Improving L-System Music Rendering Using a Hybrid of Stochastic and Context-Sensitive Grammars in a Visual Language Framework

Lim Chen Kim and Abdullah Zawawi Talib

School of Computer Sciences, Universiti Sains Malaysia,
11800 USM Pulau Pinang, Malaysia
chenkimlim@gmail.com, azht@cs.usm.my

Abstract. L-Systems have been extensively utilized in plant modeling and music rendering. However, the music generated was not very pleasant as the grammars used are very simple. This paper describes a hybrid method that generates more complex grammars for L-Systems in a visual language framework for music rendering so that the musical sounds generated can be improved and fine-tuned. The method which uses a hybrid of stochastic and context-sensitive L-Systems grammars is vital in producing harmonious musical sounds and a variety of L-System grammars for L-Systems music rendering. Based on the evaluation, the method has been rated to be useful and effective in rendering harmonious musical sounds using the visual language framework even for anyone who does not have prior knowledge in L-System music rendering.

Keywords: L-System Attributes, Stochastic, Context-Sensitive, Plant Modelling, Music Rendering, Visual Programming Framework.

1 Introduction and Related Work

L-System is a language consisting of a set of strings governed by a set of production rules while a string consists of a sequence of symbols called axioms. When the production rules are applied to the axioms, the string generated can be used to model plants and render musical sounds [1]. This paper focuses on visual generation of musical sounds using L-Systems. Stochastic and context-sensitive grammars are combined to produce a pleasant arrangement of musical notes. Consequently, the production string will render a more harmonious music. In an existing tool for music rendering called LMUSe [2], the production of a string is generated randomly, such as:

$$D<E>F(1/2)=LMNOP \tag{1}$$

which means that 50% of the time if E is between D and F, E will be replaced by LMNOP and:

$$(0.33)Z<G>H=SS \tag{2}$$

A.L. Brooks (Ed.): ArtsIT 2011, LNICST 101, pp. 46–53, 2012.

which means that there is a 33.33% chance that G will be replaced by SS if G is between and H. As the production rules are performed randomly, the string produced will generate inaccurate and different results each time. Current works in L-Systems music rendering are based on simple grammars since they have to be scripted by the programmers or users. Stochastic and Context Sensitive L-Systems grammars allow complicated grammars to be generated easily on visual language framework. Hence, a system called Visual Language Music Rendering (VLMR) has been developed to provide the flexibility in generating visually harmonious musical sounds using these complicated grammars.

1.1 Stochastic Grammar

Stochastic grammar is vital for reading the rules differently in an effort to meliorate the L-System music rendering [3]. Most of the times, every occurrence of the predecessor is simply replaced by the successor that is bounded by a set of production rules represented by the derivation symbol →. The derivation of the successor is obtained through probabilities derived from a set of predecessor grammars. The production string is generated by replacing the successor that has a higher probability i.e. a note with the highest occurrences in a set of input rules. The sum of the probability of all musical notes is equal to 1. Let's assume that 'a' is the predecessor for all productions, thus

$$\sum_{i=1}^{N} p_{ai} = 1 \quad (3)$$

Stochastic grammar also allows the representation of its attributes using the Markov model [4]. For example, chord 'D' will be replaced from chord 'E' in the production string as the probability of chord 'E' is higher compared to the other chords as given below:

$$p : \mathrm{D} \quad \begin{array}{l} \xrightarrow{3/10} \mathrm{C} \\ \xrightarrow{3/10} \mathrm{D} \\ \xrightarrow{4/10} \mathrm{E} \end{array} \quad (4)$$

At times, it is unavoidable to have equal production probability as given below:

$$p : a \quad \begin{array}{l} \xrightarrow{1/2} b \\ \xrightarrow{1/2} c \end{array} \quad (5)$$

which can also be written as:

a(.5)=b
a(.5)=c (6)

which implies that the predecessor 'a' has an equal chance to be replaced by the successors 'b' or 'c'. This particular replacement is randomly determined by VLMR.

