An automatic speech detection architecture for social robot oral interaction *

E.G. Tsardoulias
Centre of Research & Technology - Hellas
6th km Xarilaou - Thermi, 57001
Thessaloniki, Greece
etsardou@iti.gr

A.L. Symeonidis
Centre of Research & Technology - Hellas, 6th km Xarilaou - Thermi, 57001 / Department of Electrical and Computer Engineering, Aristotle University of Thessaloniki, 54124
Thessaloniki, Greece
asymeon@iti.gr

P.A. Mitkas
Department of Electrical and Computer Engineering
Aristotle University of Thessaloniki, 54124
Thessaloniki, Greece
mitkas@auth.gr

ABSTRACT

Social robotics have become a trend in contemporary robotics research, since they can be successfully used in a wide range of applications. One of the most fundamental communication skills a robot must have is the oral interaction with a human, in order to provide feedback or accept commands. And, although text-to-speech is an almost solved problem, this isn't the case for speech detection, since it includes a large number of different conditions, many of which are literally unpredictable. There are quite a few well established ASR (Automatic Speech Recognition) tools, however without providing efficient results, especially in less popular languages. The current paper investigates different speech detection strategies via the utilization of the Sphinx-4 open-source library. The first is a way to incorporate languages for which no acoustic or language model exists (Greek in our case), following the grapheme-to-phoneme concept. The speech detection model is evaluated using audio captured from a NAO v4 robot, a difficult task due to the high levels of included noise, thus denoising techniques are investigated as well.

Categories and Subject Descriptors

I.2.7 [**Natural Language Processing**]: Speech recognition and synthesis; I.2.9 [**Robotics**]: Commercial robots and applications

*Parts of this work have been supported by the FP7 Collaborative Project RAPP (Grant Agreement No 610947), funded by the European Commission

AM15, October 07-09, 2015, Thessaloniki, Greece

DOI: http://dx.doi.org/10.1145/2814895.2814931

Keywords

Social robotics, Speech detection, Sphinx-4, NAO robot, RAPP

1. INTRODUCTION

Social robotics are envisioned to physically assist people, function as their companions or even provide assistance with daily tasks. To this end, social robots must possess a number of abilities such as express or perceive emotions, communicate with high level dialogue, learn and recognize models, establish social relations, use natural cues, as well as learn or develop extra social skills.

One of the most challenging skills a robot must possess is the speech-to-text ability, i.e. to detect the succession of words a human pronounces [1]. This task is critical for a "natural" HRI (Human-Robot Interaction), since speech is one of the most fundamental ways of communication. The speech detection problem is quite challenging, as a robot must understand speech from different users, using different languages, voice pitches, pronunciations, speech speeds, as well as sound and noise levels. Several open-source speech detection tools exist; one of the most important and wildly used of them being Sphinx-4 [2], developed in Java™ by CMU (Carnegie Mellon University), Sun Microsystems Laboratories and Mitsubishi Electric Research Laboratories. Its features include the support of HMM-based acoustic models, all standard types of language models, as well as concurrent stream execution.

Current work has been performed in the context of the FP7 Collaborative project RAPP (Robotic Applications for Delivering Smart User Empowering Applications) [1], funded by the European Commission (Grand Agreement No 610947). RAPP is devoted to develop a cloud repository of applications and services that can be utilized by heterogeneous robots, aiming to assist people with a range of disabilities. This paper discusses a speech detection architecture for social robot oral interaction, which can be deployed on the cloud and is capable of supporting multiple languages. The implementation involves the employment of the NAO robot for the audio capturing, the Sphinx-4 tool for ASR and

[1]http://rapp-project.eu/

grapheme to phoneme techniques for supporting less popular languages (Greek in our case).

The remainder of the paper is organized as discussed below. Section 2 presents the state-of-the art in the social uses of NAO discussed under the prism of ASR while also focuses on ways to support an extra language in Sphinx-4 library, as well as some denoising techniques. Section 3 describes the actual system implementation, including the presentation of the overall architecture, the G2P approach followed and the denoising techniques applied on the NAO captured audio. Finally section 4 contains the experimental results and section 5 the conclusions and future work.

2. STATE OF THE ART

As aforementioned, one of the most critical tasks a social robot must perform is to successfully communicate with humans via speech. Several studies exist researching the HRI (Human-Robot Interaction) task via various methodologies, the most important of which are speech and gestures [5][6]. The NAO robot in specific, is a social robot used in several projects targeting special user groups, such as children with Autism Spectrum Disorder [7], elders suffering from mild cognitive impairment [8], or generally young users [9].

The current state-of-the-art in open-source ASR tools includes Sphinx-4 [2], which will be presented in detail in the next sections, HTK [10], a tool for building and manipulating Hidden Markov Models, Julius [11], a large vocabulary continuous speech recognition decoder and Kaldi [12] which is similar in aims with the HTK but gains more and more attention from the ASR community.

Even though ASR is a quite researched area, only a few language / acoustic models are currently officially supported. Nevertheless, several attempts have been made into extending the supported languages. Specifically for Sphinx-4, an example is [13], where isolated Swahili words recognition is performed, by examining the language structure and creating a 40 word Swahili acoustic model. Another approach is described in [14], where speaker independent isolated ASR (Automatic Speech Recognition) for Tamil language, spoken in India and Sri Lanka, is developed and tested via Sphinx-4. Other similar approaches include [15], where a Slovak speech recognition system was build upon Sphinx-4 for utilization in GSM networks and [16], where a limited vocabulary ASR system is created for the Urdu language.

