Speech Input for Live Performance
An Impromptu Dialogue Between the Computer and the Artist

Benoit Favre, Mickael Rouvier, Frédéric Béchet and Rocio Berenguer

Abstract Real time speech recognition can fuel artistic creation by providing a medium for arranging a show while it is being performed, and by forcing the artist to improvise when the ASR fails. We propose a software suite designed to integrate speech transcripts and spoken commands into a live performance. We adapt the Kaldi ASR toolkit to show conditions and build a user interface for describing and running a performance. The software is deployed by the Pulso dance company to run its Homeostatsis show.

Key words: Art-sciences, Speech recognition, Dialog systems, Adaptation, Artistic performance

1 Introduction

Art and sciences, while often seen as antagonist are in fact close matters as researchers often provide advanced technical solutions for supporting artistic concepts. The glitch (error or bug) is a concept of our modern society where technology is omnipresent, and at the same time imperfect, and fails to achieve what it was designed for. It has become a source of inspiration for artists who explore the relation between humans and their technological environment.

In this paper, we present a support tool for artistic performance which allows the artist to interact with a dialog system on the stage and let it play its role as an actor of the performance with its imperfections. While the artists taunts the system with

Benoit Favre, Mickael Rouvier, Frederic Bechet
Aix-Marseille Université, CNRS, LIF UMR 7279, 13000, Marseille, France e-mail: `first.last@lif.univ-mrs.fr`

Rocio Berenguer
PULSO, La cité des associations, 13001, Marseille, France e-mail: `creacionesinpulso@gmail.com`

commands, suggestions and feedback, the misunderstanding created by understanding errors from the system fuel the improvisation of the artist.

The system was conceived and deployed for the Homeostasis show by dance company Pulso [1], where its principal dancer explores the relation between her body and her mind, in an allegory of a failed software upgrade, materialized though a modern flamenco choreography. Video, sound and speech input are digested by the system which replies through computer generated graphics and music, as well as playing recorded speech (See Figures 1 and 2). This paper focuses on the spoken dialog aspects of the performance. Our main contributions are:

- A methodology for making dialog systems usable in live performance
- An implementation in the Kaldi toolkit and an adaptation of the models to performance conditions
- The release of generic software for artists

2 System description

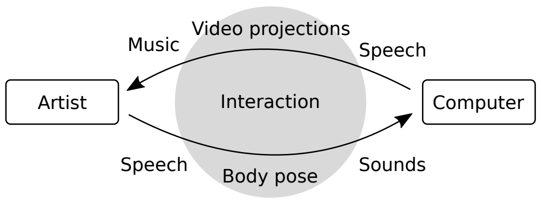


Fig. 1 The computer is a real actor in the performance as it captures multimodal input (Kinect, piezo microphones, headset) from the artist, and decides to perform actions which will impact the show (music, graphics, speech recordings). Errors made by the system force the artist to improvise.

The system presented in this study implements a dialog between a computer and a performer based on a flexible scenario described as an XML file. Similarly to the system presented in [2], the main function of this system is to align the performer's speech to the script, then to trigger actions at some specific point in the scenario. The main difference in our system is the fact that the scenario is *flexible*, leading to an interaction between the computer and the performer: the performer can jump from one section to another in the show, change the sections order, add unplanned text; the computer can perform an approximate matching between what is output by the Automatic Speech Recognition (ASR) module and the expected script, then it can make a decision on whether executing a planned action or ignoring the current message. Both the performer and the computer can make errors, leading to a dialog between them.

[1] `http://pulsopulso.com`

This section describes the software we have created so that automatic speech recognition can be integrated in a real time performance. UI parts of the software are developed in python with Gtk3 while backend parts which are more processing-intensive are implemented in C and C++. At the heart of the system, a performance description file defines the commands that can be triggered by speech. The Kaldi ASR system generates a transcript and the WFST-based SLU processes it to generate actions. Commands are sent to the main multimedia controller through the OSC protocol and a UI can be used by an assistant to recover from errors. The current software is open source and available[2].

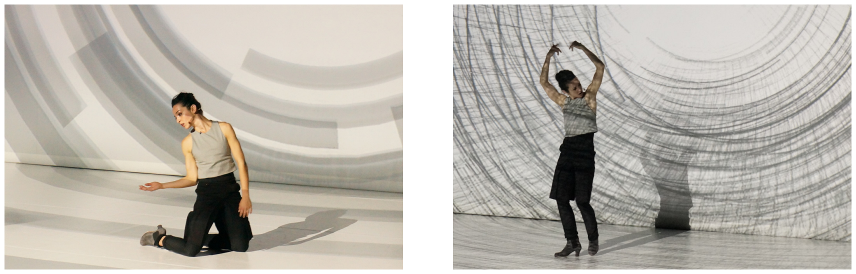


Fig. 2 Pictures of the artist using the system in the Homeostasis show.

