VOIce-based Community-cEntric mobile Services for social development

Grant Agreement Number 269954

Deliverable No D3.1

Report on state of the art and development methodology

September 2011

SEVENTH FRAMEWORK PROGRAMME

THEME ICT-2009.9.1 – International cooperation

PROJECT DELIVERABLE REPORT

Project	
Grant Agreement number	*269954*
Project acronym:	*VOICES*
Project title:	*VOIce-based Community-cEntric mobile Services for social development*
Funding Scheme:	*Collaborative Project*
Date of latest version of Annex I against which the assessment will be made:	*18 February 2011*
Document	
Deliverable number:	D3.1
Deliverable title	Report on state of the art and development methodology
Contractual Date of Delivery:	September 2011
Actual Date of Delivery:	30 September 2011
Editor (s):	
Author (s):	Paul Bagshaw, Etienne Barnard, Olivier Rosec
Reviewer (s):	Adele Botha, Max Froumentin, Filipe Cabral Pinto
Work package no.:	WP3
Work package title:	Speech Technologies
Work package leader:	NWU
Work package participants:	W3C, FT, CSIR, PTIN
Distribution:	PU
Version/Revision:	1.0
Draft/Final:	Final
Total number of pages (including cover):	29
Keywords:	Text-to-speech synthesis, automatic speech recognition, under-resourced languages

CHANGE LOG

Reason for change	Issue	Revision	Date
Document creation	0.1	Etienne Barnard	13-9-2011
First revision	0.2	Paul Bagshaw	13-9-2011
Reviewed draft	0.3	Max Froumentin, Adele Botha	19-9-2011
Reviewed draft	0.4	Filipe Cabral Pinto	21-9-2011
Release version	1.0	Etienne Barnard	21-9-2011

DISCLAIMER

This document contains description of the VOICES project work and findings.

The authors of this document have taken any and all available measures in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any responsibility for actions that might occur as a result of using its content.

This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the VOICES consortium and can in no way be taken to reflect the views of the European Union.

The European Union is established in accordance with the Treaty on European Union (Maastricht). There are currently 27 Member States of the Union. It is based on the European Communities and the member states cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice and the Court of Auditors. (http://europa.eu.int/)

VOICES is a project funded in part by the European Union.

TABLE OF CONTENTS

SUMMARY

This report contains information on the two speech technologies that play a key role in VOICES, namely *Text-to-Speech* (TTS) synthesis and *Automatic Speech Recognition* (ASR). For both of these technologies, we provide a brief overview of the building blocks employed in state-of-the-art systems, and then describe the most popular algorithms that are used in each of these modules.

For TTS synthesis, the major building blocks perform linguistic and acoustic processing, respectively. The most important algorithms for linguistic processing, related to text normalisation and pronunciation modelling, are fairly well established. Acoustic-processing algorithms, on the other hand, are more diverse, with alternatives based on parametric models, articulatory models, formant models and Hidden Markov Models all being realistic alternatives in different environments.

ASR systems can be viewed as the combination of language models, acoustic models and pronunciation dictionaries. Again, a core of widely-used algorithms has been developed for each of these functions, built around the concepts of *n*-gram language models and Hidden Markov Models as acoustic models.

Practical guidelines on the development of instances of TTS and ASR systems are provided, along with suggestions on tools that may be used for such development. The challenges of speech-technology development for under-resourced languages are given particular attention, and we demonstrate that current methods still require a substantial level of expertise for successful technology development in any language for which speech technology does not currently exist.

INTRODUCTION

Although systematic research into speech technology has been actively pursued for at least 50 years, and commercial products based on speech technology have been available for almost the same length of time, the development of general-purpose speech technology systems continues to be a challenge. In particular, current state-of-the-art approaches tend to work well only within a narrow set of constraints, and significant expertise is required to create a system for a particular task domain or target language [Huang et al., 2001]. The last of these variables is particularly challenging: since high-quality speech technology requires several specialized resources in the language for which it is designed (details are provided below), such systems have only been developed for a small subset of the approximately 7,000 languages that are spoken on earth.

The current review is intended to contribute to the development of speech technology systems in a larger set of languages, with a specific focus on Text-to-Speech (TTS) synthesis and Automatic Speech Recognition (ASR). To this end, we will provide a high-level overview of the approaches employed in current speech technology systems, discuss some of the tools that can be used to develop such systems in new languages (i.e. languages for which speech technology systems do not currently exist), and investigate some of the most pressing concerns in that must be addressed to make such development more widespread.

The difficulty involved in developing speech technology in a new language depends on a number of factors, related to how "atypical" the language is, on the one hand and the extent of resources available, on the other. In practice, a language can be considered "typical" if its phonetics, phonology, morphology, writing system, etc. are fairly similar to one or more of the languages for which extensive speech technology resources are available. As to the availability of resources, Barnard et al. [2010] defines a three-level scale for under-resourced languages, ranging from mildly under-resourced languages such as Swahili or Vietnamese, through languages with significantly constrained resources but a well-established writing system, to languages which do not have a writing system that is widely used (or none at all). For practical reasons, we will limit our attention to languages which are not highly atypical, and have at least a reasonably strong literary tradition (and thus writing system) in the remainder of this report. We therefore assume that text editors, spelling rules and similar basic linguistic resources are available in our target languages.

Our discussion is aimed at two groups of readers. Technical managers, scientists or engineers with a passing interest in speech technology and others without specialist knowledge of speech science will hopefully get a useful overview of the issues that are involved in the development of TTS and ASR systems. For students, developers and researchers with a more detailed knowledge of speech or related signals, we hope to provide guidelines that will assist in the practical development of usable systems based on speech technology. (The sections on ASR and TTS below are intended to be reasonably self-contained, so that readers who are interested in one technology but not the other should be able to read the relevant section in isolation. As a consequence, there is some overlap between the two sections.)

STATE OF THE ART IN TTS SYNTHESIS

Text-to-Speech (TTS) synthesis refers to the technology that converts an input text into a speech signal. The ability of the technology to perform this conversion successfully depends on two crucial tasks. The first one is to extract information from the text and to represent it in an enriched symbolic form. This includes a sequence of phonemes corresponding to the input text. It also includes linguistic information that is conveyed by the prosody of speech, such as variations in melody, rhythm and intensity. All these elements play a role in the correct understanding of the text when it is rendered in a spoken form. The second task links the enriched symbolic representation to an acoustic signal of speech. Its goal is to produce intelligible speech which is perceived as if it had been produced by a human speaker.

