A Greek voice recognition interface for ROV applications, using machine learning technologies and the CMU Sphinx platform

FOTIOS K. PANTAZOGLOU
Hellenic Centre for Marine Research
Institute of Oceanography
Former US base at Gournes
P.O. Box 2214, 71003 Heraklion
GREECE
fotis@hcmr.gr

GEORGIOS P. KLADIS
Hellenic Military Academy
16673 Vari Attikis
GREECE
gkladis@sse-tuc.edu.gr

NIKOLAOS K. PAPADAKIS *
Hellenic Military Academy
16673 Vari Attikis
GREECE
npapadakis@sse.gr

Abstract: Finding new technical solutions to command and control smart and high technology devices has become a necessity in our days. This is due to the fact that control of these devices, in a manual manner, may often become cumbersome for the operator especially when he/she is involved with numerous tasks. One way to overcome this, it is common practice the use of Automatic Speech Recognition (ASR) procedures, and this is the main topic of this article. In this article we present the implementation of a Greek CMU Sphinx model that can be used in Remotely Operated Vehicles (ROV) operations and applications. In particular, this work is focused in the development and training of the CMU Sphinx platform for the Greek language using well established machine learning tools and technologies .The generic Greek model and the Greek model for ROV applications are freely available in international repository via (https://gitlab.sse.gr/fpantazoglou/omilia and https://goo.gl/9v3QqG)

Key–Words: Human-machine interface, Machine learning, CMU sphinx, Pocketsphinx, Greek language, Automatic Speech recognition, Hidden Markov Models, Remotely Operated vehicles,

1 Introduction

New technologies and practices are characterizing our age with their rapid development and expansion. Automatic Speech Recognition (ASR) aims not only to improve our everyday life, but also to be used in a variety of high-tech systems and equipment. One of the oldest issues that the scientific community has dealt with is the application of ASR methods in systems .

According to [1] ASR has been used since the 50s. ASR involves the use of sophisticated microcomputers, Internet-based computing programs and programming languages, to be applied in various applications. This may improve everyday useful tasks and help in a number of more sophisticated and complicated apparatuses. CMU Sphinx [2] by Carnegie Mellon University, Sun Microsystems Laboratories, and the Mitsubishi Electric Research Laboratories, is a popular open source platform among the scientific community. Pocketsphinx is a CMU Sphinx variant developed by Huggins-Daines [3] as an answer to the rapid developments in small and smart devices like smartphones tablets microcomputers etc. Pocketsphinx is designed to be hand held and mobile-friendly, and is the main topic of this work.

*Corresponding author

Wide range of tools provided, and flexible design [2] [3] allow us to use them in speech recognition related applications development. The software including instructions is well documented and an active community is in place for commercial support. Through its 20 years age of ongoing support and development, both the CMU Sphinx and Pocketsphinx support a plethora of languages including English, German, French, Spanish, Russian, Mandarin etc. [4]. Both platforms are user friendly and due to the systematic way that those can be trained they can easily accommodate any language or specific dictionary according to the application requirements . Particularly, all the necessary tools are provided by the platforms for the creation of an accurate language model for any new task [5].

Still although there is a great number of works addressing the previous issues, only a amount is dedicated for the Greek language model as a basis, for ASR applications. For example, in [6] the authors used CMU Sphinx as part of the research project RAPP (Robotic Applications for Delivering Smart User Empowering Applications) and created a small Greek model which included only certain Greek words, and limits the use of their approach in platforms.

To overcome this a generic Greek language model was necessary as a basis for more specialized models that can be used in a variety of applications. So at first, the generic Greek language model that was created and introduced by [7]. Its development involved the use of the general principles for creating a new model according to the instructions given by the CMU Sphinx developer team. Those instructions are being used in any case a new language model is being implemented.

During the training process, a machine learning procedure, the platform is provided via sphinxtrain (https://goo.gl/YDxgeu), part of the CMU Sphinx package with audio data of the desired language and the corresponding transcribed texts to the audio data given. Through a statistical process the machine training algorithm creates the basic features of parameters for the desired language. At the same time through the decode, training results are tested in order to achieve greater accuracy. Since the training and adaptation of the Greek model is an adaptive and continuous process the authors expand on previous work by presenting a more application oriented derivative of their work. In particular this work involves, the control of a Remotely Operated Underwater vehicle (ROV).Specifically the authors trained and adapted the already created generic Greek Model so it can be used for voice control in ROV applications.

The remainder of this article is structured as follows: in Section 2 a brief literature review of ASR methods is presented.Section 3 refers to the Hidden Markov Models .In Section 4 CMU Sphinx, Pocketshinx and their design architectures are analyzed and in Section 5 a presentation of the development and implementation of the generic Greek model for the CMU Sphinx platform is included. Section 6 is dedicated to the machine learning process that resulted the ROV oriented Greek model and the paper concludes with Section 7.

2 Automatic Speech Recognition (ASR)

ASR is the process of converting a voice signal into a sequential string of words using a computational algorithm [8]. The main objective by the scientific community is to improve the accuracy of ASR used for various environments, for different types of spoken language, and speakers. The development of super processing and multi-core computing systems allowed the successful application of ASR approaches for applications [9]. ASR is being used in speech-to-speech programs, for example [10], or in daily used applications like skype (skype online translator).

ASR techniques are also widely used in voice search applications, intelligent digital assistants like Microsofts Cortana and Apples Siri and in the Gaming industry. New clever homes use ASR to allow the user to interact with the home control system and at the same time, in cars ASR based applications are used for informational or entertainment purposes [11].

One of the biggest challenges is the use of ASR for handicapped people [12] or the use of ASR for controlling high tech scientific apparatuses.

A brief presentation about the main characteristics of ASR systems and their architectures follows including also the main categories and modelling techniques that are used.

