PROSODICA REAL-TIME PROSODY TRACKER

Eric Nivel & Kristinn R. Thórisson

RUTR-CS08002

January 2008

Reykjavik University
Department of Computer Science

Technical Report

1 User Documentation

0.0 Preamble

The Prosodica proposdy tracker is software that can extract pitch and silences from live speech. Prosodica is relatively accurate, has a very small 16msec latency on extraction and requires little processing power.

This document is intended for users and developers who want to use Prosodica in their setup. Notice that Prosodica has been set up to be used with the Psyclone™ system [2], although stand-alone execution should not be complicated for an experienced C++ developer. Details on the software are provided in the Appendix. For more information please contact Eric Nivel or Kris Thórisson (eric | thorisson@ru.is).

1.1 Description

Prosodica is a software module that analyses the pitch and silences of a realtime, continuous speech, and describes the status of:

- instantaneous pitch and derivative;
- speech on/off;
- speech paused;
- humming;
- speech without a pitch: typically, the analyser has failed to assign a pitch to the sound, but still, the amplitude is high enough to differentiate it from noise;
- average ininterrupted speech segment duration;
- Average inter-segment duration.

The pitch analysis algorithm is based on the one described in [1].

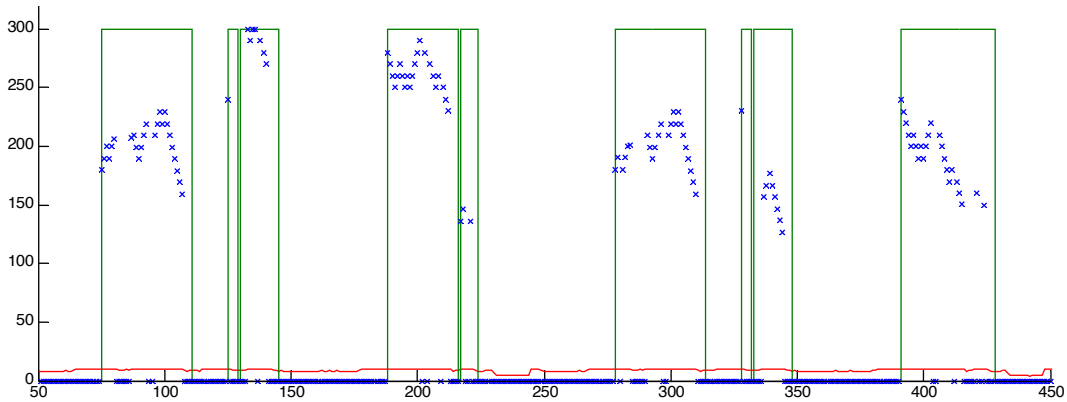

Figure 1. Sample outputs from processing continuous speech. Units: *Horizontal: samples, e.g. typically ~15ms; Vertical: frequency in Hz.* Key*: in blue: instantaneous pitch; in green: Speech.On / Speech.Off enveloppe; in red: amplitude (logarithmic scale).*

1.2 Requirements

- Windows XP SP2 or higher;
- Psyclone 1.1.7 or higher; Prosodica runs as a Psyclone module (internal or external). Details on Psyclone are available in [2].
- DirectX9 SDK or higher;
- RTAudio 3.3. Prosodica uses RTAudio [3] for accessing the audio driver.

Input: microphone in.

Output: Psyclone messages.

Performance: 15 ms latency on a dual P4 3.2GHz, 2GB RAM, 1024 bytes buffer size, typical.

1.3 Setup

Prosodica can be used either as an external (CADIA.Module.Prosodica.exe) or an internal (CADIA.Module.Prosodica.dll) module. The two corrresponding complete configurations are given in the PsySpec.xml files included in their respective projects. Notice that the following assumes familiarity with the Psyclone system.

Each version of Prosodica comes in two sub-versions: with or without Matlab plotting (CADIA.Module.Prosodica.Matlab.xxx).

The setup is specified in the (provided) PsySpec.xml configuration files. Several instances of the Prosodica internal module can run simultaneously: simply use the same dll with different names, as indicated in the commented zone in the confiduration file. Limitation: no more than one instance can use the Matlab plotting facilities.