Traditionally a TTS synthesis system is divided into two main processing blocks:

- A linguistic processing block, which aims to establish a symbolic sequence enriched with some linguistic and prosodic information from the input text;
- An acoustic processing block, which generates a speech signal from this symbolic representation.

In the following subsections, details are given of the linguistic processing modules and different approaches used to generate a speech signal are presented.

Linguistic processing

Text normalisation

Text normalisation is essentially a text-to-text transformation.

Text is represented as a sequence of encoded characters. TTS systems generally accept text encoded in Unicode, since it is designed to cover many languages. There are many cases in Unicode where two sequences of characters are canonically equivalent: the sequences represent essentially the same text, but with different actual sequences [Davis and Whistler 2010]. The text normalisation usually begins by mapping canonical equivalences to a unique sequence, thus easing the complexity of subsequent processing (lexical lookup).

An input text cannot be simply interpreted as a sequence of words and sentences. The interpretation of punctuation marks, for example, is sometimes problematic. A "." is most often considered to be a full-stop in many scripts, but it may also indicate the end of an abbreviation or it may need to be read out aloud as a word in itself (within a web page address, for example). Abbreviations need to be expressed in full form in order to be read aloud correctly. Similarly, numbers written as a sequence of digits need to be expanded into a sequence of spoken words. The text normalisation determines how an input text is to be read into words.

The way in which a sequence of digits is read aloud in words is highly language-dependent. Many languages may read numbers with the base of ten (English) whereas others may use a base of twenty (French) or a base of five (Wolof); there are also differences in word-order and the addition of conjunctions. The manner of reading numbers often depends on what the number refers to, such as telephone numbers, monetary values and years. Number reading may even depend on what is being countered (Japanese) and on its grammatical context (declination of numbers in Polish, Russian). This means that there is no common approach to processing digits during text normalisation, and that text-to-speech systems resort to rudimentary computer programming to address the problem for each supported language.

Breaking text into a sequence of words is no trivial task either. Many scripts use a white-space to separate words, but some languages (Mandarin) require a word segmentation process to disambiguate competing hypotheses. If we consider a word as being an element of language that conveys a single grammatical construct, then even when white-spaces are present, words may be concatenated ("don't" is verb "do" plus adverb "n't") and a word may contain a space (adjective locution "al dente"). Given that word segmentation requires lexical information, it is not performed during the text normalisation process. Instead, there is a simplified tokenisation process whereby text is split into sequences of characters of similar type (number, spaces, punctuation, and letters).

The input text may be enriched by a markup language to assist a speech synthesis system with text normalisation. The Speech Synthesis Markup Language (SSML) is commonly supported by TTS systems. SSML provides elements that are designed to assist explicitly with number reading (<say-as>) and word segmentation (<token>/<w>).

Determination of pronunciation from text

Determination of pronunciation is essentially a task that converts text to a sequence of phonemes.

Unfortunately for text-to-speech synthesis, a word is often not pronounced as it is written; there is no one-to-one correspondence between letters and sounds. Furthermore, there are words which are written the same be pronounced differently (heteronyms). Many languages are written in a script that does not include information that is pronounced; for example, English is written without lexical stress placement, French may be written without accents (in e-mails), Arabic is typically written without short vowel diacritics, and Hausa may be written without any indication of tone. Such information relevant to pronunciation must be inferred from another source.

A TTS system makes reference to a pronunciation lexicon, which gives all information relevant to a word's pronunciation, including a sequence of phonemes, lexical stress, tones and syllabification. However, the lexicon cannot contain the pronunciation of every possible word in a language (since there are innumerable loan words, declinations of nouns, conjugations of verbs, proper nouns, new technical jargon). The system must therefore be able to determine the pronunciation of out-of-lexicon (OOL) words. If the system's default manner of determining

OOL pronunciations is accurate, then the lexicon reduces in size as it only needs to retain exceptions to the default.

Today's TTS systems use a complex combination of grapheme-to-phoneme rules, morphological analysis and pronunciation lexicons in order to attribute one or more plausible pronunciations to each word in the input text. An explicit tie is made between the grammatical form of a word and its pronunciation. In this way, the grapheme-to-phoneme rules are not only dependent on orthographic context, but also on the grammatical form of the word to which they are applied. The morphological analysis also exploits grammatical information to restrain combinations of affixes and word roots.

The manner in which grapheme-to-phoneme conversion, morphological analysis and lexicons are developed for TTS systems depends on the available linguistic resources. If pronunciation lexicons already exist, then grapheme-to-phoneme conversion may be automatically deduced from them (Damper 2001). Otherwise, conversion rules need to be written on the basis of expert linguistic knowledge and a lexicon of exceptions laboriously composed.

In the cases that a word is attributed many plausible pronunciations, each pronunciation is associated with a distinctive (grammatical and/or semantic) tag. Disambiguation of the tags leads implicitly to disambiguation of the pronunciation. If linguistic resources are available, such as tagged textual corpora, then tag disambiguation may be established though statistical language modelling [Stolcke 2002].

The placement of pauses in the reading of text also plays an important role in its pronunciation and meaning. Punctuation marks are the most convenient source for a TTS system to determine pause placements. The SSML <break> element may also be used in enriched text. A TTS system can however go beyond these basic clues and introduce pauses and prosodic breaks from the syntactic structure of a sentence. This may be performed by rule or through the automatic learning techniques [Schmid et Atterer 2004].

Contextual modification of pronunciations

Contextual modification is essentially a phoneme-to-phoneme transformation.

The pronunciation of a word when spoken in isolation may differ when placed in the context of other words. Tone sandhi in Mandarin depends on word context; for example, the first of two tones of type 3 becomes a tone of type 2. Liaisons in French also depend on word context; for example, consonant sounds are introduced between articles and nouns beginning with vowels. Co-articulation, assimilation and epenthesis may also be implemented as context-dependent transformations of the pronunciation.

The pronunciation modifications may depend on a word's tag and on phonetic contexts (they are independent of orthographic context at this stage). The modifications are usually implemented as rules devised on expert linguistic knowledge.

