Real-time Voice Adaptation with Abstract Normalization and Sound-indexed Based Search

MIDTLYNG, Mads Alexander / MIDTLYNG, Mads Alexander

(出版者 / Publisher)
法政大学大学院情報科学研究科
(雑誌名 / Journal or Publication Title)
法政大学大学院紀要. 情報科学研究科編 / 法政大学大学院紀要. 情報科学研究科編
(巻 / Volume)
11
(開始ページ / Start Page)
1
(終了ページ / End Page)
6
(発行年 / Year)
2016-03-24
(URL)
https://doi.org/10.15002/00012917

Real-time Voice Adaptation with Abstract Normalization and Sound-indexed Based Search

Mads A. Midtlyng
Graduate School of Computer and Information Sciences
Hosei University
Tokyo, Japan
midtlyng.madsalexander.9c@stu.hosei.ac.jp

***Abstract*—This paper proposes a two-step system to conduct real-time voice adaptation in the field of speech processing. The first step includes recording and pre-processing to form a voice profile. Secondly is real-time input of the voice and adapting the input into a target voice. Concerning the fact that individual voices' structure are habitually varying, this paper suggests a method for converting them into a comparable format. The new method is called abstract normalization which cuts the voice data into smaller sounds. From the sounds are generated an abstracted, simplified version of the data using a level of abstraction along with parameter fitting. The normalized data is used to generate a sound-index which consists of a sequence hash that represents the current object in a simpler fashion. The indices are used to compare different sounds/voices for adaptation. This effectively transforms the speech-related challenges into a search problem rather than a biometric one. To assess the approach, voice profile data are compared against each other as a method to verify the sound-index. Lastly a real-time voice input using alternating levels of abstraction is run against a voice profile created with Norwegian words. The degree of adaptation success is measured in percentage, and experimental results show that while accuracy is not yet excellent, the concept was validated.**

Keywords-Voice adaptation, speech processing, voice profile, parameter fitting, search algorithm

I. INTRODUCTION

This paper introduces a new approach to perform voice adaptation (VA) in real-time. The approach has a different perspective on the problem than past research on the topic, and is intended to be coherent to implement as well as accurately adapt the voice. VA is a part of artificial intelligence (AI) which is the study about transforming one voice into another, while keeping the spoken information unchanged. For example, an input voice utters a specific message into the VA system, then, the output voice will be a distinct targeted voice, however the spoken message remains unchanged. The goal is to always keep the original message, and only affect the sound of the voice. Simply said, such a system could allow a person to use the voice of another individual. Additionally, if the adaptation can be performed without emerging as synthetic or robot-like, its potential for use could expand to any field that employs the use of voice interaction; from implementation with Text-to-Speech (TTS) systems, games, speech AI in smart devices and other entertainment.

Since this study is about the human voice, past research has tried to recreate the human speech system in experiments conducive to VA. However, our suggested approach does not consider the human speech system nor does it treat the problem as a biometric simulation. The core idea of this approach is that a voice is made up by numerous of sounds. These sounds, only when combined make up the individual voice and its characteristics, but separately they are pieces that can be used in a search problem in order to puzzle together the desired result.

Previously [1], we suggested a method called Interpolated Curve Fitting (ICF). This method also considered the voice as pieces of sounds, and the ICF method was to be applied to each sound to be managed. It would perform per-sound operations in order to adapt an input voice into the target voice. The proposed method in this paper is a redesigned process for improving accuracy and efficiency, also using the per-sound concept. In order to categorize and contain the sounds, all information about a voice is stored in a voice profile.

An individual's voice is a very varying piece of information, there are both minor and major differences that makes it difficult to compare sounds from particular sources. To overcome this challenge, we wanted to bring forth a method that could disregard these variations, but keep the general data structure intact as to not impair the data. The result is a two-step approach that includes a pre-processing step and utilizing what is called abstract normalization and sound-indexing. One of the issues we want to avoid that conventional research falls into is the need to simulate the human speech system and feed it with large amounts of training data. The amount of training data is only the initial recording of a voice profile. Our goal is to create an agile VA that can see real use. What we learned from our own past research is that unless we remodel the current voice information into a more abstracted object, the adaptation becomes unreliable in cases where the different voices are high in contrast to each other. In order to achieve this we decided to create a model that can normalize the data with a degree of abstraction which can be decided.

