Distributed Web Service Architecture Towards Robotic Speech Communication: A Vietnamese Case Study

Regular Paper

Nhut M. Pham¹, Duong N.H. Dau¹ and Quan H. Vu¹,*

¹ Artificial Intelligent Laboratory (AILAB), Ho Chi Minh City University of Science, Ho Chi Minh, Viet Nam
* Corresponding author E-mail: vhquann@fit.hcmus.edu.vn

Received 10 Jun 2012; Accepted 5 Dec 2012

DOI: 10.5772/55404

© 2013 Pham et al.; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract Robotics is emerging as one of the most prominent research areas in the world and is recently attracting the Vietnamese research community. For the purpose of determining an entrance for a Vietnamese human interacting robotics system, this paper is aimed at proposing a service-oriented architecture for location assistance robotic systems made out of loosely-coupled and distributed web services. The proposed platform consists of two major components: a Speech Processing System (SPS) and a Service Manipulating System (SMS). This architecture allows our robot to meet three critical human interaction robotic systems requirements: flexibility, scalability and reliability.

Keywords Distributed Web Service, SOA, SGMM Speech Recognition, HMM-Based Speech Synthesizer

1. Introduction

In recent years, a couple of technical architectural paradigms have been invented, forming the development of distributed and concurrent systems in robotics. The industry has tried to keep improving the ability of robots to support human beings in their daily life activities. However, pursuing a specific task while dealing with a volatile surrounding environment and ensuring a precise interaction with human beings would require a complicated multifunctional structure in which miscellaneous components of hardware and software have to collaborate in a homogeneous manner [1]. Additionally, there is an increasing need for adding real-time cognitive abilities in autonomous robot designs which can help them to operate in a partially unknown and unpredictable environment [2]. As a result, robots have gradually become highly computing intensive systems that use complex embedded resources and algorithms to handle a large number of computational and processing tasks.

In order to avoid the complexity problem, a distributed architecture has recently been proposed by the robotics research community which adopts the advantages of Internet technology and software engineering. Using a distributed platform, the computing will run asynchronously off-board the robot, saving precious on-board processing capacity [3]. However, traditional distributed applications like Message-Oriented
Architecture flexibility has been a common method for data exchange between distributed systems. The key idea behind SOA is that different companies offer their services in the form of modular and loosely coupled software components exchanging data over HTTP, which can be consumed by other companies through the Internet with no human intervention [4].

Generally, a service is a software component that contains a collection of related software functionalities that are reusable for different purposes [5]. It delivers such operations as data storage, data processing, mathematical and scientific computations, and networking. It is governed by a producer-consumer model in which a service is delivered by a service provider known as the producer, which owns the equipment for hosting, running and maintaining the service, and a client known as the consumer, which connects and uses service functionalities via a remote method invocation mechanism. Predominantly, services are implemented as Web Services (WS), which are defined by the W3C as “a software system designed to support interoperable machine-to-machine interaction over a network” [6].

In terms of WS, despite a vast number of communicating protocols (e.g., BEEP, JSON, Hessian and XLANG), SOAP – the XML-based protocol – has succeeded all of them to become the standard and the most widely used protocol in WS applications. Indeed, the latest version of SOAP (1.2) is a W3C recommendation [22], having the advantages of versatility and being easy to read, implement and integrate into existing web infrastructures. Recent years have witnessed the advent of representational state transfer-based (REST) communications [23]. However, it is still in an embryonic state and not yet fully developed, as well not being proven to be stable or efficient.

In this paper, we propose a service-oriented architecture which makes use of the .NET WS and XML technologies (called SOAP). The robot basically aims to answer a user’s question commands. Our proposed and developed multi-layered architecture generally serves to perform location service discovery using Vietnamese speech recognition and a speech synthesizer in Vietnamese speech-based robotics.

State of the art speech recognition systems are all based on a statistical pattern-matching approach. This requires that considerable acoustic-phonetic knowledge and large amounts of representative training materials – both spoken and written – are available for the training of the recognizer. A good correlation is observed between the performance of the systems and the amount of available resources. For the top languages, hundreds of hours of speech uttered by thousands of speakers have been gathered and transcribed. This is both a costly and time consuming effort. For many languages and their dialectical variants, such large scale acquisition campaigns are not feasible. Such languages are referred to as “under-resourced languages” [25]. Classical techniques would not perform well in these cases, and they also demand tremendous cost and effort even for well-resourced languages. Thus, we come to the rise of Subspace Gaussian Mixture Models (SGMMs) [11].

