Speechrecognition

Robert Feldhans

9.11.2018

Seminar Robocup

Motivation

Why even bother?

- faster and more general way to give robots commands
- a necessity for casual users
- user does not need additional hardware

Content

What is Speechrec? What does it consist of?

Content

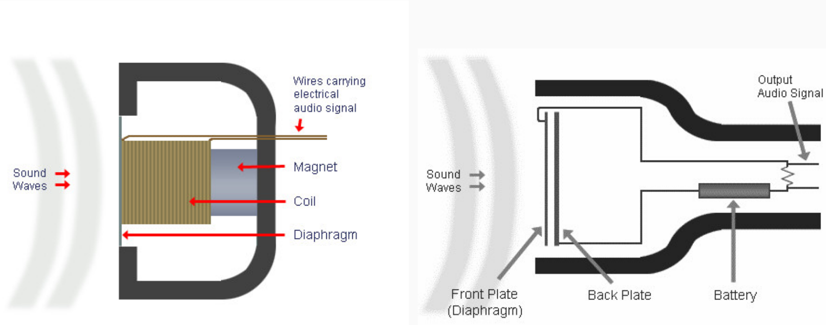
1. Hardware
2. Localisation
3. Signal Enhancing
4. Voice Activation Detection
5. Speaker Recognition
6. Speech Recognition
7. Natural Language Processing

Quick example

content...

Hardware

Microphones



(a) dynamic microphone **(b)** condenser microphone

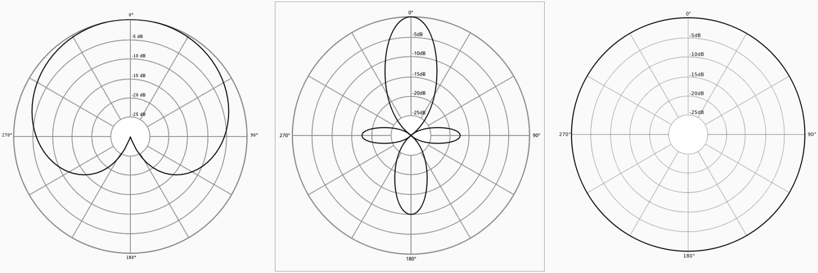
Figure 1: Different kinds of microphones. Source: http://www.onlinetuner.co

Audio

Microphones provide audio defined by:

rate:	resolution in time domain
bitrate:	resolution in quality domain
endian:	representation of signal
channel:	amount of channels
interleaving:	representation of signal by channel

Microphones: Polar Patterns



(a) cardioid pattern **(b)** directional pattern **(c)** omnidirectional pattern

Figure 2: Diffrent kinds of microphone pattern. Source: Wikipedia

Microphone Arrays

Why?

- more sensors are always better
- several microphones can be linked together to reduce noise
- by analysing the distribution of signals and frequencies in time we can even detect the direction from where a sound was emitted

Microphone Arrays II

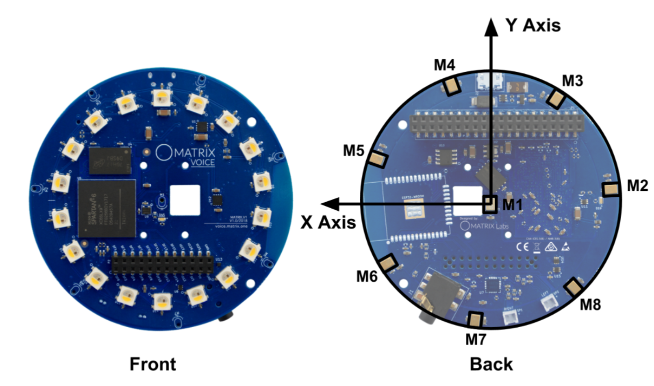


Figure 3: Matrix Voice, a microphone array. Source: https://matrix-io.github.io/matrix-documentation/matrix-voice/resources/microphone/

What do we use?

Tobi:

What do we use?

Tobi: One or two directional microphones, 2 additional for sound source localization.

Pepper:

What do we use?

Tobi: One or two directional microphones, 2 additional for sound source localization.

Pepper: An array of four omnidirectional microphones inside of the head.

Tiago:

What do we use?

Tobi: One or two directional microphones, 2 additional for sound source localization.

Pepper: An array of four omnidirectional microphones inside of the head.

Tiago: Stereo microphones in the torso (black spots under his head).

Alsa

The Advanced Linux Sound Architecture manages all your physical and virtual microphones and speakers.

- It controls volume and gain
- It controls rate and bitrate of captured audio, as well as other properties
- Extensive configuration makes basically any imaginable microphone constellation possible

Alsa

The Advanced Linux Sound Architecture manages all your physical and virtual microphones and speakers.