II. RELATED WORK

Looking back at our previous work with ICF, it gave us more insight into how diverse the voice is even with simple spoken sounds. ICF's objective was to take the data for two voices and manage their sounds; the current sound in queue from spoken

Supervisor: Professor Yuji Sato

input, and the other was the current sound from a search in a voice profile. The two sounds' temporal structure is compared, and a new set of data is generated based on the interpolated values from the two. From the new data set, an area of influence is generated, as seen in Fig. 1, and is imagined as a wide line. If a "major" percentage of the two original sounds could be identified within this area, the sounds are to be considered matching and the sound is outputted, performing the voice adaptation.

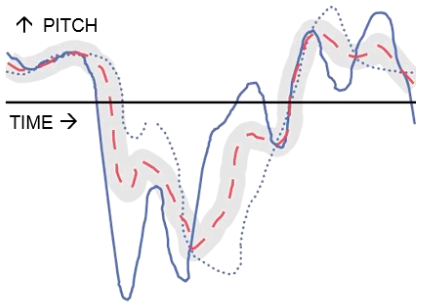

Fig. 1. Interpolated Curve Fitting (red dashed line) from the original sound A (solid line) and sound B(dotted line).

However, the drawbacks of this method is that it cannot take account for sounds that are very polar, for instance in the case of sounds from a male and female subject; their difference in voice structure would make the method too fragile and accuracy falls. This is a problem that is taken account for in the new proposed method, and it could also in theory function for adaptation between numerous of subjects at the same time. Due to the working nature of the abstract normalization, different voices from different speakers are all forced into the same "normalization space" in which they are adjusted and from that the sound-index is generated.

Past research has had a tendency to focus on the human aspect of performing VA, with the intention of creating intricate digital models that function as the different parts of the human vocal system, such as the shape of the mouth, tongue, vocal tract and other physical aspects. [2]'s results were based on two male subjects alone and a large amount of training material. If we desire a dynamic VA system that can be applied to practical use, reducing the amount of training data is important for the sake of usability and robustness. Many concurrent systems focus on the spectral conversion of the adaptation, and often apply basic adjustments such as shift pitching to simulate prosody [3] [4]. The case of simulating the human factor and all its small individual differences handicaps the goal of having a real-time system that requires little training data. Other research [5] employs a training step using interactive evolution which considers the parameters of pitch, power and length. These are then subsequently applied to real-time adaptation in order to perform prosody. Results show that evolutionary computation could get closer to a target compared to a human performing trial-and-error experiences. [6]'s approach considers mapping of the voice spectrum which is stored in a "codebook", and then codebook between speakers are compared in order to conduct the voice conversion. In its learning step, however, they employ two speakers that utter a learning set of words which is put through dynamic time warping in order to produce vectors that could be corresponding between the speakers. They also take into account pitch and power values. In our case, speaker A is learned beforehand and Speaker B is compared against A in real-time. In our case there is no relation between potential speakers A, B, C, and so on. Only the target voice exists in the system at all times, everything else is tested input.

III. Two-step Voice Adaptation in Real-time

Due to the fact that we want to create a real-time VA, minimized processing time is a crucial factor and that is why a two-step approach is used. The first step is the recording and pre-processing of a subject's voice, second is the actual voice adaptation. The result of pre-processing lets us create a voice profile that includes managed data for the subject voice, efficiently lowering processing time for the later VA stage due to the fact that half of the data is already prepared.

A. Pre-processing Step

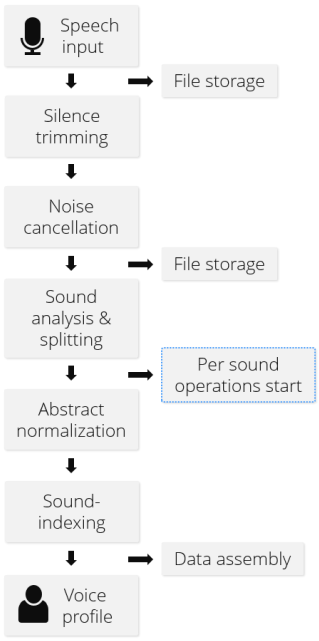

Fig. 2. The pre-processing operation.

