RIFF: A Real-Time Singing Voice Synthesizer

Nathan Stephenson, Gary Home, Hanad Elmi, Connor Bartosiewicz, Evan Coulter, Berny Guzman
Department of Electrical and Computer Engineering
George Mason University, Fairfax VA

Abstract:

A singing voice synthesizer (SVS) is an instrument that digitally generates a singing voice, which has been well-commercialized through products like VOCALOID. Despite its success, SVS is predominantly used for prerecorded music rather than as a real-time instrument. Traditional SVS systems often rely on a preset sample of lyrics and pitches as inputs to produce a vocal output.

We propose and develop RIFF, a complete hardware synthesizer and performance solution to play a synthesized singing voice, with real-time pitch and syllable inputs. This project aims to utilize embedded systems (SoC like the Raspberry Pi) and analog circuitry for the hardware synthesizer. Future research must focus on improving the expressive capabilities of real-time SVS, especially in terms of pitch modulation, emotion recognition, and timbre adjustment, to fully realize its potential as a musical instrument.

***Index terms*—VOCALOID; Singing Voice Synthesizer; Sampling; Real-time generation; Pitch modulation; Syllable input; Hardware synthesizer; Emotion recognition; Voice banks**

I. Introduction

Singing voice synthesis (SVS) utilizes a computer to directly generate singing vocals, enabling musicians to incorporate vocals into their compositions without the need for a human singer. Using SVS bypasses the challenges of hiring and recording a singer, as well as the logistical complexities of capturing high-quality vocal performances. SVS systems take lyrics and pitch as inputs to algorithms that produce realistic or stylized singing voices, which can be further manipulated for various musical genres and applications.

Current SVS systems are not synthesized in real time as music is generally prepared before a vocalist is tasked to sing it. SVS systems often require the user to preload input data for a full song, such as lyrics, pitch, and timing. Many modern software solutions such as VOCALOID [1], DiffSinger, [2], and Synthesizer V [3] have achieved SVS capabilities that produce highly realistic or stylized vocals. These platforms rely on pre-recorded voicebanks, composed of a human singer's voice, segmented into phonemes and other vocal elements [1]-[4]. The SVS system processes the input data through these voicebanks to match the required pitch and lyrics, shaping the singer's voice into the desired melody and performance style [1]-[4].

This paper presents **RIFF**, *a real-time singing vocal synthesizer (RT-SVS) Instrument For Fun* designed for intuitive input, fast processing, and high-quality output that syncs with a song's melody and lyrics. The synthesizer should match the *emotional expression* and dynamics of the performance, allowing the generated vocals to convey the intended mood and feeling of the music in real time. A conceptual flowchart depiction of a real-time SVS instrument is shown in Fig 1, illustrating how the system processes various inputs such as *pitch* and *lyrics*.

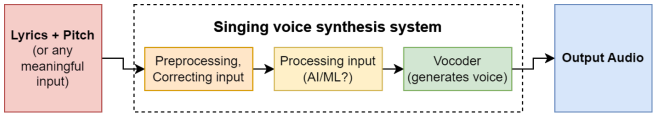

Fig 1. A high-level approach to singing voice synthesis

In order to design an effective *real-time vocal synthesizer*, we address issues that are crucial in dynamic on demand performance systems; we replicate the natural nuances of live singing, such as emotion, pitch variation, and vocal dynamics, resulting in a sound that can feel robotic and unnatural. In addition, we eliminate latency as much as possible to ensure the system works in live performances.

The rest of the paper will be formatted as follows: a background of SVS technology including real-time implementations in Section II. Section III proposes the system model, and Section IV discusses our proposed hardware and firmware implementation. A conclusion and remarks is provided in Section V.

II. Background

One of the most popular SVS solutions is VOCALOID, developed by Yamaha Corporation. Kenmochi and Ohshita of Yamaha Corporation present in their 2007 conference paper that VOCALOID is a singing vocal synthesizer that utilizes sample concatenation [1]. Sample concatenation uses recordings of a singer singing different combinations of consonants and vowels, which can be combined in succession with modified pitches to produce full melodies. Although VOCALOID is successful as a software, it has been criticized for its lower fidelity compared to other vocal software which relied on manipulating live performance vocals [11]. As such, research on realistic SVS has continued beyond VOCALOID.

