Implementation and Evaluation of Real-Time Interactive User Interface Design in Self-learning Singing Pitch Training Apps

Kin Wah Edward Lin, Hans Anderson, M.H.M. Hamzeen, Simon Lui
Singapore University of Technology and Design
{edward_lin hans_anderson}@mymail.sutd.edu.sg, {hamzeen_hameem simon_lui}@sutd.edu.sg

ABSTRACT

We present a self-learning singing pitch training tool on the smart-phone to evaluate the efficacy of the real-time interaction mechanism for improving users' intonation and timing, which are the most essential techniques in singing. It consists of (1) an intonation level classifier, (2) a scoring mechanism to help the users know how well they perform, and (3) an interactive pitch training mechanism. We stress the importance of our app's practicality, such that it serves as a guideline for implementing and enhancing similar singing training apps. Experimental results show that the synthesized singing demonstration and the visual feedback design are helpful and natural to comprehend. Our performance evaluation method shows that the score of user intonation improved by an average of 94.81% after training with our tool.

1. INTRODUCTION

Most people learn to sing by imitating professional singers while singing along to their favourite songs. Another popular method is to learn from singing tutorial videos on websites such as YouTube. This learning process depends heavily on the vocalist's ability to evaluate his or her own accuracy. Hence, there is a demand for real-time visual feedback assistance in training of vocal technique, especially the intonation. Much research work has already been conducted [1, 2]. In a review article [3], D. Hoppe et al. suggest that further quantitative investigation of the effectiveness of such visual feedback assistance is needed. They suggest investigating the efficacy of various types of visual feedback, and varying the richness of the information that feedback provides, based on the user's singing skill. Another drawback of these tools is their accessibility. Most of these tools do not have mobile app versions, they fails to take advantage of the now ubiquitous use of smart-phone.

Several intonation training apps have been developed since the Android Market and the iOS App Store were first launched in 2008. On the Android platform, Singing Lessons Voice Training [4] provides various demo videos and tips for singers to improve their vocal performance, but lacks any mechanism to receive input from the user and respond to it in real time. VoiceMatch's Sing Karaoke Voice Tuner Pro [5] records the user's voice to analyse his or her pitch range. Then it suggests songs that are appropriate for their vocal range. This app also does not provide any interaction mechanism. On the iOS platform, Smule's Sing! Karaoke [6] provides an interaction mechanism for users to know whether their pitch meets the target pitch. It has a piano-roll style notation system that scrolls from the right to the left. Based on the analysis of the user's voice, a slider-like indicator on the left moves up and down. Users then learn whether their voice meets the target pitch. They can also get feedback on the accuracy of their timing by checking if the indicator stays at the position of the pitch line and whether the pitch line comes across the indicator. Free Singing coach, songs, voice exercise, developed by sing sharp [7], combines the idea of [5] and [6]. It provides the pitch range testing, and enhances the user-interface (UI) of [6] by adding a piano keyboard on the slider-like indicator. Thus, the users not only know whether they meet the target pitch and follow the rhythm, they also know which pitch they are performing. Erol Singer's Studio - Voice Lesson [8] also provides the pitch range analysis and the same idea of the interaction mechanism mentioned in [7]. However, the performance demonstrations of both [7] and [8] are played with piano sounds, rather than with a singing voice.

In this paper, we present a singing pitch training tool on the smart-phone to study the efficacy of the real-time interaction mechanism for improving user's intonation and timing, which are the most essential techniques in singing. Our main contribution in this paper being that

Design principles and implementation issues of each component of our singing pitch training tool on iOS platform are stated and discussed. This may serve as a model for implementing and enhancing similar apps.

In Section 2, we first discuss the design principles of our singing pitch training tool, in order to define the requirements of the UI. The related technical issues are also discussed. In Section 3 we briefly describe some details of our implementation. The user-experience, the user's performance, and the evaluation methods are stated in Section 4. Finally, several possible directions for the future research are discussed in Section 5.