The linguistic processing of TTS systems is orientated towards providing a sequence of sounds (phonemes and pauses) enriched with information such as tones, syllabification, prominence and the strength of prosodic breaks. This is fed to the acoustic processing block.

Acoustic processing

Parametric approaches

In speech synthesis, parametric approaches aim to determine a set of parameters or features necessary to generate speech. These features include spectral parameters which essentially convey the informative content of the text as well as the prosodic parameters (fundamental frequency, phoneme duration and energy).

Articulatory speech synthesis

Articulatory speech synthesis can be seen as an anthropomorphic approach. Indeed, it aims to reproduce numerically the physical mechanisms involved in the speech production process. For that purpose sophisticated speech production models approximate the propagation of an air flow generated at the glottis through the vocal tract. These models are intended to characterize the air flow (periodical/noise like wave) and to follow the movements of the speech articulators such as the tongue, jaw and lips during the phonation process.

This approach yields low-quality speech and is currently used essentially for research purposes rather than for industrial deployment.

Formant-based speech synthesis

The formants correspond to the various resonances of the vocal tract. Each of these resonances produces a spectral peak with a specific frequency location and bandwidth which therefore characterizes the spectral form of the speech signal, hence the name “formant”. The formant-based speech synthesis approach relies on the fact that by determining the position of the first three or four formants and their evolution over time, it is possible to generate most of the sounds in a given language. The main difficulty of this approach is to define the evolution of these formants for all possible combinations of speech sounds. This operation is tedious since it is carried out manually by an expert who edits a certain number of rules characterizing the formant evolution. This approach, which is also referred to as “rule-based” synthesis, yields intelligible speech, but whose naturalness is poor. Despite its commercial use until the 1980’s (see e.g. the DecTalk system), this technology has been more or less abandoned.

HMM-based speech synthesis

Hidden Markov Models have shown their efficiency in the context of speech recognition, where they are considered as the standard approach. In speech synthesis, they are not considered in a decoding context but rather in a generative role. In such a paradigm, the objective is to learn statistical models given a training database of annotated speech and then to generate trajectories of acoustic parameters based on these models. From these parametric trajectories, the output signal is then synthesized using a dedicated speech decoder. Phonemic HMM models are used and can also be associated to classical clustering techniques such as decision trees to perform more detailed context dependent modelling. Typically, spectral features (e.g. Mel Frequency Cepstral Coefficients, aperiodicity coefficients) as well as the fundamental frequency are considered.

Two methodologies have been proposed for HMM-based synthesis. The first one is based on speaker dependent models estimated on a training database of a specific speaker. A second methodology is to define an average voice model, i.e. a speaker independent model learnt using a database from many speakers and then to rely on adaptation techniques to estimate the models of a specific speaker. The advantage of this speaker independent methodology is that it is able to capture the timbre of a new speaker with few adaptation data. From published results [Yamagishi J. 2009], the speaker independent methodology is recommended when the training database contains less than one hour of speech.

HMM-based synthesis has received considerable attention for more than a decade and currently concerns the majority of academic research work. Its success is due to the robustness and the flexibility of the HMM-based synthesis paradigm. Regarding the robustness issue, the intelligibility of the synthesized speech can be guaranteed. Besides, an HMM-based approach is able to cope with noisy speech data, since even in that case the estimated statistical models will enable capturing global spectral trends in the data. This robustness to noise is important since it allows for the use of recordings with data of limited quality, which is generally not the case in speech synthesis. HMM-based synthesis is also very flexible, since it allows for rapid voice creation through speaker adaptation in a speaker independent context. Moreover, it can take into account rich contextual information including supra-segmental information in order to model, for instance, different types of discourses or speaking styles. Last but not least, the footprint of a HMM-based speech synthesis system is very low (compared to concatenative speech synthesizers), which is a very strong asset for the deployment of this technology in embedded devices with low capacity.

By essence however, the HMM-based synthesis paradigm suffers from a strong limitation. Indeed, the quality of speech produced by an HMM-based synthesizer is intrinsically bounded by the parametric description of the data and the analysis-synthesis algorithm. In current HMM-based synthesizers, the speech representation by means of MFCC and aperiodicity

coefficients does not allow for high quality speech. Such a quality is not enough for a deployment of this technology in IVR systems; where higher-quality proposed by all industrial TTS players are generally preferred.

Concatenative approaches

Another approach for speech synthesis is to build speech from natural speech recordings. More specifically, given a database recorded by a professional speaker, speech segments corresponding to phonological units are extracted and labelled and consequently stored in an acoustic inventory. At synthesis, these units are then selected, adapted and then concatenated to produce the expected speech synthesized message. The key idea of this concatenative approach is to avoid acoustic modelling as in parametric approaches, relying on the acoustic quality of the recorded data itself. In the following subsections, the two main concatenative technologies that have led to massive industrial deployment are presented.

Diphone-based synthesis

In designing a concatenative speech synthesis system, the first question to answer is what kind of speech segments should be considered. Using phones as basic speech units leads to very bad speech quality with very low intelligibility. This is due to the fact that preserving phone transitions is essential for preserving speech intelligibility. This observation led to the choice of diphones as the basic speech units: a diphone is a speech unit spanning from the stable part of a phone up to the stable part of the next phone and thus preserves the transition between two phones.

The first systems based on diphone concatenation were developed in the late 1970's. At that time, due to computational limitations, only a single instance of each diphone was considered. The difficulty is that these diphones must be concatenated without artefacts and so, in such a framework, it is necessary to produce diphones in a mean configuration. This is achieved by recording non-sense phonetic sequences referred to as logatomes.

In this diphone-based approach, the major issue is in conveying an acceptable prosody. This has two strong implications on the output speech quality. First, the prosody must be predicted, which includes the location of pauses, the prediction of numerical fundamental frequency contours and the determination of the durations of each phoneme. As mentioned previously, only coarse prosodic information can be predicted, which yields synthesized sentences with a normative (thus intelligible), but also monotonous and unnatural prosody. Second, the sequence of selected diphones must be modified to satisfy the desired target prosodic patterns. For that purpose, a prosodic modification algorithm such as TD-PSOLA [Moulines E. 1990] can be used. However such signal modifications degrades the output speech quality in such a way that the resulting speech cannot be considered as natural.

To summarize, diphone-based synthesis is a low cost concatenative speech synthesis approach that was deployed by most industrial TTS players.