The recent introduction of machine learning to SVS further improved the realism of computer-generated human vocals. In 2021, Hono et. al proposed Sinsy[4] which utilizes machine learning, includes components to improve quality such as pitch correction, and introduces configurable parameters such as vibrato. Sinsy provided a foundation for practical state-of-the-art SVS, leading to several improved designs, as well as commercial products like Synthesizer V [3]. However, there are limitations to Sinsy and its derivatives: first, their input systems do not directly support non-lyrical expressions such as growling and shouting, and second, they are not intended for real-time use, prioritizing high fidelity instead of speed.

While real-time machine learning based SVS has not yet been achieved, concatenative real-time SVS has been explored. In a 2010 report, Kenmochi and Yoshioka from Yamaha Corporation demonstrated VOCALOID's potential for embedded systems with a prototype "VOCALOID-board" that could take lyric and pitch inputs from a computer and produce a voice output in real-time [7]. However, since VOCALOID-board failed to improve VOCALOID's sound quality, the vocals sounded unrealistic. Yamaha later introduced the NSX-1, a low-power IC chip designed for real-time vocal synthesis [10]. Our testing revealed that the NSX-1 lacks support for syllable transitions, resulting in disjointed vocals.

Emotion and expression control remain challenging for real-time systems. Capturing the nuances of human singing, such as vibrato, pitch fluctuations, and emotional depth, requires processing that can change dynamically during a performance. Although modern solutions like Synthesizer V [3] allow controlling intuitive attributes such as "Powerful," "Soft," and "Clear," existing real-time solutions do not tap into the potential of controllable expression, resulting in synthesized vocals that sound robotic or unnatural when compared to offline SVS systems.

III. System Model

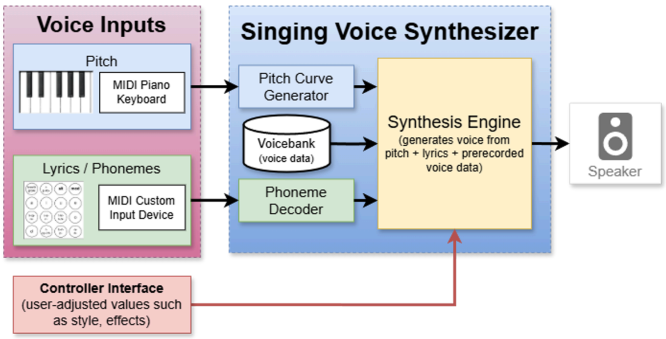

Fig 2. Our proposed *real-time singing vocal synthesizer* system model

For RIFF, we propose a system model which takes pitch and lyric inputs and processes them through a *pitch curve* generator, phoneme decoder, and synthesis engine to produce real-time human vocal output. We include a controller interface which allows users to adjust parameters like voice type, emotion, and mixer balance, ensuring the synthesized vocals match the emotional tone and dynamics of the performance.

RIFF is capable of synthesizing multiple voices through *voicebanks*, databases consisting of recordings of several syllables in different pitches by a human singer. The voice data consists of extracted features from the waveforms such as the fundamental frequency (F0), and is eventually reconstructed into an appropriate waveform by the SVS system.

Once the voicebank is loaded, the voice synthesis process is as follows:

Step-1) The system accepts note input from a human interface device such as a MIDI keyboard, and extrapolates information such as note pitch and length.

Step-2) Past and present pitch information is used to generate a *pitch curve*, which smoothly transitions the pitch from one note to another. The *pitch curve* is also influenced by auxiliary parameters from the *controller interface*, such as vibrato.

Step-3) The *volume envelope* of the voice is computed based on parameters such as the note velocity and length.

Step-4) The system accepts lyric input from another human interface device which produces phonemes, or parts of syllables.

Step-5) Past and present lyric information is used to determine the current syllable and voice data associated with it.

Step-6) If transitioning between syllables, the voice data is crossfaded between the previous and current syllables.

Step-7) The calculated pitch and voice data is inputted into a *phase vocoder*, which reconstructs vocals from extracted features stored in a database.