2.1 ASR Design Architecture

The typical architecture of an ASR system is depicted in Fig. 1. The system consists of four parts: A signal processing and extraction part, the Acoustic Model (AM), the Language Model (LM) and the Hypothesis Search (HS).The first part receives as input the audio signal.

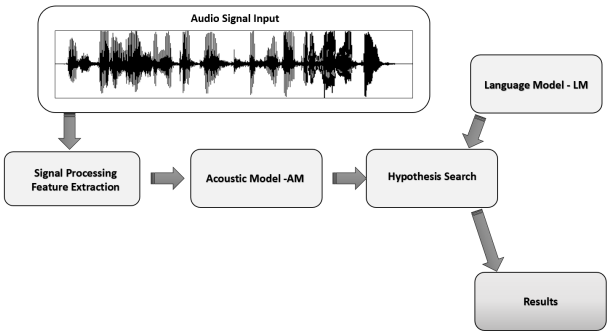

Figure 1: A typical architecture of an ASR system.

In particular, the input signal is filtered from noise and any disturbances, and is transformed into the frequency domain thus producing the principal features to be fed to the AM. The AM incorporates knowledge about acoustics and voice, and a score is generated for the variable length sequence.

The LM calculates the probability of a hypothetical word sequence, creating the LM score. Thereafter, the HS part combines the two scores produced and outputs the sequence of words with the greatest score.

2.2 ASR Categories

ASR systems are categorized and characterized by the expressions that are recognized. Those can be classified according to [13] as :

- Single word recognition systems. Systems that recognize individual words or single utterances. Typically, a period of silence is necessary in both parts of the sampling window while accepting individual words or expressions at a time.
- Expression recognition systems. These systems are similar to the above, but they allow the simultaneous processing of expressions as long as there exists a minimum gap between them.
- Continuous speech. These systems allow natural constant flow of speech by the user. It is one of the most difficult systems in creation as it contains techniques that define the different limits of expressions.
- Spontaneous speech. Is the speech that is natural and without rehearsal. A speech recognition system of spontaneous speech needs to be able to handle a variety of physical features of human speech, words that are used in the spoken language such as ”ah”, ”eh” etc.

CMU Sphinx and Pocketshpinx as platforms, are compatible with all ASR categories. The accuracy of the developed models is subject to the training of the model.

2.3 Modelling techniques

Different types in modelling techniques are used in the ASR process, in order to generate speaker models using the speaker specific feature vector. In their work [14] these are grouped as : The acoustic-phonetic approach [15], Pattern Recognition approach [20], Template based approaches [16], Dynamic Time Warping (DTW) approach [21], Knowledge based approaches [14], Statistical based approaches [20], Learning based approaches [14], Artificial intelligence approaches [14], Stochastic Approaches [14], and approaches based on Deep Neural Networks (DNN) [17].
A brief description of each modeling technique follows .

- **Acoustic-phonetic approach.** Based on the theory of vocal phonetics and assumptions.
- **Pattern Recognition approach.** Method is divided into two separate parts. Pattern training and recognition. Method based on HMM and can be applied to a sound, a word, or phrase.
- **Template based approaches.** Here in the input speech is compared to a set of pre-recorded words in order to find the best match.
- **Dynamic Time Warping.** DTW is an algorithm that measures similarities between two sequences that may vary in time or speed. It has been successfully applied to audio data.
- **Knowledge based approaches.** A knowledge base for a language is encoded into a machine. Being able to acquire this knowledge covering all cases of a language and then successfully using it, is not a trivial task.
- **Statistical based approaches.** Oral speech is statistically modeled using automated trained statistical processes. The main disadvantage of statistical models is that they must receive preliminary models of modeling. In ASR HMMs have been implemented with great success.
- **Learning based approaches.** To overcome the drawbacks of applications using HMM, learning methods such as neural networks and genetic algorithm programming can be introduced. In these learning models it is not necessary to provide clear rules or other knowledge.
- **Artificial intelligence approach.** This is a hybrid methodology between the acoustic-phonetic approach and the pattern recognition approach. However, this approach has had limited success, mainly due to the difficulty of quantifying existing knowledge.
- **Stochastic Approach.** Probability models are used to deal with uncertain or incomplete information. The most popular stochastic approach is the HMM, characterized by one of a finite state and a set of output distributions.
- **Deep Neural Networks Approaches.** With the implementation of DNN, error rates have been significantly reduced, due to computer power, the existence of large training data sets, and the better understanding of these models. DNN are capable of modeling complex non-linear relationships between inputs and outputs. This ability is very important for high quality and accurate acoustic modeling.

In particular, CMU Sphinx and Pocketsphinx are based on modeling with the use of HMMs, using Statistical based approaches.

3 Hidden Markov Models

A Hidden Markov Model, is one the machine learning model on which CMU Sphinx and Pocketsphinx are based. In his work Baum [18] first described Hidden Markov Models (HMM) defining them as a collection of states connected by transitions.As noted by Jurafsky [19] we use them in speech recognition, since they allow us to talk about observed events and hidden events in our probabilistic model. In his tutorial

Rabiner [20] describes he following components that specify a HMM:

1. **N**, the number of states in our model (starting and ending state included). Q=$q_1 q_2 ... q_N$
2. **M**, the number of distinct observation symbols per state which correspond to the physical output of the system being modeled.
3. **A**. An N x N state transition matrix, where a_{ij} represents the transition possibility from state i to state j.
   $a_{ij} = P[q_{t+1} = S_i \mid q_t = S_i], \quad 1 \leq i, j \leq N.$
4. $\mathbf{b_i}$. The observation output distribution in state i.
5. π. The initial state distribution $\pi = \{\pi_i\}$ where
   $\pi_i = P[q_1 = S_i], \quad 1 \leq i \leq N.$

Rabiner described in his work, the the steps which lead to the generation of an observation sequence $O = O_1 O_2 O_T$ from the HMM when the appropriate values of N, M, A, B and π are given. Those steps are:

1. According to to the initial state distribution π, choose an initial state $q_1 = S_i$.
2. Set t = 1.
3. According to the symbol probability distribution in state S_i, choose $O_t = v_k$.
4. Transit to a new state $q_{t+1} = S_j$ according to the state transition probability distribution for state S_i.
5. Set $t = t + 1$. If $t < T$ return to step 3 otherwise terminate the procedure.