Figure 2 demonstrates the course that creates a voice profile from a voice recording. The voice is recorded by microphone with the guidance of our program, where the voice subject reads a manuscript. The recorded file is then put through individual stages of noise cancellation, trimming, analysis, normalization and sound-indexing before all of the useful data is stored in a voice profile. The prominent stages including the abstract normalization and sound-indexing is used to qualify a sound before it could be used in the adaptation stage. Half of the used data is already prepared and could be loaded into the program. The remaining data is the inputted voice which we can't anticipate because it is unrelated to the voice profile, thus has to be handled as-is upon input.

Although a very basic procedure for the voice adaptation to work, it is one of the most crucial ones. The target voice we

want to create a voice profile from has to read a manuscript into a microphone, using the developed software. Compared to traditional approaches, the grammar and semantics are not important, only the combination of words is. Ideally, for the voice to include the highest possible potential of unique sounds, we need to combine words that can create phonemes and utterings to be used in the voice profile. Both a raw original recording and an optimized, cleaned file is stored for later use.

B. *Real-time Voice Adaptation Step*

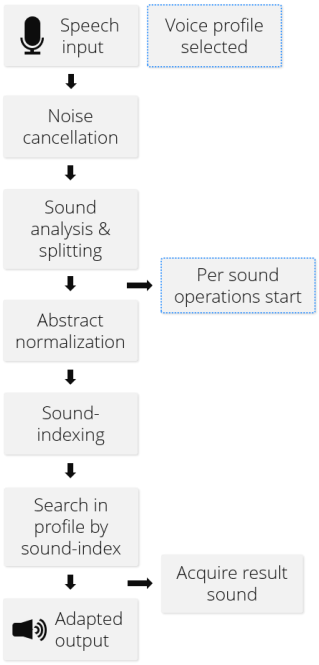

Fig. 3. The voice adaptation operation's management of sound data.

In preparation to perform the actual VA, a voice profile is loaded into memory with the use of our developed software. The system is then ready to receive input from a speaker using a connected microphone. Figure 3 shows the course of action as speech is inputted. The operations are similar to the pre-processing stage. The voice is cut into sounds, and each of the sounds is queued for abstract normalization and then sound-indexing. In the current stage, a sound is made by cutting the voice into 10ms long fragments, incrementing 5ms between each cut, so that the variation between two sounds could also be included into the voice profile. For the human ear, a 10ms sound makes no sense, but if many of them are stitched together to form a longer segment, we'd be able to hear something that makes sense.

Once complete, the sound-index value between the inputted sound and the collection of sounds from the voice profile are compared. The sounds that come close, or even is a complete comparable match are then successfully adapted by the output of the voice profile's sound. As this happens in real time, the outputted sounds form words, sentences, depending on the input speaker's message – taking no accounts for semantics, grammar nor language.

While it in theory can function with several languages at once, it is apparent that if the voice profile does not include the proper pronounced sound, attempting to match the sound becomes futile. Although due to the fact that phonemes exist, different words can invoke the same pronunciations, which is why the design of the manuscript is genuinely important.

IV. Abstract Normalization and Sound-Indexing

A. *Abstract Normalization*

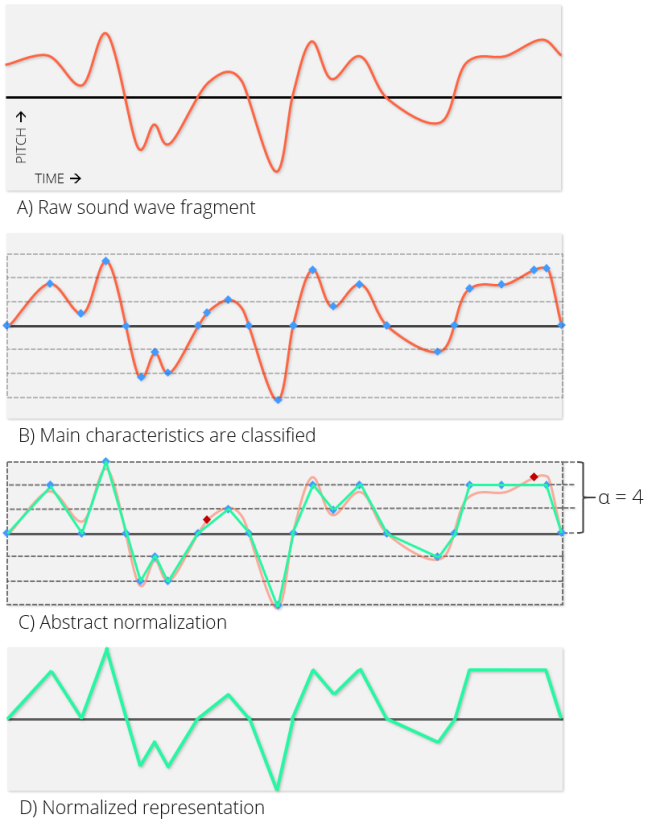

Fig. 4. Abstract Normalization of a sound. (From original sound wave to normalized representation).