Unit selection speech synthesis

With the increase in computational resources, it becomes possible to increase the size of the acoustic inventory. Thus, it is possible to store many instances of each speech units recorded in several linguistic and prosodic contexts. By doing so, the paradigm of the concatenative approach changes drastically. The objective becomes to select a sequence of speech units that best match the desired synthesis context. The underlying idea is that if the recorded speech database sufficiently covering a given speaker space can be obtained, then it should be possible to select and concatenate the units that fit the synthesis context, thus avoiding prosodic modification. In this framework, the selection process plays a key role and aims to minimize a cost function consisting of a target cost (measuring the appropriateness of a candidate unit to the target information) and a concatenation cost (measuring the discontinuity between two adjacent units) as suggested in [Hunt A. 1996]. The selected units are then simply concatenated with some smoothing in order to reduce potential mismatches. This way, the speech units are kept almost intact and consequently the naturalness of the voice can be maintained. It must be noted that in a unit selection speech synthesis system, the prosodic prediction module does not play the same role as in the previous diphone-based synthesis approach. Indeed, it is not used to provide numerical prosodic target that will be applied to the speech signal, but rather produce prosodic information (either of symbolic or numeric nature) that will be used subsequently during the unit selection process. Otherwise stated, the prosodic information guides the selection rather than imposes numerical prosodic patterns. Consequently, the output prosody will – to a large extent – reflect the prosodic content of the speech corpus.

Unit selection approaches heavily rely on the recorded corpus, hence their denomination "corpus-based" speech synthesis. This implies that particular care must be dedicated to the design of the text corpus to be recorded. To this end, greedy approaches can be used to optimize the unit coverage as in (François H. 2002). Moreover, the recording itself must be very well-controlled, hence the need for supervision during the recording, so that the speaker follows the awaited phonological sequence with a correct prosody and a stable timbre. The recorded corpus is then labelled and segmented into the required speech units and annotated according to the linguistic and prosodic information needed by the selection module. Most of these processing stages can be automated, but depending on the quality of the recording, some checking and manual correction may be needed.

Current unit selection systems are able to produce high to very high quality synthetic speech using between one and ten hours of recorded speech. The unit selection technology was considered a major breakthrough, since it really increased the adoption of TTS technology in many services. However, it must be noted that despite its tremendous success, unit selection speech synthesis suffers from some drawbacks. First, the quality of the output speech is more variable than with other technologies (diphone or HMM-based speech synthesis). This

problem can be attributed to an insufficient coverage of the acoustic database or to some linguistic or prosodic defects. To circumvent such problems interfaces have been designed in order to post-process the output of a speech synthesis system. In an interface such as Speech Online [2011], a user can correct the placement of pauses, the phonetisation or even can discard the speech units that led to artefacts. Using such an interface, it is then possible to generate very high quality synthetic speech that can be used as an alternative to pre-recorded cues in any voice service. A second drawback of unit selection speech synthesis is that the voice creation process is tedious and costly. This can be a serious shortcoming for applications where personalized voices are needed. Third, the prosody is constrained to the prosodic style contained in the corpus. Current technology is based on an acoustic database recorded according to a "neutral reading style". This style may be sufficient in many applications including IVR, but is certainly not enough for using speech synthesis in applications such as video games or audio books. The next breakthrough in speech synthesis is certainly the control of the expressiveness in speech synthesis.

Acoustic data recording and processing

An important issue in the development of a TTS system is related to the recording and processing of the acoustic data. This concerns the concatenative synthesis, but also the HMM-based approach.

In a concatenative speech synthesis framework, the need for high quality and controlled acoustic data is crucial. For this purpose, high quality microphones and recording devices must be used. Moreover, recording generally requires the presence of a supervisor. Indeed, it is essential that the recorded data corresponds to the awaited data, since corpora are generally designed so as to maximize a certain linguistic – and consequently acoustic – coverage, while minimizing the number of sentences to be recorded. So the role of the supervisor is essentially to check the pronunciation of the recorded utterances. An important point is that by doing so, the processing of the data requires few or even no manual intervention. Specifically, if the recorded data matches the awaited phonetic sequence, then the segmentation of the speech data can be done automatically; e.g. by means of standard toolkits such as HTK.

In HMM-based speech synthesis, the constraints on the data can be somehow relaxed, since the data is not actually stored; it is only used to provide generative models for speech synthesis. Stated another way, sufficient acoustic coverage is needed no matter how it is attained. This enables the use of very large databases not dedicated to speech synthesis. Another asset of HMM-based speech synthesis is that the quality and the homogeneity of the data is not a crucial issue since an HMM-based synthesizer can cope with data recorded in various conditions. Note that the speech data must be labelled in phonemes (and, if necessary, richer linguistic information when more complex context dependent HMMs are considered), which may require a tedious laborious annotation task.

As mentioned previously, HMM-based synthesis with an independent speaker model is a very attractive solution for creating new voices. However, in the case of new languages, and

especially under-resource languages, such an approach is questionable. The main issue is related to the so called average voice model. Ideally, such a model should be build for each new language so as to cover the phonological space as well as the prosody of the language. In the case of under-resourced languages, this approach is too costly and so a compromise has to be found. At the phonological level, approximations can be done in order to relate the necessary phonemes to those of other languages for which acoustic models are available. The same kind of approximation can be considered on the prosodic level, but it is then probable that the underlying prosodic models may not be suited to the language under scope. These approximations may reduce the overall quality of an HMM-based speech synthesis system. Thus, the fundamental question is whether such approximations can lead to acceptable TTS solutions, even in a low-cost development strategy.

Recommendations in TTS for the development of under-resourced languages

The development of a speech synthesiser for under-resourced languages can be considered as a real challenge since it requires the adaptation of linguistic processing as well as the recording and annotation of speech corpora.

Recommendations about which technology to use essentially depend on the applicative context. To a large extent, the linguistic modules do not depend on the acoustic processing modules. Thus, the distinction between existing technologies (HMM-based, diphone-based or unit selection speech synthesis) is rather on the acoustic level and more specifically may depend on the acoustic data that will be needed in a given applicative context.