1.2 Context-Sensitive Grammar

In order to generate a variety of musical notes, VLMR has been extended to include context-sensitive grammar which is concerned with the other symbols surrounding the symbol to be replaced. The context-sensitivity rule can be portrayed as follows:

$$A<B>C=DEF \tag{7}$$

which implies that B will be replaced by DEF provided that A is on the left of B and C is on the right of B. The rule can also be one-sided at times. For example:

$$J<K=XYZ \tag{8}$$

means that K is replaced by XYZ when J is on the left of K and there is no symbol to the right of K, or at the end of the production string. The same theory applies for:

$$R>S=PQRST \tag{9}$$

which means that R is replaced by PQSRT if R is at the beginning of the production string and there is no symbol to the left of R but there is S on the right.

1.3 Previous Work

Our work is inspired by the previous work of Siew and Talib [5] namely Visual Language Plant Modeling system (VLPM) which is a visual language framework for plant modeling using L-Systems. The user-friendly visual language framework is to cater those users with no prior knowledge in both programming and L-Systems. The user can easily generate plant model by using the framework which consists of the L-System attributes and grammars that are represented by icons. As for LMUSe [2], it is a complete and sophisticated system that is compatible with L-System grammars specifically for music rendering. However, music cannot be rendered if one does not know the L-System grammar that is supposed to instruct the direction, movement, operations of the stack, increment and decrement of musical pitch, and tempo. In LMUSe, as proposed by Worth and Stepney [6], it is possible to produce more 'pleasing' or harmonious musical sounds by using stochastic and context-sensitive grammars. However, it requires the users to key in the L-System grammar and to have extensive knowledge in L-System. In this paper, we propose a visual language framework for creating more harmonious musical sounds that can easily be used by the users who have little or no knowledge on L-Systems. By chance, VLMR integrates parts of VLPM with pieces from LMUSe since VLPM provides iconic representations which match the musical notes with the attributes of plant models [5].

2 Proposed Method

In the visual programming domain, the emphasis of the research depends on the application of visual formalism. From the view of programming it is regarded to be more effective than textual formalism. The overall implementation approach is designed based on the data flow programming model. The usage of data flow model has a few advantages. It results in easier debugging process because it allows immediate access to the program state. Besides, data flow model also supports automatic parallelization which is vital for music rendering. As the processes encountered in the implementation of VLMR involve concurrent processing scenarios, the data flow model is well suited in this case. The data flow model can run repetitively for the next event and this model starts with the string input as shown in Figure 1.

In the method, firstly, the user needs to input the rules (lower and upper scales) using the visual language framework by selecting, and dragging and dropping the icons on the window for defining the rules. The selected icons will be read by the L-System editor that runs behind the interface. The editor then converts the rules by matching them with the L-system grammars and stored the converted rules in an array. As the rules are too short to generate harmonious musical sound, the rules can be mutated using the stochastic L-System grammar. Rewriting the converted rules (mutation) then takes place (if required) so that a longer piece of rules is generated [7][8][9]. The mutated rules are also able to create a plant model. The L-System editor will then pass the rules to the string generator. Then, the string generator which follows the context-sensitive L-System grammar, outputs (interprets) the production rules based on the mutated rules [10]. Finally, harmonious musical sound can be played with the aid of MIDI library [11].

