MUMT-307 Final Project Report

Formant-Wave-Function Synthesis

Yifan Huang

2022.04.22

1 Introduction

FOF (Formant-wave-function) synthesis is a method that directly uses the function of time to calculate the amplitude of the waveform of a signal. FOF stands for "Function d'onde formantique" in French. This Synthesis technique is first developed by Xavier Rodet at IRCAM in 1980 [1]. It is inspired by the source filter model of vocal synthesis that models the excitation of the resonant frequencies in the vocal tract by the glottis and was first used to synthesize vocal sound in the CHANT project of IRCAM [2]. The term "formant" in FOF not only meant in a human voice but meant in a broad sense, the resonance of the sound in the spectra domain. The FOF synthesis is not only used to create very realistic vocal sounds, but also some instrumental sounds.

This project will focus on realizing the FOF synthesis algorithm in real time using STK (for implementation) and MATLAB (for analysis in the early stage) based on two papers in 1980 [1] and 1984 [2], explore the nature of the FOF algorithm in both time and spectra domain, dig into the usage of the parameters in the function in a real practice situation and analyze the advantages and the limitations of the FOF synthesis.

This report is organized as follows: in section 2, we will go over some related works of implementation of FOF. Section 3 will show the details of the FOF algorithm and the implementation. Section 4 will show the principles of FOF synthesis techniques and the implementation. Section 5 will provide a thorough discussion on the advantages and limitations of the FOF synthesis and personal experience in this project. Finally, the conclusion will be given in Section 6.

2 Related works

After developing the FOF and using it in the CHANT program, FOF was implemented by a unit-generator to Music 11 as a part of VOCEL (VOiCe Eleven) project in 1988 by Michael Clarke. This implementation allows different fundamental frequencies for each formant. In 2003 [3], Clarke and Rodet implement the FOF synthesis to Max [4].

More recently, a different implementation approach was taken by Michael Jørgen Olsen et al. [5]. They proposed a hybrid filter-wavetable oscillator technique to implement the FOF synthesis. In this research, since the amplitude of a FOF is an exponentially decaying amplitude envelope with smoothed attack, an impulse train is fed into a second-order filter to generate this envelope. Then, this amplitude envelope is multiplied by a sinusoidal wavetable oscillator to generate the FOF wave bursts that can be used for FOF synthesis. This research is implemented using the FAUST (Functional Audio Stream) audio programming language. Also, similar to the original time-domain FOF implementation, this approach allows users to separately control the bandwidth and skirt width of the formant region generated in the frequency domain by the FOF synthesis.

3 FOF Algorithm and Implementation

3.1 FOF Algorithm

Generally, when we synthesize a sound, we could use an excitation function followed a parallel or cascaded filter. In the FOF method, the filter is replaced by a unique formula describing the output of the filter. Here, when the input is an impulse, the output, or we can say the impulse response of one formant can be calculated in the following formula:

$$s(n) = 0, k \leq 0$$

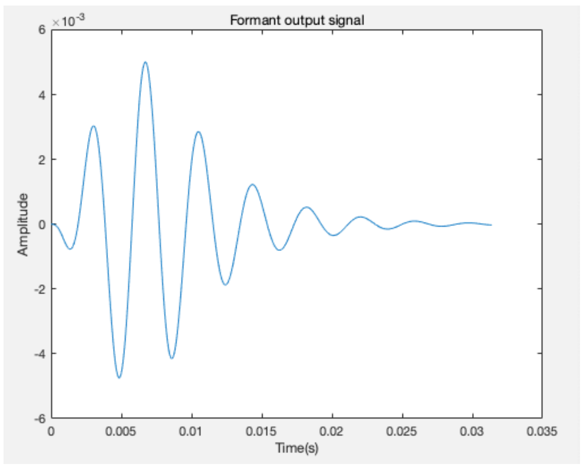
$$s(n) = \frac{1}{2}(1 - \cos(\beta nT))e^{-\alpha nT}\sin(\omega_c nT + \varphi)\,, 0 \leq n \leq k$$