Regarding the approaches for speech synthesis, it is very difficult to formulate definite recommendations. On one hand, HMM-based synthesis seems an attractive solution due to its flexibility. For instance, an HMM-based speech synthesis solution can be implemented using data not necessarily dedicated to speech synthesis, provided that such data is labelled and yields sufficient acoustic coverage. On the other hand, concatenative approaches can yield more natural synthesised speech. More specifically, the quality of a unit selection system essentially depends on the data and consequently can be optimized depending on the applicative domain.

For limited domain applications, a unit selection system can be developed with a limited amount of speech data (numbers, dates, or any other application dedicated lexicons). This is typically the case for the deployment of services where sentences consisting of slot and fillers are vocalized. In that respect, unit selection synthesis is particularly well suited. Indeed, when designing such a service, the fixed parts of the messages is recorded and can be reproduced as-is during synthesis. Then the variable parts (numbers, dates, …) are synthesised using dedicated data (i.e. a sub-corpora built using numbers, dates). This way very high synthesis quality can be reached provided that the concatenation between fixed and variable parts can be controlled.

For unlimited domain, according to Van Niekerk, Barnard and Schlünz [2009], it seems that HMM-based approach is preferable when very little data is available (e.g. 100 utterances), while unit-selection seems to yield higher quality when a sufficient amount of data (~30 minutes) is available.

STATE OF THE ART IN AUTOMATIC SPEECH RECOGNITION

Automatic Speech Recognition (ASR) refers to a set of technologies that allow digital systems to respond to the contents of human speech. Given the importance that speech plays in human communication, it is not surprising that ASR has a large range of potential applications. These include command-and-control applications (allowing a person to control a digital system with spoken commands), dictation (in which documents are created by converting speech to a textual representation), information access (typically used to allow telephone callers or non-literate users to access information sources by speech), and several others.

Background

As mentioned above, current ASR systems are found in a wide range of applications. To contextualise the class of ASR approaches that are the focus of the current review, it is useful to distinguish between three classes of ASR applications:

- *Embedded applications* are those which run on special-purpose digital systems (e.g. household appliances, telephone handsets, laboratory equipment). Such applications typically require the recognition of a relatively small set of command words / phrases, spoken in isolation, and are specifically characterized by their need to execute within significant constraints of computational resources (memory and computer cycles). Depending on the details of the application, embedded systems may require speaker-independent or speaker-specific recognition.

- *Desktop applications* are executed on a general-purpose computer system – typically a personal computer containing a sound card with attached microphone. These applications often require very large recognition vocabularies, for tasks such as dictation, but typically have more generous computational resources available. Desktop applications can generally also be tuned to the particular user of the personal computer, which improves their performance considerably.

- *Server-based applications* allow a large number of users to use the same ASR system, by executing the recognizer on a compute server. This is the most common arrangement for telephone-based systems, which are mostly speaker-independent systems (since the identity of the caller is generally not known), and require medium-to-large vocabulary dictionaries for typical information-access applications.

Since the current review is mostly focused on applications in the developing world, where server-based architectures are most likely to be useful, the final category above will be our core concern. However, most of the principles and practices that we discuss are also applicable to the other two classes.

The overwhelming majority of server-based ASR systems are built around three language-specific components, as shown in Fig. 1 below.

- The *language model* describes the content that the system is expecting as input – thus, for a system that is designed to provide the market prices for a range of products which are traded on each of several markets, the language model would represent sentences such as "The price of maize in Delhi" or "Milk in Uttar Pradesh", etc.

- The *pronunciation dictionary* specifies how each of the words in the language model should be pronounced, in terms of simpler units. Those units are typically linguistically-motivated entities such as phones, phonemes or syllables.

- The *acoustic model* is a representation of the sounds corresponding to each of these entities – thus, in a syllable-based system, it represents the sounds corresponding to each of the syllables that can occur in the target language.

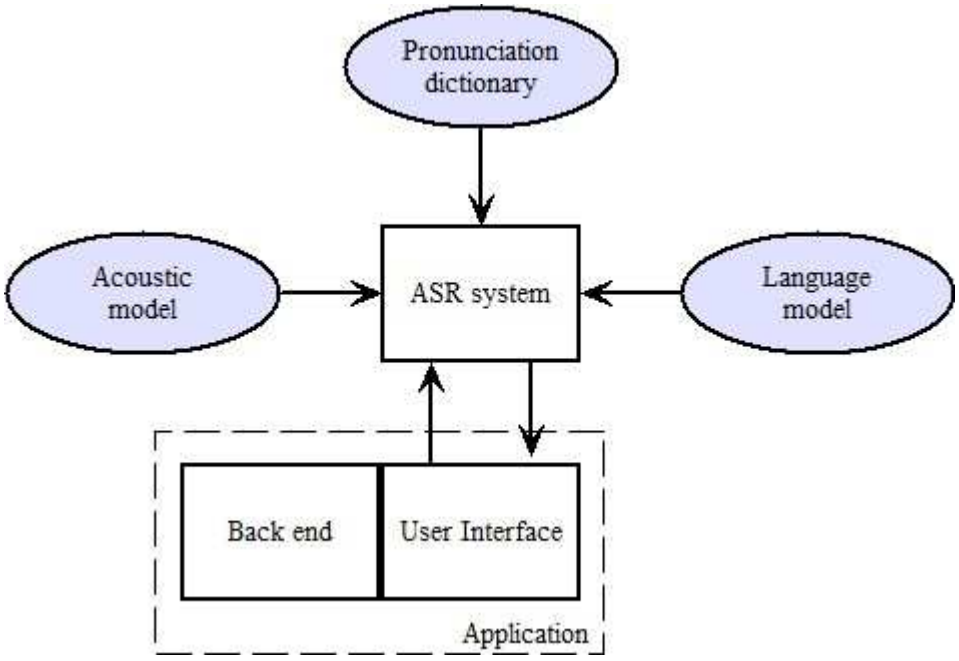

Figure 1: Block diagram of a speech-enabled application; the most important language-specific components are shaded.

The remaining components in Fig. 1 are generally much less language dependent (though possibly not entirely so – for example, the user interface is likely to contain some language-dependent components such as recorded prompts). In the remainder of this section, we therefore summarize the properties of each of these subsystems. We start by providing a somewhat more detailed description of state-of-the art representatives of each component, before shifting our attention to approaches that are particularly relevant in resource-constrained environments.

