次世代ロボット知能化技術開発プロジェクト

施設内生活支援ロボット知能の研究開発

発話推定モジュール(SpeechDetector)

マニュアル

Ver. 2.1

2012 年 1 月

九州工業大学

改変履歴

Ver.	改版日	改変内容
1.0	2010/3	新規作成
1.1	2010/11	ドキュメントの改訂
2.0	2011/12/14	OpenRTM-1.0.0 対応、モジュール名変更
2.1	2012/1/24	ドキュメントから不要な情報を削除

1. はじめに 4
1.1. 本文書について 4
1.2. 対象プラットホーム 4
1.3. OpenRTM-aist-C++ 関係 4
2. 発話推定モジュール(SpeechDetector) 5
2.1. インストール 5
2.2. ビルド 6
2.3. 起動 6
2.4. ポートについて 8
2.5. 動作検証 8

1. はじめに

1.1. 本文書について

本書は、「次世代ロボット知能化技術開発プロジェクト」の「施設内生活支援ロボット知能研究開発」において構築した音声認識モジュールについてのマニュアルです。本書は RT ミドルウェア（以下 RTM）、RT コンポーネント（以下 RTC）を用いたロボットシステム開発者を対象に記述されており、RTM、RTC や関連ツールに関する一般的な知識を持つことを前提とします。

OpenRTM-aist official webside: http://www.openrtm.org/openrtm/ja/

1.2. 対象プラットホーム

本モジュールは、以下の環境で動作確認しています。

- Windows XP professional 32bit
- Windows Vista Business 32bit/64bit
- Windows 7 Professional/Enterprise 32bit/64bit

1.3. OpenRTM-aist-C++ 関係

以下のソフトウェアをインストールしてください。

- Microsoft Visual Studio 2008 Professional Edition/Express Edition
- OpenRTM-aist-1.0.0-RELEASE_vc9_100212.msi
- OpenCV_1.0.exe

2. 発話推定モジュール(SpeechDetector)

USB カメラの映像に写るユーザが発話しているかどうかをモジュールのための RTC(Consumer)です。画像の入力や処理のため OpenCV を用いています。別途 OpenCV-1.0 をインストールしてください。本モジュールでは、含まれるファイルが Win32 を前提としていますので、他の OS で動作させる場合には注意して下さい。
モジュールのディレクトリは SpeechDetector となります。

このモジュールは、OpenCV に接続されている USB カメラなどのキャプチャーデバイスから画像を取得し、発話しているかどうかを推定し、サービスポートから認識結果を出力します。このため、モジュールは SpeechDetector.idl で定義されるサービスポート(コンスマー)として機能します。

```
interface SpeechDetector {
   /*!
    * 会話を開始したときに呼び出されます。
    */
   void speechBegin();

   /*!
    * 会話を終了したときに呼び出されます。
    */
   void speechEnd();

   /*!
    * 解析された画像フレーム毎に呼び出されます。
    * @param frame 現在のフレーム数
    * @param mouth_x 口領域の画像上の x 座標。未発見の場合は-1。
    * @param mouth_y 口領域の画像上の y 座標。未発見の場合は-1。
    * @param mouth_w 口領域の幅のピクセル数。未発見の場合は-1。
    * @param mouth_h 口領域の高さのピクセル数。未発見の場合は-1。
    * @param of 発話推定結果
    * @param sad 発話推定結果
    * @param result 発話推定結果。正は発話中、負は非発話中
    */
   void onAnalyze(in long frame, in long mouth_x, in long mouth_y, in long
mouth_w, in long mouth_h, in float of, in float sad, in float result);
};
```

画像フレーム毎 onAnalyze()に解析結果が出力され、会話の開始時と終了時にそれぞれ speechBegin()と speechEnd()が呼び出されます。

2.1. インストール

RTC 再利用センター登録の zip ファイルをダウンロードしてください。

音声認識モジュール: SpeechDetector.2.0.zip

以下の手順で eclipse にプロジェクトをインポートします。

1. eclipse を起動する。
2. ファイルメニューから「インポート」を選択し、「一般」の「既存のプロジェクトをワークスペースへ」を選ぶ。
3. 次に進み、「アーカイブ・ファイルの選択」の欄にチェックし、「参照」で

SpeechDetector.2.0.zip を選択する。

4. 「完了」を選択する。
5. 環境に合わせて、SpeechDetector/user_config.vsprops ファイルを調整する。
   詳細については OpenRTM のドキュメントを参照してください。
6. Microsoft Visual Studio より次のソリューションファイルを読み込ませる。
   SpeechDetector/SpeechDetectorConsumer_vc9.sln