In essence, the target contribution of this paper is to provide an effortless, easy, fast and rapid development approach for building a robotic speech interaction service, which is not available in other works. That is, it should be effortless in relation to constructing an ASR system with limited training resources, SOAP for easy and fast deployment, and SOA for rapid software development in a distributed environment. Otherwise, it would require a heavy burden in terms of costs of both time and money to produce such systems. Such an approach can be applied for all languages - Vietnamese is just one case where it is compulsory because of the lack of language resources.

The rest of this paper is organized as follows. Section 2 briefly describes our system architecture for robot processing and communication. Section 3 is conducted by introducing a noise reduction scheme, a voice activity detection scheme and an advanced acoustic modelling technique for under-resourced languages. Section 4 presents our state-of-the-art Vietnamese speech Synthesizer based on Hidden Markov Modelling. Section 5 explains the distributed WS integration. Section 6 summarizes the experiment that was performed using the implemented architecture. And finally, Section 7 concludes the discussion.

2. System Architecture

The robot basically aims to answer a user’s question commands in specific circumstances. Therefore, the system architecture consists of two major components: a Speech Processing System (SPS) and a Service Manipulating System (SMS), as follows:

In Fig. 1, we can see that a user is able to interact with the robot via a speech-based channel to pass his command on to the recognition terminal in SPS; then, the recognition terminal sends the command to the SMS throughout the WS. After manipulating the request by keyword spotting and knowledge mining, SMS will return a response message to the synthesizer terminal in SPS, which will subsequently render into speech wave form in order to answer the user’s question. All of the messages from the
robot to SPS and SMS are transferred via the standard Service-Oriented Architecture Protocol (SOAP).

Figure 1. System Architecture

SPS is a multi-function system and its goal is to perceive human speech through speech recognition and response naturally by the speech synthesizer service. It also optimizes the robot’s scalability by providing a mechanism to determine which the best service for speech processing that the robot can be connected with. As we have deployed several speech-synthesizer services that are utilized for a couple of Vietnamese voices, the robot can be programmed so as to easily coordinate with the desired voice.

SMS implements the knowledge processing and decision-making methods, so we called it the “robot brain”. Since the task is quite trivial for natural language processing, the main objective of SMS is to realize what the user is asking by using keyword spotting and then allocate the knowledge from its preset database. Key notes for the knowledge mining are:

- First of all, we create some knowledge for common user location assistance requests and store it in a relational database management system.
- For unpredicted requests, we introduce a simple distant knowledge discovery method which makes use of the Internet as a tool. Like search bots, the robot will compose the answer by looking into thousands of Vietnamese information-carrying websites, as described in [7].

3. Automatic Speech Recognition

3.1 Voice Activity Detection

An important problem in many fields of speech processing is the determination of the presence of speech processing in a given signal. This classification of speech/non-speech channels will significantly improve the quality of ASR performance. However, in order to generate better speech quality, we firstly adopted Winner filtering as standard for spectral noise suppression. We have utilized the Qualcomm-ICSI-OGI front end tool [8], as given by the following equation for the suppression filter:

$$|H(f)|^2 = \max \left( \frac{|X(f)|^2 - \alpha |N(f)|^2}{|X(f)|^2}, \beta \right)^2$$ (1)

where $|X(f)|^2$ is the clean speech power spectrum at time $t$ and $|N(f)|^2$ is the noise power spectrum. Secondly, and for real-time implementation in the case of the robot’s environment, we have used the AMR VAD approach [9]. The block diagram of the VAD algorithm is depicted in Fig. 2.