By knowing the two model parameters (N and M) the specification of observation symbols and the specification of the three probability measures (A, B, π) we can specify an HMM. In general for our convenience we use the compact notation λ = (A, B, π) to indicate our complete set of parameters.

Rabiner [20] also introduced the idea that HMM should be characterized by three fundamental problems [19] :

Problem 1 (Likelihood): Given an HMM λ = (A, B) and an observation sequence O, determine the Likelihood P $(O \mid \lambda)$.

Problem 2 (Decoding): Given an observation sequence O and an HMM λ = (A, B), discover the best hidden state sequence Q.

Problem 3 (Learning): Given an observation sequence O and the set of states in the HMM, learn the HMM parameters A and B.

To solve the three fundamental problems we use different algorithms as presented in detail by [19]. Those are, the Forward Algorithm for the Likelihood problem the Viterbi Algorithm for the Decoding problem and the Forward-Backward (or Baum-Welch) Algorithm for the Learning problem.

4 CMU Sphinx and Pocketsphinx platforms

CMU Sphinx has been developed jointly by Carnegie Mellon University, Sun Microsystems Laboratories and Mitsubishi Research Laboratories [2]. It has been developed in Java and supports all types of audio models represented via HMM. CMU Sphinx is highly flexible, supports all types of HMM-based acoustic models and multiple search strategies. It is an open source project that supports most common languages and its code is freely available at SourceForge[1].

At the same time, it is easier to exploit the capabilities of computing machines with multiple processors in the process of training for our model with large sets. CMU Sphinx allows the user to alter the language model by parametrizing only one part of the system and without affecting the overall architecture.

Pockesphinx is a lightweight version of the CMU Sphinx. It was first intended to be a version OS SPHINX-II for use on mobile devices and hardware that lacked on performance specifications. However, as Huggins-Daines showed [3], although its reasonable performance it needs a new design architecture.

This is developed in ANSI C programming language, and the requirements were namely, application portability, simple implementation, small code and data footprint, security, memory performance [22]. Due to the specifications of the working platform addressed in this work, a laptop computer of modest computing power (Intel core Duo with 4 gbite RAM) is used for Pockesphinx.

4.1 CMU Sphinx Design Architecture

Lamere in his work [2] gave a very detailed presentation of the CMU Sphinx platform and its architecture. The design Architecture of CMU Sphinx and the inter dependencies communications among its blocks is depicted in Fig. 2. The main blocks are the Frontend, Decoder and Knowledge base. All blocks, except those in the Knowledge Base, are independently replaceable custom made software modules written in Java.

In Frontend module speech is received and parametrized. The Decoder incorporates the linguist,

[1] https://goo.gl/nzRDmF

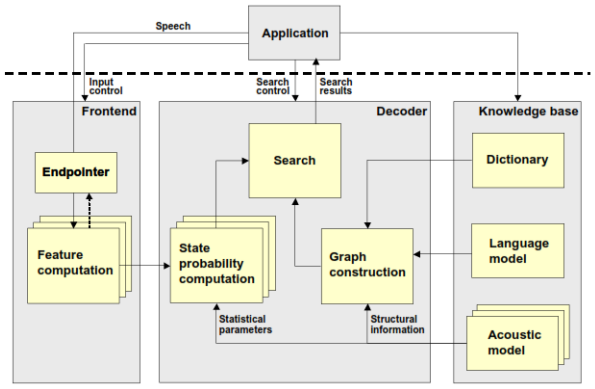

Figure 2: CMU Sphinx design architecture (Lamere et al., 2003).

the search manager, and the acoustic scorer while the Knowledge base includes the Acoustic (AM) and Language model (LM) as well as the Lexicon (or Phonetic Dictionary).

The front end block and its several communicating blocks, is shown in Fig. 3.

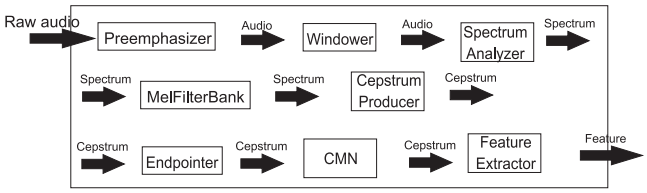

Figure 3: CMU Sphinx Front end architecture (Lamere et al., 2003).

Each block is activated via the excitation of its input and processes the information to determine whether it is voice data or a control signal. The control signal sets the beginning or ending of the speech, something very important for the decoder. If the incoming data is speech then they are processed and stored temporarily for the next block.

One of the design features of the program is the fact that all the block outputs can be used simultaneously. The input data is not necessarily an input of the first block. It can be introduced to numerous blocks. It is also possible for any of the blocks to be replaced for filtering purposes, reduction of noise, etc. The characteristics calculated using independent sources of information can be fed directly to the decoder, or alongside the speech signal, or by skipping it all together. The entire system allows continuous operation, and has the ability to function in a very flexible way.

The decoder incorporates the search manager, the linguist and the acoustic scorer blocks. The Search manager is responsible for the construction and search of a probability tree for the best matching case. The search tree construction is based on the information provided by the linguist. Additionally, the search manager communicates with the audio scorer to obtain the scores for the incoming audio data. For a more detailed treatment of the algorithms used the reader is referred to the Viterbi [23] and Bushderby [24] works.