2.2. ビルド

1. ビルドには、Microsoft Visual Studio 2008、OpenRTM の環境(OpenRTM C++、OmniORB など)と OpenCV が必要となります。
2. SpeechDetectorConsumer のソリューションのビルドを実行して下さい。
3. ビルドが完成したら、
   SpeechDetector/SpeechDetectorConsumer/Debug,Release
   SpeechDetector/SpeechDetectorConsumerComp/Debug,Release
   に実行ファイルや DLL ファイルが作成されます。
4. モジュールを動作させる対象のディレクトリには、OpenCV の DLL ファイルなどをコピーします。
   必要なファイルについては、SpeechDetector/components を参照して下さい。

なお、SpeechDetector/components に、Win32 向けにビルド済みのファイルを格納しています。

2.3. 起動

SpeechDetector の起動には、USB カメラが必要ですので、予め用意し設定などを完了しておいてください。

起動時の設定ファイルである SpeechDetector/rtc.conf の内容を確認して下さい。

- corba.nameservers: CORBA ネームサーバのホスト名とポート番号
  例: corba.nameservers: localhost:5005
- exec_cxt.periodic.rate: RTC コンテキストの実行周期。
  値が大きいほど周期が早くなります。計算機の負荷に影響します。
  例: exec_cxt.periodic.rate: 30.0

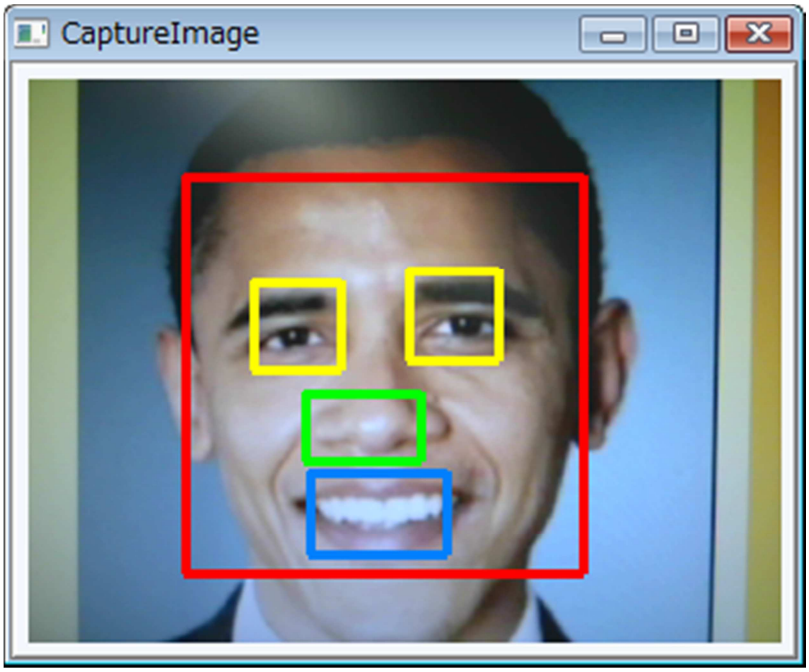
次にコマンドプロンプトを開き、SpeechDetector に移動します。

先にビルドした SpeechDetectorConsumerComp.exe を実行します。

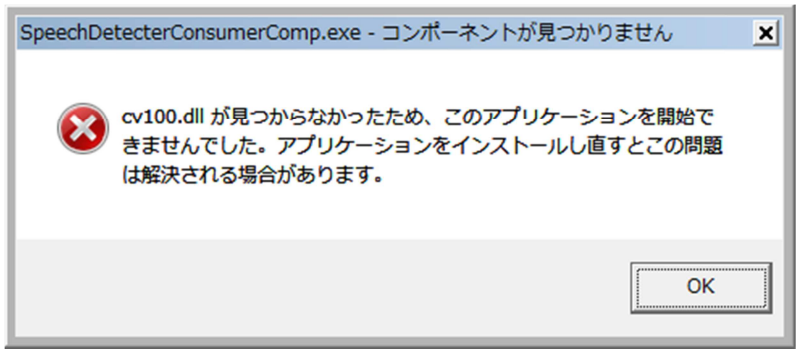
```
C:${SRP_HOME}/SpeechDetector>
components¥SpeechDetectorConsumerComp.exe
```

（ここでは、SpeechDetector/components にビルド済みのファイルがコピーされているとしています。）

正常に起動しても何も変化がありませんが、RTC を活性化すると次のようにカメラ画像を描画するウィンドウが表示されます。赤枠が顔部分、黄色が目部分、緑が鼻部分、青が口部分の検出結果を示しています。これらの枠が表示されない場合は、照明や顔との距離を調整してみてください。