In our case, we are interested in ASR applications using the NAO robot. Regarding its audio capturing capabilities, NAO robot is equipped with four microphones placed on its head, whose performance is quite low as described in [17]. There, a speech recognition artificial language is adopted and its differences to English are investigated regarding ASR, in the cases where the NAO head is stationary or moving. An interesting research concerning the quality of the NAO microphones was performed by Heinrich and Wermter in [18], where the robustness of an ASR with a multi-pass decoder is investigated for three audio capturing devices: a headset, a ceiling boundary microphone and the NAO microphones. The results confirm the poor NAO audio capturing abilities, as the experimental true positives ratio for three different audio decoders were 2-6% for the NAO microphones, 22-42% for the ceiling microphone and 63-77% for the headset.

It is evident that in order to acquire successful results in ASR with the NAO captured audio, certain pre-processing techniques must be employed. These include signal-noise spectral subtraction [19], soft or hard signal gating [20], or even wavelet domain Wiener filtering at various decomposition schemes with different windowing / thresholding configurations [21].

Next the proposed system implementation is presented.

3. SYSTEM IMPLEMENTATION

The RAPP architecture has been designed as a distributed system separated in two main modules: the cloud part and the robots. The robots can download robotic applications (RApps) and these can invoke cloud services, one of which is the ASR module developed in this work. It should be stated that NAO provides a speech recognition software by NUANCE [2], which needs to be fed with a set of words in order to operate and is restricted by the robot's currently installed language. In our case we desire a flexible speech recognition module, capable of multi-language extension and good performance on limited or generalized language models. For this reason, Sphinx-4 was selected. Since we aim for speech detection services for RApps regardless of the actual robot at hand, we decided to install Sphinx-4 in the RAPP Cloud and provide services for uploading or streaming audio, as well as configuring the ASR towards language or vocabulary-specific decisions. This allows for speech detection support, even for robots with low computational abilities (such as NAO), since ASR is delegated in the cloud. A diagram showcasing the RAPP Cloud Speech detection architecture is presented in Figure 1. Next, the RAPP ASR functionalities are described in detail.

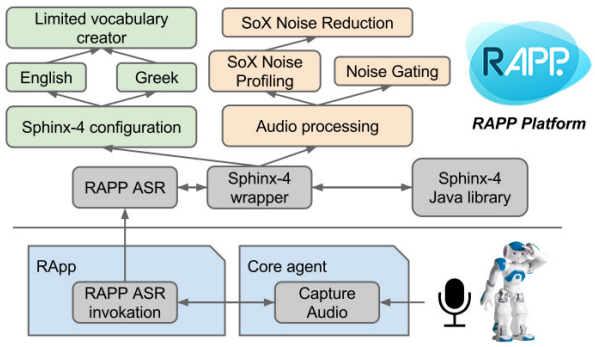

Figure 1: RAPP Cloud ASR architecture (Audio capturing in cyan, pre-processing in orange and ASR configuration in green)

3.1 RAPP ASR using Sphinx-4

Before describing the actual implementation, it is necessary to investigate the Sphinx-4 configuration capabilities. In order to perform speech detection, Sphinx-4 requires:

1) An acoustic model containing information about *senones*. Senones are short sound detectors able to represent the specific language's sound elements (phones, diphones, triphones etc.). In order to train an acoustic model for an unsupported language an abundance of data must be available. CMU Sphinx supports a number of high-quality acoustic models,

[2]http://www.nuance.com/index.htm

including US English, German, Russian, Spanish, French, Dutch, Mexican and Mandarin. In the RAPP case we desire multi speakers support, thus as Sphinx-4 suggests, 50 hours of dictation from 200 different speakers are necessary [3], including the knowledge of the language's phonetic structure, as well as enough time to train the model and optimize its parameters. If these resources are unavailable (which is the current case), a possibility is to utilize the US English acoustic model and perform grapheme to phoneme transformations.

2) A language model, used to restrict word search. Usually n-gram (an n-character slice of a bigger string) language models are used in order to strip non probable words during the speech detection procedure, thus minimizing the search time and optimizing the overall procedure. Sphinx-4 supports two language models: grammars and statistical language models. Statistical language models include a set of sentences from which probabilities of words and succession of words are extracted. A grammar is a more "strict" language model, since the ASR tries to match the input speech only to the provided sentences. In our case, we are initially interested in detecting single words from a limited vocabulary, thus no special attention was paid in the construction of a generalized Greek language model.

3) A phonetic dictionary, which essentially is a mapping from words to phones. This procedure is usually performed using G2P converters (Grapheme-to-Phoneme), such as *Phonetisaurus*[4] or *sequitur-g2p*[5].

G2P converters are used in almost all TTSs (Text-to-Speech converters), as they require pronunciation rules to work correctly. An example of a G2P study for the Greek language is described in [22], where a rule-based method for associating Greek graphemes to phonemes is presented. In the current work, we decided to not use a G2P tool, but investigate and doument the overall G2P procedure of translating Greek graphemes directly into the CMU Arpabet format (which Sphinx supports).

After providing some insights on the Sphinx configuration capabilities, the description of the RAPP ASR architecture follows. As evident in Figure 1, two distinct parts exist: the NAO robot and RAPP Cloud part. Lets assume that a robotic application (RApp) is deployed in the robot, which needs to perform speech detection. The first step is to invoke the *Capture Audio* service the Core agent provides, which in turn captures an audio file via the NAO microphones. This audio file is sent to the cloud *RAPP ASR* node in order to perform ASR. RAPP ASR is a ROS[6] node responsible for servicing the ASR invocations either from a robot, or from other RAPP cloud nodes. It should be stated that the RApp-to-RAPP Platform communication is performed via the RAPP API, utilizing the HOP framework [23]. The most important module of the RAPP ASR is the *Sphinx-4 wrapper*. This node is responsible for receiving the service call data and configuring the Sphinx-4 software according to the request. The actual Sphinx-4 library is executed as a separate process and Sphinx-4 wrapper is communicating with it via standard *stdin / stdout* streams.