According to [25], the linguist creates the search graph that the decoder uses when searching for words, while concealing the complexities associated with creating this graph. The search graph is created using the language structure, as represented by a given LM and the topological structure of the AM. The linguist can also use a dictionary (usually a pronunciation dictionary) to match words from the LM to data sequences of the AM. When creating the search graph, the linguist can incorporate sub-units of arbitrary length frames. The design allows the user to provide different configurations for various systems and recognition requirements. For example, a simple number recognition application can use a simple linguist that keeps the entire memory search process, while larger applications with vocabularies of the order of 100.000 words use smarter linguists by storing only a certain number of data from the find process in memory. The aim of the acoustic scorer is to calculate the probability of the output state or the probability density values for the different states for any given input vector.

The requirement is the communication potential with the Frontend blocks, to obtain the extracted attributes. The reviewer keeps all information about the probability densities of the output states while the search block ignores, whether the scoring is done with continuous, semi-continuous or discrete HMM. Finally, the design allows the user to include conventional Gauss scoring procedures, or scoring approaches based on game theory, for example.

4.2 Pocketsphinx Design Architecture

The Pocketsphinx decoder architecture has been derived mainly from Sphinx-II and its architecture design [22]. The design architecture is depicted in Fig. 4.

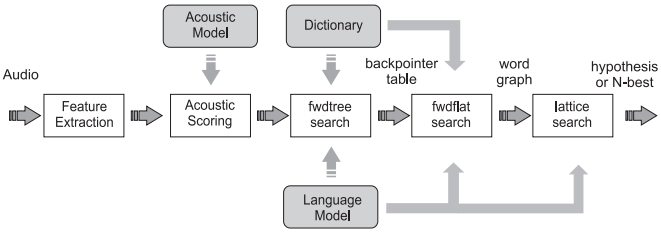

Figure 4: Pocketsphinx Decoding Architecture (Daines, 2011).

During the decoding phase, Pocketsphinx uses a multi-search approach [2]. This involves the, FWDTREE, FWDFLAT and BESTPATH stages.

The FWDTREE search method is based on a dictionary that has a tree structure. It separates the search graph in the vocabulary tree and in the word sequence. In order to have a fast computational process, the method uses a large number of adjustable, width wise beams. In particular, a separate beam is applied to the last phoneme of each word from all HMMs. The FWDTREE search method is quick but not very accurate.

The FWDFLAT search method looks for a grid created by the FWDTREE but does not re-rank it on the same grid. It essentially reassesses the fact that the active vocabulary for search has already been assumed by the FWDTREE. Thus, the total number of words as a case is clearly more limited than the original number of words. Hence, a dynamic association of each phrase with only that set of words is possible during decoding, which is attractive for portable devices of modest specs.

The BESTPATH lattice generation (best-path) method is a rewrite of the word table word grid that has been generated by the FWDFLAT search. The recalibration methodology is based on the Dijkstra shortest path algorithm [26]. In fact, the first search method is principal and the other two are used for corrective and complementary purposes.

5 Implementation of the generic Greek model

The flow chart for the entire process is depicted in Figure 5 and a description of each phase follows.

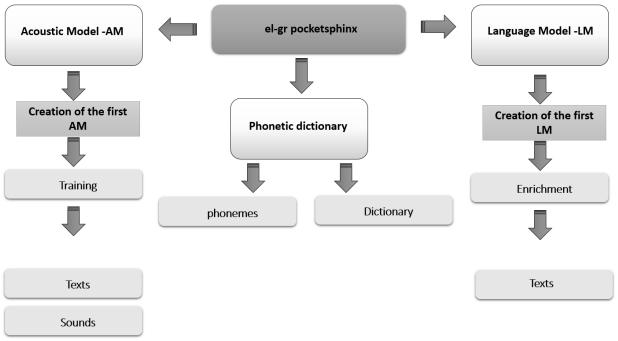

Figure 5: Implementation of the generic Greek model-flow chart.

5.1 Implementation of the Greek language model

Language Models calculate the probability of a hypothetical word sequence, creating the LM score. They can describe more complex languages, like Greek, and are ideal for cases where we have free form, such as verbal speech, inputs.

There are two ways to create the language model. The former involves small and limited vocabularies by making use of the on-line tool developed by Carnegie Mellon University [3] .This method also assumes the existence of a generic model. By the latter, for larger language models (i.e. 20000 words) and in the absence of a generic model, like in the Greek language case, then the steps in the sequel can be performed.

1. Installation of the CMUCLMTK package of programs that are provided by the Carnegie Mellon University.

2. Preparation of the necessary texts which need to be of specific form. In particular, there are no capital letters, and punctuation marks other than tones. Separate phrases are labeled by the $< s >$ and $< /s >$ symbols and there are no number symbols. Still the number symbols can be represented in verbal form. For the Greek language model, the training data were created from various sources: texts from daily news, books of different themes, and subtitles from documentaries. These adopt 'simple spoken' language avoiding any technical terms or specialized vocabulary. It is noted that it is possible to improve the language model by adding additional data. At present, the Greek language model consists of over 19000 words that represent the most common words used in daily speaking.

3. Thereafter all text samples are transformed into binary format via the tools provided from the CMUCLMTK package and the LM is created.

It is noted that for the application considered to be used in ROV operations a specific language model was created by the proper selection of particular words/phrases.

5.2 Implementation of the Greek phonetic dictionary

One of the most important parts of the generic Greek model is the Phonetic dictionary. It is essential, to convert Greek words into a form that is compatible with the CMU Sphinx and Pockesphinx software.