2.1 ASR module

For an artistic performance based on speech recognition to be successful, the ASR system has to provide a sufficient level of performance which out-of-the box systems, such as smart phone or desktop speech recognition do not provide. Loud music, environment reverberation and ambient noise, as well as accented speech and emotion-bearing speech are the main challenges we faced. In addition, real time recognition with a very low delay and high reliability requirements preclude the use of off-site services such as those provided by current industry leaders. On the other hand, the number of speakers is generally closed, and the text of the performance can be limited to a small vocabulary—predefined word sequences can be chosen for triggering actions and interactions. In the context of the *glitch* concept, ASR errors are beneficial in that they force the artist to improvise, however, too many errors lead to chaos. In our system we had to compromise between freedom of expression and control of the speech-related interactions to provide interesting material for artistic performance.

In our experiments we used the ASR Kaldi toolkit [3]. The show is transcribed using the online decoder in Kaldi. By “online decoding” we mean decoding where

[2] `http://pageperso.lif.univ-mrs.fr/~benoit.favre/homeostasis`

the features are coming in real time, and you don't want to wait until all the audio is captured before starting the online decoding. The online decoding is based on the neural net based setup, which means that a Deep Neural Network (DNN) is used as acoustic model. The acoustic models are trained using the TED-LIUM corpus (118 hours) and an in-domain corpus (less than 1 hour), made of performances (repetitions) from the Homeostasis show.

Because of the online-decoding context, no adaptation through batch decoding was possible. We proposed instead an on-the-fly adaptation method where the adaptation is done continuously by collecting the necessary statistic and normalizing the acoustic features using the i-vector paradigm. More details can be found in [5] and [4].

2.2 Spoken Language Understanding

The goal of the Spoken Language Understanding (SLU) module is to translate word sequences into actions. For each section in a scenario, a script contains all the text that could be recognized as well as the actions that have to be triggered when a given word sequence is recognized. Two kinds of actions are considered: the *jump* actions that exit the current section in the show and move to another section; the *external* actions that send to the multimedia server a command for displaying or playing a given image, video or sound.

In our SLU module this translation process is implemented by finite state transducers using the OpenFST library. Each section of the show, described in an XML document, is turned into a transducer that implements the characteristics of the section (fixed order between the subsection or free order, repetition allowed or not, . . .). In order to be robust to ASR errors, this transducer is augmented with alternative paths that can model word deletion, insertion or substitution.

When a new word is recognized by the ASR module, it is added to the ASR buffer. If the distortion between a word sequence in the script and the ASR buffer is below a threshold, the current pointer in the script is moved to a new position, possibly triggering an action and the buffer is erased. If no acceptable match is found, the word is sent to the multimedia server with no action attached to it. All these operations are made through the OpenFST composition operator between the ASR buffer and the transducer corresponding to the current section.

2.3 Scenario description format in XML

The text and commands that will be used during the artistic performance must be described in an XML file which sets the various necessary constraints on ASR and SLU. A performance consists of a list of `sections` each of which can be entered with a special command recognized at any time regardless of the state of the SLU.

In addition, a fallback command is defined to end a section and return to the out-of-section state. Actions can be bound to start and end of sections. Then, a `section` is made of `sequences` which contain recognizable text with `keywords` that generate actions. A `keyword` can contain any number of words and the associated action is generated as soon as the whole phrase is recognized. `Sequences` can follow a strict or variable order, with or without repetitions. `Sequences` without a strict order are ended with a special `#end` action.

The performance description editor, showed in Figure 3 consists in a text editor with XML syntax highlighting, and a validator which checks the created file. Each time the performance is modified, models need to be recompiled. A pronunciation lexicon is created from a basic dictionary [3], words manually adapted to the accent of the artist, and a CRF-based phonetizer for OOV words, trained on the dictionary using CRFSuite [1]. All the word sequences from the performance description are concatenated to train a basic 3-gram language model. Then, the Kaldi search space is created by composing the acoustic model targets, the lexicon and the language model. This procedure is completely automatized to remove the hassle for artists to freely modify the lexicon and language model used for the show.

