Special Lecture (406)
Spoken Language Dialog Systems
Working with the CSLU Speech Toolkit

Rolf Schwitter
schwitt@ics.mq.edu.au
PhD Scholarship at Macquarie University

• A Natural Language Interface to a Logic Teaching Tool.
• HyperProof is a popular computer-based logic teaching tool.
• It comes with the largest-selling introductory logic textbook, Language, Proof and Logic by Jon Barwise and John Etchemendy.
• Goal:
  – Make it possible to write logic problems in English.
  – Make it possible to express the resulting proofs in English.
• More information:
Today's Agenda

- Recap: CSLU Speech Toolkit
- Preference Settings
- Login Object
- Subdialogs
- Digit Recognizer
- Natural Speech Prompts
- Speech Grammars
Recap: CSLU Speech Toolkit
Recap: CSLU Speech Toolkit

Used Objects:

- Start
- Generic
- Subdialogue
- Goodbye
Recap: CSLU Speech Toolkit

Speech Recognition
(double click red port)

anything_else

Speech Synthesis
(double click icon)
Audio Parameters

- Beep before recognition
- Maximum Record Duration: 4 s
- Leading Silence Duration: 5000 ms
- Trailing Silence Duration: 300 ms
- Record Backoff: 150 ms
- Voice Detection Threshold: 20 %stdv

[Calibrate]
Audio Parameters

• Maximum Record Duration
  – specifies how long the recogniser will record an utterance

• Leading Silence Duration
  – specifies how long the recogniser continues to record if it is detecting only silence

• Trailing Silence Duration
  – specifies how long the recogniser will continue to record after the user stops speaking (→ natural pauses)
Audio Parameters

- Record Backoff
  - specifies the time between beep and recording
- Voice Detection Threshold
  - set this value via microphone calibration
Preference Setting: Repair

- The recogniser can only choose between defined vocabulary.
- With Repair turned "off"
  - the recognizer is forced to decide
    between the vocabulary regardless of the confidence score.
- With Repair turned "on"
  - the recognizer is allowed to reject
    all vocabulary when it is not confident.
- This is called "Out of Vocabulary Rejection".
Preference Setting: Repair

- Select Preferences > General tab and turn Repair "on".

![Global Preferences window](image)
Preference Setting: Recognition

- The following options are available for individual objects:
Preference: Recognition

- Out of Vocabulary Rejection Median
  - a lower number rejects more and a higher number less
  - a high number is more forgiving of incorrect pronunciation.

- Word Spotting Median
  - a low number spots less and a high number spots more.

- Grammar Garbage Threshold
  - rejection setting for grammar type recognizers
Subdialogs

- A subdialog is like a mini-canva.
- You can build structures of states within them to hide information.
- When you drop the subdialog object on the canvas a menu appears.
- Select "New Master Subdialog" from the menu.
- This will create a new empty subdialog that you can rename.
Example: Subdialog
Example: Subdialog

- Double clicking on the "login_subdialog" state opens a subdialog.
Example: Subdialog

- The "login dialog" can be encapsulated as a subdialog:
Example: Subdialog

• Configure the subdialog objects as follows:
  
  ![sign_in]
  Please sign in.

  ![login]
  "Add a few names".

• The prompt for the feedback state might be:

  ![feedback]
  Hello $user, nice to hear you again.
Recognizing Digits

- The digit object has a specialized grammar to recognise digits.

```
$dig = zero | oh | one | two | three | four |
     | five | six | seven | eight | nine;
$sep = *sil%* ["any%"] *sil%*;
$grammar = [$sep] < $dig [$sep] > [$sep];
```
Recognizing Digits

- The digit object has only one output port.