[2]https://goo.gl/N7bk3n

[3]http://www.speech.cs.cmu.edu/tools/lmtool-new.html

Initially, such a conversion involves a specific alphabet. The alphabet used internationally on the CMU Sphinx platform is ARPABET [4] . ARPABET is the voice representation developed by ARPA (Advanced Research Projects Agency) in the framework of speech comprehension programs. Each phoneme of the English alphabet is matched with a separate ASCII character string. For the case of the Greek vocabulary creation two sub-tasks are performed. By the former, a sufficient number of Greek words is needed to create the dictionary, and by the latter it is necessary to convert this dictionary into a compatible form to be used by the system. This is included in the sequel.

5.2.1 Dictionary creation

For the creation of the dictionary, open source software and applications are preferred. It is noted that it is not possible to include all words of our daily spoken language in the dictionary. For example, scientifically terms, toponyms or surnames are not included. Still, those are added so that it responds effectively during the recognition process.

The first source of words used is provided by the Institute for Language and Speech Processing as a result of the work by researchers [27] .

Furthermore, the Greek lexicon [5] of the open source program OpenOffice/LibreOffice is used. The words are edited in order to avoid duplication of records and they are converted to binary form to be used by the Sphinx CMU platform.

5.2.2 Word to phoneme conversion mechanism-The Greek phonemes

After the creation of the dictionary, the words are transformed into a set of phonemes compatible with the CMU Sphinx and Pocketshpinx system. Initially the Greek phonemes are defined and the transformation mechanism is deployed. In the Greek language there exist 24 letters and 25 phonemes:

- **Greek letters** $\alpha, \beta, \gamma, \delta, \epsilon, \zeta, \eta, \theta, \iota, \kappa, \lambda, \mu, \nu, \xi, o, \pi, \rho, \sigma, \tau, \upsilon, \phi, \chi, \psi, \omega$
- **Greek phonemes** $\alpha, \epsilon, \iota, o, o\upsilon, \beta, \gamma, \delta, \zeta, \theta, \kappa, \lambda, \mu, \nu, \pi, \rho, \sigma, \tau, \phi, \chi, \mu\pi, \nu\tau, \gamma\kappa, \tau\sigma, \tau\zeta$

There are phonemes that contain two letters, and there are letters such as η, ι, which correspond to the same phoneme i. There exist the double letters ξ, ψ, and double letters corresponding to a single phoneme such as the one corresponding to ϵ. In the Greek alphabet we have the final sigma ς, and double letters such as $\beta\beta, \kappa\kappa, \lambda\lambda$ etc. Finally, the $\alpha\upsilon$ and $\epsilon\upsilon$ correspond to a different vocal imaging, according to the letter that follows, as shown by Tsardoulias [6].

Taking into account all the above and the phonemes that CMU Sphinx uses, the list of Greek phonemes used in the Greek model is :

- **Greek CMU Sphinx phonemes** AE, AA, B, CH, D, DH, EH, F, G, HH, IH, JH, K, KS, L, M, N, OW, P, PS, R, S, SIL, T, TH, UW, V, W, Z, ZB, ZD, ZDH, ZL, ZM, ZN, ZR, ZV,ZW

Provided the list of Greek phonemes it is easy to create the phonetic dictionary with the use of an available algorithm. The specific steps during this procedure in ARPABET format are :

1. Conversion of all letters to minuscule.
2. Replacement of the double and triple combinations of letters with their corresponding phrases in the following order: $o\upsilon, o\acute{\upsilon}, \sigma\mu\pi, \sigma\nu\tau, \sigma\gamma, \sigma\beta, \sigma\delta, \sigma\mu, \sigma\nu, \sigma\lambda, \sigma\rho, \mu\pi$ $\nu\tau, \gamma\kappa, \gamma\gamma, \tau\varsigma, \tau\zeta, \sigma\varsigma, \kappa\kappa, \beta\beta, \lambda\lambda, \mu\mu, \nu\nu, \pi\pi, \rho\rho$ $\tau\tau$.
3. Replacement of the double special vowel combinations with the corresponding vocals in the following order: $\alpha\iota, \alpha\acute{\iota}, \epsilon\iota, \epsilon\acute{\iota}, o\iota, o\acute{\iota}, \upsilon\iota, \upsilon\acute{\iota}, \alpha\upsilon, \alpha\beta, \epsilon\upsilon, \epsilon\beta, \epsilon\acute{\upsilon}, \acute{\epsilon}\beta$ $\alpha\upsilon, \alpha\phi, \alpha\acute{\upsilon}, \acute{\alpha}\phi, \acute{\epsilon}\phi$.
4. Replacement of all other remaining letters with the corresponding phonemes.

After the completion of the Greek dictionary, a necessary step is the calibration and training of the Greek Acoustic Model (AM).

5.3 Training of the Greek AM

The most cumbersome task in the overall process in the creation of the generic Greek model is the AM. This is a machine learning process that leads to a series of statistical records that contains information about speech units. In fact it is the "mapping" of the language used in voice recognition services. For this mapping to be performed the system is fed with a large number of audio files which are matched to the written transcribed text. Depending on the calculational power and amount of data this is the most cost effective step for the training of the AM. For a near to complete successful process, the more the speakers and the variety of subjects spoken included, the better the training outcome. For example, using in the system sound data that refer only to scientific or economic

[4] http://www.speech.cs.cmu.edu/cgi-bin/cmudict
[5] http://www.elspell.gr/

field, may yield a poor performance in speech recognition. Audio data is required to be of specific format for use in the Pocketsphinx and sphinxtrain [6].