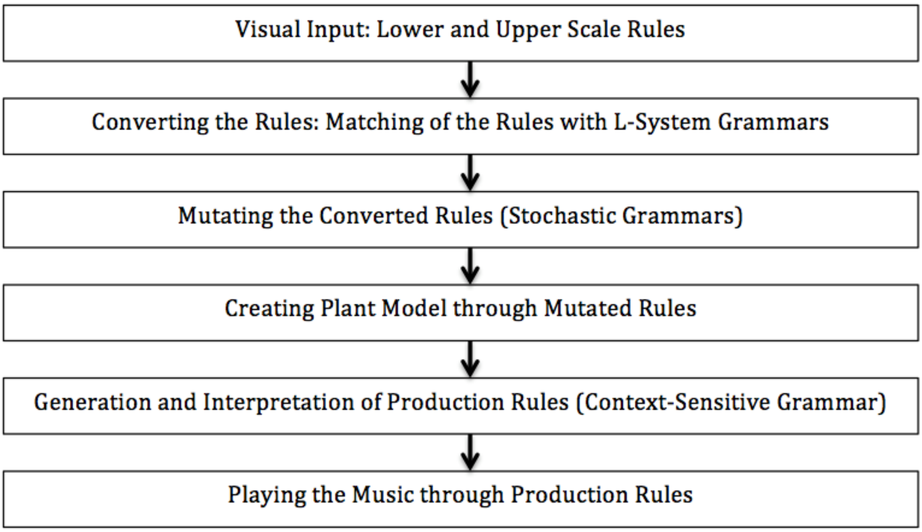

Fig. 1. The Overall Method: Data Flow in VLMR

3 Implementation

The visual language framework was developed using the code that was taken from a functioning Java L-System application. The software is available for public use and so is LMUSe for music rendering [12]. The functioning Java L-System application is created to allow users to draw trees easily in a visual environment [13]. This system is written in Java [14] and the development toolkit used is Eclipse Galileo IDE. The system overview of the architecture of VLMR is shown in Figure 2. The implementation code of Mutator.java where the stochastic grammar is used to mutate the rules is shown in Figure 3 while the implementation code of Production.java where the context-sensitive grammar is used to generate a variety of production rules is shown in Figure 4.

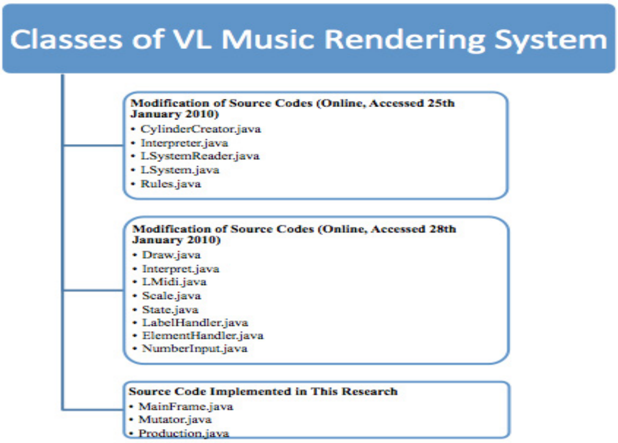

Fig. 2. Classes of VLMR

```
private void makeBTNActionPerformed (java.awt.event.ActionEventevt) {
if (count == 1) {       String backup = convertedrumeATXT.getText ();
                        Vector rulesVector = textToRules(); //mutate process
                        Mutatormutator = new Mutator();
                        rulesVector = mutator.do_mutation(rulesVector,0,1); //one-point swap
                        rulesToText(rulesVector); }
convertedruleATXT.setText(backup);
count++;}
```

Fig. 3. Example of How Mutation is Performed (Stochastic Grammar)

```
private void makeBTNActionPerformed (java.awt.event.ActionEventevt) {
        InString = MainFrame.axio;
        Production p = new Production(inString,3,"[]<>"); //Recursive
        intnumrecursionsdone = 0;
        numrecursionsdone += p.run();
        p.interruptime = true;
        p.run();
        outString = p.getProduction();
        productionruleATXT.setText(outString);}
```

Fig. 4. Example of How Production Rules are Obtained (Context-Sensitive Grammar)

Based on L-System, the plant modeling can then be interpreted for music rendering [6] through the interpretation of the L-Systems grammars. A snapshot of the VLMR system is shown in Figure 5. The visual language framework is implemented with musical icons that can be dragged and dropped and is written in Java [14] using Netbeans IDE in order to integrate with the context-sensitive and stochastic L-Systems grammars that were taken partially from LMUSe system. As shown in Figure 5, the VLMR system consists of the Rules (Upper Scale and Lower Scale), Music Toolbox (Key, Transpose Up, Transpose Down, Play, Posh and Pop), Tree's Attribute (Angle, Left Branch's Size and Right Branch's Size), Converted Rules and Production Rules. The rest of the method is implemented under Functions Menu of the interface namely ConvertRule (Converting the Rules: Matching of the rules with L-System Grammars), Mutate (Mutating the Converted Rules (Stochastic Grammars)), Make (Creating Plant Model through Mutated Rules), Interpret (Generation and Interpretation of Production Rules (Context-Sensitive Grammar)) and Play (Playing the Music through the Production Rules). The music can be easily rendered by clicking 'Play' button on VLMR as shown in Figure 5.