Copyright: ©2014 Kin Wah Edward Lin, Hans Anderson, M.H.M. Hamzeen, Simon Lui et al. This work is supported by Multi-platform Game Innovation Centre (MAGIC), funded by the Singapore National Research Foundation under its IDM Futures Funding Initiative and administered by the Interactive and Digital Media Programme Office, Media Development Authority.

2. DESIGN PRINCIPLES

We want to design a real-time interactive tool on smart-phones to enhance user's intonation. In order to make our app interactive, the visual cues that indicates the user's current pitch, should be fast enough to respond in real-time. Moreover, users should be able to immediately interpret the clue in a non-ambiguous way. In other words, once the visual clue reacts to the pitch change, it should alert the user quickly enough that the user knows (1) what pitch he or she is currently performing, (2) compare to previous pitch, whether the current pitch is higher or lower, and (3) whether the current pitch meets the target pitch. A typical interactive way in which a user gradually tunes a pitch indicator (which has a pitch range background) up and down to match-up with the melody line at the right time, fulfills this design requirement. P. Hmlinen et al. [1] suggest that the total delay between voice input and visual feedback consists of (1) audio hardware and driver, (2) pitch estimation algorithm, and (3) video hardware and driver. Since the hardware configuration is fixed in the smart-phone, the only component we can adjust the delay is the pitch estimation algorithm. P. Hmlinen et al. point out that more reliable pitch estimation causes more delay, resulting in less responsive UI and thus poor user-experience. Hence, among the pitch estimation methods in the literature, we adopt the element wise product of Fast Fourier Transform (FFT) and Cepstrum in [9]. Its running time is O(NlogN) with N audio samples and it has 91% accuracy. Comparing the running time of this pitch estimation method with the audio samples buffering time, the running time is negligible. Then the only concern is about the audio samples buffering time. However, if the buffer size is too small, the pitch estimation method will then be very sensitive to any tiny short sound, resulting a stability problem of the visual feedback. The relationship between the user-experience of the pitch indicator and the buffering configuration is studied and presented in Section 4.

To make our app accessible to users studying without a teacher, users should be able to obtain our app easily. Smart-phone is the most desirable platform because of its ubiquitous use. In addition, S. Lui [10] reports that electronic device users nowadays are device sensitive; using the best features of each kind of device to match different suitable tasks. For instance, 90% of users send email with desktop PCs, which have physical keyboards and relatively large displays, compared to smart-phone. 73% of users use navigation activities on smart-phones because of their mobility. Since listening, singing and looking at the visual feedback are the only actions of this self-learning process, a smart-phone is an appropriate device for learning intonation and tempo. However, not all smart-phones are suitable for our app. As mentioned before, audio hardware and driver also contribute to the delay between voice input and visual feedback. Hence, at this beginning stage of development, we should first implement our app on a smart-phone with low-latency audio. iOS devices currently have the lowest audio latency (around 5.8ms) [11]. For this reason, we decided to develop our app on the iOS platform.

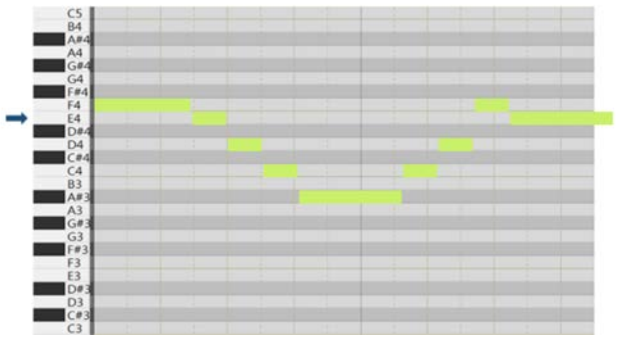

Figure 1. Intonation Level Classifier with Song - Fly Me to the Moon.