- It controls volume and gain
- It controls rate and bitrate of captured audio, as well as other properties
- Extensive configuration makes basically any imaginable microphone constellation possible

Honorable mention: pulseaudio

Alsa

The Advanced Linux Sound Architecture manages all your physical and virtual microphones and speakers.

- It controls volume and gain
- It controls rate and bitrate of captured audio, as well as other properties
- Extensive configuration makes basically any imaginable microphone constellation possible

Honorable mention: pulseaudio

- *many* extensions (called modules) to suit all kinds of needs
- eg. virtual devices over network, on the fly filtering, etc.

Localisation

Sound Source Localisation

Basic Idea:

- more than one microphone is required
- four microphones are required for "exact" localisation
- can be seperated into 2 sub-problems: time delay estimation and position estimation
- time delay estimation guesses the time differences with which a sound arrives at different microphones
- position estimation tries to locate the soundsource based on these guesses

Time Delay Estimation

Approaches:

- time domain approximation
- frequency based
- statistical approaches

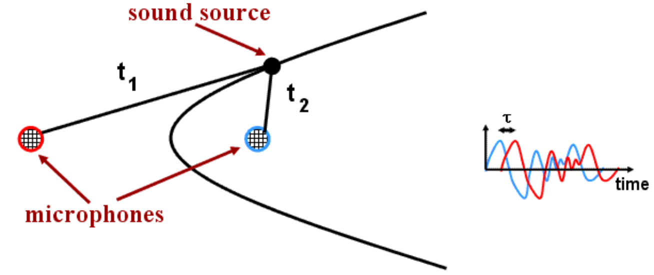


Figure 4: Source: https://cecas.clemson.edu/ stb/research/acousticloc/

Position estimation

Sound source localisation is not exact

- TDE is just an estimation (duh)
- because of superposition TDE may provide false positives
- position estimation can be interpreted as (n-1)-dimensional optimization problem
- optimization problems can be solved by eg. gradient descent or conjugate gradients

(see Mario Botsch's Scientific Computing course for more information about solving these problems)

Signal Enhancing

Beamforming

content...

Voice Activation Detection

What we use

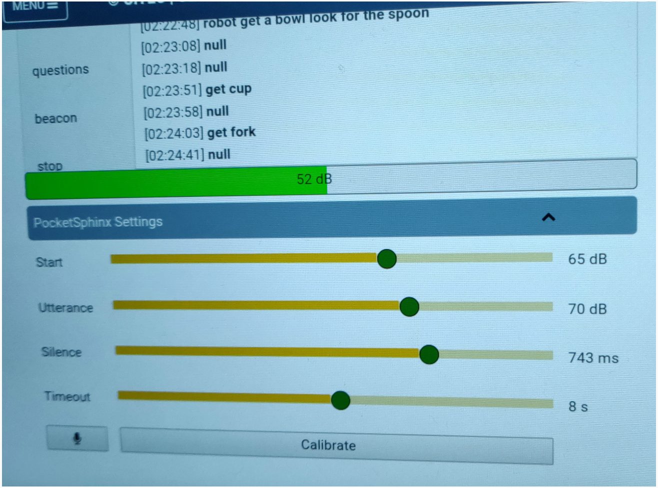


Figure 5: Double threshold voice activation detection

What we use II

A voice activation detection based on audio loudness with three states:

idle
: Start in this state

starting
: switch to this state if the audio $>$ StartDb and stay here as long as audio $>$ UtteranceDb

ending
: switch to this state if the audio $<$ UtteranceDb and stay as long as specified via Silence, then return to idle

A maximum audio length can be specified via Timeout

Other approaches

Based on...

loudness
: based on decibel calculation, it will only take into account the single most extreme value in an audio frame

energy
: in contrast to loudness-based approaches, energy calculation will take all values in an audio frame into consideration

frequency
: will calculate frequencies and search for those typically used by human speech

Speaker Recognition

Speech Recognition

Approaches

There are three big approaches for consumer/ robotics speech recognition:

- Hidden Markov Models
- Deep Learing
- Online Services

Hidden Markov Models

Hidden Markov models are statistical models where a system is assumed to be a markov process with hidden states.

- imagine a statemachine where the transitions are modeled by probabilities
- states in this statemachine can be roughly understood as phonemes
- a most probable path through the model can be computed for a given sequence of signals or data
- this path is can then be mapped to a spoken word or sequence of words
- states are unknown (hidden) and assumed via probabilities

Sphinx

A HMM based group of application which use

- a speech model (models phenomes)
- a dictionary (maps phonemes to words)
- a language model (probabilities of word sequences)

Recent versions can provide quasi-free speech recognition.