Step-8) The volume envelope of the voice is computed based on parameters such as the note velocity and length, and is applied to the resulting waveform.

A. Pitch

To naturally track vocal pitch, we generate a pitch curve, similar to approaches used in VOCALOID. Easing functions help create smooth pitch transitions, mimicking the natural movement of vocal cords. We also modulate pitch to add expression: slower modulation creates vibrato, while faster rates produce effects like growls and screams.

B. Lyrics

Lyrics, consisting of sequences of words, are made up of combinations of syllables. A single syllable may be split up into several phonemes, the basic units of speech. We assume that lyrics will be inputted in real-time through one or more button inputs, and that a *phoneme decoder* will process these inputs and interpret them as syllables or phonemes.

C. Synthesis Engine I

The synthesis engine is based on a vocoder which accepts F0 as a parameter, which enables more natural pitch control of the voice compared to resampling or directly manipulating the voicebank samples.

D. Comparison with past works

System	**Real-time**	**Input method**	**Synthesis method**
VOCALOID4	No	Piano roll with annotated lyrics	Sample concatenation
Synthesizer V	No	Piano roll with annotated lyrics	Neural vocoder
Yamaha NSX-1	Yes (pre-programmed lyrics)	MIDI keyboard	Real-time sample concatenation
RIFF	**Yes**	**MIDI keyboard with lyric controller**	**Real-time vocoder**

Table I: Comparison of current synthesizer systems

Table I above explores the comparison of current synthesizer systems, analyzing their real-time capabilities, input methods, synthesis techniques, and expression control. Systems like VOCALOID4, Synthesizer V and OpenUTAU DiffSinger rely on either sample concatenation or neural vocoders for synthesis [1][3][5]. These systems offer expression control through pitch curves, vibrato, and other vocal parameters, but are not suitable for live performances due to latency and lack of real-time input.

In contrast, systems such as Yamaha NSX-1 and our project RIFF offer real-time synthesis with MIDI input. While Yamaha NSX-1 uses a smooth pitch curve, it struggles with natural transitions between syllables, resulting in a choppy sound [10]. Project RIFF, however, allows for dynamic expression control, including *pitch curve*, vibrato, and growl.

IV. Implementation

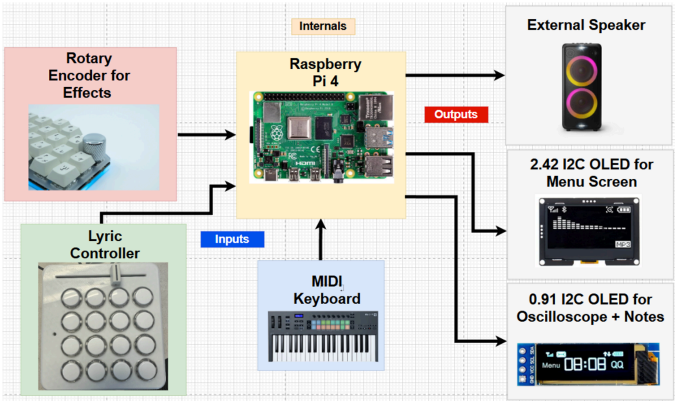

Fig 3. Our proposed *real-time singing vocal synthesizer* hardware model

For our implementation, we propose a model as shown in Fig. 3 which accepts inputs from a MIDI keyboard and a custom lyric controller. We develop a frontend with basic controls including rotary encoders to control vocal effects. We utilize communication protocols such as I2C to allow for the user to interact with the display and make changes based on user input such as switching between voices and seeing the output signal.

A. Lyric Controller

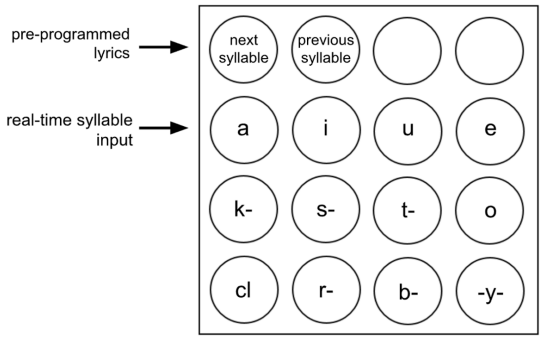

Fig 4. Lyric controller input mapping

To input syllables into the SVS system, a 4x4 push button matrix is utilized, where combinations of inputs represent syllables. Fig 4. depicts the input mappings, where consonants and vowels are dedicated to specific buttons. In this case, we assume not all combinations are compatible.