$$s(n) = e^{-\alpha nT}\sin(\omega_c nT + \varphi)\,, n \geq k$$

This formula is a sinusoid function shaped by an envelope. This envelope is a damped exponential amplitude, and the initial discontinuity of the envelope is smoothed by multiplication by (1/2)(1-cos(βnT)) for a duration of k samples, where k = π/(βT) controls the amplitude envelope attack time as well as the skirt width of the formant region in the frequency domain, α controls the bandwidth of the formant region and ωc controls the center frequency of the formant region. Here the term skirt width measures the extent of the spectral shape far from the maximum (-40dB in the spectrum), and the bandwidth measures the extent close to the maximum (-6dB in the spectrum). The counterpart of π/β (width of the skirts in the spectral domain) is the attack time in the time domain, as the counterpart of α/π (damping coefficient) is the bandwidth in the spectral domain. The bandwidth and skirt width can be controlled independentaly of each other.

3.2 FOF Algorithm Analysis and Implementation

I first use MATLAB to analyze the algorithm and the parameters and also check my ideas for implementing the algorithm. Here is the result of one formant "Impulse response" (Center frequency = 260, Attack time= 0.01s, Initial Phase = pi, Bandwidth = 70, Sample rate = 44100, Amplitude = 0.029), as shown in figure 1,2. We can clearly see the envelope in the time domain and the center frequency in the spectra domain controlled by our parameters.



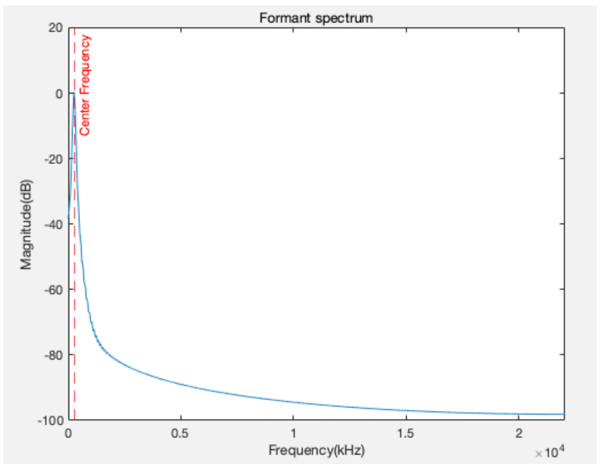
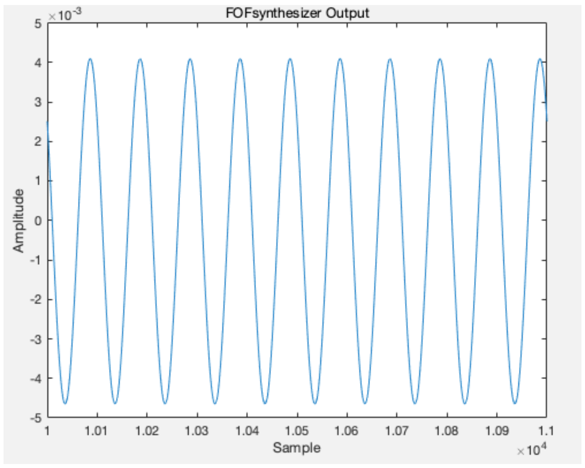


Figure 1,2 Formant impulse response in time and spectra domain

However, when we want to construct a natural sound, we often use an impulse train as the input of the FOF. In other words, we have to generate a FOF burst in the fundamental frequency of the impulse train and sum up all the bursts as the output. Since the FOF burst may not “end” by the start of the next period, this may lead to overlap adding. In my implementation, I use the table look-up method to implement the overlapping add. Since the FOF is actually infinite, I cut off the burst when the amplitude decay to T60 to form the table. Therefore, at each time point, we just need to find all the possible values in the table, add them together and we will get the result. Here is a result of one formant output (Fundamental Frequency (Impulse train frequency) = 440, Center Frequency = 300, Attack Time = 0.006s, Bandwidth = 50Hz, Sample rate = 44100, Amplitude = 0.029), shown in the figure 3, 4. As for the different fundamental frequencies and the Sample rate that may lead to “fractional index” or “fractional delays”, I use the closest integer to deal with this problem, but this should be improved in the future.