Figure 2. Block diagram of the ARM1 VAD algorithm

The VAD algorithm uses the parameters of the speech encoder to compute the Boolean VAD flag (VAD_flag). Samples of the Input frame ($s(i)$) are divided into sub-bands and the level of the signal in each band (level[n]) is calculated. The inputs for the pitch detection function are open-loop lags ($T_{op}[n]$), which are calculated by the open-loop pitch analysis of the speech encoder. The pitch decision function computes a flag (pitch), which indicates the presence of pitch. The tone decision function calculates a flag (tone), which indicates the presence of an information tone. Tones are detected based upon the pitch gain of the open-loop pitch analysis. The pitch gain is estimated using autocorrelation values ($t_0$ and $t_1$) received from the pitch analysis. A Complex Signal Detection function calculates a flag (complex_warning), which indicates the presence of a correlated complex signal, such as music. Correlated complex signals are detected based on the analysis of the correlation vector available in the open-loop pitch analysis. The VAD decision function estimates the background noise levels. An intermediate VAD decision is calculated based upon the comparison of the background noise estimate and levels of the input frame (level[n]). Finally, the VAD flag is calculated by adding a hangover to the intermediate VAD decision.

3.2 Acoustic Modelling for Under-resourced Languages

To cope with the problem of limited training data, Subspace Gaussian Mixture Model (SGMM) acoustic modelling techniques [10][11] are used. In contrast to the
usual approaches that deploy a set of universal phones to cover multiple languages, the approach of SGMM uses distinct phone sets but shares a large amount of parameters across languages. In SGMM, HMM-states’ feature distributions are Gaussian Mixture Models (GMMs) with a common structure, constrained to lie in a subspace of the total parameter space. The parameters that define this subspace can be shared across languages/domains. Formally defined, the feature distribution of a HMM-state \( j \) is given by:

\[
p(x | j) = \sum_{i=1}^{I} w_{ji} N(x; \mu_{ji}, \Sigma_i)
\]  

where \( x \) is the feature vector and \( N(x; \mu, \Sigma) \) is the Gaussian function. This might look a little similar to the conventional GMM, however, the difference lies in the way of representing mixtures. An intuitive illustration for both models can be seen in Fig. 3. For SGMM, a particular state \( j \) is associated with a vector \( v_j \) which determines the means and weights as follows:

\[
\begin{align*}
\mu_{ji} &= M_{ij} v_j \\
w_{ij} &= \frac{\exp w_{ij}^Tv_j}{\sum_{l=1}^{L} \exp w_{lj}^Tv_j}
\end{align*}
\]

where \( M_i \) and \( w_i \) are shared across all state distributions. In addition, the covariance matrices \( \Sigma_j \) are globally shared as well. Together, \( M_i, w_i \) and \( \Sigma_j \) form the set of globally shared parameters, as opposed to the state-specific vectors \( v_j \).

![HMM structures](image)

**Figure 3.** HMM structures

To achieve a balance between the amount of shared and state-specific parameters, the notion of a “sub-state” [11] was introduced. Instead of just one state-vector, the feature distribution for a state can be represented by a mixture of \( M \) vectors, each with its own weight \( c \). Fig. 4 gives a clearer picture on this notion. In this case, the feature distribution of a state \( j \) is given by:

\[
p(x | j) = \sum_{m=1}^{M} c_{jm} \sum_{i=1}^{I} w_{jmi} N(x; \mu_{jmi}, \Sigma_i)
\]

Utilizing SGMM, one can deal with the problem of limited training data for under-resourced languages. Indeed, the set of globally shared parameters \( \{ M_i, w_i, \Sigma_j \} \) can be trained on out-of-language data, while the state-specific vectors \( \{ v_j \} \) can be trained on a limited amount of in-language data. In the experiments for this paper, English and Vietnamese are selected as the targets for well-resourced and under-resourced languages respectively (i.e., the English corpus serves as the out-of-language data and the Vietnamese corpus serves as the in-language data).

![SGMM with sub-states](image)

**Figure 4.** SGMM with sub-states

### 3.3 Keyword Spotting

In contrast to voice commands by keywords, our system provides a flexible means for voice communication by natural language interaction. One can ask a question of the robot as they would do so with a person. By saying something such as “I need directions for the cafeteria” or “where is the AI-Lab,” or simply “department of computer science,” the system will find the suitable locations and fetch them.

To achieve this goal, a keyword spotting mechanism is proposed to pick out important terms from a complete sentence. Let \( A \) be the set of keywords for a location and \( B \) stand for the set of grammar terms. The finite state machine (FSM) depicted in Fig. 5 is used to render the keyword spotting functionality. In this sequence, location terms \( (A) \) are always required for queries while grammar terms \( (B) \) are optional and can be disposed of. If the “end” state is not reached, a null query will be assumed.

