
global current_item
current_item += 1
rtc_result="(select %d)" % current_item
</script>
    </rule>

    <rule>
      <key>prev_item</key>
      <script host="ppt">
global current_item
current_item -= 1
rtc_result="(select %d)" % current_item
</script>
    </rule>

    <rule>
      <key>(いち|に|さん|よん|ご|ろく|なな|はち|きゅう|じゅう) 番目</key>
      <script host="ppt">
global current_item
items ={u"いち":0, u"に":1, u"さん":2, u"よん":3, u"ご":4, u"ろく":5, u"なな":6, u"はち":
v = rtc_in_data.split(' ')[0]
print v
current_item = items[v]
rtc_result="(select %d)" % current_item
</script>
    </rule>
      <rule>
      <key>(ppt|スライド)</key>
      <statetransition>ppt_mode</statetransition>
    </rule>
  </state>

   <state name="ppt_run_mode">
       <rule>
      <key>(next|次へ|次の スライド)</key>
      <command host="ppt">(next)</command>
    </rule>
    <rule>
      <key>(prev|前へ|前の スライド)</key>
      <command host="ppt">(previous)</command>
    </rule>
    <rule>
      <key>(最初の スライド)</key>
      <command host="ppt">(first)</command>
    </rule>
    <rule>
      <key>(最後の スライド)</key>
      <command host="ppt">(last)</command>
    </rule>

      <rule>
      <key>(end|終了)</key>
      <command host="ppt">(end)</command>
      <statetransition>ppt_mode</statetransition>
```

```
    </rule>
   </state>


  <state name="video_mode">
    <rule>
      <key>(list|リスト)</key>
      <command host="mplayer">(openList)
     </command>
      <script host="mplayer">
global current_item
current_item=0
rtc_result="(select %d)\n" % current_item
</script>
    <statetransition>video_list_mode</statetransition>
    </rule>

    <rule>
      <key>close</key>
      <command host="mplayer">(close)</command>
    </rule>
      <rule>
      <key>(play|再生)</key>
      <command host="mplayer">(play)</command>
      <statetransition>video_run_mode</statetransition>
    </rule>


    <rule>
      <key>(ppt|スライド)</key>
      <statetransition>ppt_mode</statetransition>
    </rule>

  </state>

  <state name="video_list_mode">
      <rule>
      <key>(open|オープン)</key>
      <command host="mplayer">(open)</command>
      <script host="mplayer">
global current_volume
current_volume = 10
rtc_result="(volume %d)\n" % current_volume
</script>
<statetransition>video_mode</statetransition>
    </rule>

    <rule>
      <key>select</key>
      <script host="mplayer">
global current_item
current_item=0
rtc_result="(select %d)\n" % current_item
</script>
    </rule>

    <rule>
      <key>next_item</key>
      <script host="mplayer">
global current_item
current_item += 1
```

```
rtc_result="(select %d)\n" % current_item
</script>
    </rule>

    <rule>
      <key>prev_item</key>
      <script host="mplayer">
global current_item
current_item -= 1
rtc_result="(select %d)\n" % current_item
</script>
    </rule>
    <rule>
      <key>(いち|に|さん|よん|ご|ろく|なな|はち|きゅう|じゅう) 番目</key>
      <script host="mplayer">
global current_item
items ={u"いち":0, u"に":1, u"さん":2, u"よん":3, u"ご":4, u"ろく":5, u"なな":6, u"はち":
v = rtc_in_data.split(' ')[0]
current_item = items[v]
rtc_result="(select %d)" % current_item
</script>
    </rule>
      <rule>
      <key>(play|再生)</key>
      <command host="mplayer">(play)</command>
      <statetransition>video_run_mode</statetransition>
    </rule>

      <rule>
      <key>(video|ビデオ|ビデオプレーヤ)</key>
      <statetransition>video_mode</statetransition>
    </rule>
  </state>

  <state name="video_run_mode">


    <rule>
      <key>(ff|早送り)</key>
      <command host="mplayer">(forward)</command>
    </rule>

    <rule>
      <key>(bw|巻き戻し)</key>
      <command host="mplayer">(backward)</command>
    </rule>

    <rule>
      <key>(pause|一時停止)</key>
      <command host="mplayer">(pause)</command>
       <statetransition>video_mode</statetransition>
    </rule>