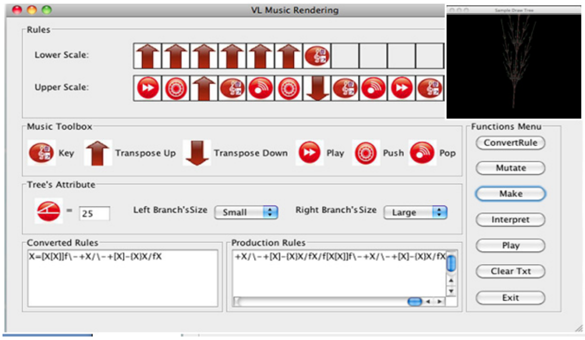

Fig. 5. A Snapshot of the VLMR System

4 Evaluation and Discussion

A survey together with a set of experiments were carried out to determine the acceptability and the quality of the techniques and methods derived in this research. The respondents were required to listen to the musical sounds rendered by LMUSe as well as VLMR and then make comparison between both sounds based on the level of harmony. The harmony of the musical sound is defined as the complication of note played per tab on the instrument. Ten respondents who are experts in music as well as ten respondents who are non-experts in music who mainly originate from the School of Computer Sciences and the School of the Arts, USM were asked to complete the questionnaires. In addition, a total of twenty respondents who are neither experts in L-System nor have any experience with L-System were also surveyed subjectively on the ease of using both systems.

Table 1 shows the results of the evaluation of LMUSe and VLMR by the respondents. The rating is based on a scale of 1 to 5 where 5 is the highest. The mean rating is calculated as follows:

$$\text{Mean Rating} = \frac{\text{Rating of Respondent 1+......+ Rating of Respondent 10}}{\text{Number of Respondents}} \quad (10)$$

For the level of harmony of the musical sounds generated, the musicians and non-musicians have rated 3.7 and 3.1 respectively for LMUSe, and 4.3 and 4.8 respectively for VLMR. On the ease of generating the harmonious musical sounds, the musicians and non-musicians have rated 2.5 of and 1.7 respectively for LMUSe, and 4.5 and 4.2 respectively for VLMR. The 20 respondents who are neither expert in L-Systems nor have any experience with L-Systems also wrote down some feedbacks on their opinion about the systems.

Table 1. Result on the Evaluation of LMUSe and VLMR

Question	System	Mean Rating	
		Musicians	Non-Musicians
Level of Harmony in the Musical Sounds	**LMUSe**	3.7	3.1
	VLMR	4.3	4.8
Ease of Generating Harmonious Musical Sounds	**LMUSe**	2.5	1.7
	VLMR	4.5	4.2

From the results of the objective and subjective evaluations, some useful and effective feedbacks were obtained. First of all, the respondents think that the hybrid method of stochastic and context-sensitive grammars in VLMR has significantly enhanced the level of harmony in the musical sounds rendered compared to LMUSe. Secondly, VLMR is easier to use in generating harmonious musical sounds compared to LMUSe. Thus VLMR has a great potential as an edutainment tool for both L-Systems and basic music learners.

5 Conclusion and Future Work

As compared to the previous research on L-Systems music rendering, this research has provided a variety of L-Systems grammars by combining stochastic and context-sensitive L-Systems. Firstly, the hybrid models allows rendering of a more harmonious musical sound for music rendering. Secondly, it is found that VLMR is easy to use in producing harmonious musical sounds. Thus, the visual language framework used in VLMR is well-suited to allow non-experts to understand, learn and use L-Systems in particular for L-System music rendering [15].