Sphinxtrain is responsible for the training of the AM. Training is necessary since the AM for a new language is required. Specifically, the audio data is in MS WAV format, mono at 16 kHz 16 bit. A range of different sound sources were chosen to achieve thematic and sound pluralism. These are:

- Recordings of individuals. Individuals were asked to read either texts or speak fluently and there were recorded according the general instructions given by the CMU Sphinx development team.
- News recordings. A large amount of data texts of the current news, from various thematic sections, on selected, which were then pronounced in a synthetic female language with the help of the Google translate service and were recorded via the Audacity Free Software program [7].
- Free Audio books. Free Audio-books were used which are provided by the Greek Ministry of Education [8].
- Audio data from Academic Lectures. A very important source of good audio data was the open lectures of Hellenic universities that are in the Open course repository [9] . These were edited to extract only the sound needed, and then edited to become compatible according to the data specifications.
- Meetings of the Greek Parliament [10] . The repository of audiovisual material of the Greek Parliament has a variety of different subject materials, as well as many different speakers. These were processed and stored in the appropriate format for the sphinxtrain training mechanism.

Overall, the audio data material used for the training process had a total duration of 358:09:38 hours. The overall accuracy of the generic model is derived from the combination of all individual statistics that occur during the training process. Having the individual statistics, one can intervene in the training and hardware parameters, and then introduce recurrent training processes until a better accuracy is achieved for the underpinned application.

The implementation of the generic Greek model is the necessary step in order to develop a smaller use case oriented Greek model that can be used in ROV applications. The description of this model follows.

[6]https://goo.gl/noanMQ
[7]http://www.audacityteam.org/
[8]http://ebooks.edu.gr/new/
[9]http://opencourses.gr
[10]http://www.parliament.gr/

6 The Greek model for ROV applications

ROVs are a family of submarine robotic vehicles that allow tasks to be performed in conditions and environments that are neither friendly nor easily accessible by humans. Connected via an umbilical cable with the support ship, they allow the operator to undertake scientific missions that are characterized by high precision in sampling and data quality. One of the most valuable tools of an ROV, is its manipulator system.

Figure 6: Hydro-Lek manipulator during MaxROVER operations (https://goo.gl/DSmoz6).

Via such a system it allows the operator to perform from simple tasks (i.e. picking objects) to more complicated procedures (i.e. maintenance, sampling and surveying of the sea-bed) (Fig. 6). The manipulator systems are remotely controlled by the operator. The most basic manipulator system consists of a control device, arms, joints and a gripper as the end-effector where different tools can be used. The most sophisticated systems have complex electronic control systems and a large number of degrees of freedom in movement.

One of the most important part of the system is the operator's interface system to perform tasks. The handling systems may vary according to the company and depending on the degrees of freedom of the manipulator. These may include from simple levers for operation of the manipulator to more sophisticated handling systems.

Such operations require dexterity and a good knowledge of the system by the operator for the handling. Usually, in ROV tasks, the shift job is six hours, thus adds a further degree of difficulty to the entire process. The handling performed is needed to be accurate and at the same time may require repetition. In this context, an ASR system that combines sim-

ple rules, ease of use, repetition, can decisively aid to further increase the productivity of the tools used by the ROV and its operation by the user. Thus the user is able to operate the existing control and command systems on board using simple voice commands, and focus in other tasks. Voice commands are clearly simpler and can improve the ease of operation in a number of applications.

Having this in mind the already developed generic Greek model, illustrated in the previous section, is used for the creation of a smaller ROV oriented Greek model. Since we need higher accuracy we have to adapt the already created generic Greek model using the technique already presented by the CMU Sphinx development team. A smaller pruned language model (LM) is also needed since large LMs are not helping in the overall accuracy.

From our working experience commands like *stop, turn left, turn right, take sample, open, slow, more, less, close, add waypoint, mark*, etc. can be used in a set of voice commands that can ease the work of a ROV team. Therefore as a first step in the adaptation process, new phrases with text that includes the potential command words, where used to create the new pruned LM. The new, much smaller in size, LM is created with the process already described in Section 5.1.

Besides the LM, the new adopted AM that fits our ROV operations is fed with the new training material for more accurate performance. The new training, material is used from the recording of 30 phrases (depicted in Fig. 7 translated in English) by a large number of individuals. In those phrases, the principal commands that an ROV system operator team is naturally using is included. The phrases don't have any conceptual meaning .

The number of individuals chosen for the data were divided equally with respect to genre. Thereafter the recorded and edited data set was introduced to the sphinxtrain program for the adaptation process that actually recalculates all the statistical information and characteristics that is needed from the Acoustic Model (AM).

During the testing process of the new ROV adapted model, the accuracy of the model was in most cases 100% as depicted in Fig. 8.

HCMRs [11] MaxROVER (Fig. 9) is an Observation/Light Work Class ROV that is certified to dive and work until 2000 m depths. It is been widely used for research projects and has the capability to monitor and record marine habitats, to take samples and conduct precise measurements of scientific interest.

The Manipulation system of MaxROVER consists from two simple 5 DOF manipulators provided by Hydro-Lek [12] .Those are controlled with the help of a simple and standard 6 joystick system. Extra tools like a built in line cutter can be used to break, cut, or lift objects. Especially designed sampling and measurement apparatuses can be added to the MaxROVER and this gives an extra flexibility in the number of tasks it can undertake[13]. A built-in sampling basket is not present and this fact differentiates the sampling strategy with the use of separated baskets that are deployed in coordination with the ROV deployment. This adds more working load to the ROV operating team and reduces the number of simultaneous tasks that it can perform during a single dive.

Usually the HCMR ROV operating team is 4 technical scientists. A dive-supervisor, a pilot, a sonar and arm manipulation operator and a data log responsible. Since the system is mainly operated in a manual mode, a high degree of collaboration is needed between the ROV operating team members the scientists on board and the support ship personnel.

