**Landmark Processing Suite Manual**

1. Introduction
   * Data includes various phonetic objects (i.e. phones, landmarks, accents) and their relationships.
   * [diagram]

This landmark processing package involves four major components:

1. external knowledge source including CMU lexicon (cmudict.0.7a), phone-transition-to-landmark transition table (lm\_predict\_table.py ), conversion between different landmark notations;
2. data model including TextGrid which contains Praat display and general query functions (TGProcess.py), and ExtendedTextGrid class (ExtendedTextGrid.py) which manages specific context data
3. GUI display, i.e. Praat
4. Data mining tools for context-dependence discovery, i.e Python Orange machine learning tool

This manual presents its functionalities so that it can be used and understood by people without extensive programming knowledge . However, this version is still very preliminary and contains various design issues, mainly:

1. Data storage. Data is stored as python objects. This makes it hard to separate database from the programs and python platform
2. Encapsulation. Inter-dependencies in modules, mainly that between data model and display, will make it difficult to update the software.

For these reasons, this version is intended to be used as-is and as a reference for designing future systems; massive extension on this package is not recommended (until at least issue (ii) is resolved).

1. Usage
2. Preparation

External knowledge, such as lexicon and landmark conversion rules, are converted into python objects by LexiconExtract.py, … The executable is tables.py which runs through all the required procedures.

1. Playing ExtendedTextGrid objects

1) Read file

\*.textgrid (praat) file: ExtendedTextGrid(f='conv07.textgrid')

\*.pkl (TextGrid python object) file: ExtendedTextGrid.readObject('conv07.pkl')

2) Predict landmarks given words, hand-labeled landmarks, and comments (presumbly

named "Words", "Landmarks", "Comments" respectively):

- Run tg.putPhns()

- Run tg.predictLM()

3) Align observed and predicted landmarks

- Run tg.convertLM() to change the format of hand labels (ignore the warnings for now)

- Run tg.linkLMtoWords("pred. LM") and tg.linkLMtoWords("act. LM")

- Run tg.alignLM()

4) LM context in praat as a tier

- lm\_tier.links("Words"), lm\_tier.links("phones") etc

5) LM alignment as a tier

- tg.aligned()

6) Write file

- tg.writeGridToPath('conv07') (notice omission of extension name; both .textgrid and

.pkl will be created.)

- alias: tg.saveAs('conv07')

- save with the original name: tg.save()

7) Write out context in a tab-delimited format

- tg.saveTab()