Next, a singing demonstration is required for unsupervised study. Having a professional singing teacher to provide a suitable demonstration for our app would be expensive and time-consuming. As the commercial singing synthesizer, Vocaloid [12], is flexible and accurate, we used it to create the singing demonstrations for our app. Listening to examples from a virtual teacher is not enough for unsupervised self-study. Students also need to know how well their intonation currently is, by comparing against their own previous performance and also against their peers. Therefore, we need a scoring system. By using the scoring mechanism, users should be able to easily, unambiguously, and fairly do the performance comparison. In this early stage of development, we would like to make the scoring mechanism as simple as possible so that it minimizes the delay between voice input and visual feedback, and it also fulfils the requirement of performance comparison. The details of such a requirement are further discussed when we present our scoring mechanism in Section 3.2.

One last issue about designing our app is that since we want to minimize the delay as much as possible, we avoid expensive harddisk I/O operations, especially during pitch training. In other words, the singing voice is not recorded. In our app development, we found that the harddisk I/O operations significantly affect the audio sample buffering, which subsequently affects the accuracy of pitch estimation.

3. IMPLEMENTATION

With the design principles in mind, we implemented our singing pitch training tool with iOS's graphic rendering and animation infrastructure called Sprite Kit [13]. We deployed it on iPod Touch (5th generation) with iOS 7.0.4 and Apple 1GHz dual core A5 CPU. It consists of (1) an intonation level classifier, (2) a scoring mechanism to let the users know how well they perform, and (3) an interactive pitch training mechanism. Each lesson begins with a synthesized singing demonstration, follows by a tuning note. Then the user sings to tune a pitch indicator, to match with the melody of the song. In this section, we briefly describe the compact UI in the 4 inch iPod Touch Retina Display and the corresponding implementation issues of each component.

3.1 Intonation Level Classifier

The UI of the intonation level classifier is shown in Figure 1. There is a score panel on the right. A keyboard label with English naming convention of a 12-tone chromatic scale is located at the left. The pitch range is from C3 to C5. The first two sentences of some popular songs such as Fly Me to the Moon are synthesized by Vocaloid. When the user starts using the classifier, the notes in song, represented by green bar, move from the right to the left. When each note hits the keyboard, the corresponding synthesized singing voice is played. In this way, a singing demonstration is played. After the demonstration is finished, it is the user's turn to sing. The same sequence of notes move from the right to the left again. Before the first note hits the keyboard, the synthesized singing voice of the first note is sung and this serves as the tuning note. When user is singing along with this sequence of notes, their voice is continuously captured and then the pitch is estimated, regardless of whether the voice matches with the correct words. In addition, since our focus is the visual clue for helping the user to get the right pitch, lyrics is therefore omitted. Based on the pitch estimation, the blue arrow moves to the corresponding position of the keyboard. When the blue arrow hit the green bar, the green bar shimmers. In this fashion, user gets familiar with our app interactive interface and the visual clue response. Finally, the score of user performance is displayed. The scoring mechanism is described in the sequel section. We briefly classify the users into three categories, namely Expert, Average and Beginner.

3.2 Scoring Mechanism

The scoring mechanism works by scoring the user's performance on a pass / fail basis at regular time intervals. For each time interval, a score of one indicates that the user's pitch is within the acceptable range for that time interval; zero indicates incorrect pitch. We implemented the UI of the app using Apple's Sprite Kit graphics API, which calls a function to update the entire screen once, around every 0.0167 seconds. To calculate the score, we use two variables: `maxScore` and `yourScore`. Each time Sprite Kit calls its graphics update function, we increment `maxScore` and if the user's current pitch is within acceptable range, we also increment `yourScore`. At the end of the lesson, the following formula gives the user's performance score:

$$\frac{\texttt{yourScore}}{\texttt{maxScore}} \cdot 100\%$$

This scoring mechanism is simple. It fulfils the delay minimization requirement and the score provides an easy, unambiguous, fair comparison between different users. However, this schema is too strict about the intonation. Users have to maintain their pitch for the whole period of each note, even when they need to breathe or would like to glissando style transition between notes. Otherwise, they are penalized. Since this schema provides insight for future work, we adopt it in this early stage of development. We discuss plans for future improvement in Section 5.