Regarding the Sphinx-4 configuration, the user is able to select the ASR language and if they desire ASR on a limited vocabulary or on a generalized one stored in the RAPP cloud. If a limited vocabulary is selected, the user can also define the language model (the sentences of the statistical language model or the grammar). The configuration task is performed by the *Sphinx-4 Configuration* module. There, the ASR language is retrieved and the corresponding language modules are employed (currently Greek or English). If the user has requested ASR on a limited vocabulary, the corresponding language module must feed the *Limited vocabulary creator* with the correct grapheme to phoneme transformations, in order to create the necessary configuration files. In the English case, this task is easy, since Sphinx-4 provides a generalized English vocabulary, which includes the words' G2P transformations. When Greek is requested, a simplified G2P method is implemented, which will be discussed in section 3.2. In the case where the user requests a generalized ASR, the predefined generalized dictionaries are used (currently only English support exists).

The second major task that needs to be performed before the actual Sphinx-4 ASR is the audio preparation. This involves the employment of the *SoX* audio library, a cross platform command line utility, able to perform several audio processing algorithms in a wide variety of file encodings. In our implementation, SoX is used for extracting a noise profile by performing DTF on an input file that contains silence, and then applies a standard subtraction of the input audio and noise profile FFTs for denoising. It should be stated that since the RAPP Platform must function for different users under different environmental noises, we store a different noise profile for each user. Next, noise gating is applied, in order to cut off the remaining noise and finally the audio file is provided as input to the Sphinx-4 library in order to perform ASR. The resulting words are extracted and transmitted back to the RApp, as a response to the HOP service call. The audio processing procedure will be explained in detail in paragraph 3.3.

In the following paragraph the Grapheme to Phoneme rules for the Greek language defined are discussed, as well as the successive steps performed to reach a valid Sphinx-4 configuration.

3.2 Greek G2P rules

The current chapter aims to describe the step-by-step G2P process of transforming Greek words into the 39 Arpabet-based phonemes Sphinx-4 supports[7].

3.2.1 Greek language structure and phonemes

There is an abundance of sources for the Greek pronunciation, thus it is redundant to describe the pronunciation conventions in detail. Nevertheless, some insights on the Greek language structure will be given [24]. The Greek language has 24 letters (α, β, γ, δ, ϵ, ζ, η, θ, ι, κ, λ, μ, ν, ξ, o, π, ρ, σ, τ, υ ϕ, χ, ψ, ω) and 25 phonemes (the structural sound components that define a word's acoustic properties). These are α, ϵ, ι, o, $o\upsilon$, β, γ, δ, ζ, θ, κ, λ, μ, ν, π, ρ, σ, τ, ϕ, χ, $\mu\pi$, $\nu\tau$, $\gamma\kappa$, $\tau\sigma$ and $\tau\zeta$.

The Greek alphabet has not separate letters for each phoneme, as there are phonemes that contain two letters ($\mu\pi$). Additionally, letters that sound the same exist, or in other words correspond to the same phoneme (e.g. the letters η, ι, υ correspond to the ι phoneme). The letters ξ, ψ are called *double*,

[3] http://cmusphinx.sourceforge.net/wiki/tutorialam
[4] https://code.google.com/p/phonetisaurus/
[5] http://www-i6.informatik.rwth-aachen.de/web/Software/g2p.html
[6] http://www.ros.org/
[7] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

since they contain two phonemes each (e.g. $\xi = \kappa + \sigma$). Furthermore, two-digit vowels exist (except for $o\upsilon$ which is a phoneme) that correspond to specific phonemes. These are $\alpha\iota \rightarrow \epsilon$, $\epsilon\iota \rightarrow \iota$, $o\iota \rightarrow \iota$ and $\upsilon\iota \rightarrow \iota$. A special letter case is ς which is the same as σ, but substitutes it when placed in the end of a word. Another special category is the *common consonants* which are simply two same letters in succession. Their pronunciation is the same as of the single letter case. These are $\beta\beta, \kappa\kappa, \lambda\lambda, \mu\mu, \nu\nu, \pi\pi, \rho\rho, \sigma\sigma, \tau\tau$. Finally, there are two special vowel combinations ($\alpha\upsilon$ and $\epsilon\upsilon$) the pronunciation of which depends on the next letter. If this is one of (π, τ, κ, θ, χ, σ, ξ, ψ), then υ is pronounced like ϕ. On the other hand, if the next letter is one of ($\delta, \gamma, \zeta, \lambda, \rho, \mu, \nu$) or a vowel, υ is pronounced as β.

Greek words have two basic distinct elements, apart form the letters. The first is the *acute accent* or *Greek tonos*, which is combined with one or two vowels in a word and indicates where the word is accented. The acute accent can be also be applied on the last of the two vowels of a diphthong (two digit entities that contain two phonemes - e.g. $\alpha\iota$). The second one is the *diaeresis* which can appear in the second letter of a diphthong, only if this is ι or υ, indicating that the vowels are pronounced separately. In some cases the acute accent and diaeresis can be applied together in a single vowel. In the proposed G2P analysis the letters with acute accent or diaeresis are treated the same as the normal letters, except for the case where we have diphthongs, where the pronunciation differs.