To solve the problem discovered with ICF, we suggest an approach called abstract normalization, as seen in Figure 4. This method was inspired from the graphical term 'level of detail' (LoD), where an object's level of detail is decided by how far it is viewed from. We take this idea and create an abstraction level of α that we establish and apply to a current sound in the queue. The level of abstraction decides how complex, or simple the normalized object becomes. Since different voices are always varying, a method such as ICF is too optimistic, hence the introduction of this method that takes more considerations about the vocal differences.

Considering it is a normalization, certain aspects of the current sound is adjusted, so that all data passed through this process can be compared on equal grounds. In the normalization process they are all aligned with the center line. What happens during the normalization is that first, the level of abstraction is declared.

The level dictates the different levels of pitch which is allowed for the sound wave's main characteristics. The main characteristics are also decided, these include the highest and lowest amplitude points, as well as every part of the wave crossing the center line. Once the main characteristics of the sound has been declared, these are then forced into what is known as the normalization space, which is a defined maximum and minimum pitch. Anything over or under is forced to align with the defined levels. To imagine what the level of abstraction does, imagine a level $\alpha = 4$. This means that under and over the center line will be 4 evenly distributed thresholds which points will snap into, based on proximity. For the normalization, the actual pitch values are not relevant, only the main characteristics chosen by the algorithm. Differences between a set of 3 points are evaluated, if the middle point could be found along this line and is too small of a variable; it is discarded. Finally, we are left with a set of points describing the sound wave's original shape. From the set of points now declared, the normalized representation of the original sound is generated. The current set of points are used to create a sound-index which is used for comparing sounds against each other.

B. *Sound-Indexing*

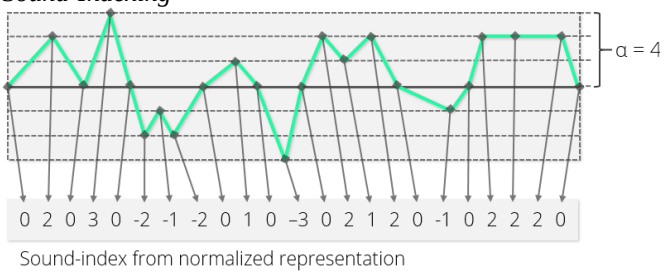

Fig. 5. The sound-index represents a normalized sound.

This method (1) is applied to a sound that has gone through abstract normalization, and the complexity of the indexing, as seen in Fig. 5, is related to the abstraction level applied to the sound. We propose this along with abstract normalization in order to make the search-and-compare aspect as effective as possible. A voice profile consists of a large number of small sounds, in order to make the search instantaneous we propose sound-index as a search format. It should be noted that the complexity of the sound-index is in respect to the level of abstraction. Comparing Fig. 5 with Figure 6 shows that if there is a higher abstraction level, the sound-index contains more information about its structure. This is essentially a threshold function that chooses the closes point between several available. Fundamentally, we can consider the sound-index as a hash where the sequence is based on the abstraction object's composition. The sequence is the current sound's signature, and all the processed sound's signature is stored in the voice profile. This information is used during the voice adaptation stage when the current inputted sound in the queue attempts to find a comparable sound in the voice profile.

We suppose that a sound-index is denoted as $Q(t_1 t_2, \cdots, t_i, t_n)$. Where $i \in 1, 2, \cdots, n$ *and* $t_1 t_2, \cdots, t_i, t_n$ represents the time series. A sound point (main characteristic) is denoted as $p(t_i, \lambda_i)$. In other words, during the time i, the point has a temporal value of λ_i. For each line l that is above the zero marker, it is denoted as l_j where $j \in 1, 2, \cdots, \alpha - 1$. Lines that are below the zero marker are denoted as l_k where $k \in -1, -2, \cdots, \alpha + 1$. The distance between the point $p(t_i, \lambda_i)$ and l_j is defined as θ_j and similarly for point $p(t_i, \lambda_i)$ and l_k is defined as θ_k. If we can establish that line l_j or l_k has the highest proximity to point $p(t_i, \lambda_i)$, then the subscript (j or k) of the line is defined as the index of the point. This is effectively a way to threshold and snap points to their respective closest line. The corresponding equations are shown as below:

