

USABILITY OF CURRENTLY AVAILABLE SOFTWARE FOR THE APPLICATION OF VARIOUS SYNTHESIS ALGORITHMS

¹Julie Tan & ²Minni Ang

Music Department

Faculty of Human Ecology

Universiti Putra Malaysia

43400 UPM Serdang

tel. 03-9486101 ext 1738

fax. 03-9161689

¹julie@music.upm.edu.my

²minni@music.upm.edu.my

ABSTRACT

There are various types of music synthesis methods, which can be applied in computers using different music synthesis software currently available. Early synthesizers were based on oscillators, filters and envelope generators which were linked together by cords and patch bays. With recent advances in software technology, professional quality sounds can now be generated using many different software synthesis techniques. Due to the accelerated development in music synthesis software in the recent years, there are an increasing number of new softwares available now. The abstract nature of software makes it difficult to characterize, and design changes are often introduced into software before, during and after its pure code-production phase. Moreover, different software has been invented separately or specifically for different music synthesis techniques. This paper presents a methodology to test music synthesis softwares for their usability for specific music synthesis tasks, according to the different software interfaces. The softwares used includes specialized graphical user interface synthesis software, digital audio editing software which can be used for music synthesis purposes, and command prompt type software. Similar music synthesis tasks can be performed by software perceived as being vastly different in approach. The aim of this research is to identify the best software for specific synthesis purposes from the aspects of usability, efficiency and user-friendliness. Comparison of various synthesis techniques and parameters defined aid in determining this output.

Keywords: Software synthesis, music software, synthesis algorithms, software quality, software testing.

1.0 INTRODUCTION

"Music synthesis" refers to the method using a voltage or computer controlled oscillator to produce a fundamental waveform, with changes to the waveform being made using different techniques to produce different sounds. During recent years, the musical world has rapidly changed, with many new music synthesis techniques developed, and with the

arrival of Pentium and equivalent microprocessors in computers. The increasing general importance and scope of software applications is evident in almost every facet of modern life. These have attracted a wide range of musicians who wish to create their own instruments.

Music software production has evolved from being something of a black-box art form, into an engineering discipline in which there has been a very rapid evolution of techniques. The momentum of change in this field therefore makes the process of basing new estimates on anything other than fairly recent developments very suspect. The evaluation of software can be viewed as a basis for increasing music software production efficiency or quality and for providing a rational means of measuring and estimating development costs.

2.0 LITERATURE REVIEW

In the early years, a lot of effort went into custom-built hardware design, where components were wired from point-to-point with inexpensive tools, to troubleshoot problems hindering high performance. Most of these projects did not include the development of software environments [1]. This can be traced through innovations by inventors such as Leon Theremin, Les Paul, Robert Moog, and Donald Buchla to bring new and unusual sounds into musical practice [2]. Often the hardware technology of the 1990s uses the software technique of the 1970s and 1980s to realize the musical concepts of the 1960s [3]. However, the musical world of hardware is gradually falling into following the increased power and availability of software to musicians. This software-based synthesis takes advantage of the full capabilities of general-purpose processors and is gradually replacing the instruments for both studio and live performance [2]. The instruments or modules are connected using virtual patch cords, and function exactly like the hardware before, but is portable, more reliable and easier to handle. The advantages of software compared to hardware-based synthesis are the space and cost savings, ease of upgrading, programmability and lack of physical limitations. There are also some disadvantages in software synthesis. For example, a high-powered computer is required, and latency (the time between when a note is pressed and when it sounds) is also an issue. Basically, the concept of "soft" resists a standardisation of rules, but instead depends on the relationship of composers or performers to systems, the flow of information, and also with the network. In reality, sound may only be produced when a computing procedure (synthesis algorithm) can be described for its generation. This has given great impulse to the search for algorithms for sound synthesis [4]. Such algorithms have been available since early 1970s [5]. Today, there are many methods of sound synthesis and a wide array of synthesizers employing various methods is in use. Major methods include Additive synthesis, Subtractive synthesis, Frequency Modulation (FM), Ring Modulation (RM), Amplitude Modulation (AM), Granular synthesis, Physical Modeling, Spectral Modeling and others [6]. Nowadays, commercial products such as synthesizer keyboards or software synthesizers use different kinds of synthesis techniques in order to produce more realistic sound instruments. Due to the accelerated development in music synthesis software, there are an increasing number of new softwares with different interfaces available now. Thus, the objective of this paper is to present a methodology to test the

usability of the various synthesis softwares using different synthesis technique. These softwares are grouped according to the software interface. Various synthesis techniques and parameters defined are compared to assist in determining software usability issues. As a result the most appropriate software for different synthesis techniques will be identified.