There are several other smaller cases of altering pronunciation, but the ones presented are the most basic and commonly presented. Detailed information on the Modern Greek Language orthography and phonology can be found in [25].

3.2.2 G2P transformation and Sphinx-4 configuration

The successive G2P transformation steps are denoted in table 1. In essence these include:

1. The upper case letters are substituted by their corresponding lower case.
2. The two-letter phonemes and the common consonants are substituted with the corresponding CMU phonemes.
3. The two special vowel combinations ($\alpha\upsilon$ and $\epsilon\upsilon$) are substituted with $\alpha\phi$, $\alpha\beta$, $\epsilon\phi$ or $\epsilon\beta$, according to their next letter.
4. The two-digit vowels are substituted with the corresponding CMU phonemes.
5. Specific rules are applied which involve σ when the next letter is one of (γ, β, δ, μ, ν, λ, ρ, $\mu\pi$, $\nu\tau$).
6. The final step is to replace the remaining Greek letters with their corresponding CMU phonemes.

It should be noted that better matching techniques for adjusting CMU phonemes with similar pronunciation (e.g. AA, AE, AH, AO, AW, AY) were not investigated. Some G2P examples for a few Greek words are presented in figure 2.

Since the Grapheme to Phoneme description has reached its end, the Sphinx-4 library can be configured. In the limited vocabulary case, a dictionary file containing the words along with their phonetic transcription is created. Additionally, since our study case concerns single word recognition, the language model file simply contains the isolated words. If a generalized model was concerned, the dictionary file would be created in the same way but the language model file should contain characteristic sentences of the language we plan to support. Either way, Sphinx-4 requires the dictionary file to be in ARPA format, which is finally transformed to binary DMP format, utilizing a set of tools Sphinx provides.

Table 1: Greek G2P

Step 1			
$A \rightarrow \alpha$	$B \rightarrow \beta$	$\Gamma \rightarrow \gamma$	$\Delta \rightarrow \delta$
$E \rightarrow \epsilon$	$Z \rightarrow \zeta$	$H \rightarrow \eta$	$\Theta \rightarrow \theta$
$I \rightarrow \iota$	$K \rightarrow \kappa$	$\Lambda \rightarrow \lambda$	$M \rightarrow \mu$
$N \rightarrow \nu$	$\Xi \rightarrow \xi$	$O \rightarrow o$	$\Pi \rightarrow \pi$
$P \rightarrow \rho$	$\Sigma \rightarrow \sigma$	$T \rightarrow \tau$	$\Upsilon \rightarrow \upsilon$
$\Phi \rightarrow \phi$	$X \rightarrow \chi$	$\Psi \rightarrow \psi$	$\Omega \rightarrow \omega$
Step 2			
$o\upsilon \rightarrow UW$	$\mu\pi \rightarrow B$	$\nu\tau \rightarrow D$	$\gamma\kappa \rightarrow G$
$\gamma\gamma \rightarrow G$	$\tau\sigma \rightarrow CH$	$\tau\zeta \rightarrow JH$	$\sigma\sigma \rightarrow S$
$\kappa\kappa \rightarrow K$	$\beta\beta \rightarrow V$	$\lambda\lambda \rightarrow L$	$\mu\mu \rightarrow M$
$\nu\nu \rightarrow N$	$\pi\pi \rightarrow P$	$\rho\rho \rightarrow R$	$\tau\tau \rightarrow T$
Step 3			
$\alpha\upsilon \rightarrow \alpha\phi$ if next letter $\in (\pi, \tau, \kappa, \theta, \chi, \sigma, \xi, \psi)$			
$\alpha\upsilon \rightarrow \alpha\beta$ if next letter $\in (\delta, \gamma, \zeta, \lambda, \rho, \mu, \nu$ *or vowel*$)$			
$\epsilon\upsilon \rightarrow \epsilon\phi$ if next letter $\in (\pi, \tau, \kappa, \theta, \chi, \sigma, \xi, \psi)$			
$\epsilon\upsilon \rightarrow \epsilon\beta$ if next letter $\in (\delta, \gamma, \zeta, \lambda, \rho, \mu, \nu$ *or vowel*$)$			
Step 4			
$\alpha\iota \rightarrow EH$	$\epsilon\iota \rightarrow IH$	$o\iota \rightarrow IH$	$\upsilon\iota \rightarrow IH$
Step 5			
$\sigma\beta \rightarrow Z\ V$	$\sigma\gamma \rightarrow Z\ W$	$\sigma\delta \rightarrow Z\ DH$	$\sigma\mu \rightarrow Z\ M$
$\sigma\nu \rightarrow Z\ N$	$\sigma\lambda \rightarrow Z\ L$	$\sigma\rho \rightarrow Z\ R$	$\sigma\mu\pi \rightarrow Z\ B$
$\sigma\nu\tau \rightarrow Z\ DH$			
Step 6			
$\alpha \rightarrow AA$	$\beta \rightarrow V$	$\gamma \rightarrow W$	$\delta \rightarrow DH$
$\epsilon \rightarrow EH$	$\zeta \rightarrow Z$	$\eta \rightarrow IH$	$\theta \rightarrow TH$
$\iota \rightarrow IH$	$\kappa \rightarrow K$	$\lambda \rightarrow L$	$\mu \rightarrow M$
$\nu \rightarrow N$	$\xi \rightarrow K\ S$	$o \rightarrow OH$	$\pi \rightarrow P$
$\rho \rightarrow R$	$\sigma \rightarrow S$	$\tau \rightarrow T$	$\upsilon \rightarrow IH$
$\phi \rightarrow F$	$\chi \rightarrow HH$	$\psi \rightarrow P\ S$	$\omega \rightarrow OH$
$\varsigma \rightarrow S$			