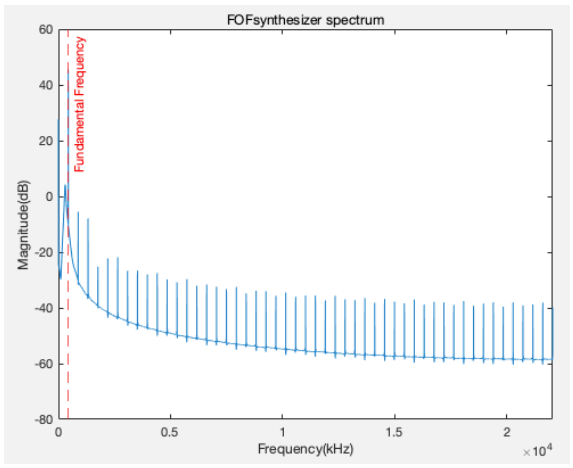
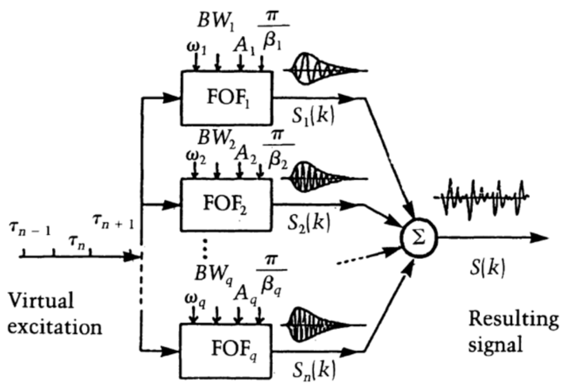


Figure 3,4 One formant output in time and spectra domain

4 FOF Synthesis and Implementation

FOF synthesis technique is based on the FOF algorithm. It synthesizes a sound by superposing the contributions of each formant. Therefore, we could sum the different FOFs together to obtain a target sound. As shown in figure 5, it is the structure of the FOF synthesis that we input an impulse train to a parallel FOF and add all the results of the functions to get the resulting signal.



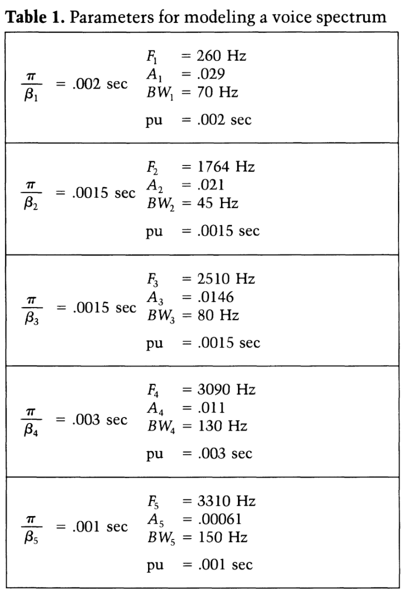


Table 1. Parameters for modeling a voice spectrum

$\frac{\pi}{\beta_1}$ = .002 sec	F_1 = 260 Hz A_1 = .029 BW_1 = 70 Hz pu = .002 sec
$\frac{\pi}{\beta_2}$ = .0015 sec	F_2 = 1764 Hz A_2 = .021 BW_2 = 45 Hz pu = .0015 sec
$\frac{\pi}{\beta_3}$ = .0015 sec	F_3 = 2510 Hz A_3 = .0146 BW_3 = 80 Hz pu = .0015 sec
$\frac{\pi}{\beta_4}$ = .003 sec	F_4 = 3090 Hz A_4 = .011 BW_4 = 130 Hz pu = .003 sec
$\frac{\pi}{\beta_5}$ = .001 sec	F_5 = 3310 Hz A_5 = .00061 BW_5 = 150 Hz pu = .001 sec

Figure 5 FOF synthesis structure Figure 6 Parameters used in FOF synthesis

After checking my ideas in MATLAB, I implement them in STK. I design a C++ class for each formant that could be easy to use in the future. And I packed the FOF algorithm and overlap adding illustrated in the last section into this class and the main function of the code is used for FOF synthesis. As for the parameters, I used the parameters from Rodet’s paper [1], as shown in figure 6 to synthesize the human voice. The result of the sound is the real-time output of the program. From my perspective, this human voice generated is very “real”. Also, I have another non-real-time version implementation in MATLAB for can be accessed in the appendix.