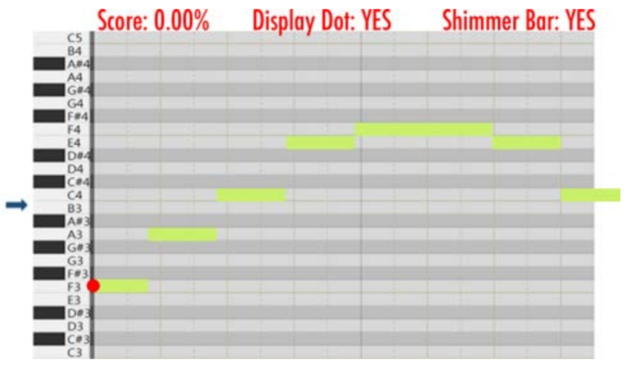
Score: 0.00% Display Dot: YES Shimmer Bar: YES

Figure 2. Pitch Miss in Training.

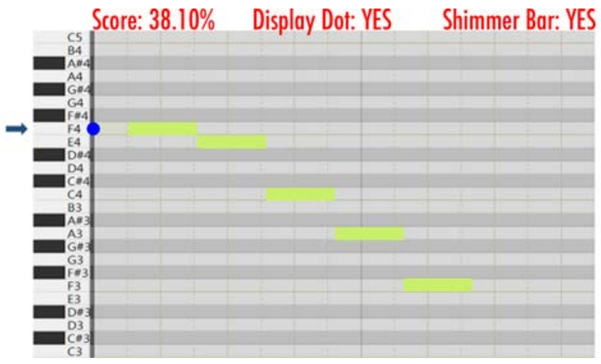
Score: 38.10% Display Dot: YES Shimmer Bar: YES

Figure 3. Pitch Hit in Training.

3.3 Pitch Training

Our app includes 4 typical pitching exercises for intonation practice, namely Major Scales, Minor Scales, Major Arpeggios and 7ths Arpeggios. Table 1 gives an example of the forward notes sequences in these exercises in F. In these exercises, notes are played forward and backward.

Major Scales	F	G	A	B♭	C	D	E	F
Minor Scales	F	G	A♭	B♭	C	D♭	E♭	F
Major Arpeggios	F	A	C	F				
7th Arpeggios	F	A	C	E	F			

Table 1. Four Pitch Training Exercises in the Key of F.

Figure 2 and Figure 3 depict the UI features of pitch training. It is similar to the intonation level classifier. The differences are that users can configure the tempo of the song (the speed of the moving notes), and three visual feedback cues are added. Basically, these visual cues can be considered as the reinforcement features [14]. Reinforcement learning is a technique from the field of behaviour psychology, in which users alter their decision-making in order to maximize some notion of cumulative reward. Typically, the system only knows the correct desired behavior, but lack the exact steps or procedure to guide the user towards success. Our tool uses the reinforcement features described below to train the user to maximize their reward, the performance score. Reinforcement features used in our tool are (1) the performance score indicator, (2) pitch miss red dot and pitch hit blue dot and (3) the shimmering green note bar. The performance score indicator display the current performance score. When the user misses the right pitch, a red dot is displayed. If the user gets the right pitch, then a blue dot is displayed and the green note bar shimmers. These reinforcement features are enabled or disabled and we discuss the effectiveness of the combination of reinforcement features in the following section.

4. EXPERIMENT

We performed 4 tests to gather users' feedback on the UI design of our app and evaluated users' intonation skill. Ten male users and ten female users with age ranging from 19 to 32 were invited to perform these user-experience tests. Each user was asked to use the app in a separate and quiet room. They were asked to use the app with mouth-to-microphone distance of 15-18cm and listened to the singing demonstrations with earphones. There is no input from the speaker to the microphone. After each test, the users rated the visual feedback cues with 7 being the best and 1 being the worst (See Table 2).