Διαίρεση (division)	DH IH EH R EH S IH
Ποινή (penalty)	P IH N IH
Εξαιρετικά (exceptionally)	EH K S EH R EH T IH K AA
εύστοχος (well-aimed)	EH F S T OW HH OW S
Καλυτερεύω (improving)	K AA L IH T EH R EH V OW
ωριαίος (hourly)	OW R IH EH OW S
Χάσμα (chasm / gap)	HH AA Z M AA
Αυγό (egg)	AA V W OW
επιστήμονας (scientist)	EH P IH S T IH M OW N AA S

Figure 2: G2P indicative results

3.3 Employing Sphinx-4 with NAO

Given the Sphinx-4 configuration, the second problem at hand is related to the NAO captured audio and its properties, regarding the Sphinx-4 interoperability. As already mentioned, the NAO robot is equipped with four microphones, which are placed on its head[8] (figure 3) having an

[8]http://doc.aldebaran.com/1-

electrical bandpass of $[300Hz - 8KHz]$.

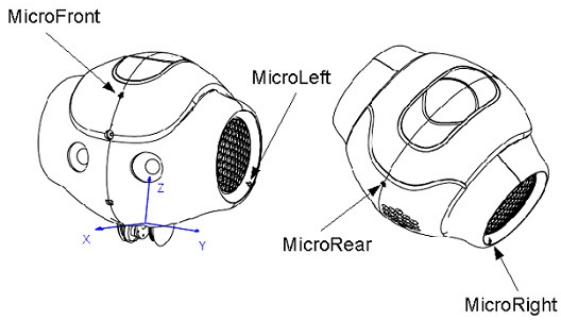

Figure 3: Microphones placement on NAO head

NAO is able to record a single audio file at a time (wav or ogg), either from all microphones (4 channels at 48kHz) or from any single microphone (1 channel, 16kHz). In the conducted experiments, the recordings were performed using only the front microphone. Additionally, the one-channel audio is the most appropriate selection, since Sphinx-4 requires single channel wav files, with a 16kHz sample rate and 16 bit little-endian format. A comparison of two 3-second audio files captured by a headset and from the front NAO microphone in which one word is dictated, can be seen in figure 4. It is evident that the NAO captured audio contains considerable background static noise, being probably the result of a cooling fan that also exists in the NAO head. The problem raised is that the high noise levels cause Sphinx-4 to fail by producing no output.

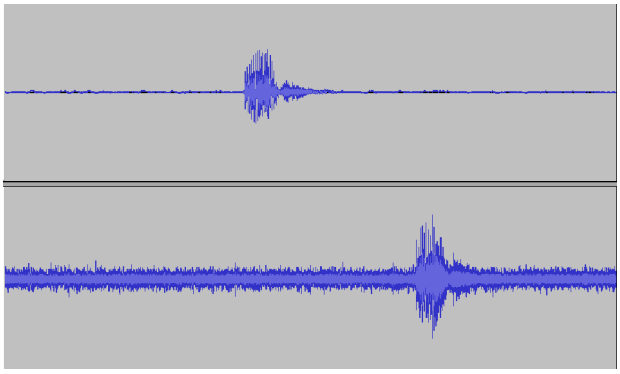

Figure 4: Top: A word captured by a headset microphone. Bottom: The same word captured by the NAO front microphone

3.3.1 Audio noise reduction using SoX

The first step towards denoising the signal is the obvious one, which involves the creation of the noise frequency profile, and then performing a frequency subtraction between the signal and the noise. This approach can be described as static, since a static noise profile is assumed. This assumption is generally valid, as the noise is NAO-specific and seems to be constantly existent and independently of the recording environment. In the current implementation we employed SoX library (Sound eXchange). Specifically, the *noiseprof* tool was used for extracting the noise frequency coefficients via DFT, and then the *noisered* plugin is applied in order to perform frequency subtraction (with a reduction coefficient of 0.12). An example of the performed noise reduction is presented in figure 5.

14/_images/hardware_microposition.png

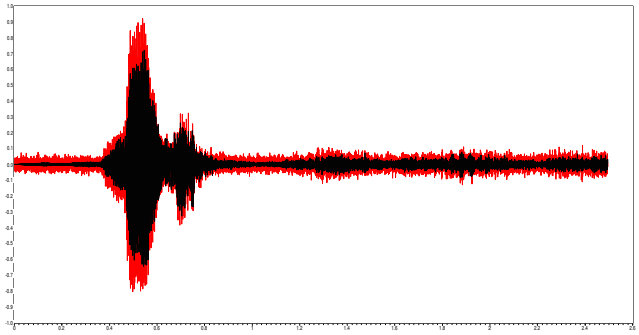

Figure 5: Audio signal before (red) and after (black) the SoX-based noise reduction

As evident from figure 5, this method did not succeed to eliminate all of the noise, since the reduction coefficient was quite small. Generally speaking, it is recommended to not perform an aggressive noise removal, or in other words not over-clean the noise, since the actual voice can be distorted. Nevertheless, the specific denoising process cannot be considered effective audio-wise, but the results in the Sphinx-4 compatibility are stunning, since the success rate increases almost 10 times. After the noise reduction process, Sphinx-4 managed to detect the correct word in most of the cases (the detailed results will be presented in chapter 4).

3.3.2 Audio signal gating

Even though the aforementioned approach had positive results in static, laboratory environments, the performed tests in real conditions failed quite often. The main reason was that in real environments several different noise sources exist, such as other people talking in the background, sounds from a radio or television, or even doors opening and closing. It is obvious that these events cannot be predicted in order to change the noise profile on the fly, thus another technique must be applied.