In addition to the algorithm implementation, I add some audio effects to make it more natural. I add the reverberation, vibrato, chorus effect, and amplitude modulation into the main program. The combination of chorus effect and reverberation and the combination of vibrato, amplitude modulation, and reverberation are both good. Users could turn on or turn down the effects easily using the parameters in my code.

5 Discussions

5.1 Advantages of FOF

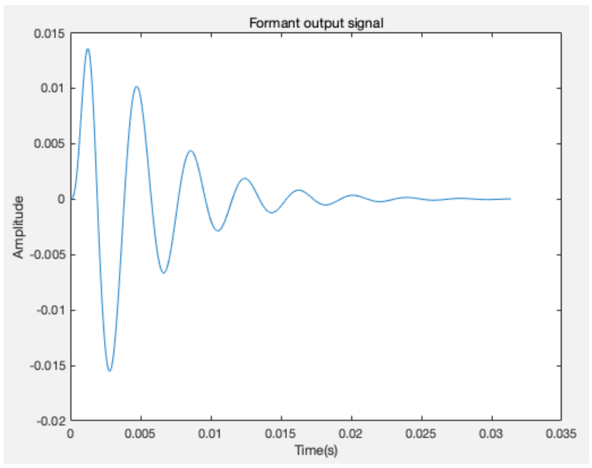
Generally speaking, FOF can obtain the synthesis of very high quality for a low calculation cost that it could be performed in real-time which make me realize it in real-time STK. Also, the precision required in the calculation is at most that required at the output, therefore, no risk of numerical overflow exists.

At the same time, the calculation of the control parameters is simple. Like skirt width and bandwidth, we can simply do a spectrum analysis of the target sound (such as using FFT), and we can get the parameters from the spectra plot.

Also, the FOF has a wide range of usage. It can be used to synthesize very realistic vocal sounds, and also can be used to reconstruct instrumental and other sounds.

Compared to synthesizing using filters, FOF is continuous by construction. Since a new FOF burst is constructed at each period, it allows us to adjust the variable center frequency in a very detailed way and do not need to worry about the discontinuity like using a filter. FOF is also really good at synthesizing a rich sound in a certain frequency band without having to specify each partial.

From my perspective, in practice, the usage of parameters also provides great advantages. One of the advantages of FOF is from parameter β. Since parameter β governs the skirt widths, we can simply generate forms that would require a filter of a higher order or unique functions of excitation for each filter component. Here are the result only of different β (Attack time = 0.002s and Attack time = 0.01s), as shown in figure 7, 8.



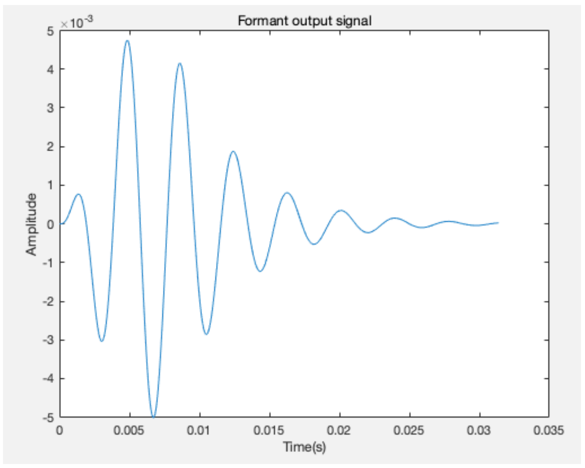


Figure 7,8 Formant impulse response of Attack time = 0.002s and 0.01s

5.2 Limitations of FOF

Although the FOF performs really well in synthesizing a sound, there are still some limitations.