Core components of an ASR system

Each of the shaded components in Fig. 1 has been – and continues to be – the subject of intensive worldwide research, and numerous approaches have been proposed in each case. Given the continued activity in these topics, no single methodology or approach can plausibly be claimed to be optimal, but in each case a small set of core concepts apply to most of the state-of-the-art systems. Our attention will therefore be limited to these core ideas – for a recent and comprehensive overview, see Gales and Young [2007].

A dichotomy that has played a significant role in ASR research and development (and in language technology more generally) derives from a fundamental divide in pattern recognition, namely that between rule-based and statistical approaches. For the former class of approaches, the system developer specifies the rules that describe each of the entities in the system (in an appropriate formalism), whereas the latter relies on automatic approaches to extract such descriptions from a set of exemplars or training samples. We will see that this dichotomy is relevant to each of the core ASR components.

Language models

Recall that the purpose of a language model is to specify the textual representation of utterances that will be recognized by an ASR system. This corresponds exactly to the capabilities of languages in the Chomsky hierarchy, and the specification languages (grammars) of such formal languages are commonly employed to describe language models. The lowest levels of the hierarchy (regular grammars and context-free grammars) are most widely used, and both rule-based and statistical are employed widely.

For rule-based language models, the designer is required to envision all the likely utterances that a speaker is likely to produce in a particular situation, and then to capture those utterances in a convenient description language. For example, the market-price application mentioned above may contain a prompt "*For which product do you wish to hear the price?*", and the corresponding rule based grammar will contain entries such as "*$product_name*", "*For $product_name*" and "*The price of $product_name*", where *$product_name* specifies a variable that can further be expanded to "*wheat*", "*maize*", "*milk*", etc.

For small, constrained tasks such a grammar is reasonably straightforward to develop and evaluate. Also, for tasks such as the recognition of dates, numbers or amounts, rule-based models can be created with guaranteed coverage of all reasonable utterances. However, as the complexity of the task increases, rule-based models become arduous to develop and error prone in practice.

Statistical language models, in contrast, require a corpus of task-relevant text in the target language. Statistical algorithms then automatically extract a grammar description for that task. Most practical grammars are based on so-called n-gram models – that is, the probability of sequences of n words occurring together, where n ranges from two to six (depending on the application). Various refinements to this basic model have gained acceptance, for example

class-based n-grams, in which similar words are grouped together. These models work extremely well if trained with large enough corpora. Astoundingly large corpora are therefore used for applications such as dictation – corpora containing hundreds of millions, or even billions of words are not uncommon.

Acoustic models

Modern acoustic modelling typically consists of three stages. In the first, known as "feature extraction", the raw acoustic signal is chopped into a sequence of equal-sized frames, and each frame is converted to a representation that is suitable for statistical modelling – the so-called "feature representation". Thereafter, statistical models are used to compute the likelihood that each frame corresponds to each of the basic elements (e.g. phonemes) that are used to describe the speech of the target language. Finally, a search process is used to assemble these frame-based likelihoods into probabilities for larger units such as words, phrases or utterances.

Although numerous workable approaches for each of these stages have been proposed, they are not as diverse as the language-modelling approaches summarized above. In particular, almost all modern approaches are statistical in nature, and most popular feature representations use some variant of a smoothed short-time spectrum to represent speech. The search step is invariably based on some variant of dynamic programming, which assumes that successive frames are conditionally independent of one another.

Pronunciation dictionaries

Since a pronunciation dictionary simply lists the sequence of basic units (e.g. phonemes) that constitute each of the words that a system can recognize, a straightforward approach would simply be to create all pronunciations manually. This was indeed the approach taken in almost all early ASR systems, but it suffers from a number of drawbacks. For a system containing more than a few thousand words, the dictionary-creation process can be quite time consuming, and ensuring consistency across such a large list of manually generated pronunciations is a non-trivial task. Also, the addition of new words to the system then requires the assistance of someone able to create pronunciations that are consistent with those already in the dictionary.

For all these reasons, more recent systems typically contain a component that can automatically predict the pronunciations of words, which is typically known as a letter-to-sound (or grapheme-to-phoneme) converter. In some languages, which have a very regular relationship between the written and spoken forms of a word, the letter-to-sound system can be used by itself, but irregularities (especially of loan words) are sufficiently frequent for this to be uncommon. Most modern systems therefore have a core of manually-produced dictionary entries, combined with a letter-to-sound system that can be used to predict pronunciations for any words not in that dictionary.

Most current letter-to-sound systems are trained from examples (since manually written rules have proven to be insufficiently accurate for various languages). As is true of many ASR components, a diverse set of approaches achieve comparable accuracies in practice, with the Joint-Sequence Model [Bisani and Ney, 2008] currently being the most popular approach.

ASR in under-resourced languages: State of the art

The statistical components described above (n-gram language models, acoustic models, letter-to-sound converters) all require language-specific inputs as training data. However, the algorithms that are used to convert these inputs into a useable ASR system are, to a large extent, language independent. This dichotomy translates into a substantial opportunity for the development of ASR for under-resourced languages, since the massive investment that went into the development of ASR algorithms for languages such as English, French and German need not be repeated for the under-resourced languages. (Of course, we must be careful not to force technology that was created in the developed world into mismatched developing-world circumstances – such "technology drop" has been responsible for many of the failures of IT in the developing world. We return to this matter below.)

The non-statistical ("rule-based") components, namely manually-produced pronunciation dictionaries and rule-based language models are even easier to develop in under-resourced languages. These components are created manually using tools such as text editors or simple graphical user interfaces, and it is generally a straightforward matter to translate these tools into a new language.

The default process for creating an ASR system in a new language, based on the approaches described above, is summarized in Fig. 2. Most of the steps in Fig. 2 are readily accomplished with standard tools such as text editors or scripting languages, but specialized software is very useful for some of the more complex processes. Fortunately, powerful open-source packages exist to address each of these issues, and we next discuss some of the packages that can be used to create state-of-the-art systems.