We assume that the the audio part where the speech exists has a higher intensity (or acoustic energy) than the rest of the signal (which in theory contains the background noise). This assumption is logical, as we expect the voice in interest to sound louder than the surrounding noises. The proposed approach is a form of signal hard gating, using the concept of rms (root mean squared) method. As already known, a signal's rms value is an indication of the average amplitude over time. Our approach involves squaring the amplitude of each point of the waveform, calculate the mean value and hard gate the signal's square based on this mean, multiplied by a coefficient. In mathematical notation, if S is the audio signal, its mean audio power value is calculated by

$$rms^2 = \overline{S^2} = \frac{1}{||S||} \sum_{\forall i} S[i]^2 \quad (1)$$

The next step is to perform a hard gating in the S signal, based on the individual sample's power when compared with

the MIV value:

$$S[i]^2 < c \cdot rms^2 \Rightarrow S[i] = 0, c > 0 \quad (2)$$

An example of the proposed signal gating is presented in figure 6. In the current implementation we used $c = 0.15$, which is quite low, in order not to distort the actual speech. If the gating fails to remove the remaining background noise due to its strength, it is highly possible for Sphinx to not produce any result. In that case we perform another gating with a higher c coefficient and invoke Sphinx-4 again.

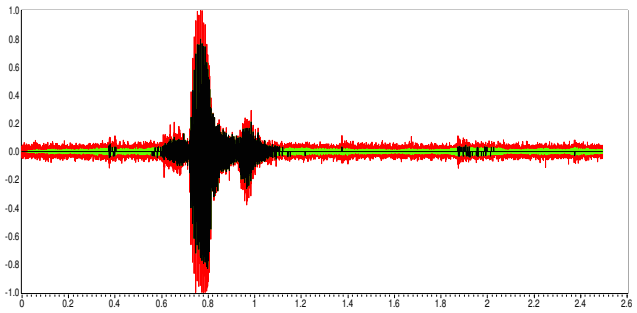

Figure 6: Red: Original signal, Green: After SoX based noise reduction, Black: After signal gating

One worth-mentioning comment is that the gating procedure can create sputter noise, as the selected gating has a single threshold. Nevertheless, the experimental results show that even though this kind of noise is irritating to the human ear, it does not affect Sphinx-4.

4. EXPERIMENTAL RESULTS

In order to validate the correctness of our decisions, two datasets were used. The first set involves recordings acquired during the early evaluation of the robotic applications in the RAPP project. Specifically, a cognitive game application was tested on 10 elders from the New Moudania Seniors' Center (3 men, 7 women), where NAO was dictating a short story and each elder was asked ten questions in order to trigger their memory. In this dataset, the only valid answers were $\nu\alpha\iota$ (yes) and $o\chi\iota$ (no), thus a two-word vocabulary was used. The number of recordings was 366, with a duration of 3 seconds each. Unfortunately, the original audio files captured from NAO were not stored, but the ones after the audio noise reduction using SoX were available, thus this dataset will be used to showcase the usefulness of the signal gating step. It should be mentioned that the conditions in the Seniors' Center were quite noisy, as the environment was not isolated; more than 15 persons were in the same room drinking their coffee or chatting.

The second dataset consists of 130 audio files recorded via NAO from 10 different people that work in the laboratory (8 men, 2 women). This dataset contains recordings of words chosen from a 10-word vocabulary, thus it is perfect to showcase how the error rates scale when the dictionary size increases. The 10 word vocabulary contains the words $\nu\alpha\iota$ (yes), $o\chi\iota$ (no), $\alpha\rho\kappa\epsilon\tau\alpha$ (enough), $\epsilon\iota\mu\alpha\iota$ (I am), $\phi o\upsilon\rho\nu o\varsigma$ (oven), $\gamma\iota\alpha\tau\rho o\varsigma$ (doctor), $\kappa\alpha\tau\sigma\epsilon$ (sit), $\chi\alpha\pi\iota\alpha$ (pills), $\rho o\mu\pi o\tau$ (robot) and $\sigma\tau\epsilon\iota\lambda\epsilon$ (send). Again, the environment was not noise sterile, as background speech may exist. Additionally, since the original recordings are available, the usefulness of both denoising steps will be presented.

4.1 Denoising steps validation

In the first set of experiments both datasets will be used, employing a two-words vocabulary that includes $\nu\alpha\iota$ (yes) and $o\chi\iota$ (no). The Seniors' Center dataset contains 366 samples whereas the Laboratory dataset contains 63 audio files. The results are presented in tables 2 and 3.

Table 2: Denoising steps validation - Laboratory dataset (63 samples) - 2 word vocabulary

	Original	After SoX	After signal gating
Success rate	7.94%	74.62 %	93.66 %
Error rate	1.58 %	12.69 %	1.58 %
No recognition rate	90.48 %	12.69 %	4.76 %

Table 3: Denoising steps validation - Seniors' Center dataset (366 samples) - 2 word vocabulary

	After SoX	After signal gating
Success rate	75.69 %	90.44 %
Error rate	13.11 %	4.10 %
No recognition rate	11.20 %	5.46 %

The experimental results indicate that Sphinx-4 cannot properly operate when its input is the original NAO audio file, since the success rate is almost 8%, while in the same time the 90% of the samples produced no output. This is due to the high noise levels the captured audio contains. When the SoX spectral denoising is applied, the success ratio reaches 75%, while the error rate and the no-recognition rate is almost 12.5%. It is obvious that after the frequency-based noise removal, Sphinx-4 was able to perform ASR, even though enough errors occurred due to the remaining noise. Finally, after the signal gating, the success ratio reached 94%, the errors were almost eliminated and the no-recognition rate was reduced to 5%. Thus, we can assume that Sphinx-4 was able to better distinguish the words when a large percent of the noise was eliminated, and was not affected by the sputtering noise that the gating procedure left as residue. The results of the Seniors' Center dataset are similar, thus no special commenting is required.