We also incorporate a simplified approach where lyrics are preprogrammed beforehand, and the user can progress forwards or backwards in the sequence of syllables using the top row of buttons.

B. Voicebank

Voicebanks that are utilized are designed for UTAU, a free software for SVS which aims to mimic the technology in VOCALOID. The voicebanks in UTAU are distributed as audio samples with oto.ini files associated with them.

The oto.ini files utilized in this project are made up of their file name and the phonemes/syllables that are related to it. These files consist of two different types of lyrics, CV (consonant-vowel) and VCV (vowel-consonant-vowel), which represent the different syllables that humans make when speaking. VCV files represent the transition between a vowel and another syllable, allowing for a more concise and smooth sounding voice. Generally, a vocalist is recorded several times transitioning between various samples, and by splicing sections of these audio files and adjoining them according to the oto.ini, the synthesizer can form full words.

Figure 4 shows an example of one of these oto.ini files. These files consist of 6 separate regions: the initial offset, consonant, cutoff, pre-utterance, stretched area, and overlap. The offset is the time of a given syllable in the audio file. The consonant length represents how long the unvoiced part of a syllable lasts. The note start is where the syllable aligns when synchronized to a beat. The stretched area is where the audio file is drawn out for longer notes. The crossfade end is where the previous audio file is fading out to fade in with the following audio file. In our project we created our own CV voicebank called hanadUTAU and another goal of RIFF is to allow user to input different voicebanks.

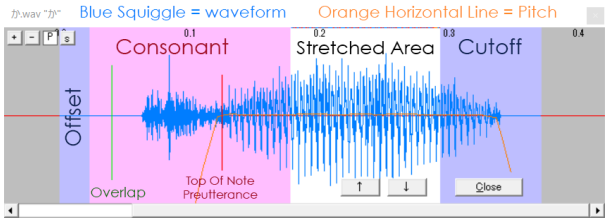

Fig 5. Example of audio file that consists of 6 regions (the initial offset, consonant, cutoff, pre utterance, stretched area, and overlap). The blue signal is the waveform of the audio and the orange line is the pitch of the audio file.

C. SoC (Singer on a Chip)

A significant hurdle for real time SVS is the trade-off between quality and speed. Systems like DiffSinger and Sinsy, each rely on neural networks to generate realistic vocals that focus on high quality sound generation, but take longer to produce. These systems cannot be used for real-time use, where the priority is to generate vocals as quickly as possible to match live performances. To address these challenges, hardware-based solutions have emerged as a viable path toward real-time SVS. Using a dedicated SoC allows real-time systems to achieve the speed needed for live performance.

D. PCB Design & Fabrication

For our real-time Raspberry Pi 4-based vocal synthesizer, we used a PCB to interface between the user inputs on from the switches and knobs to allow the Raspberry Pi to take in real time inputs to make changes to the voice allowing the user to freely adjust the voice in real time. Figure 6 below shows the user interface of the system in which we have designed our PCB around allowing proper connections between each of the different GPIO pins for the switches and knobs to the I2C pins for the displays.

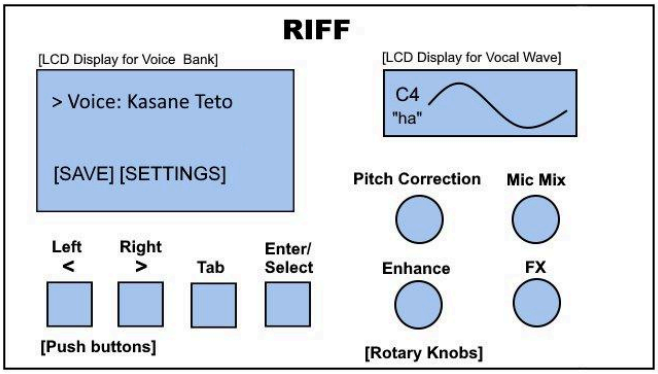

Fig 6. RIFF's User Interface

The PCB allows immediate adjustments in synthesis parameters sent to the synthesis engine inside the Raspberry Pi 4 such as pitch correction, reverb & delay, and a range of different effects to the voice. PCB also allows the user to change menu options such as switching between different voices, reloading different voicebanks, and also allowing saving of vocal settings for the user to use again in the future.