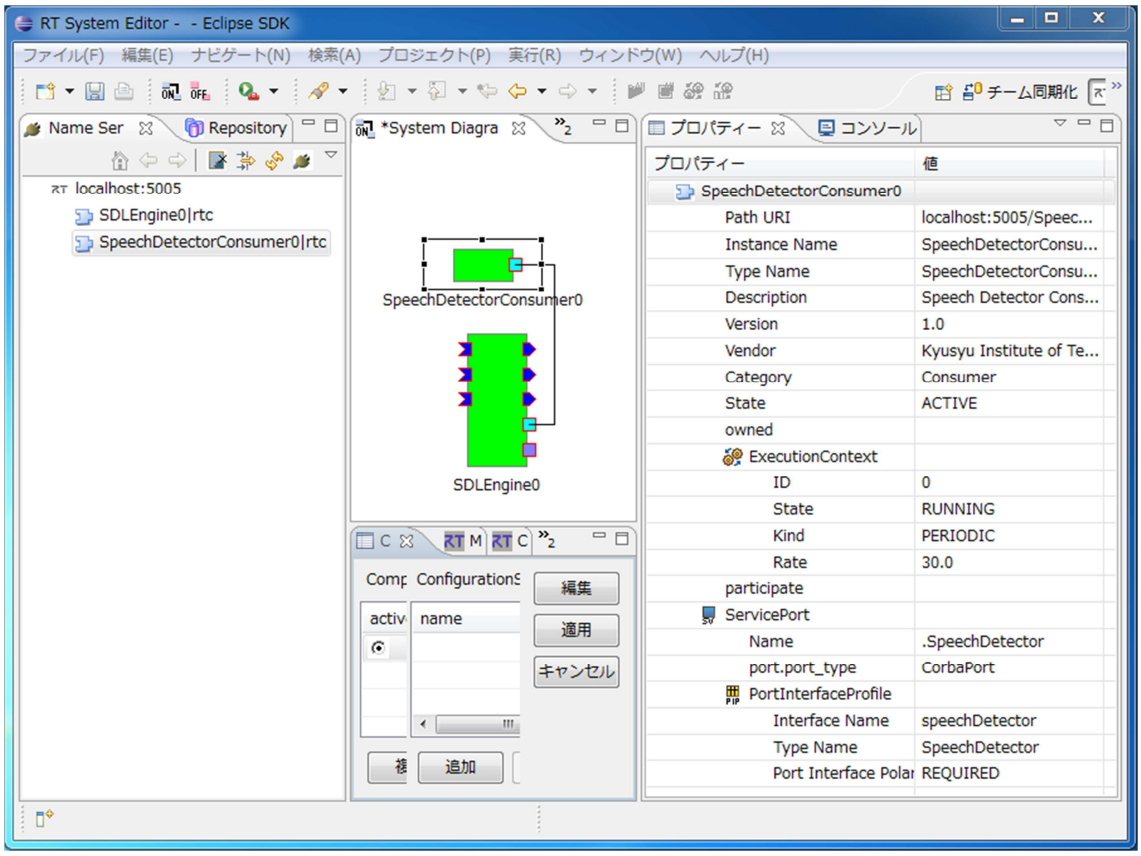


DLL ファイルが不足している場合は起動時に次の画面が表示されます。



2.4. ポートについて

SpeechDetector.idl で定義されるサービスポート(コンスーマ)のみです。次の動作検証で説明する作業計画モジュールと接続した状態を以下に示します。



2.5. 動作検証

ここでは、作業計画モジュールと組み合わせた動作検証方法を紹介します。

(1) 前準備

1. ネームサーバを起動します。

   以下の方法を参考にネームサーバを起動してください。

   (ア) Eclipse の SDLEngine プロジェクト内の OmniNames.bat を実行する。

   ※ 環境変数 OMNI_ROOT が設定されていることを確認してください。

   ※ 必要に応じて OmniNames.bat を書き換えてください。

2. 作業計画モジュールのコンソールを起動

   (ア) Eclipse の SDLEngine プロジェクト内にある

   jp.ac.kyutech.SRP.Console クラス

   を Java アプリケーションとして起動します。

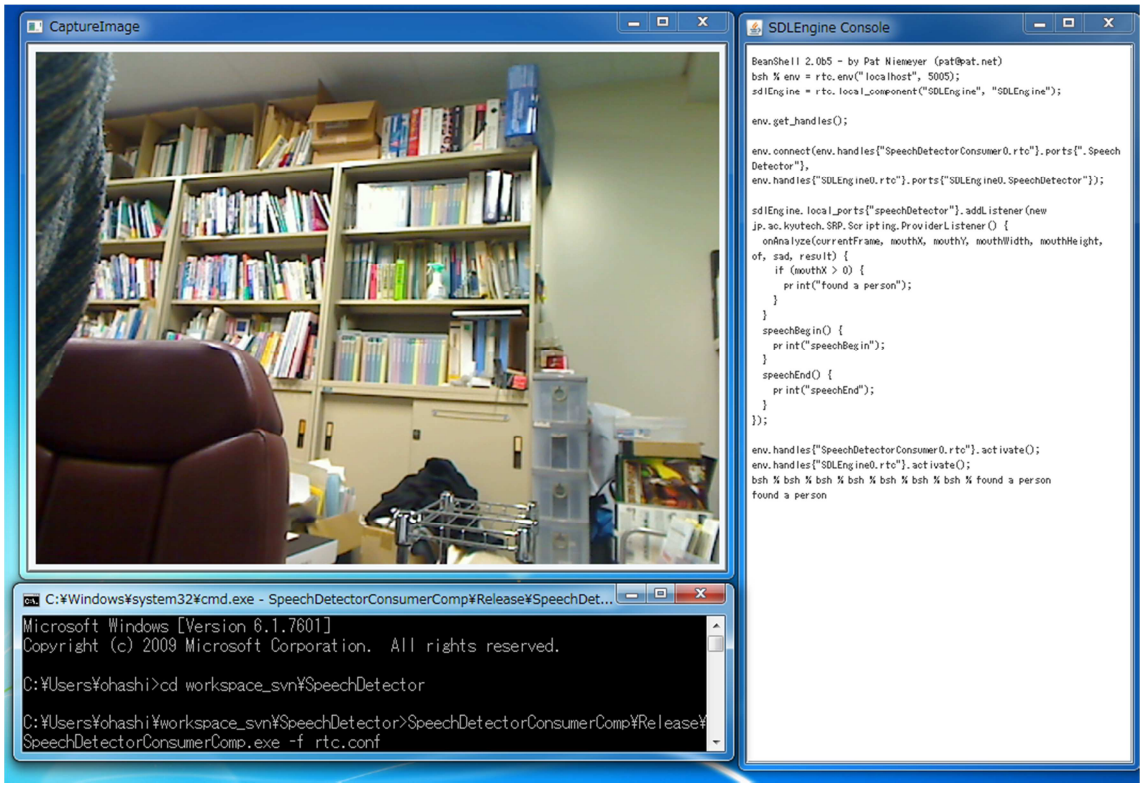
(2) デモ

1. スクリプトの投入

行動計画コンソールに以下のスクリプトをコピペしたのち enter を入力して実行して下さい。

```
// name サーバの設定
env = rtc.env("localhost", 5005);
// SDLEngine をローカルオブジェクトとして起動
sdlEngine = rtc.local_component("SDLEngine", "SDLEngine");
// name サーバに登録されているオブジェクトのハンドラを取得
env.get_handles();
// SpeechDetectorConsumer0.rtc のポートと SDLEngine0.rtc のポートを接続
env.connect(env.handles["SpeechDetectorConsumer0.rtc"].ports[".SpeechDetector"],
                env.handles["SDLEngine0.rtc"].ports["SDLEngine0.SpeechDetector"]);
// sdlEngine の SpeechDetector ポートに対応するリスナーを設定（実装に相当）