It should be stated that in the current ASR application we are mostly interested to maximize the ASR success rate and minimize the error rate, while the no-recognition rate can range in reasonable levels. This is due to our capability to program the robot accordingly, in order to ask the user again if their answer was not recognized. On the other hand a high error rate would be extremely negative for obvious reasons.

4.2 Vocabulary expansion investigation

The second set of experiments investigate the proposed methods' performance when the vocabulary size increases. Two experiments were performed using a 6-word vocabulary and the whole 10-word vocabulary the Laboratory dataset contains. The results are presented in tables 4 and 5. The results clearly indicate that the ASR error rate increases

along with the vocabulary size. For this result two culprits may be responsible: either the G2P procedure fails to correctly distinguish (acoustically) similar words' phonemes or the denoising procedure corrupts the actual words. In order to investigate the actual effect of these two causes, a final experiment was conducted, where the G2P procedure was isolated. Specifically, the 2,6 and 10 words tests were performed with the same amount of samples, but this time the audio was captured from a headset's microphone, thus the words were clearly dictated and no background noise existed. The results are presented in table 6.

Table 4: Vocabulary expansion validation - Laboratory dataset (94 samples) - 6 word vocabulary

	Original	After SoX	After signal gating
Success rate	6.38 %	43.62 %	62.11 %
Error rate	0.00 %	40.43 %	28.42 %
No recognition rate	93.62 %	15.95 %	9.47 %

Table 5: Vocabulary expansion validation - Laboratory dataset (130 samples) - 10 word vocabulary

	Original	After SoX	After signal gating
Success rate	5.38 %	30.77 %	48.48 %
Error rate	1.54 %	50.00 %	40.76 %
No recognition rate	93.08 %	19.23 %	10.76 %

Table 6: Vocabulary expansion validation - Headset microphone test

	2 words 63 samples	6 words 94 samples	10 words 130 samples
Success rate	100.00 %	95.78 %	90.77 %
Error rate	0.00 %	2.11 %	7.69 %
No recognition rate	0.00 %	2.11 %	1.54 %

As obvious, the main reason of failure was the denoising method's effect on the actual words' phonemes, since the headset experiment indicated that the G2P can produce high success ratios when the audio is "clean". Specifically, it has been noticed that the dictated word in the denoised audio file has its limits (begin and ending) distorted. This has an erroneous ASR as a result, as many times the individual word's phonemes are not audible or recognizable in the audio signal. Nevertheless, the 7.69% error rate in the noise-free 10-words experiment shows that there are cases where Sphinx-4 cannot distinguish similar words. This can be partially corrected by a more detailed G2P process, but erroneous cases will always appear, since our problem is single-word recognition. Thus, Sphinx cannot be statistically helped by a language model, which could contain the proper words succession.

5. CONCLUSIONS & FUTURE WORK

The current paper presents a complete speech recognition tool for social robots. Specifically, we present a RAPP-based Sphinx-4 ASR application that involves a limited vocabulary, G2P techniques by which Greek words are supported, as well as audio denoising techniques necessary for the successful recognition of words in audio files captured by NAO. Our approach handles the issue of noisy environments and multi-user cases.

Results have indicated that the G2P approach can be a "quick and dirty" technique to perform automatic word recognition in languages Sphinx-4 does not currently support, avoiding the generally painful acoustic model training. Nevertheless, it was proven that this approach cannot be scaled for using a large vocabulary set, since the incorrect phonemes' transition probabilities that exist in the used English model prohibit the correct recognition of other languages' words. An additional conclusion is that the audio captured from the NAO robot's microphones is, in general, not suitable for speech recognition using Sphinx-4 due to the high levels of inherent noise. We indicated that the application of different denoising techniques can eliminate this defect, since high success rates were achieved in a small sized vocabulary. Nevertheless, the proposed denoising approaches distort the dictated words, something that becomes more and more apparent when the dictionary size increases. A final comment regarding the Sphinx-4 operation is that, when the configuration was maintained constant, successive speech recognition calls using the same audio file could provide different results. Specifically, it was quite usual for Sphinx not to be able to provide an output, but on resubmission of the same file a result was extracted.

As far as future work is considered, more elaborate denoising techniques must be investigated. Generally, when spectral denoising is concerned, the Audacity tools are preferred over SoX, since they have much better performance. A future approach may utilize parts of Audacity's source code or port its denoising methods into a ROS package, since no command line tool exists. Another interesting approach could be the gating applied in spectral information, or even Wiener filtering techniques involving DWT and Wavelet packets. In order to overcome the G2P problems, a different language's acoustic model can be used, whose phonemes are closer to Greek (such as a Spanish model). Of course the best (and generally proposed) course of action would be to train a Greek acoustic model for a limited vocabulary. Finally, if we still want to avoid the acoustic model training path, a more elegant and structured approach to G2P can be followed, either by refining the pronunciation rules or by utilizing a G2P tool, extracting the dictionary phoneme transcription in IPA format and then transforming them to the appropriate CMU phonemes. It should be noted that the proposed G2P methodology is not Greek-specific, but can be applied to other languages that are not currently supported via an acoustic / language model in Sphinx.