2.1 Software Metrics

There are many softwares available in the market today. However, it is difficult to choose software, which accommodate the user's needs. According to McCall, the criteria for software metrics should be analysed and compared to determine desire software. The parameters or metrics [8] used are adapted from McCall's [7] criteria of quality, defined in Table 1. The parameters used by McCall are then broken-down into smaller features, for ease and simplicity of comparison (Table 2).

Table 1. McCall's Criteria of Software Quality

| Parameters | Features |
|-------------|--|
| Usability | Ease of use of the software. |
| Efficiency | The amount of computing resources required to perform a user-defined function. It is concerned with the use of resources, such as processor time. |
| Correctness | A system satisfies its functional specification. |
| Reliability | The extent to which a system can be expected to reproduce its function over a period of time without failure. |
| Flexibility | The ease of making changes required. |
| Testability | The ease of testing the program, to ensure that it is error-free and meets its specification. |

Table 2. Different Parameters in Software Evaluation

| Components | Features |
|-------------|-----------------------|
| Usability | Handling ability |
| | Learning requirements |
| | Installation |
| Efficiency | Efficiency |
| Testability | Simplicity |
| | Complexity |
| | Traceability |
| Reliability | Accuracy |
| | Consistency |
| | Modularity |
| Flexibility | Consistency |
| | Expandability |
| | Modularity |
| Correctness | Completeness |
| | Traceability |

3.0 METHODOLOGY

The music synthesis techniques to be used are first determined. Music synthesis softwares are then identified, for the purpose of testing their usability for specific music synthesis tasks, according to the different software interfaces. The softwares used in this research according to the software interface include specialised graphical user interface synthesis software, digital audio editing software and command prompt-type software.

The next step is the experiment phase, where synthesis techniques are applied into these softwares one by one. The parameters are applied and changed according to the needs of the technique in use. Finally the results produced are analysed and compared according to the different facets of software quality, to determine the usability of these softwares. Figure 1 is a summary overview of the software testing methodology.

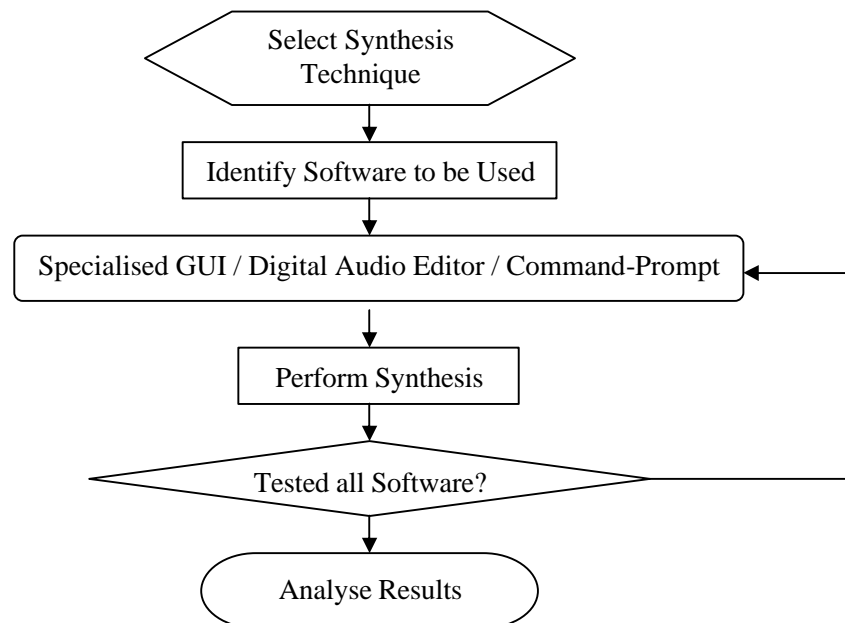


Figure 1. Software Testing Methodology

3.1 Perform Synthesis Technique

Additive synthesis technique is discussed here using the different groups of software interface. The parameters used for each softwares includes the type of wave (*sine wave*) and frequency (220 Hz, 660Hz, 1100Hz, 1540Hz and 1980Hz).