In order to further improve the method, the use of accurate deterministic and non-deterministic L-Systems should be investigated.

Acknowledgments. This work presented is supported by USM Short Term Grant No. P3710 and scholarship from Mini Budget 2009 for Malaysian researchers.

References

1. Lim, C.K., Talib, A.Z.: A Visual Language Framework for Music Rendering Using L-System. In: Proceedings of the 3rd WSEAS International Conference on Visualisation, Imaging and Simulation (VIS 2010), Portugal, pp. 47–52 (2010)

2. Sharp, D.: LMUSe, Available from World Wide Web, http://www.geocities.com/Athens/Academy/8764/lmuse/lmuse.html (accessed on January 28, 2010)
3. Stepney, S., Beaumont, D.: Grammatical Evolution of L-Systems. In: Proceedings of 11th Conference on Congress on Evolutionary Computation, Trondheim, Norway, pp. 2446–2453 (2009)
4. McCormack, J.: Grammar Based Music Composition. In: Stocker, et al. (eds.) Complex Systems 1996: From Local Interactions to Global Phenomena, pp. 321–336. IOS Press (1996)
5. Hwa Siew, B., Zawawi Talib, A.: Visual Language Framework for Plant Modeling Using L-System. In: Badioze Zaman, H., Robinson, P., Petrou, M., Olivier, P., Schröder, H., Shih, T.K. (eds.) IVIC 2009. LNCS, vol. 5857, pp. 696–707. Springer, Heidelberg (2009)
6. Worth, P., Stepney, S.: Growing Music: Musical Interpretations of L-Systems. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 545–550. Springer, Heidelberg (2005)
7. Majherová, J.: Virtual Plants in High School Informatics. In: Conference (ICL 2007), Villach, Austria, pp. 1(7)–7(7) (2007)
8. Pradal, C., Dufour-Kowalski, S., Boudon, F., Fournier, C., Godin, C.: OpenAlea: A Visual Programming and Component-Based Software Platform for Plant Modelling. Functional Plant Biology 35(10), 751–760 (2008)
9. Prusinkiewicz, P., Lindenmayer, A.: The Algorithmic Beauty of Plants. Springer-Verlag New York Inc. (1990)
10. Sato, K.: Design Information Framework, Context Sensitive Design and Human-Centered Interactive Systems. In: Conference on Human Factors in Computing Systems, Vienna, Austria, pp. 1588–1589 (2004)
11. Bresson, J.: OpenMusic MIDI Documentation. Ircam Software Documentation (2004)
12. Teresi, S.: Scott Teresi's Website, Available from World Wide Web, http://teresi.us/html/main/programming.html (accessed on January 25, 2010)
13. Ijiri, T., Owada, S., Igarashi, T.: The Sketch L-System: Global Control of Tree Modeling Using Free-Form Strokes. In: Butz, A., Fisher, B., Krüger, A., Olivier, P. (eds.) SG 2006. LNCS, vol. 4073, pp. 138–146. Springer, Heidelberg (2006)
14. Java Technology, Available from World Wide Web, http://www.sun.com/java (accessed on January 20, 2010)
15. Menzies, T.: Evaluation issues for Visual Programming Language. Handbook of Software Engineering and Knowledge Engineering 2, 93–101 (2000)
16. Bruno, F., Jose, C.L., Marcio, C.P.: L-Systems, Scores, and Evolutionary Techniques. In: Proceedings of 6th Sound and Music Computing Conference, Portugal, pp. 113–118 (2009)
17. Prusinkiewicz, P.: Score generation with L–systems. In: Proceedings of the International Computer Music Conference, Den Haag, Netherlands, pp. 455–457 (1986)
18. Bresson, J., Agon, C.: Sound Writing and Representation in a Visual Programming Framework. In: DMNR 2006 Doctoral Research Conference, Digital Music Research Network, Goldsmiths College, University of London, United Kingdom (2006)