1.4 Running Prosodica

Start Psyclone with the adequate PsySpec.xml file as its argument. If the external module is used, start it manually and provide two command line arguments, the address and port of Psyclone (default values: localhost 10000).

1.5 Parameter Specification

Prosodica can be configured to run according to several parameters. These are defined in the PsySpec.xml configuration files. Parameters are either static – i.e. will be read once, at startup time -, or dynamic – i.e. their values can be set at runtime using the Psyclone API.

The PsySpec.xml configuration files provide default values for all of the parameters.

ModuleID - static

Integer. Identifies an instance of Prosodica.

DeviceID - static

Integer. Identifies the sound card driver. When Prosodica starts, it displays a list of the available drivers; just pick up the one you need (featuring at least one input)

BufferSize - static

Integer, in bytes. 768 bytes seems to be the minimum, and 1024 the best value. In general, the extra latency brought by choosing a higher value is not worth the improvement in accuracy. Inferior values – e.g. 512 bytes – lead to significant loss of accuracy.

SamplingFrequency - static

Integer, in Hz. Choose it accordingly to your driver's capabilities.

AmplitudeThreshold - static

Double, in [0,1]. Tuning parameter for the pitch analysis. 0.85 suits 99% of the cases.

NoiseThreshold - static

Double, in. Under that value the signal will be considered as noise, and will contribute to the emission of a Speech.Off or Speech.Pause message.

PitchHighPass – dynamic

Double, in. Value under which the pitch value is considered unsignificant (i.e. zero).

PitchLowPass – dynamic

Double, in. Value over which the pitch value is kept down to this value.

PitchDeltaConstraint – dynamic

Double, in. If |last_pitch_value-current_pitch_value|>constraint, limits the current value by the constraint.

PitchMonitorWindow – static

Integer, in samples. Series of pitch values are recorded in a circular buffer. The monitor window is the size of the buffer.

PitchSamplingWindow – static

Integer, in samples. <= PitchMonitorWindow.

PauseThreshold – dynamic

Integer, in ms. Minimum duration of a silence before being considered a pause in speech.

SegmentIntervalThreshold - dynamic

Integer, in ms. In a continuous speech, the pitch can fall to zero for a short period. If that duration is under the value of this parameter, no Speech.Off will be issued: instead the speech will be considered as being uninterrupted, but will carry a zero pitch value.

SpeechOnDelay - dynamic

Integer, in samples. Defines the number of samples required to trigger a SpeechOn.

HumPitchThreshold - dynamic

Double, in. The pitch variation under which the pitch shall stay in order to be a candidate for a hum. Input signal sensitive.

HumDurationThreshold - dynamic

Double, in ms. The minimum duration during which a hum shall be sustained to yield an effective hum information in Speaking messages.

1.6 Message Specification

in: Psyclone.System.Ready

Starts the module.

Arguments: none.

Thereafter, all message types end with *.id*: this corresponds to the ModuleID parameter set in the early stage of the configuration. Message data appear under the form
<data argName="value" ... />, following the XML syntax.

in: RU.Prosodica.Stop.id

Stops the module.

Arguments: none.

Thereafter, in addition of their own specific data, each message conveys the following *common data*:

timestamp: int64, from 01.01.1970.

latency: unsigned short, in ms.

out: RU.Input.Prosodica.Speech.On.id

Sent whenever speech is detected, from a speech-less state.

Arguments:

pitch: float, in Hz, optional

speechNoPitch: in {0,1}, optional

out: RU.Input.Prosodica.Speech.Off.id

Sent whenever no speech is detected anymore from a speech state.

Arguments:

pitch_value_count: `integer`, number of pitch value pairs (see below)

array of pitch value pairs, indexed by i: pitch_value_i (float, in Hz) pitch_time_i (int64 ms)

averageInterSegmentDuration: long, in ms, reset after each pause

averageSegmentDuration: long, in ms, reset after each pause

out: RU.Input.Prosodica.Speaking.id

Sent every sampling time (e.g. 15 ms) whenever a pitch or a transition in speech is detected, i.e. humming, speech with no pitch.