3.1.1 Graphical User Interface Software

The software *WaveCraft* was chosen to represent this group of software. In this software, blocks are used to represent modules. Five oscillators were selected and connected using a mixer and later to a file dump to generate sound (Figure 2).

3.1.2 Digital Audio Editor

Cool Edit Pro 1.1 was chosen to represent this group of software. Tones were selected under the generate menu, to generate sound. The sound was then played back (Figure 2).

3.1.3 Command Prompt-Type Software

For command prompt-type software, *Matlab* was chosen. The sound here was generated using equations defined at the command prompt, as listed in table 3 below. The software interface is shown in figure 2.

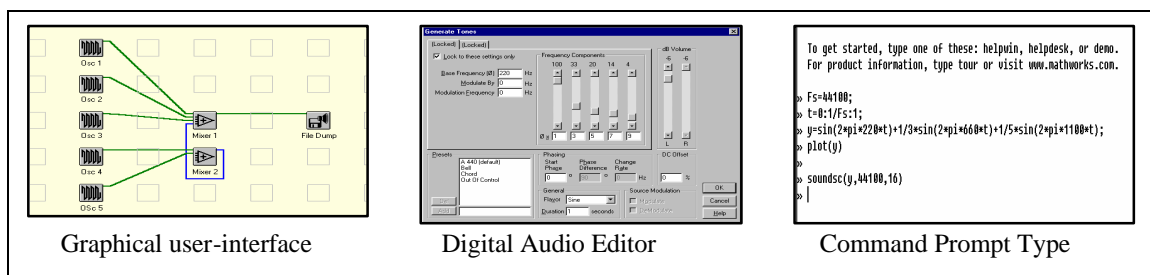


Figure 2. Software Interfaces

Table 3. Equations used for Command Prompt-Type software.

| Equation | Description |
|---|---|
| $F_s=44100$ | Sampling rate |
| $t=0:1/F_s:1$ | x axis value |
| $y=\sin(2\pi*220*t)+1/3*\sin(2\pi*660*t)+1/5*\sin(2\pi*1100*t)+1/7*\sin(2\pi*1540*t)+1/9*\sin(2\pi*1980*t)$ | y axis equation to generate the same type of sound quality as the other groups of software interface. |
| <code>plot(y)</code> | Graph produced from the equation. |
| <code>soundsc(y,44100,16)</code> | To listen to the sound and the type of sound generated. |

3.2 Music Synthesis Software’s Usability Metrics

The metrics and definitions used by McCall are very general and can be used by all types of softwares. Thus, the metrics will only be used here as a guide to produce new definitions or terms to suit music synthesis purposes. The “TLM table” (Table 4) below is the new metrics developed for this purpose and it is used to analyse the results produced from the application of various synthesis algorithms into different music synthesis softwares.

Table 4. TLM's Metrics for Music Synthesis Software

| Components | Features | Details |
|------------------------|-----------------------|----------------------------|
| Usability | Handling Ability | Menu levels |
| | | Shortcut keys |
| | | Customisable toolbars |
| | | Balloon help |
| | | Tool palettes |
| | | Customisable colours |
| | | Mouse right-click function |
| | Learning requirements | Help file |
| | | Tutorials |
| | | Searchable help |
| | | Wizards |
| | | Tip of the day |
| | | Website support |
| | | Newsgroup/ mailing list |
| Printed/ Online manual | | |
| Efficiency | Processing time | |
| | Number of resources | CPU usage |
| | | Memory usage |
| | | Hard disk usage |
| Testability | Complexity | Questionnaires |
| Flexibility | Expandability | Customising parameters |
| Reliability | Accuracy | Bit resolution |
| | | Sampling rate |
| Correctness | Completeness | Questionnaires |
| | Traceability | Trouble-shooting wizards |
| | | Error logs |