    <rule>
      <key>(stop|停止|ストップ)</key>
      <command host="mplayer">(stop)</command>
       <statetransition>video_mode</statetransition>
    </rule>

      <rule>
      <key>(fullScreen|全画面)</key>
```

```
      <command host="mplayer">(fullScreen)</command>
    </rule>
  </state>
</seatml>
```

システムを起動する

最初にダウンロードした Seminar-OpenHRI.zip を解凍し、[Seminar-OpenHRI] ディレクトリに移動します。[rtm-naming.bat] と [RTSE.bat] を実行して、ネーミングサービスとRTシステムエディタを起動しておきます。

次に [Seminar-OpenHRI] > [0 SpeechDemo]ディレクトリに移動し、[0 StartDemo.bat] を実行します。[0 StartDemo.bat]で次のアプリケーションとRTコンポーネントが起動します。

MediaPlayer
: 動画再生するアプリケーションです。MediaPlayer.exe が起動します。

PowerPointServer
: Microsoft Office PowerPointを編集するアプリケーションで、PowerPointServer.exe が起動します。Microsoft Office PowerPointがインストールされていない場合、ウィンドウが表示されますので [OK]を押下します。

PortAudioInputコンポーネント
: AudioInputコンポーネントはマイクから音声データを取得し出力するコンポーネントで、portaudioinput.exe が起動します。

Juliusコンポーネント
: Juliusコンポーネントは文法ファイルを元に日本語･英語の音声を認識するコンポーネントです。juliusrtc.exe が音声認識の文法ファイル [kinect_demo2.grxml] を指定して起動します。

SEATコンポーネント
: SEATコンポーネントは文法ファイルを元にパラフレーズマッチングする対話制御コンポーネントです。SEAT.exeが状態遷移モデルファイル[kinect_demo2.seatml]を指定して起動します。

アプリケーションとRTコンポーネントが起動したら、[1 ConnectRTC.bat] と [2 ActivateRTC.bat] を実行します。データポートの接続と、コンポーネントが Activate になり、システムの起動が完了します。

システムを終了する場合は [3 DeactivateRTC.bat] [4 DisconnectRTC.bat] を、コンポーネントを終了する場合は [5 DemoExit.bat]を実行し終了します。ネーミングサービスやRTシステムエディタを終了する時は、手動で終了してください。

動作例

次に、パソコンにマイクを接続し、音声認識文法モデルで定義した文（「スライド」、「ビデオ」など）を発話します。

システム起動直後は、操作パターン図の [ppt_mode] の状態になります。操作パターンは「リスト」「実行」「再生」「ビデオ」「ビデオプレーヤ」です。

「リスト」はpptデータを選択する [ppt_list_mode] に、「実行」「再生」はPowerPointを操作する [ppt_run_mode]に状態遷移します。「ビデオ」「ビデオプレーヤ」は MediaPlayerを操作する[video_mode] に状態遷移します。

『 リスト 』とマイクに発話すると、pptデータのリストが表示され、[ppt_list_mode]に状態遷移します。[ppt_list_mode]は、「いち番目」～「じゅう番目」でリストのデータを選択したり、「オープン」「スライド」でpptデータを開いて[ppt_mode] に状態遷移します。

『 さん番目 』と発話してpptデータを選択した後、『 オープン 』又は『 スライド 』と発話し[ppt_mode]に状態遷移します。この時、リストウィンドウは表示されたままですが、[ok] か [キャンセル]を押下して閉じることも可能です。なお、選択したpptデータに影響はありません。

次に『 実行 』又は『 再生 』と発話すると、PowerPointでデータが開き、[ppt_run_mode] に状態遷移します。[ppt_rum_mode]は、「前へ」「前のスライド」「次へ」「次のスライド」「最初のスライド」「最後のスライド」「終了」でPowerPointのスライド操作をします。

PowerPointの操作を終了する際は、『 終了 』と発話すると[ppt_mode]に状態遷移します。この時、 PowerPointは起動したままです。

[ppt_mode]の状態に戻ると、「ビデオ」「ビデオプレーヤ」で[video_mode]に状態遷移し、MediaPlayerを使って動画を再生することが出来ます。

操作パターンの図を参考に、操作してみてください。