![Keyword spotting FSM](image)

**Figure 5.** Keyword spotting FSM
4. Speech Synthesizer

The Tokuda framework [12], which provides a compact runtime engine for lightweight deployment, is applied in order to build the response voice, with modifications on language-specific details for Vietnamese. Fig. 6 gives an outline view on the system flow, parts of which will be explained in the following subsections.

![Figure 6. HMM-based speech synthesis](image)

4.1 Acoustic models for HMM training

The modelling of acoustic data is designed in the usual approach, as for Chinese [13] in which each syllable is decomposed into an initial and a final part. While most Vietnamese syllables consist of an Initial and a Final, some of them only have a Final. The initial part always corresponds to a consonant. The final part includes a main sound plus tone and an optional ending sound. This decomposition results in a total number of 44 monophones, as depicted in Fig. 7. Next, context-dependent models (quinphones) are built straightforwardly; an example of this can be seen in Fig. 8.

![Figure 7. Initial-final units for Vietnamese](image)

Finally, the tree-based state tying technique [12] is applied with a set of 52 questions that are designed based on Vietnamese phonetic knowledge, and 682 questions designed based on prosodic information (which will be discussed in Section 4.2). A decision tree is built using a top-down sequential optimum procedure, starting from the root node of the tree. Initially, all data with the same central unit is pooled together at the root node, which is also the only leaf node. Each leaf node is then split according to the designed questions and the process is repeated until the likelihood of an increase is smaller than a predefined threshold.

4.2 Prosodic information

The prosodic factor affects the output speech’s prosody, and thus is important when being designed. Along with quinphones, the approach of this paper makes use of tonal and contextual information on different levels in order to compile the prosodic factor, as follows:

a) Phoneme level:
- Quinphone representation.
- Position in the current syllable (forward, backward).

b) Syllable level:
- Tones of the previous, current and next syllable.
- Number of phonemes in the previous, current and next syllable.
- Position in the current word (forward, backward).
- Position in the current phrase (forward, backward).

c) Word level:
- Part-of-speech of the previous, current and next word.
- Position in the current phrase (forward, backward).
- Position in the current utterance (forward, backward).
- Number of [phonemes, syllables] in the previous, current and next word.
- Exclamation and interrogation flag for the current word.

d) Phrase level:
- Number of [syllables, words] in the previous, current and next phrase.
- Position in the current utterance (forward, backward).

e) Utterance level:
- Number of [syllables, words, phrase] in the utterance.
This prosodic design is packed into 682 questions for the tree-based clustering in Section 4.1, and also for parsing the input text.

4.3 Speech synthesis

Having constructed the HMM set from the training phase, the system is ready for speech synthesis. An arbitrary text that has been given to the TTS system is first put through the text analysis procedure. This step consists of normalizing the text and parsing it into quinphones with prosodic information, called "labels". According to the labels, a sentence HMM is constructed by concatenating the correspondent context-dependent HMMs. Unseen models will be synthesized by clustering in the decision tree. State durations of the sentence HMM are determined so as to maximize the likelihood of the state duration densities. According to the obtained state durations, a sequence of spectral and pitch values is generated from the sentence HMM by using the speech parameter generation algorithm [14]. Finally, the STRAIGHT vocoder [15] is responsible for synthesizing a speech waveform from the generated parameters. The whole idea is depicted in Fig. 9.

![Figure 9. Speech synthesis procedure](image)

5. Distributed Web Service Integration

5.1 SOA-based Web Service

Service Oriented Architecture (SOA) represents the evolution of component-oriented programming, making it possible to encapsulate, reuse and distribute business functionality with the added benefit of interoperability across technology platforms [16]. Adopting a service-oriented approach to system design provides a cleaner separation between business logic and the protocols by which that business logic is reached – making systems more flexible, optimized for the deployment scenario and easier to maintain.

Service-oriented computing defines a paradigm whose goal is to achieve loose coupling among interacting software entities, thus minimizing artificial dependencies [1]. The key concept of this paradigm is the service: a unit of work executed by a service provider to achieve the results desired by a service consumer. The provider and the consumer are both simply roles played by software entities on behalf of their owners. Therefore, service consumers are considered as end users and provided with client tools or other services. The interaction pattern among service providers and consumers is illustrated in Fig. 10. The most important achievement of the SOA-based distributed environments is that shared resources (mainly applications and data) are available on demand as independent services that can be accessed without knowledge of their underlying platform implementation in robotics; we have discovered that a WS-based service marks an ideal solution to all the problems of traditional computing.