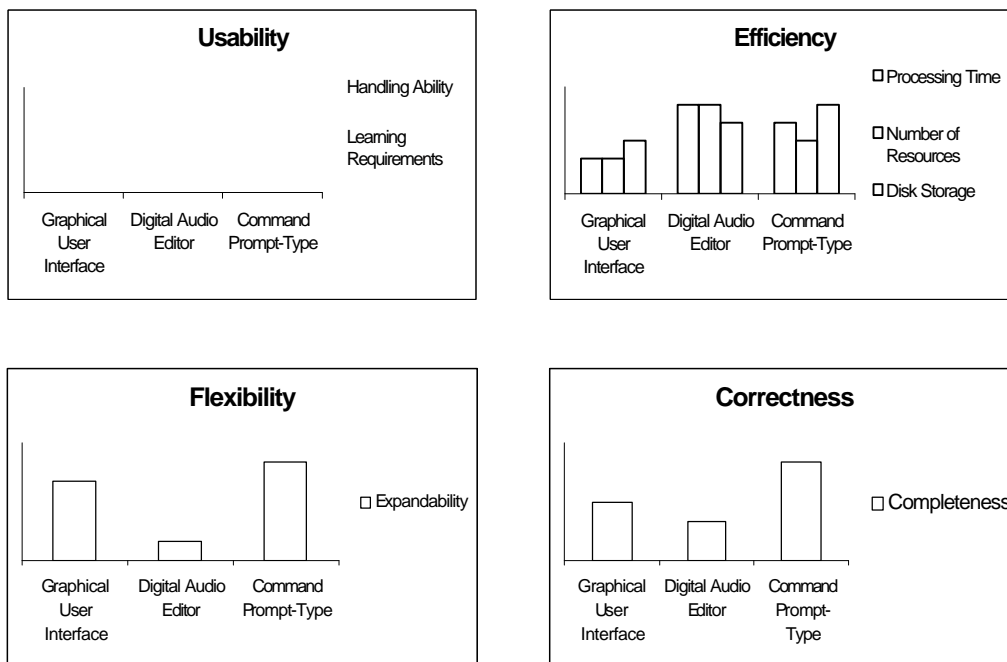
Below are the definitions for the new parameters:

1. **Usability** – How much effort does the user need to learn and use the software? (Can the user operate it easily?). *Handling ability* looks at the user's attitude towards the system because most prefer on-hand manuals on how to use the software. *Learning requirements* (Do these softwares provide any examples, tutorials or printed manuals to guide users?).
2. **Efficiency** – The amount of resources (CPU, memory and hard disk usage) and time needed to perform a user-defined program.
3. **Testability** – How much effort or how do users feel towards the software? This is because different people will have different conceptions towards different types of softwares. Testability in terms of complexity is gauged using questionnaires.
4. **Flexibility** – Does the software provide any customisable parameters? It is defined as the ability to modify and change the module or program at any time.

5. **Reliability** – Does it work accurately as expected? Musical point of view is taken into consideration here. For example, the higher the bit resolution the better is the tone quality. This also applies for the sampling rate.
6. **Correctness**– Does it provide full implementation of the functions required? (Completeness). *Traceability* is the ability to trace back problems or errors to prevent the same error from happening again in the future.

4.0 EXAMPLE RESULTS

The tables below illustrate the results obtained for the preliminary tests using Additive Synthesis as described in section 3.1.



5.0 DISCUSSION

From the graphs above digital audio editor provides better handling ability tools, though command prompt type software provides more tutorials, examples and others. With respect to efficiency, digital audio software processes results faster but uses more resources than the other softwares. Though the command prompt type of software uses more disk space storage and is more complex, it is however more flexible, reliable and the results produced are more complete. The digital audio editor is simpler to use, but the parameters used do not provide much room for expansion. Graphical User Interface software are more stable for all the metrics used for comparison.

6.0 CONCLUSION

The methodology for comparison of currently available software for the application of various synthesis algorithms was developed and presented. Early example results, where the additive synthesis technique was used, indicate the methodology is useful.

REFERENCES

- [1] Dannenberg, R.B. (1996). "A Perspective on Computer Music". *Computer Music Journal*. 20(1): Pp.52-56.
- [2] Chris, B. (1996). "Bringing Digital Music to Life". *Computer Music Journal*. 20(1): Pp.28-32.
- [3] Paul, B. (1996). "Abstracting the Future: The Search for Musical Constructs". *Computer Music Journal*. 20(3): Pp.24-27.
- [4] De Poli. G. (1996). "In Search of New Sound". *Computer Music Journal*. 20(2): Pp.39-43.
- [5] Goebel, J. (1996). "Freedom and Precision of Control". *Computer Music Journal*. 20(1)" Pp.46-48.
- [6] Miranda, E.R. (1998). "Computer Sound Synthesis for the Electronic Musician". Oxford: Focal Press.
- [7] Alan, C.G. (1992). "Software Quality". *Theory and management*. London: Chapman & Hall.
- [8] Arthur, L.J. (1985). "Measuring Programmer Productivity and Software Quality." New York: Wiley.