Value	Interpretation
7	Entirely Natural and Helpful
6	Very Natural and Helpful
5	Quite Natural and Helpful
4	Somewhat Natural and Helpful Somewhat Unnatural and Unhelpful
3	Very Unnatural and Unhelpful
2	Quite Unnatural and Unhelpful
1	Entirely Unnatural and Unhelpful

Table 2. Possible Visual Feedback Test Responses.

The 1st test aims to understand the relationship between the users experience with the pitch indicator and the buffer configuration of the pitch estimation algorithm. For each buffer configuration, each user sang several notes, and at the same time observed the blue arrow (the pitch indicator). Users kept on doing this until they formed a clean opinion about the efficacy of the blue arrow. The result of the test is shown in Table 3. Each row shows that, given the buffer size and the percentage of frame overlap, the period of buffering sufficient audio samples is measured, the running time of estimating pitch is recorded and the average of users' rating is stated. It is found that the buffer size of 8192 audio sample points and 50% of frame overlap gives the most pleasant visual feedback for the pitch indicator. The delay between voice input and visual feedback is low and the buffer size is large enough to provide a stable estimation.

In the 2nd test, we use the intonation level classifier to classify the users into 3 different groups, namely Expert, Average and Beginner, on a curve. Top 10% of them were classified as Experts, the next 30% is classified as Average, and the remaining 60% were classified as Beginners. Since the result was calculated after every user had finished their tests, they did not know in which group they were classified. But they were told in advance that they will be graded

Buffer Size	Overlap %	Period ms	Estimate Pitch ms	Ave Rating
4096	0	185.76	0.670	4.69
4096	50	92.88	0.673	5.00
8192	0	371.52	0.901	5.13
8192	50	185.76	0.514	5.18
16384	0	743.04	1.340	4.60
16384	50	371.52	1.343	4.80

Table 3. User-Experiences on Pitch Indicator and Buffer Configuration.

Exercise	Overall Ave	
	Rating	Score
Major Scales	5.05	10.82
Minor Scales	4.35	7.86
Arpeggios	5.00	5.22
Arpeggios 7th	4.80	9.68

Table 4. User-Experience and Average Score of Four Pitch Tuning Exercises.

Reinforcement Features	Ave Rating	Expert	Average	Beginner
Score	4.20	4.50	4.17	4.17
Red/Blue Dot	5.00	4.00	5.50	4.92
Green Bar	4.05	3.00	4.17	4.17
Score, Red/Blue Dot	5.45	5.50	6.00	5.17
Score, Green Dot	4.85	5.50	5.00	4.67
Red/Blue Dot, Green Bar	4.90	5.50	5.00	4.75
Score, Red/Blue Dot, Green Bar	5.75	6.50	6.17	5.42

Table 5. User-Experience on Reinforcement Features.

Skill Level	Before Training Score		
	Min	Ave	Max
Expert	16.5	18.16	19.82
Average	11	13.82	16.13
Beginner	0.64	6.50	10.76
Skill Level	**After Training Score**		
	Min	**Ave**	**Max**
Expert	32.27	33.42	34.57
Average	20.51	24.81	30.27
Beginner	1.77	14.03	19.01

Table 6. Group Performance Before and After Training.

on a curve with the scheme mentioned above. They were also told in advance that after they finished practising the

Overall	Ave Score	Score Std Dev
Before Training	9.86	5.10
After Training	19.21	8.33

Table 7. Overall Performance Before and After Training.

3rd test, they would be graded again in the fourth test. This motivated them to improve their score. The result of this 2nd test serves as the benchmark *before* pitch training. And it will be discussed along with the 4th test, which serves as the benchmark *after* pitch training.

In the 3rd test, users were first asked to practice and rate each pitch training exercise. The result is shown in Table 4. The results show that users do not have a strong preference for any particular pitch training exercise. Next they were free to choose one of the pitch training exercises to practice and they were also free to adjust the tempo. But once the exercise was chosen, it was fixed and they had to test all combinations of the reinforcement features, with their exercise choice. The set of Reinforcement features was randomly chosen for each user, so that it mitigated the bias of favouring the later set due to the learning effect. Table 5 shows the result and the classification is based on the result of the 4th test. Users generally preferred the set having the pitch miss red dot and pitch hit blue dot (Red/Blue Dot). This feature worthy of further investigation when designing reinforcement features for breathing and vibrato.