![Figure 10. SOA interaction model](image)

Typically, WSs are supported by a set of Web-based technologies [17][18], including:

- Standard data representation in XML, as well as language and protocol definition in XML;
- Standard interface definition in WSDL;
- Standard binding/communication protocol, such as SOAP and MIME;
- Ontology definition languages, such as RDF and OWL;
- Service registry and repository, such as UDDI and ebXML;
- Application composition languages, such as BPEL and WS-CDL.

Fig. 11 presents an example of a WS which is applied in a local communication. From this perspective, the service registry and repository are not necessary. In general, there are only two major parties involving the interaction service workflow that aims to make use of loose-coupling. Firstly, the requester (i.e., the consumer) and the provider create a handshake which helps them to identify each other before entering the main transaction. Secondly, they both sign a XML-based contract so as to physically agree on a semantics and Web Service Description (WSD), which will be the constraint in forming their message definition and manipulation. In the third step, each time one of them needs to transfer a command, they have to input the semantics into a WSDL.
packet and send it to the agent. Finally, the agent will interact with the corresponding party.

**Figure 11.** An example of using the web service

### 5.2 Web Service Implementation

The WS in our system is based on C# and the terminal program is compiled in Windows Communication Foundation (WCF), the Microsoft platform for SOA [16]. It is a rich technology foundation that aims at building distributed service-oriented applications for enterprise and the Web. The fourth version of .NET (3.0), officially launched with Windows Vista in January 2007, introduced WCF along with Windows Workflow Foundation (WF) in order to support service composition. This marked the release of Microsoft's first class Web services platform, simplifying the design, implementation and deployment of services with essential linking for scalability, performance, security, reliable message delivery, transactions, multithreading and asynchronous messaging.

Fig. 12 is a code snippet extracted from the source-code of the SOAP-based WS in the SMS system whose aim is to convert the speech into text.

```csharp
namespace Service
{
  [ServiceContract]
  public interface ISmsService
  {
    [OperationContract]
    public string ASK(string textString);
  }

  public class SmsRecognitionService : ISmsService
  {
    public string ASK(string textString)
    {
      try
      {
        byte[] byteFromText = new byte[textString.Length / 2];
        for (int i = 0; i < textString.Length; i += 2)
        {
          byteFromText[i / 2] = byte.ParseHex(textString.Substring(i, 2));
        }
        string filename = CreateFilename()
        string filePath = GetFilePath(\"Vov\" + fileNames);
        File.SaveFile(data, filePath, filename);
        File.Delete(filePath, true);
        return "true";
      }
      catch (Exception ex)
      {
        return ex.Message;
      }

      private string CreateFilename()
      {
        return "ExampleFile.txt";
      }
    }
  }
}
```

**Figure 12.** An example of the speech recognition method

The service communication process can be described as below:

- **Step 1:** The recognition terminal in SPS invokes a function called ASR2 located in a WCF WS. The request is always in the SOAP protocol and encapsulates the SOAP request document that defines the parameters of the function. The document will be sent as an HTTP packet via the network.

- **Step 2:** The SMS system receives the request message in the SOAP format; it first validates the correctness of the XML structure and is then converted from the SOAP format into the protocol of the destination WS - in this case .NET WCF - using the XML protocol translator.

- **Step 3:** The XML translator passes the parameters to the ASR2 method by converting them in a programming language implemented for the method. The method is executed and the return value is sent to the translator.

- **Step 4:** The translator passes the result to the SMS manager, who packages it as the SOAP response document.

- **Step 5:** The return document is sent to the SPS system as a HTTP packet.

**LocationAssistanceRobotService**

The following operations are supported. For a formal definition, please review the *Service Description*:

- ASR2
- GetText
- NOENormalization
- ASR2ToInteraction
- SPSToASR2

**Figure 13.** Various methods exposed by the SPS and SMS

Fig. 13 delineates the list of functionalities exposed by the SPS and SMS systems and originally implemented in the WSs.

### 6. Experiments

This section focuses on three main experiments: evaluations of the recognizer, synthetic speech quality and a response timing test. All of them are conducted on the dataset described below.