As for the implementation of the FOF, since FOF is actually infinite, it is difficult to implement that we can’t have an infinite table in the implementation of overlap adding. In this way, we have to find an “end” when using a table lookup in implementation. In this project, I use the amplitude of T60 to be the end of the table. Also, because of the infiniteness of FOF, we can not simply use a filter to implement. We also have to cut the waveform at some point to deal with the finiteness.

5.3 Implementation Experience

Generally, it is my first time to do algorithm implementation from a paper to result in signal processing field. For me, it is hard at the beginning, since it is difficult to understand the true meaning of each function and the nature of each parameter in the function. Also, it is very abstract just looking at the paper. Luckily, everything gets clearer when I started to do a little. Therefore, I think in the algorithm implementation task, understanding by doing could also be a good way.

As for my implementation procedure, it's not very smooth. At first, I spent a lot of time implementing them in Max/MSP for its great GUI, but I failed in implementing the overlap adding in this situation. Also, as I illustrated before, IRCAM has an external Max object of FOF, I implement the synthesis using their object but still find some "click" in the output. Finally, I turned to STK to implement my algorithm in real-time and come up with the lookup table method for overlap adding. For me, I think it is not easy to find the most ideal tool to implement the algorithm at first, but it will show after many precious trials.

Also, figuring out the true physical principle of the algorithm is very helpful since different papers may use different physical units to illustrate the algorithm. Physical units are also a part that would be difficult to debug. Therefore, understanding the physical principle would be useful.

Often looking back to the papers will be illuminated. For me, I fully figured out the usage of the initial phase and skirt width just when I looked back at the papers after finishing my implementation. Almost every time I review the papers, I would get something new. From my perspective, reviewing the papers are the most helpful way in my exploration of algorithm implementation.

5.4 Future Improvement

As for the possible improvement, since the real sound especially for human voice always has changing frequencies and FOF allows the implementation of variations frequencies, I may implement possible variable center frequency or variable fundamental frequency to explore more potentials of FOF synthesis.

In the meantime, thanks to my classmates who suggested I use gen~ to implement overlap adding in Max, I will try to figure out that in the future.

Also, depending on the fundamental frequencies we choose, it may lead to the fractional sample index and may need to do interpolation in implementing overlap adding using a table lookup. Therefore, in the future, I would like to do some implementation on fractional delays and interpolation to conduct a more precise and sophisticated implementation of FOF.

6 Conclusions

In this report, I provide an introduction to FOF synthesis and go over a brief review of previous studies. The FOF synthesis details are illustrated and are implemented using STK and MATLAB in this project. The limitations, advantages, and implementation experience, are discussed. I have learned a lot in this study. Many possible improvements could be done in the future.

References

[1] X. Rodet, “Time-Domain Formant-Wave-Function Synthesis,” *Comput. Music J.*, vol. 8, no. 3, pp. 9–14, 1980.

[2] X. Rodet, Y. Potard, and J. Barriere, “The CHANT Project: From the Syntesis of the Singing Voice to Synthesis in General,” *Comput. Music J.*, vol. 8, no. 3, pp. 15–31, 1984.

[3] J. M. Clarke and A. Purvis, “VOCEL: New implementations of the FOF synthesis method.pdf,” in *Proceedings of the International Computer Music Conference*, 1988.

[4] M. Clarke and X. Rodet, “Real-time FOF and FOG Synthesis in MSP and its Integration with PSOLA,” in *Proceedings of the International Computer Music Conference*, 2003.

[5] M. J. Olsen, J. O. Smith, and J. S. Abel, “A hybrid filter-wavetable oscillator technique for formant-wave-function synthesis,” in *13th Sound and Music Computing Conference Proceedings*, 2016, pp. 366–372.

Appendix

All Code could be accessed in the following link:

https://github.com/HuangYifan-Emma/FOF-synthesizer.git