In a common use scenario we can use the Greek model for ROV operations in a number of tasks un-

```
<s> start straight ahead </ s> (1)
<s> turn right turn left slowly </ s> (2)
<s> start stop keep going back </ s> (3)
<s> speed increase speed decrease </ s> (4)
<s> one hundred eighty degrees </ s> (5)
<s> turn forty five degrees </ s> (6)
<s> take sample grab item </ s> (7)
<s> open close network close claw </ s> (8)
<s> course one hundred twenty degrees </ s> (9)
<s> quieter sixty degrees </ s> (10)
<s> still a little tense </ s> (11)
<s> reduce opening arm volume </ s> (12)
<s> close left-hand arm </ s> (13)
<s> front right rear right </ s> (14)
<s> make circle ninety degrees open lights </ s> (15)
<s> leave object open claw </ s> (16)
<s> two hundred and seventy degrees </ s> (17)
<s> start stop </ s> (18)
<s> one more one less </ s> (19)
<s> start two stops two </ s> (20)
<s> two more two less </ s> (21)
<s> start three stops three </ s> (22)
<s> three more three less </ s> (23)
<s> start four stops four </ s> (24)
<s> four more four less </ s> (25)
<s> starts five stops five </ s> (26)
<s> five more five less </ s> (27)
<s> three hundred sixty degrees closed lights </ s> (28)
<s> top right down left slowly slowly </ s> (29)
<s> sample object open closed </ s> (30)
```

Figure 7: Set of phrases used for the training of the ROV Greek model.

[11] https://www.hcmr.gr

[12] http://www.hydro-lek.com/

[13] https://bit.ly/2zJLp4q

```
ξεκίνα πορεία ευθεία σιγά  (user-271)
ξεκίνα πορεία ευθεία σιγά  (user-271)
Words: 4 Correct: 4 Errors: 0 Percent correct = 100.00% Error = 0.00% Accuracy = 100.00%
Insertions: 0 Deletions: 0 Substitutions: 0
στρίψε δεξιά στρίψε αριστερά σιγά σιγά  (user-272)
στρίψε δεξιά στρίψε αριστερά σιγά σιγά  (user-272)
Words: 6 Correct: 6 Errors: 0 Percent correct = 100.00% Error = 0.00% Accuracy = 100.00%
Insertions: 0 Deletions: 0 Substitutions: 0
σταμάτα ξεκίνα συνέχισε κάνε πίσω  (user-273)
σταμάτα ξεκίνα συνέχισε κάνε πίσω  (user-273)
Words: 5 Correct: 5 Errors: 0 Percent correct = 100.00% Error = 0.00% Accuracy = 100.00%
Insertions: 0 Deletions: 0 Substitutions: 0
αύξησε ταχύτητα μείωσε ταχύτητα  (user-274)
αύξησε ταχύτητα μείωσε ταχύτητα  (user-274)
Words: 4 Correct: 4 Errors: 0 Percent correct = 100.00% Error = 0.00% Accuracy = 100.00%
Insertions: 0 Deletions: 0 Substitutions: 0
```

Figure 8: Part of the training report for the ROV Greek model.

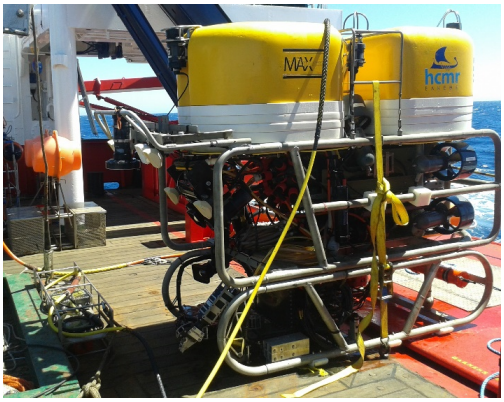

Figure 9: HCMR MaxROVER ROV.

dertaken by the pilot, the sonar and arm manipulation operator and the data log responsible. Simple small computers like the Raspberry Pi 3 (which already supports Pocketsphinx) can be used to automate some of the work done.

The pilot would demand the ROV for example to follow a certain path ,following way-points, via voice commands like start, stop, take right turn, heading 60 degrees etc. The sonar and arm manipulation operator can use voice commands to take samples and conduct measurements in a more easy and fast way. Finally the data log responsible can with the help of voice commands and special programmed applications handle a bigger amount of collected data at the same time. It is noted that the accuracy of recognition is influenced greatly by factors such as the noise of the surrounding environment, the quality of the speech of the speaker and the stress in the speech during operations.

7 Conclusions

Human machine interaction technologies like ASR, gaining momentum the past decades. Voice recognition and voice communication technologies have contributed significantly to a range of our day-to-day activities. Taking advantage of the rapid growth and the spread of technology, we have at our disposal a range of tools that enable us to improve both our daily routine and our performance in a series of very demanding tasks.

CMU Sphinx and Pocketsphinx are ASR platforms that allow the development of those tools via a systematic methodology, which is well documented and easy to reproduce. Due to the systematic method, which can be easily followed, a Greek voice recognition interface for ROV applications was described in this work. Although, the special characteristics and features of the Greek language it was proved herein that the CMU sphinx can easily accommodate such challenges, and may be used and cater any special requirements according to the application concerned. The interested reader is referred to [7] for a detailed treatment of the implementation of the generic Greek model for CMU Sphinx.

After the introduction of the generic Greek model for CMU Sphinx [7] the authors used the methodology provided by the CMU Sphinx development team to further implement a Greek model that can be used in ROV operations and applications. This was necessary since we needed a specialized model for a small vocabulary application (ROV control).

The development of the ROV oriented Greek model, including the new adapted Acoustic Model, the Voice Dictionary and the new Language Model was performed using a modest computing system.