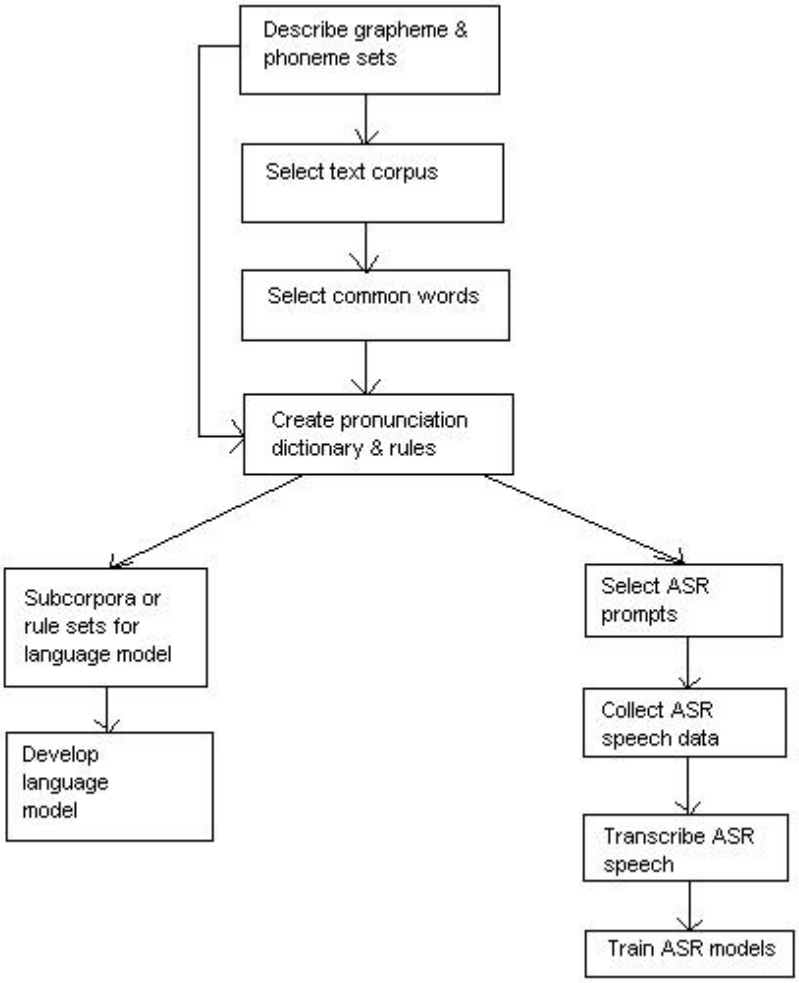

Figure 2: Typical ASR development process

Tools for the creation of statistical language models

The two basic steps used for the creation of statistical language models are (a) selecting or creating a relevant text corpus and (b) calculating the parameters of an appropriate language model, based on that corpus. For well-resourced languages, there are generally several sources (commercial and free) of suitable text corpora, but for under-resourced languages the ASR system developer often has to create a usable corpus from more or less appropriate sources. Fortunately, a bootstrapping approach for the development of such corpora based on Web content has been developed, and has been shown to give good results for languages in which sufficiently diverse content is available on the Web [Schultz, 2008]. For less-resourced languages, such content is often not available. With somewhat more effort, electronic content in these languages can often be gathered from sources such as government agencies, universities and publishers.

Once a suitable text corpus has been obtained, the process of parameter calculation is well supported by a number of toolkits, such as the SRILM toolkit (developed at the SRI Speech Technology and Research Laboratory – see http://www.speech.sri.com/projects/srilm) and HTK (from Cambridge University Engineering Department – see http://htk.eng.cam.ac.uk). Such packages generally contain several highly sophisticated tools, since they were developed to support cutting-edge research. However, using them for more straightforward language modelling is generally fairly easy to do, and a number of tutorials are included with the packages to guide the novice user through the development process.

Tools for the creation of acoustic models

The creation of acoustic models is again (broadly) a two-step process, and – as with language models – the more challenging part is often the conceptually simple step of creating a suitable corpus of speech. An additional challenge here is that orthographic transcriptions of the speech are generally required for the creation of acoustic models (that is, the words corresponding to the recorded speech must be transcribed). For some under-resourced languages, appropriate corpora are available from sources such as the Linguistic Data Consortium (LDC) or commercial suppliers such as Appen. In others, collections of recorded speech are available on Web sites such as news archives or cultural repositories, sometimes with (more or less accurate) transcriptions. However, the majority of languages will require a dedicated data collection effort. Several software open-source packages are available to support both the recording of speech corpora and the transcription of the recordings, for example Praat [Boersma and Weenink, 2001], which is suitable for corpus creation using recordings on a personal computer and Woefzela [De Vries et al., 2011], which is used for data collection using mobile telephones.

The two most prominent toolkits for creating acoustic models once a suitable transcribed corpus has been created are HTK and CMU Sphinx (developed at Carnegie Mellon University – see http://cmusphinx.sourceforge.net/*).* Both toolkits contain a number of components to assist in the modelling process, and tutorials included with the toolkits as well as numerous hints and tutorials on the Internet can be useful to assist new users in developing such models.

Tools for the creation of pronunciation dictionaries

As described above, the pronunciation component of an ASR system generally consists of a manually-created list of pronunciations along with a rule set that can be used for letter-to-sound conversion. As is the case for the resources discussed above, ready-made pronunciation lists are valuable when available, but scarce in most under-resourced languages. When such lists are available, they can also be used as training data for the creation of letter-to-sound rules.

DictionaryMaker [Davel & Barnard, 2008] (http://dictionarymaker.sourceforge.net/) is an open-source package that can be used to create both manual pronunciation dictionaries and letter-to-sound rules. The system does not require the knowledge of an expert linguist nor

any programming skills. It is suitable for languages (such as English, Turkish or Zulu) in which the writing symbols (letters) roughly correspond to phonemes.

Additional requirements

The principles summarized in the Introduction along with the tools described in the previous subsection make it possible to develop an ASR system in a new language with a manageable amount of effort. A more precise analysis of the required effort requires information about the extent of available resources in the target language, the details of the required recognition system (vocabulary size, diversity of user population, etc), but a rough estimate of approximately one to two person years per language holds for many languages. However, such estimates assume that the development is in the hands of highly experienced speech technologists, and for many applications of ASR systems the availability of such skilled personnel becomes the major limiting factor.

A crucial need for the wider availability of ASR systems is therefore that this bottleneck should be alleviated. Fortunately, recent years have seen much progress in this regard: the toolkits that we have mentioned above have each played a substantial role in reducing the complexity of developing the specific components for which they are relevant. However, two related issues are not resolved by these tools:

- None of the toolkits is specifically aimed at the needs of under-resourced languages. As a consequence, their documentation and tutorials often make assumptions about resources and skills that are not suitable for such languages.