6. REFERENCES

[1] Gulzar, Taabish, Anand Singh, Dinesh Kumar Rajoriya, and Najma Farooq. "A systematic analysis of automatic speech recognition: an overview." Int. J. Curr. Eng. Technol 4, no. 3 (2014): 1664-1675.

[2] Walker, Willie, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, and Joe

Woelfel. "Sphinx-4: A flexible open source framework for speech recognition." (2004).

[3] Woodland, Phillip C., Julian J. Odell, Valtcho Valtchev, and Steve J. Young. "Large vocabulary continuous speech recognition using HTK." In Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on, vol. 2, pp. II-125. IEEE, 1994.

[4] Lee, Akinobu, and Tatsuya Kawahara. "Recent development of open-source speech recognition engine julius." In Proceedings: APSIPA ASC 2009: Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference, pp. 131-137. Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference, International Organizing Committee, 2009.

[5] Stiefelhagen, Rainer, Christian FÃijgen, Petra Gieselmann, Hartwig Holzapfel, Kai Nickel, and Alex Waibel. "Natural human-robot interaction using speech, head pose and gestures." In Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on, vol. 3, pp. 2422-2427. IEEE, 2004.

[6] Sidner, Candace L., Christopher Lee, Cory D. Kidd, Neal Lesh, and Charles Rich. "Explorations in engagement for humans and robots." Artificial Intelligence 166, no. 1 (2005): 140-164.

[7] Shamsuddin, Syamimi, Hanafiah Yussof, Luthffi Idzhar Ismail, Salina Mohamed, Fazah Akhtar Hanapiah, and Nur Ismarrubie Zahari. "Initial response in HRI-a case study on evaluation of child with autism spectrum disorders interacting with a humanoid robot Nao." Procedia Engineering 41 (2012): 1448-1455.

[8] Psomopoulos, Fotis, Emmanouil Tsardoulias, Alexandros Giokas, Cezary Zielinski, Vincent Prunet, Ilias Trochidis, David Daney et al. "Rapp system architecture." In Assistance and Service Robotics in a Human Environment, IEEE/RSJ International Conference on Intelligent Robots and Systems. 2014.

[9] Baxter, Paul, Tony Belpaeme, Lola Canamero, Piero Cosi, Yiannis Demiris, Valentin Enescu, A. Hiolle et al. "Long-term human-robot interaction with young users." In IEEE/ACM Human-Robot Interaction 2011 Conference (Robots with Children Workshop). 2011.

[10] Evermann, Gunnar, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell et al. The HTK book. Vol. 2. Cambridge: Entropic Cambridge Research Laboratory, 1997.

[11] Lee, Akinobu, and Tatsuya Kawahara. "Recent development of open-source speech recognition engine julius." In Proceedings: APSIPA ASC 2009: Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference, pp. 131-137. Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference, International Organizing Committee, 2009.

[12] Povey, Daniel, Arnab Ghoshal, Gilles Boulianne, LukÃąÅą Burget, OndÅŹej Glembek, Nagendra Goel, Mirko Hannemann et al. "The Kaldi speech recognition toolkit." (2011).

[13] Kimutai, Shadrack K., Edna Milgo, and David Gichoya. "Isolated Swahili Words Recognition using Sphinx4."

[14] Radha, V. "Speaker Independent Isolated Speech Recognition System for Tamil Language using HMM." Procedia Engineering 30 (2012): 1097-1102.

[15] Vojtko, Juraj, Juraj Kacur, and Gregor Rozinaj. "The training of Slovak speech recognition system based on Sphinx 4 for GSM networks." In ELMAR, 2007, pp. 147-150. IEEE, 2007.

[16] Ashraf, Javed, Naveed Iqbal, Naveed Sarfraz Khattak, and Ather Mohsin Zaidi. "Speaker independent Urdu speech recognition using HMM." In Informatics and Systems (INFOS), 2010 The 7th International Conference on, pp. 1-5. IEEE, 2010.

[17] Mubin, Omar, Jacob Henderson, and Christoph Bartneck. "You just do not understand me! Speech Recognition in Human Robot Interaction." In Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on, pp. 637-642. IEEE, 2014.

[18] Heinrich, Stefan, and Stefan Wermter. "Towards robust speech recognition for human-robot interaction." In Proceedings of the IROS2011 Workshop on Cognitive Neuroscience Robotics (CNR), pp. 29-34. 2011.

[19] Boll, Steven F. "Suppression of acoustic noise in speech using spectral subtraction." Acoustics, Speech and Signal Processing, IEEE Transactions on 27, no. 2 (1979): 113-120.

[20] Donoho, David L. "De-noising by soft-thresholding." Information Theory, IEEE Transactions on 41, no. 3 (1995): 613-627.

[21] Dimoulas, C., G. Kalliris, G. Papanikolaou, and A. Kalampakas. "Novel wavelet domain Wiener filtering de-noising techniques: application to bowel sounds captured by means of abdominal surface vibrations." Biomedical signal processing and control 1, no. 3 (2006): 177-218.

[22] Chalamandaris, A., S. Raptis, and P. Tsiakoulis. "Rule-based grapheme-to-phoneme method for the Greek." trees 18 (2005): 19.

[23] Serrano, Manuel, Erick Gallesio, and Florian Loitsch. "Hop: a language for programming the web 2. 0." In OOPSLA Companion, pp. 975-985. 2006.

[24] Petrounias, Evaggelos. "Modern Greek Grammar and Comparative Analysis." University Studio Press, Thessaloniki, Greece (1984).

[25] Papanastasiou, G., "Modern Greek spelling: history, theory, practice", Institute of Modern Greek Studies, Manolis Triandaphyllidis Foundation, Thessaloniki, Greece (2008).