- Any branching on the recognition result must be done with a conditional object.
Example: Digits
Natural Speech Prompts

• RAD can substitute synthetic speech by recorded voice.
• The recorded speech can be aligned with the talking heads so that lip, face and body movements are in sync.
• Double click on the desired state to open its configuration dialog, select the Recorded tab and then Edit . . .
• This will open BaldiSync with natural speech.
• BaldiSync allows you to record your voice, enter your transcription, align the text and audio, test it, and save the file.
Natural Speech Prompts

![bsync - Baldi with natural speech](image)

- Sound
- Sound energy
- Spectogram
Speech Grammar

- So far we have only used single words, phrases and built-in features (such as *any or *sil) in the recognition states.
Speech Grammar

- Finite-state grammars are more flexible and allow generalizations.
- For example, the user may respond to
  - What size and type of pizza do you want?
  with
  - a small pepperoni
  - a medium vegetarian pizza
  - a large cheese one
Speech Grammar

• Speech grammar should describe all possible utterances.
• We can start with three variables:
  – one that represents the size of the pizza,
  – one that represents the topping choices, and
  – one that combines all possibilities of the first two.
• For example:

```
ssize = small | medium | gigantic;
$topping = vegetarian | pepperoni | cheese;
$pizza = [*sil%% | *any%%] $size $topping [*sil%% | *any%%];
```
Speech Grammar

- The names of the variables (or non-terminals) in this grammar are $size, $topping, and $pizza.
- The string associated with $size is small | medium | gigantic.
- The | character means "or".
- *sil and *any are special built-in features of RAD.
- *sil is used to recognize silence.
- *any is like a garbage/noise collector.
## Syntactic Rules

<table>
<thead>
<tr>
<th>Character</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>[]</td>
<td>delimit parts of the grammar which are optional</td>
</tr>
<tr>
<td>&lt;&gt;</td>
<td>delimit parts of the grammar which can be repeated one or more times</td>
</tr>
<tr>
<td>()</td>
<td>delimit parts of the grammar which can be repeated zero or more times</td>
</tr>
<tr>
<td></td>
<td>or</td>
</tr>
<tr>
<td>%%</td>
<td>indicates that word will not appear in the result, even if recognized</td>
</tr>
<tr>
<td>%</td>
<td>will substitute the next word in the result if the first word is recognized</td>
</tr>
</tbody>
</table>
Example: Speech Grammar
Configuration

• Configure the "PizzaType" state.
• The prompt for this state is
  – What size and type of pizza do you want?
• Double click on the red port of the "PizzaType" state.
• This will open the vocabulary dialog.
Example: Adding Grammar Rules

- In the vocabulary dialog select the check box Grammar, provide a name for the grammar, and add the grammar rules.
Example: Extracting Words

- After entering the grammar, select "Extract Words".
- All individual words will appear in the vocabulary area.
- Select "Update All" to get the pronunciation strings.
Example: Confirmation

• Click on the "PizzaType" object, select the "OnExit" tab, and add the following Tcl code to the textfield:

  ```tcl
  set pizzaSize [lindex $PizzaType(recog) 0]
  set pizzaTopping [lindex $PizzaType(recog) 1]
  tts "You ordered a $pizzaSize $pizzaTopping pizza."
  ```

• The first two lines assign the two words to individual variables.
• The last line uses those two variables to synthesize a confirmation to the user.
• Note: `lindex` retrieves an element from a list.
Building a Pizza Ordering Service

• Specification
  – Pizzas are three sizes: small, medium, and large.
  – Pizzas may have one or more of these toppings:
    cheese, pepperoni, sausage, peppers, pineapple, and tomatoes
  – Drinks are three sizes: small, medium, and large.
  – Drinks may be Coke, Pepsi, Diet Pepsi, lemonade, or water.
  – An order may contain zero or more pizzas and zero or more drinks.
Building a Pizza Ordering Service

- Construct a conceptual model (call flow diagram)
- Construct scenarios (Wizard of Oz experiment)
  - human experimenter simulates the automated system
  - experimenter reads prompts (from the spec)
  - caller provides information
  - observer records interaction
- Develop the pizza ordering application