- Given the very specific tasks for which they were designed, none of these tools provides a complete overview of the process of ASR development.

During the current project, we will address both of these shortcomings by creating tutorials as well as working examples that span the entire process of ASR development, from the definition of grapheme and phoneme sets to their use in practical applications.

Another type of intervention, which requires a somewhat larger scope than that, provided by the VOICES project, is the creation of resources that allow for the efficient sharing of resources across languages. Without such sharing it is unlikely that a meaningful fraction of all the world's languages will reap the benefits of speech technology. If, however, speech technologists across the globe can create open resources that are specifically aimed at spanning most of the major language families on earth, and tools are developed that make it possible to port ASR resources and systems efficiently between closely related languages, meaningful coverage of the world's languages becomes much more feasible.

ASR in the context of the VOICES project

We have discussed the basic principles behind the operation of modern ASR systems, and provided some guidance on the open-source tools that can be used to develop such systems

in languages with limited resources. These guidelines will be used during the rest of the current project to guide our development of ASR systems for the use cases developed in the other Work Packages of the project.

Given the initial requirements that have emerged out of those Work Packages, one potential concern is that these state-of-the-art methods may be unnecessarily complicated for those specific purposes. In particular, the requirements specify small and fixed recognition vocabularies – for such limited, static systems, whole-word recognizers can be developed with substantially lower effort than the more flexible ASR systems we have discussed [Huang et al., 2001]. A number of factors argue against such an inflexible approach, however. Most importantly, any changes to the vocabulary of a whole-word system require substantial effort – thus, changes to the user interface that may be suggested by user trials will be much harder to implement if they happen later in the development cycle. Also, the subword-based approaches can easily be applied to a variety of tasks besides that for which it is initially developed. Finally, it is hoped that experience with state-of-the-art methods when applied to under-resourced languages will be useful beyond the scope of the current project. For all these reasons, we believe that the methodology suggested in this review will be the right one to follow for the VOICES project.

CONCLUSION

We have summarized the two types of speech technology most relevant to developing-world applications, namely ASR and TTS. Several other speech technologies, such speaker recognition, language identification and topic classification follow similar principles, and the background provided here should be useful preparation for readers who want to delve into the literature relevant to those domains.

Our survey has made it clear that the basic principles for the development of useable TTS and ASR systems are now well understood, and several tools that support such development are now available. However, a fairly detailed understanding is still required to deploy those tools successfully, and we have not yet reached the stage where any skilled computer programmer can develop useable speech-based systems. (This is to be contrasted, for example, with the development of Web sites or databases, where a much wider range of developers are empowered by currently available tools.)

Another major deficiency in the current understanding of speech technology is related to its deployment in the developing world. Whereas speech technology in the first world is already a highly profitable industry, with widespread impact, there are hardly any successful, sustainable products or services based on speech technology in less-developed countries. Thus, a significantly improved understanding of issues related to business models, user interfaces and similar matters is equally important if these technologies are to have widespread impact outside the developed world.

BIBLIOGRAPHY

Barnard E, Schalkwyk J, Van Heerden C and Moreno P J, "Voice search for development," in *Proceedings of the 11th Annual Conference of the International Speech Communication Association (Interspeech)*, Makuhari, Japan, pp. 282-285, 2010

Boersma P and Weenink D, "Praat, a system for doing phonetics by computer." *Glot International* 5(9/10): 341-345. 2001.

Bisani M and Ney H. "Joint-Sequence Models for Grapheme-to-Phoneme Conversion". *Speech Communication*, 50(5):434-451, 2008

Damper, R. I. *Data-Driven Techniques in Speech Synthesis.* Springer, 2001.

Davel M and Barnard E, "Pronunciation prediction with Default&Refine", *Computer Speech and Language*, Vol. 22, pp. 374-393, Oct 2008

Davis, M and Whistler K. "Unicode Collation Algorithm." Unicode Technical Standard #10, 2010.

De Vries N, Badenhorst J, Davel M, Barnard E and De Waal, A. "*Woefzela* — An Open-Source Platform for ASR Data Collection in the Developing World" ," in *Proceedings of the 12th Annual Conference of the International Speech Communication Association (Interspeech)*, Florence, Italy, pp. 3177-3180, 2011

François H, Boëffard O. "The greedy algorithm and its application to the construction of a continuous speech database." *International concerence on Language Resources and Evaluation (LREC).* 2002. 1420-1426.

Gales M and Young S. "The Application of Hidden Markov Models in Speech Recognition." *Foundations and Trends in Signal Processing* 1(3): 195-304, 2007.

Huang X, Acero A and Hon H-W, *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*, Prentice Hall PTR, 2001

Hunt A, Black A W. "Unit selection in a concatenative speech synthesis system using a large speech database." *IEEE ICASSP.* 1996. 373-376.

Moulines E, Charpentier F. "Pich-synchronous waveform processing techniques for Text-to-Speech synthesis using diphones." *Speech Communication* 9 (1990): 453-467.

Schmid H, and Atterer M. "New Statistical Methods for Phrase Break Prediction." *International conference on Computational Linguistics.* 2004.

Schultz T, "Multilingual Speech Processing in the context of Under-resourced Languages" in *SLTU: Workshop on Spoken Language Technologies for Under-Resourced Languages*, Hanoi, Vietnam, 2008

Speech Online. 2011. http://baratinoo.elibel.tm.fr/spo/login?next=/spo/index.html.

Stolcke A. "SRILM - An Extensible Language Modelling Toolkit." *Intl. Conf. Spoken Language Processing.* 2002.

Van Niekerk D R, Barnard E, and Schlünz G. "A perceptual evaluation of corpus-based speech synthesis techniques in under-resourced environments." *Annual Symposium of the Pattern Recognition Association of South Africa (PRASA).* Stellenbosch, South Africa, 2009. 71-75.

Yamagishi J, Nose T, Zen H, Ling Z H, Toda T, Tokuda K, King S and Renals S. "Robust speaker-adaptive HMM-based Text-to-speech synthesis." *IEEE Trans. on Audio, Speech and Language Processing* 17, no. 6 (2009): 1208-1230.