Figure 7 shows the connections of the PCB which are connected on to a 40 pin connector that connects to different GPIO and I2C pins to allow for the I/O interface to work in tandem with each other and enable intuitive user interaction. The PCB was designed on KiCad and was fabricated and shipped through JLCPCB.

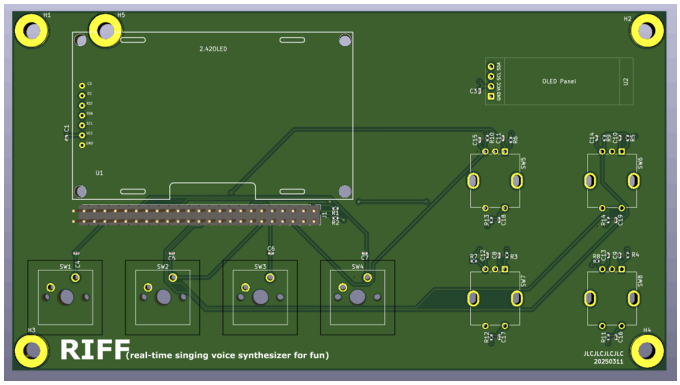

Fig 7. RIFF's PCB

E. Synthesis Engine II

The pitch and lyric input is received from USB ports on Raspberry Pi 4 which are sent to the Synthesis Engine, using the USB-MIDI specification. Preprocessing includes debouncing and ignoring duplicate input; once the inputs are cleaned they are translated into pitch and syllables.

During the onset and duration of any played note, ADSR envelopes are used to control the pitch and volume, which consists of 4 phases: Attack, Decay, Sustain, and Release (Fig. 8).

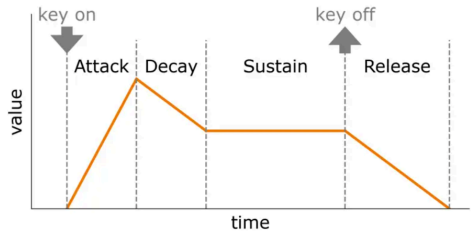

Fig 8. ADSR Envelope

In the case of the Synthesis Engine it is programmed to control how quickly or slowly the attack & decay occurs in terms of both volume and vibrato; by altering these rates, the synthesizer can modulate the expression of the voice by controlling the sharpness/softness of the phonemes.

Once the pipeline completes the three main preprocessing modules—Phoneme, Pitch, and Volume—the synthesis stage begins. For synthesis, we use WORLD, a vocoder known for producing natural sounding, human speech and becoming the standard by balancing quality and real-time performance [9]. Although neural vocoders have since surpassed WORLD in quality, its lightweight, machine-learning-free design is ideal for embedded systems. We plan to use WORLD or a variant of its implementation on the Raspberry Pi 4 to synthesize vocals.

The voicebank samples are stored as extracted features which are inputs to the WORLD vocoder: F0, spectral envelope, and aperiodicity. The WORLD components for the current and past lyrics are crossfaded and synthesized into a waveform by the WORLD vocoder [9]. Finally, the waveform is outputted to a speaker via a 3.5mm 4-pole composite audio video output.

F. Cost Analysis

RIFF Components	**Cost per Unit**
Raspberry Pi 4	$35
OLED 2.42"	$17
OLED 0.91"	$2.80
4 Knobs	$6.86
4 switches	$1.86
PCB Fabrication	$3
3D Printed Case	$3
Total	$69.34

Table II: Hardware Cost Analysis for RIFF

Table II displays the cost of the system at a component level in which the total material cost of RIFF is $69.34. We utilized these components to allow us to reach a lower cost alternative to competitor real-time systems. As stated we used the Raspberry Pi 4 for its portability of design as well as its ability to allow us to synthesize in real time with other I/O tasks running parallel such as the OLED displays and switches/knobs for control of the system. As outlined the Raspberry Pi 4 takes a majority of the cost of the system and different improvements can be made on the cost from the embedded system chosen such as an inexpensive microcontroller.