Both the generic Greek model of CMU Sphinx and the Greek model created for ROV operations can be found in (https://gitlab.sse.gr/fpantazoglou/omilia and https://goo.gl/9v3QqG) for the interested reader.

Nowadays microcomputers (like Raspberry pi) are used for various applications in maritime applications [28]. Those can be easily programmed and are a capable environment where CMU Sphinx can be deployed. Future work of the authors will involve the deployment of the presented Greek model on a Rasperry Pi for the control of manipulators for ROVs. The new voice guided control system, will aid towards reducing the work load of ROV pilots, improving accuracy and productivity.

Acknowledgements: The authors would like to thank Mr. Nickolay Shmyrev, developer and support officer for the CMU Sphinx platform. Mr. Shmyrev contributed decisively with his experience and enthusiasm during the training and customization phase of the implementation of the Greek Model.

References:

[1] T. B. Martin, H. J. Zadell, E. F. Grunza, M. B. Hmcher, and D. R. Reddy, Speech Recognition by Machine: A Review, *Proc. IEEE Conf. Rec. Soc. Amcr. ZYans. Comput. Tech. Rep* vol. 6443, no. 2, 1974, pp. 541–546.

[2] P. Lamere et al., The CMU SPHINX-4 speech recognition system, *IEEE Intl. Conf. Acoust. Speech Signal Process. (ICASSP 2003), Hong Kong* vol. 1, 2003, pp. 2-5.

[3] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky, Pocketsphinx: A Free, Real-Time Continuous Speech Recognition System for Hand-Held Devices *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.* vol. 1, 2006, pp. 185-188.

[4] CMU Sphinx Acoustic and Language Models, https://bit.ly/2SzLkYd, 2018.

[5] CMU Speech Recognition Toolkit, https://bit.ly/2QC00bL, 2018.

[6] E. G. Tsardoulias, A. L. Symeonidis, and P. A. Mitkas, An automatic speech detection architecture for social robot oral interaction, *Proc. Audio Most. 2015 Interact. With Sound*, 2015.

[7] F. K. Pantazoglou, N. K. Papadakis, and G. P. Kladis, Implementation of the generic Greek Model for CMU Sphinx speech recognition toolkit, *eRA-12 International Scientific Conference*, 2017.

[8] M. Anusuya and S. Katti, Speech recognition by machine: A review, *Int. J. Comput. Sci. Inf. Secur.*, vol. 6, no. 3, 2009, pp. 181-205 .

[9] D. Yu, Automatic Speech Recognition: A Deep Learning Approach, *Springer*, 2014.

[10] D. Stallard et al., The BBN TransTalk Speech-to-Speech Translation System, *English, no. June*, 2009.

[11] M. L. Seltzer, Y. C. Ju, I. Tashev, Y. Y. Wang, and D. Yu, In-car media search, *IEEE Signal Process. Mag.*, vol. 28, no. 4, 2011, pp. 50-60.

[12] A. Samah and A. A. Osman, Controlling Home Devices for Handicapped People via Voice Command Techniques, 2015.

[13] K. Geetha and E. Chandra, Automatic Speech Recognition - An Overview, *Int. J. Eng. Comput. Sci.*, vol. 2, no. 3, 2013, pp. 633-639.

[14] S. K. Gaikwad, B. W. Gawali, and P. Yannawar, A Review on Speech Recognition Technique, *Int. J. Comput. Appl.*, vol. 10, no. 3, 2010, pp. 16-24.

[15] Hemdal, J. F., G. W. Hughes, A feature based computer recognition program for the modeling of vowel perception, *Models for the Perception of Speech and Visual Form, MIT Press(Models for the Perception of Speech and Visual Form)*, 1967.

[16] R.K.Moore, Twenty things we still dont know about speech, *Workshop on Progress and Prospects of speech Research and Technology*, 1994.

[17] G. Hinton et al., Deep Neural Networks for Acoustic Modeling in Speech Recognition, *IEEE Signal Process. Mag.*, no. November, 2012, pp. 82-97.

[18] Baum and L., An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process, *Inequalities*, vol. 3, 1972, pp. 1-8.

[19] D. Jurafsky and J. Martin, Hidden Markov Models, *Speech Lang. Process.*, no. Chapter 20, 2017, p. 21.

[20] L. R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, *Proc. IEEE*, vol. 77, no. 2, 1989, pp. 257-286.

[21] Senin, P., Dynamic Time Warping Algorithm Review, *Science*, December 2007, pp. 1-23

[22] D. H. Daines, An Architecture for Scalable , Universal Speech Recognition *Thesis Committee: c 2011 David Huggins Daines*, 2011, p. 131.

[23] A. J. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, *IEEE Trans. Inf. Theory*, vol. 13, no. 2, 1967, pp. 260-269.

[24] R. Singh, M. K. Warmuth, B. Raj, and P. Lamere, Classification with Free Energy at Raised Temperatures, *EuroSpeech*, no. 2, 2003, pp. 1773-1776.

[25] W. Walker et al., Sphinx-4: A Flexible Open Source Framework for Speech Recognition, *Smli*, no. TR-2004-139, 2004, pp. 1-9.

[26] E. W. Dijkstra, A note on two problems in connexion with graphs, *Numer. Math.*, vol. 1, no. 1, 1959, pp. 269-271.

[27] A. Protopapas, M. Tzakosta, A. Chalamandaris, and P. Tsiakoulis, IPLR: an online resource for Greek word-level and sublexical information, *Lang. Resour. Eval.*, vol. 46, no. 3, sep 2012, pp. 449-459.

[28] G.Divya Priya , Mr.I.Harish, Raspberry PI Based Underwater Vehicle for Monitoring Aquatic Ecosystem, *International Journal of Engineering Trends and Applications (IJETA)*, vol. 6, iss. 2, Mar-Apr 2015.