2.4 Open Sound Control

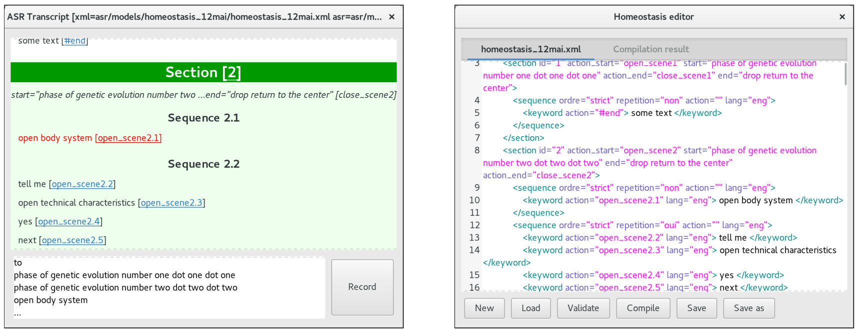
Open Sound Control (OSC) is a widely used protocol for controlling real time multimedia performance in the artistic community. It consists in a UDP protocol for exchanging commands over the network with high resolution timing and convenient naming schemes. Transcripts generated through Kaldi and actions triggered by the SLU module are both sent as OSC packets as soon as they occur so that they can be processed by the the computers of the multimedia setup to perform related actions or display the raw transcript. Two types of OSC commands are sent:

- `WORDS(utterance_id): text` which corresponds to the latest transcript `text` for the utterance number `uttrance_id`. The best partial hypothesis is sent and modified as soon as the ASR systems consumes more audio frames.
- `ACTION: action_id` which corresponds to a recognized action by the SLU module. The `action_id` is replaced by the action field defined in the performance description file.

2.5 User interface

A user interface was created for controlling the rolling out of actions: the sections, subsections and keywords from the performance description file are displayed in a convenient UI which shows the transcript as it is generated and actions as they

[3] CMUDict from `http://www.speech.cs.cmu.edu/cgi-bin/cmudict`.



some text [#end]

start="phase of genetic evolution number two ...end="drop return to the center" [close_scene2]

open body system [open_scene2.1]

tell me [open_scene2.2]

open technical characteristics [open_scene2.3]

yes [open_scene2.4]

next [open_scene2.5]

```
to
phase of genetic evolution number one dot one dot one
phase of genetic evolution number two dot two dot two
open body system
...
```

```
    <section id="1" action_start="open_scene1" start="phase of genetic evolution number one dot one dot one" action_end="close_scene1" end="drop return to the center">
      <sequence ordre="strict" repetition="non" action="" lang="eng">
        <keyword action="#end"> some text </keyword>
      </sequence>
    </section>
    <section id="2" action_start="open_scene2" start="phase of genetic evolution number two dot two dot two" end="drop return to the center" action_end="close_scene2">
      <sequence ordre="strict" repetition="non" action="" lang="eng">
        <keyword action="open_scene2.1" lang="eng"> open body system </keyword>
      </sequence>
      <sequence ordre="strict" repetition="oui" action="" lang="eng">
        <keyword action="open_scene2.2" lang="eng"> tell me </keyword>
        <keyword action="open_scene2.3" lang="eng"> open technical characteristics </keyword>
        <keyword action="open_scene2.4" lang="eng"> yes </keyword>
        <keyword action="open_scene2.5" lang="eng"> next </keyword>
```

Fig. 3 Screenshots of the live-show and editor UI.

are performed (See Figure 3). Technical staff can resynchronize the SLU module at any time by clicking a section or keyword. All missing actions are performed up to that keyword. This way, if the system starts failing in the middle of the show, it's possible to control how much chaos is introduced.

3 Conclusion

In this paper we have described a system for integrating speech recognition in artistic performance, showing how to adapt the Kaldi system to the constraints of a live show, and providing software that can be directly used by artists.

As for future work, we would like to implement support for different spoken languages, allow more interactions over OSC, in particular to command the system, and improve the flexibility and accuracy of the SLU module. We also envision that the computer could play a much more involved role in the performance by being impersonated by an avatar.

References

1. Okazaki, N.: Crfsuite: a fast implementation of conditional random fields (crfs) (2007). URL `http://www.chokkan.org/software/crfsuite/`
2. Peters, S., Bui, T.H.: A speech-driven e-book technology prototype demo. In: 2014 IEEE Spoken Language Technology Workshop (SLT 2014), South Lake Tahoe, Nevada, USA (2014)
3. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlíček, P., Qian, Y., Schwarz, P., et al.: The kaldi speech recognition toolkit (2011)
4. Rouvier, M., Favre, B.: Speaker adaptation of DNN-based ASR with i-vectors: Does it actually adapt models to speakers? In: Interspeech, Singapore (2014)
5. Saon, G., Soltau, H., Nahamoo, D., Picheny, M.: Speaker adaptation of neural network acoustic models using i-vectors. In: Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on, pp. 55–59. IEEE (2013)