G. Task Distribution

Our group consists of 4 Computer Engineers and 2 Electrical Engineers with tasks divided but overlapping and peer reviewed. The task distributions are as follows:

- Nathan Stephenson (CE): Project Manager, created the project idea and system model, and will overlook both prototype software, firmware and hardware development. Overlooks any necessary edits for reports and presentations.

- Hanad Elmi (CE): Embedded Systems developer and researcher, designing CAD models for final instrument, in charge of team organization for deadlines, meeting times, mock presentation setup, etc.
- Gary Home (EE): Hardware Network Engineer and Technical writer, designs the physical components in CAD
- Connor Bartosiewicz (CE): Software developer in ZMQ and IPC communications between different programs.
- Berny Guzman (CE): FX Developer.
- Evan Coulter (EE): Equipment management and tester, will overlook the necessary power requirements and testing the product's interface.

Fig 9. RIFF synthesizer with custom lyric controller and commercial piano keyboard

V. Summary

We developed RIFF, a real-time singing voice synthesizer that is designed for live performance and expression. Figure 9 shows the complete system with a custom lyric controller and MIDI keyboard. Our approach addresses the requirements of low latency, high quality, and intuitive, real-time input. RIFF is the first of its kind for its inexpensive cost, real-time control without the use of preprogrammed lyrics, and portability compared to its competitors which are higher in cost and rely on preprogrammed lyrics.

References:

[1] H. Kenmochi and H. Ohshita, "VOCALOID- Commercial singing synthesizer based on sample concatenation," in Proc. Annu. Conf. Int. Speech Commun. Assoc., 2007, pp. 4009–4010.

[2] J. Liu, C. Li, Y. Ren, F. Chen, and Z. Zhao, 'DiffSinger: Singing Voice Synthesis via Shallow Diffusion Mechanism', *arXiv [eess.AS]*. 2022.

[3] "Synthesizer V," https://dreamtonics.com/synthesizerv/, accessed: 2024-10-25.

[4] Y. Hono, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Sinsy: A Deep Neural Network-Based Singing Voice Synthesis System," IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 29, pp. 2803–2815, 2021, doi: https://doi.org/10.1109/taslp.2021.3104165.

[5] J. Liu, C. Li, Y. Ren, F. Chen, P. Liu, and Z. Zhao, "DiffSinger: Singing Voice Synthesis via Shallow Diffusion Mechanism," *arXiv:2105.02446 [cs, eess]*, May 2021, Available: https://arxiv.org/abs/2105.02446

[6] "Vocaloid 2 Sweet Ann by PowerFX - Singing Synth Plugin VST3 Audio Unit," *KVR Audio*, 2019. https://www.kvraudio.com/product/vocaloid_2_sweet_ann_by_powerfx, accessed: 2024-10-15.

[7] "eVY1 module," *Aides-tech.com*, 2014. https://evy1.aides-tech.com/en/, accessed: 2024-10-25..

[8] Ikinamo, "Yamaha Vocaloid Keyboard - Play Miku Songs Live! #DigInfo," YouTube, Mar. 20, 2012. https://www.youtube.com/watch?v=d9e87KLMrng, accessed: 2024-10-25.

[9] M. MORISE, F. YOKOMORI, and K. OZAWA, "WORLD: A Vocoder-Based High-Quality Speech Synthesis System for Real-Time Applications," IEICE Transactions on Information and Systems, vol. E99.D, no. 7, pp. 1877–1884, 2016, doi: https://doi.org/10.1587/transinf.2015edp7457.

[10] Yamaha, "YMW820 NSX-1," 2012. https://sandsoftwaresound.net/wp-content/uploads/2017/07/YMW820_data_sheet_en.pdf, accessed: 2024-10-25.

[11] S. A. Bell, "The dB in the .db: Vocaloid Software as Posthuman Instrument," *Popular Music and Society*, vol. 39, no. 2, pp. 222–240, June. 2015, doi: https://doi.org/10.1080/03007766.2015.1049041.