#### 6.1 Datasets and Setups

For experimental purposes, the VOV [13] and the King-ASR-090 speech corpora (shown in Table 1) are compiled to evaluate the recognizer. Both corpora are converted to an identical format of 16 KHz, 16 bits, mono. They are further parameterized into 12-dimensional MFCC, energy, plus their delta and acceleration (39 length front-end parameters).

The corpora are then divided into subsets for training and testing three target ASR systems (i.e., English baseline,
Vietnamese baseline and the Vietnamese SGMM system. Table 2 summarizes the training and test sets used for each system.

|            | Duration | # Speakers |
|------------|----------|------------|
| VOV        | -20 hours| 20         |
| King-ASR-090 | -56 hours| 50         |

Table 1. ASR corpora

| Training | Test |
|----------|------|
| hours    | corpora | hours Corpora |
| English baseline | 53 | 53h Kg (Kg) | 3 | 3h Kg |
| Vietnamese baseline | 17 | 17h VOV (V) | 3 | 3h V |
| Vietnamese SGMM | 73 | 56h Kg + 17h V | 3 | 3h V |

Table 2. ASR training and test sets

Language models (trigrams) for the recognizers are built by interpolating individual models trained from the web text corpus and the training data’s transcriptions.

Another corpus for building the TTS voice is collected from the Vietnamese broadcaster – 8h of speech from one female speaker. The corpus is segmented into 4,376 utterances and transcribed manually. Speech signals are sampled at 16 KHz and stored in 16-bit mono PCM format.

Services are deployed on the server side with hardware configurations of 3.2 GHz x 4 cores CPU and 32 GB RAM. In addition, the Kaldi framework [11] [24] serves as the main tool for training ASR models and decoding waveforms.

6.2 Evaluation Metrics

The performance of an ASR system is typically measured in terms of the word error rate (WER):

\[
\text{WER} = \frac{S + I + D}{N}
\]

(7)

Where N is the total number of words in the test-set and S, I and D are the total number of substitutions, insertions and deletions, respectively. This is the edit distance between the automatically generated transcription and the reference transcription that was manually transcribed. This paper makes use of the word accuracy rate (WAR), which is defined as \(\text{WAR} = 1 - \text{WER}\), to report the performances of the recognizers.

The evaluation metrics for ASR are quite straightforward; however this is not the case for TTS. A challenge in evaluating speech synthesis systems is the lack of universally agreed objective evaluation criteria and data. However, in [19] the ITU-T P.85 standard was verified to be reliable and is thus used as an evaluation scheme for the speech synthesizer. Listeners are presented with synthetic speech and are asked to rate it on five scales: overall impression, comprehension, articulation, pronunciation, and pleasantness of the voice. Each of these scales ranges from 1 (poor) to 5 (good). Table 3 shows the rating scales and their corresponding interpretations.

6.3 Transcription Evaluation

In this experiment, the recognizers are evaluated on the task of speech transcription. Performances are reported for three different systems: English baseline, Vietnamese baseline and a Vietnamese SGMM system. The English and Vietnamese baseline recognizers are based on conventional 3-state left-to-right HMM triphone models, with 18 Gaussians per state. The SGMM system’s shared parameters are estimated using data from both English and Vietnamese, while the state-specific parameters are trained on Vietnamese data only. An SGMM configuration with 400 shared Gaussian components (I = 400), 40-dimensional state-vectors and 12 sub-states per state is used.

|            | # Sub-states | % Parameters | % WAR |
|------------|--------------|--------------|-------|
| English baseline | - | 0 | 2.732K | 88.6 |
| Vietnamese baseline | - | 0 | 4.714K | 84.1 |
| Vietnamese SGMM | 39K | 952K | 1.631K | 90.3 |

Table 4. Transcription performances

Table 4 summarizes the performances of the recognizers. Using SGMM, an absolute improvement of 6.2% WAR over the Vietnamese baseline is achieved. This is obvious due to the disproportionately small amount of training data. The results confirm the benefit of SGMM in taking advantage of resources in other languages whenever only a small amount of training data is available.