$$Q(t_1 t_2, \cdots, t_i, t_n) = q(t_1) q(t_2), \cdots, q(t_i), q(t_{2n}) \quad (1)$$

$$\theta_j = |\lambda_i - l_j| \quad (2)$$

$$\theta_k = |\lambda_i - l_k| \quad (3)$$

$$\min(\theta_j) = \min(|\lambda_i - l_j|) = j \quad (4)$$

$$min(\theta_k) = \min(|\lambda_i - l_k|) = k \quad (5)$$

$$q(t_i) = \begin{cases} \min(\theta_j) & \text{if } \lambda_i > 0 \\ \min(\theta_k) & \text{if } \lambda_k < 0 \end{cases} \quad (6)$$

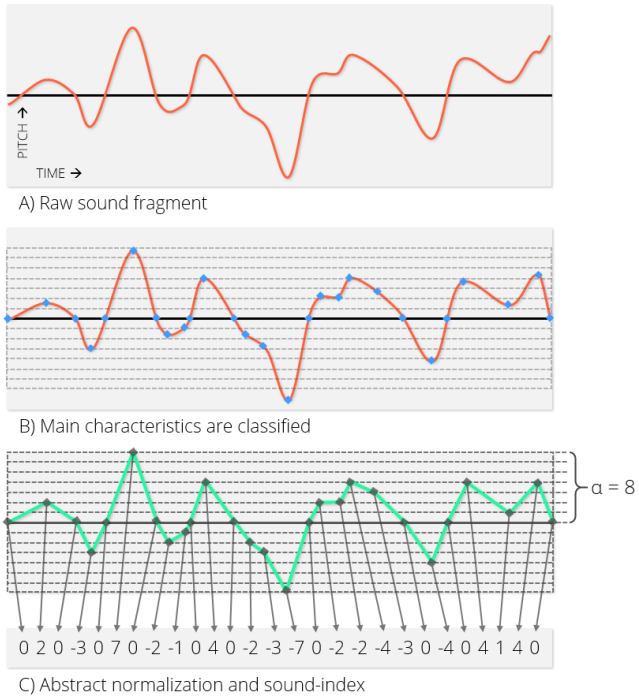

Fig. 6. Sound-index. (From normalized sound with abstraction level of 8).

V. EXPERIMENTS

A. Experimental methods

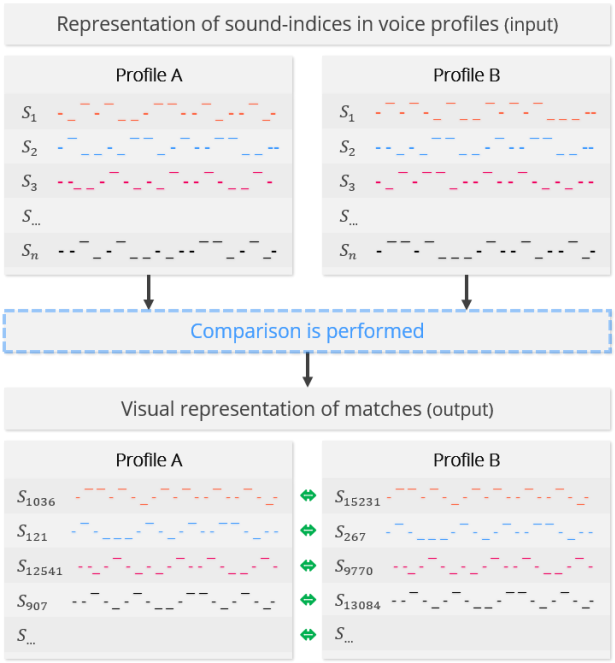

Fig. 7. Data validation representation of how *n* sounds are imagined. (After normalization and sound-indexing in a voice profile).