sdlEngine.local_ports["speechDetector"].addListener(new jp.ac.kyutech.SRP.Scripting.ProviderListener() {
  onAnalyze(currentFrame, mouthX, mouthY, mouthWidth, mouthHeight, of, sad, result) {
    // 口の X 座標が正ならば人を見つけたと判断する
    if (mouthX > 0) {
      print("found a person");
    }
  }
  speechBegin() {
    print("speechBegin");
  }
  speechEnd() {
    print("speechEnd");
  }
});
// SpeechDetectorConsumer0.rtc と SDLEngine0.rtc を活性化する
env.handles["SpeechDetectorConsumer0.rtc"].activate();
env.handles["SDLEngine0.rtc"].activate();
```

次の図では、左上に USB カメラから取り込んだ画像と認識結果を表示するウィンドウ、右に SDLEngine のコンソールのウィンドウ、左下が cmd.exe から SpeechDetectorConsumer コンポーネントを起動したウィンドウを示します。



USB カメラに人の顔の正面が写ると左右の目、鼻、口を検出し、顔領域を推定します。いずれかの領域が検出されると onAnarize() が呼びだされます。また、口領域の開閉により発話の開始と終了を検出し、speechBegin() と speechEnd() がそれぞれ呼び出されます。

以下の図では、両目と口が検出されていて、左下の SpeechDetectorConsumer コンポーネントに検出結果が表示され、SDLEngine のコンソールにも onAnarize() が呼び出されて且つ口領域の x 座標が正のときに “found a person” と表示され、speechBegin() と speechEnd() が呼び出されたときにそれぞれ “speechBegin” と “speechEnd” が表示されている様子です。先のスクリプトの print() の部分を必要な処理に書き換えてご活用ください。

終了するには、各ウィンドウの×を押して閉じてください。