6.4 Synthesis Evaluation

In this experiment, a comparative evaluation is carried out between the speech synthesizer and two earlier Vietnamese TTS systems: VnSpeech [20] and VietVoice [21]. A set of 200 sentences is put in the line for synthesis using each system respectively, and then presented to 10 native listeners. The listeners are asked to rate the output speech of each system on ITU-T P.85 rating scales. Average rating scores and their accompanied standard deviations are shown in Table 5. Furthermore, Fig. 14
plots 95% confidence intervals for the ratings of each system. Obviously, the synthesizer outperforms VnSpeech and VietVoice in all respects.

| Rating scale            | VnSpeech | VietVoice | The synthesizer |
|-------------------------|----------|-----------|-----------------|
| Overall impression      | 2.68 ± 0.13 | 3.03 ± 0.17 | 4.34 ± 0.23    |
| Comprehension           | 2.43 ± 0.21 | 3.28 ± 0.12 | 4.82 ± 0.06    |
| Articulation            | 3.12 ± 0.18 | 2.9 ± 0.19  | 4.42 ± 0.17    |
| Pronunciation           | 2.12 ± 0.22 | 3.14 ± 0.08 | 4.16 ± 0.11    |
| Voice pleasantness      | 2.43 ± 0.15 | 2.98 ± 0.23 | 3.95 ± 0.05    |

Table 5. TTS Rating scores

6.5 Running-time Evaluation

As a robot supporting service, the response time of the whole system is crucial. In order to be deployed, response timing must be real-time equivalent or even better. This experiment measures the service’s running time for each communication session, including both ASR and TTS computations. 50 subjects are asked to communicate with the service using random utterances. Processing durations are logged and an average response time of 0.939 seconds can be derived. Fig. 15 plots the timing performances on 50 loops.

6.6 System Trial

This experiment measures the service performances in a concrete field of application – location assistance in the University. The lexicon for ASR consists of 92 keywords (e.g., cafeteria, office, faculty, department, computer science) and 27 grammar terms (e.g., I want, to, direction, please, give). The keywords are necessary for making queries while the grammar terms allow for natural language interactions.

Figure 16. System trial performances

Inquiry sessions are collected from the same 50 subjects as in the last experiment. Each subject was asked to make inquiries to the system within a lexical range, using a wireless close-talk microphone attached to the ear. Trials were performed 10 times for each subject in an open-space environment (i.e., the university corridor). Fig. 16 shows the average performances acquired for each test. Not much difference can be seen between trials. In the best case, WAR reaches 94.35% (93.51% on average). Despite being suppressed, the noises present in the environment affected the outputs. However, the results are quite promising, with the compensation of a small vocabulary size.

6.7 User Survey

Finally, a survey was carried out among test subjects, including 37 men and 13 women. The subjects are asked about the robotic service’s usefulness and user-friendliness. Each aspect is rated from 1 (poor) to 5 (outstanding). Table 6 shows the rating scale used for the survey.

| Score | Interpretation |
|-------|----------------|
| 1     | poor           |
| 2     | mediocre       |
| 3     | moderate       |
| 4     | good           |
| 5     | outstanding    |

Table 6. Rating scale for the survey

Fig. 17 and Fig. 18 plot the proportions of user responses on the survey. 2/3 of the responses agree that the robotic
system is useful (i.e., 66% users rated for “good” and “outstanding”), while the remaining 1/3 of the users did not. Most of the negative feedbacks state that the system is prone to error when exposed in a noise-dense environment. Furthermore, the subjects also complained about the problem of out-of-vocabulary. On the other hand, the ratings on user-friendliness seem to have a uniform distribution. A restriction on spoken words and grammars must have been the reason for negative feedback.

8. Acknowledgements

This work is part of the VNU key project No.B2011-18-05TD, supported by the Vietnam National University at HCMC.