We define two methods of experimentation that is used in this paper, namely data validation as demonstrated in Figure 7, and speaker VA. Data validation is used to verify to which degree abstract normalization and sound-indexing is working. In other words, it compares the stored data between voice profiles to see if they could match using the proposed method. Speaker VA can be considered a proof of concept. It shows that the randomly inputted voice data could be found in the stored voice profile before outputted again. For this paper, voice profiles were based on a ~2 minute long recording segment and a manuscript in the Norwegian language. Two voice subjects there used, both male. On average, a normal person can speak around 150 words per minute. Due to that fact and that the recording is relatively short, only certain words were attempted adapted. However, with a recording time of approximately 2 minutes, around 12000 sounds were stored in the voice profile (10ms each). The level of abstraction was 10 for the voice profile only.

Current experiments are done with an abstraction level of 10, 20 and 40, to determine which levels of abstraction is the most suitable. Note that at this early stage it is uncertain which value is the best, due to the fact that the quality of the voice profile plays a large a role as the abstract normalization, both must be "just right". If the voice profile data is lacking, the level of abstraction is impartial. On the other hand, if the voice profile data is good, but the level of abstraction is too low or too high, the VA is unreliable. Creating the perfect recipe for a voice profile recording as well as fine-tuning the level of abstraction will continue in future work.

All things considered, this research is about authentic vocal output, thus a very ideal experiment is to perform listening tests with various judges. Finally, if the outputted voice could be deemed as a real person's speech it would be a great advance for this research. Before that point, however, smaller, other tests are performed to test the voice profile quality, the algorithms functionality and proof of concept.

B. Test environment

In order to test, visualize and demonstrate the suggested approach, a multi-thread system is developed using C++ functionality in Microsoft Visual Studio 2013. The test system's specifications are seen in Table 1. Various libraries are incorporated into the solution to take care of different tasks. The major ones include Chromium Embedded Framework [7] which integrates WebKit. In our case it is used for rendering custom made HTML pages alongside with CSS and JavaScript for graphically visualizing the voice adaptation, offering a user friendly recording process and more. Another library is FMOD [8] which is a sound engine with a wide reputation and use in many known applications for its ease of use and power.

TABLE I. TEST ENVIRONMENT

OS	Windows 7 64 bit edition
CPU	Intel Core i5-4690K 3.5 – 3.9 GHz
GPU	AMD Radeon R9 2080X 3GB DDR5
RAM	16GB
Tools	Dynamic USB Microphone (44.1 kHz/48 kHz sample rate) Developed VA software (C++, CPU calculation)

C. Experimental results

As expected, the data comparison experiment shows both matching and not matching data. Refer to Fig. 7 to see how the comparison can be imagined. In the Norwegian voice profile, 11953 "*sounds*" were categorized based on the recording that was made.

For the Speaker VA experiment presented in Figure 8, a speaker A's input was tested against the Norwegian voice profile in a case where the speaker uttered the most used words and similar words that were used in the manuscript. The speaker uttered each word 20 times and the VA system attempted output. In some cases, only partial utterings were outputted by the system, indicating that sounds were missing from the voice profile, or that the stored profile data should have used a different level of abstraction. There were also cases where the wrong sound was outputted. This is most probably because the level of abstraction was too low to distinguish different sounds because details of the sound is ignored the lower the level is. On the contrary if the level is too high, the index becomes too complex for practical search. Furthermore, even if the voice profile and the input data is managed with different levels of abstraction, the sound-index is obviously difficult to compare on a symbol-to-symbol basis. The sound-index actually represents the *shape* of the sound and that is how the system compares it, by controlling the sounds' high and low points. There is still room for improvements, as seen by the results. Also consider that this experiment was done on a word-to-word basis, instead of a fluid continuous speech, which is purposed for future

research and larger voice profiles, ideally between 5 and 6 minutes long, in order to contain as many as possible phonemes and combinations of sounds.

Voice Adaptation results with different α

Words			Success rate with 20 attempts		
#	norwegian	(eng)	α = 10	α = 20	α = 40
1	og	(and)	9.5%	43%	61%
2	i	(i n)	29%	19%	32%
3	det	(the)	59%	51%	48.5%
4	på	(o n)	44%	70%	68%
5	s o m	(a s)	79%	55%	44%
6	e r	(i s)	66.5%	65%	71%
7	en	(one)	35%	41%	15%
8	t i l	(t o)	80%	43%	61%
9	a v	(o f)	73%	62%	59%
10	han	(him)	68%	48%	6%

Fig. 8. Results from the Norwegian voice profile experiment. (Blue shows the best results and red the poor results).