Deep Learning & DeepSpeech

Deep Learning

- *very* big field
- computationally expensive concept to abstract high level information out of big chunks of data

Deep Learning & DeepSpeech

Deep Learning

- *very* big field
- computationally expensive concept to abstract high level information out of big chunks of data

DeepSpeech

- ML architecture by Baidu
- several implementations, e.g. by mozilla using tensorflow
- detects free speech, but is somewhat phoneme based
- currently one of the best architectures in regards to word error rate

Google/Bing/... Online Speechrec

Commercial "Cloud" services provided by companies like Google, Amazon, Microsoft, etc.

- kind of a blackbox
- need *fast* internet connections
- typically better results than local speechrec
- not free, can be quite expensive if used extensively

Grammar vs Grammarless/Freespeech

Grammar based recognition

- by restraining accepted inputs the results can be more precise
- mapping result to action can be very easy
- must be supported by approach *and* implementation
- best suited for controlled environments/ use cases

Freespeech

- usually seen in deep learning approaches
- any spoken word can be detected
- go-to apporach for extremely complex use cases

Corrected Spelling vs Phoneme based recognition

Corrected spelling

- needs a dictionary
- guarantees correct spelling (important for further processing)
- makes a very robust speech recognition (esp. in combination with grammars)

Phoneme bases recognition

- does not need any kind of dictionary
- adapts better to new/unique words
- usually embraced by deep learning approaches

What do we use?

For everyday use:

What do we use?

For everyday use:

(Pocket-)Sphinx

For special cases:

What do we use?

For everyday use:

(Pocket-)Sphinx

For special cases:

Google Speechrec

Natural Language Processing

Idea

Just recognizing what was said does not solve all our problems.
We need to *understand* what was said.

Idea

Just recognizing what was said does not solve all our problems.
We need to *understand* what was said.

- direct mapping?

Idea

Just recognizing what was said does not solve all our problems.
We need to *understand* what was said.

- direct mapping?
- keywordspotting?

Idea

Just recognizing what was said does not solve all our problems.
We need to *understand* what was said.

- direct mapping?
- keywordspotting?
- how to do planning?

Idea: Create the grammar tree of a sentence, thus making it easier to extract information.

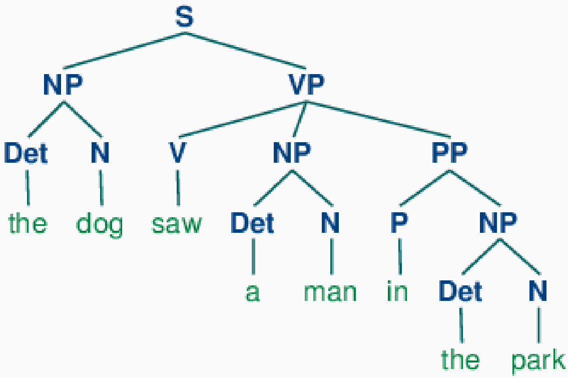


Figure 6: Example of a grammar tree. Source: https://www.nltk.org/book/ch08.html

Beyong Grammar

More intensive forms of analysis can involve:

- Statistical analysis of sentences/phrases
- tagging of phrases
- dependency parsing
- tokenization

What do we use?

With Pocketsphinx:

What do we use?

With Pocketsphinx:

Not that much :(

With Google Speechrec:

What do we use?

With Pocketsphinx:

Not that much :(

With Google Speechrec:

spaCy, a python library which can...

- create a grammar tree of a sentence
- classify phrases and words (in context)
- abstract information out of text
- analyse the similarity of two sentences

spaCy example

From Wikipedia:

"RoboCup is an annual international robotics competition proposed and founded in 1996 (Pre-RoboCup) by a group of university professors (among which Hiroaki Kitano, Manuela M. Veloso, and Minoru Asada). The aim of such a competition consists of promoting robotics and AI research, by offering a publicly appealing, but formidable challenge."

spaCy example

From Wikipedia:

"RoboCup is an annual international robotics competition proposed and founded in 1996 (Pre-RoboCup) by a group of university professors (among which Hiroaki Kitano, Manuela M. Veloso, and Minoru Asada). The aim of such a competition consists of promoting robotics and AI research, by offering a publicly appealing, but formidable challenge."

spaCy:

RoboCup ORG, 1996 DATE, Hiroaki Kitano ORG, Manuela M. Veloso PERSON, Minoru Asada PERSON, AI GPE

Thanks for the Attention!

Discussion

Sources