In the 4th test, users were asked to use the intonation level classifier once again. But this time, the test song was different from the 1st test. Table 6 and Table 7 show their performance before and after pitch training. The results show that our scoring mechanism is able to quantify users' intonation skill. Based on our grading mechanism, the score of user's intonation improved by 94.81% on average.

5. FUTURE WORK AND DISCUSSION

This study provides an insight for us to enhance the current app. Although our scoring mechanism is so strict that requires users to sing in a mechanically precise way, avoiding all expressive use of pitch such as vibrato or glissando, it is able to distinguish people who have singing experience from those who have none. From our experimental data, participants with choir experience, have an average score of 11.58% and 26.80% in the 1st and 4th test respectively. On the other hand, participants with no choir experience, have an average score of 8.93% and 15.12% in the 1st and 4th tests respectively. The experienced participants performed really well and deserve to get full marks, but they can only achieve around 20% because of the strictness of our scoring mechanism. This inspires us to study further what exact components of voice data contribute to the intonation. It helps us to polish our scoring mechanism, so that it allows users to sing more expressively. This study inspires us to study other singing techniques, such as vibrato, in a similar way. We will revise our interface in terms of aesthetics and provide some historic representations of pitch, and be prepared to publish apps in all other app stores when the new audio frameworks in Android or Windows Phone become more efficient. We will also study the user experience between Vocaloid versus a real singer.

6. REFERENCES

[1] P. Hamalainen, T. Maki-Patola, V. Pulkki, and M. Airas, "Musical computer games played by singing," in *Proc. Int. Conf. on Digital Audio Effects*, 2004.

[2] O. Mayor, J. Bonada, and A. Loscos, "Performance analysis and scoring of the singing voice," in *ASE 35th International Conference*, 2009.

[3] D. Hoppe, M. Sadakata, and P. Desain, "Development of real-time visual feedback assistance in singing training: a review," *J. of Computer Assisted Learning*, vol. 22, pp. 308–316, 2005.

[4] Andys Apps, "Singing lessons voice training," https://play.google.com/store/apps/details?id=com.a192988968150b0f49b62d262a.a64419678a.

[5] VoiceMatch, "Sing karaoke voice tuner pro," https://play.google.com/store/apps/details?id=com.audio.pro.

[6] Smule, "Sing! karaoke", https://itunes.apple.com/us/app/sing!-karaoke-by-smule/id509993510?mt=8.

[7] Karaoke Music App Lab, "Free singing coach, song, voice exercises," https://itunes.apple.com/us/app/free-singing-coach-songs-voice/id772052329?mt=8.

[8] Erol Studios, "Erol singer's studio - voice lessons," https://itunes.apple.com/us/app/erol-singers-studio-voice/id502780186?mt=8.

[9] G. Peeters, "Music pitch representation by periodicity measures based on combined temporal and spectral representations," in *Proc. Int. Conf. on Acoustic, Speech and Signal Processing*, 2006.

[10] S. Lui, "A compact spectrum-assisted human beatboxing reinforcement learning tool on smartphone," in *Proc. Int. Conf. on New Interface for Musical Expression*, 2013.

[11] TouchMusic, http://www.musiquetactile.fr/android-is-far-behind-ios/.

[12] H. Kenmochi and H. Ohshita, "Vocaloid commercial singing synthesizer based on sample concatenation," in *Proc. Annual. Conf. on Internation Speech Communication Association (INTERSPEECH)*, 2007.

[13] Apple Inc, "Sprite kit, ios graphic rendering and animation infrastructure," https://developer.apple.com/library/ios/documentation/GraphicsAnimation/Conceptual/SpriteKit_PG/Introduction/Introduction.html.

[14] Sutton, R. S. and Barto, A. G., *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press, 1998.