VI. Discussion

While many past experiments regarding VA or "voice conversion" are relatively preceding, most of the results were done in a similar fashion with just minor sounds, and in some cases just utterings of the alphabet. Our proposed system is still not complete. There are many factors to take account for, including design and length of the recording manuscript, length of a sound, sound increment, normalization space definition, level of abstraction value and the strictness of the sound-index search. All things considered, the proposed method produced results and was able to output a sound from the voice profile based on the input of the speaker. The data and test conditions in this case were adequately uncomplicated, so we want to improve the system for more ambitious tests along the way.

Something to be vary of is that in our experiment, many of the words have quite simple pronunciations, however depending on the dialect and speaking habits it can sound different. For example, word #10 in the list will have a very large variance in structure if spoken with normal speed versa slower talking speed. This is probably the cause as to that word had very contrasting results with the different levels of abstraction. The lower the level of abstraction is, the simpler the data we're left with, that is probably why it was easily compared with $\alpha = 10$ versa $\alpha = 40$ which contains more information, thus is harder to match. An obvious fact is that with a very simple word that produces a simple sound, both high and low levels of abstraction will yield similar results due to the fact that the sound doesn't have a detailed structure to begin with.

One aspect that was theorized before testing is the difference between a high and lower level of abstraction. We studied the fact that if the abstraction level is too low, then it is highly likely that one sound can be abstracted into a similar or same object as a totally different sound, due to the fact that it was abstracted too much. On the contrary if the level of abstraction is higher, the sounds retain even miniscule details, which makes the sound more similar to its original state. That is why we want to perform more experiments to find the optimal value, or range of values for the abstraction level, as well as optimize the recording process of a voice profile to include more, but not redundant data. It is possible that some other challenges are uncovered along the way to pursue an excellent VA, nevertheless, a primary vision for this research is to have VA that is more capable and easier to deploy than previous research, which is not restricted by biometric factors.

VII. Conclusion

We created a new way to handle voice adaptation by treating it as a search problem rather than a biometric challenge. Our proposed method splits a recorded voice profile into tiny sound fragments, and processes each sound by applying abstract normalization from which a sound-index is generated. The core idea of treating the voice as a series of categorizable sounds lets us ignore many of the speaker individual-differences that are always varying and unpredictable. The level of abstraction dictates the complexity or simplicity of the outputted data. The sound-index, which describes the sound is then used for search and comparison against another voice/sound. While accuracy is not yet excellent, future work will strive to prove this method with more ambitious experiments.

VIII. Future Work

The goal is to have a fully robust system that can handle real-time speech and output in harmony with the speaker. Components for consideration are how to induce accuracy without having to overcomplicate the user by expanding the recording stage. Planned experiments include longer and bigger voice profiles using more alternating levels of abstraction as well as more challenging speech data in other languages as well. Finally we want to perform cross-gender VA and speaker tests to see if impartial subjects can distinguish between VA output and a real speaker.

References

[1] M. Midtlyng and Y. Sato, "Real-time voice adaption by applying pattern recognition to audio stream," International Symposium on Artificial Life and Robotics (AROB) 20th. Beppu, pp. 502–505, January 2015.

[2] Y. Stylianou, O. Cappé, E. Moulines, "Continuous probabilistic transform for voice conversion," IEEE Transaction on Speech and Audio Processing, vol. 1, pp. 285-288, Seattle, May 1998.

[3] Y. Chen, M. Chu, E. Chang, J. Liu and R. Liu, "Voice Conversion with Smoothed GMM and MAP Adaptation," Eurospeech, Geneva, 2003

[4] A. Kain and M.W. Macon, "Spectral voice conversion for Text-to-Speech synthesis," in Proc. of ICASSP, pp. 285-299, 1998.

[5] Y. Sato, "Voice quality conversion using interactive evolution of prosodic control," Applied Soft Computing Journal, pp. 187-190, Elsevier, 2004.

[6] M. Abe, S. Nakamura, K. Shikano and H. Kuwabara, "Voice conversion through vector quantization," in International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 655-658, New York, 1988.

[7] Chromium Embedded Framework. [Online]. Available: https://bitbucket.org/chromiumembedded/cef. [Accessed: 07- July- 2015].

[8] FMOD. [Online]. Available: http://www.fmod.org/. [Accessed: 15- Sept- 2014]