9. References

[1] M. Amoretti, M. Reggiani, Architectural paradigms for robotics applications, Advanced Engineering Informatics 24(1), pp.4-13, 2010.
[2] D. Vernon, G. Metta, G. Sandini, A survey of artificial cognitive systems: implications for the autonomous development of mental capabilities in computational agents, IEEE Transaction on Evolutionary Computation 11(2), pp.151–180, 2007.
[3] R. Hartanto, Joachim Hertzberg, Offering existing AI planners as Web services, GI Workshop Planen und Konfigurieren, 2005.
[4] V. M. Trifa, C. M. Cianci, D. Guinard, Dynamic control of a robotic swarm using a service-oriented architecture, 13th International Symposium on Artificial Life and Robotics (AROB 2008), Beppu, Japan, 2008.
[5] F. Hoque, e-Enterprise: business models, architecture, and components, Cambridge University Press, 2000.
[6] H. Haas, A. Brown, Web Services Glossary, W3C, http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211, retrieved April 2011.
[7] H. Nguyen, T. Tran, N. Le, N. Pham, Q. Vu, iSago: the Vietnamese mobile speech assistant for food court and restaurant location, RIVF-VLSP, 2012.
[8] A. Adami, L. Burget, S. Dupont, H. Garudadri, F. Grezl, H. Hermansky, P. Jain, S. Kajarekar, N. Morgan, S. Sivadas, Qualcomm‐isci‐ogi features for asr, Proceedings of the ICSLP 1, pp.4-7, 2002.
[9] ETSI EN 301 708 V7.1.1, Digital cellular telecommunications system (Phase 2+): Voice Activity Detector (VAD) for Adaptive Multi-Rate (AMR) speech traffic channels; General description (GSM 06.94 version 7.1.1 Release 1998).
[10] L. Burget, P. Schwarz, M. Agarwal, P. Akyazi, K. Feng, A. Ghoshal, O. Glembek, N. Goel, M. Karafiat, D. Povey, A. Rastrow, R. Rose, S. Thomas, Multilingual acoustic modeling for speech recognition based on subspace Gaussian mixture models, Proceedings of ICASSP’10, 2010.
[11] D. Povey, L. Burget, M. Agarwal, P. Akyazi, K. Feng, A. Ghoshal, O. Glembek, N. Goel, M. Karafiat, A. Rastrow, R. Rose, P. Schwarz, S. Thomas, Subspace Gaussian mixture models for speech recognition, Proceedings of ICASSP’10, 2010.
[12] K. Tokuda, H. Zen, A. Black, An HMM-based speech synthesis system applied to English, IEEE Speech Synthesis Workshop, 2002.
[13] H. Nguyen, Q. Vu, Selection of basic units for Vietnamese large vocabulary continuous speech
recognition, Proceedings of the 4th IEEE International Conference on Computing & Communication Technologies, Research, Innovation, and Vision for the Future, pp.320-326, 2006.

[14] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, T. Kitamura, Speech parameter generation algorithms for HMM-based speech synthesis, Proc. ICASSP 2000, pp.1315–1318, 2000.

[15] H. Kawahara, M. Morise, T. Takahashi, R. Nisimura, T. Irino, H. Banno, Tandem-STRAGHT: A temporally stable power spectral representation for periodic signals and applications to interference-free spectrum, F0, and aperiodicity estimation, Proc. ICASSP 2008, Las Vegas, pp.3933-3936, 2008.

[16] M.L. Bustamante, Windows Communication Foundation: Application Deployment Scenarios, MSDN Homepage, retrieved May 2008.

[17] Y. Chen, W.T. Tsai, Introduction to programming languages: Programming in C, C++, Scheme, Prolog, C#, and SOA, second edition, Kendall/Hunt Pub, 2006.

[18] T. Erl, Service-oriented architecture: concepts, technology, and design, Prentice Hall, 2005.

[19] Y. Alvarez, M. Huckvale, The reliability of the ITU-T P.85 standard for the evaluation of text-to-speech systems, Proceedings of the ICSLP’02, Denver, pp. 329-332, 2002.

[20] H.M. Le, VnSpeech, http://www.vnisp.com, retrieved August 2009.

[21] T.H. Le, VietVoice, Virtual Voice Inc., http://noitiengviet.ca, retrieved August 2009.

[22] SOAP Version 1.2 Part 1: Messaging Framework (Second Edition), http://www.w3.org/TR/soap12-part1, retrieved April 2007.

[23] D. Benslimane, S. Dustdar, A. Sheth, Services mashups: the new generation of Web applications, IEEE Internet Computing 12(5), Institute of Electrical and Electronics Engineers, pp.13–15, 2008.

[24] The Kaldi project, http://kaldi.sourceforge.net, retrieved January 2012.

[25] V.B. Le, L. Besacier, Automatic speech recognition for under-resourced languages: application to Vietnamese language, IEEE Transactions on Audio, Speech, and Language Processing 17(8), pp.1471-1482, 2009.