Arguments:

pitch_value_count：`integer`, number of pitch value pairs (see below)

array of pitch value pairs, indexed by i: pitch_value_i (float, in Hz) pitch_time_i (int64 ms); optional

hum: integer, in {0,1}; optional

speechNoPitch: integer, in {0,1}; optional

out: RU.Input.Prosodica.Speech.Pause.id

Sent whenever a pause in speech is detected. A pause is a long interval (duration specified by the *PauseThreshold* parameter) occurring after a speech off message.

Arguments: none.

2 Developer Documentation

The following assumes a fair degree of familiarity with Psyclone.

2.1 Requirements

VisualStudio C++ 2005, SP1

RTAudio 3.0.3 – Prosodica uses the DirectSound configuration.

2.2 Compilation

Two projects are provided (including configuration files), for building Prosodica in both its external and internal module incarnations. These projects are named InternalModule and ExternalModule. They depend on a third one: Core.

2.2.1 Setting the environment variables

The following environment variables are used by the VisualStudio project configurations:

- DXSDK_DIR: directx sdk install directory
- MATLAB_DIR: matlab install directory; only if you use Matlab
- PSYCLONE_DIR: psyclone install directory: for the debug command line
- PSYCLONE_SDK_DIR: psyclone sdk install directory: for both internal and external modules
- PSYCLONE_CPP_AIR_DIR: psyclone CppAIRPlug install directory; for the external module
- RT_AUDIO_INC_DIR: RTAudio include directory
- RT_AUDIO_LIB_DIR: directory containing RTAudioLib.lib

2.2.2 Building Prosodica

The project can be compiled to use Matlab plotting facilities.

To switch it on:

- in Core.h, define the preprocessor directive USING_MATLAB.
- link Core against libeng.lib and libmx.lib
- rebuild everything

To switch it off:

- in Core.h, undefine the preprocessor directive USING_MATLAB.
- remove the link dependencies on libeng.lib and libmx.lib in Core
- rebuild everything

2.2.3 Running Prosodica

Start Psyclone with the PsySpec.xml file as its argument. If the external module is used, start it manually when Psyclone is up and running.

References

[1] http://online.physics.uiuc.edu/courses/phys199pom/NSF_REU_Reports/2005_reu/Real-Time_Time-Domain_Pitch_Tracking_Using_Wavelets_Final_Presentation.pdf

[2] Psyclone website: http://www.cmlabs.com/psyclone

Downloads: http://www.mindmakers.org/projects/Psyclone

[3] RTAudio 3.3, website: http://www.music.mcgill.ca/~gary/rtaudio

APPENDIX – Software Package

The software package contains the following items:

- Binaries:
  - RTAudio.lib – RTAudio 3.3 library, provided in both release and debug versions;
  - CADIA.Module.Prosodica.Matlab.dll – Prosodica internal module, using Matlab for plotting the output; both in debug and release versions;
  - CADIA.Module.Prosodica.Matlab.exe - Prosodica external module, using Matlab for plotting the output; both in debug and release versions;
  - CADIA.Module.Prosodica.dll - Prosodica internal module, *not* using Matlab for plotting the output; both in debug and release versions;
  - CADIA.Module.Prosodica.exe - Prosodica external module, *not* using Matlab for plotting the output; both in debug and release versions;
- Configuration files:
  - InternalModule\PsySpec.xml – configuration file for the Prosodica internal module;
  - ExternalModule\PsySpec.xml – configuration file for the Prosodica external module;
- Source code: provided as a Visual Studio solution containing 3 projects:
  - Core: the main algorithm and hooks to the audio device;
  - InternalModule: encapsulation of Prosodica in a Psyclone internal module;
  - ExternalModule: encapsulation of Prosodica in a Psyclone external module.

  PitchTracker.h and .cpp defines the main template class for instantiating Prosodica either in an internal (see InternalModule\cranks.cpp) or an external module (see ExternalModule\main.cpp).

  WaveletPitchProcessor.h and .cpp defines the pitch tracking algorithm.

  PitchTracker.Evaluator.h and .cpp defines the prosody analysis, exploiting data acquired by the WaveletPitchProcessor.

  AudioDevice.h and .cpp encapsulates the RTAudio library.

  MatlabEngine.h and .cpp provide utility classes for sending data and commands to an instance of Matlab.