Introduction to Generative Musical Morphing

Dr. Umberto Roncoroni Osio
Universidad de Lima, Lima, Perú.
www.digitalpoiesis.org
hroncoro@ulima.edu.pe

Abstract

This paper is about musical morphing, a technique that is still waiting to develop its full potential. The goal of its generative upgrading is to engender new music as the pairing of musical fathers and mothers. To do this, generative grammars and especially L-systems are used, since they store information in strings like DNA does. First, will be introduced some basic concepts. In the second section will be discussed some technical problems, since interpolations between pitches or durations using MIDI codes are not mathematically precise and cause unpredictable behaviors and the algorithm must consider harmony, rhythm, counterpoint and different musical forms. The last section of this paper, on the basis of Mole's theory of aesthetic information, will describe an experimental morphing architecture and implementation, considering its levels of complexity. The conclusions will stress the importance of interface design to improve the L-systems musical notation for the efficiency of the process and of the human machine interaction.

1. Introduction

Remixing and sampling are popular methodologies in audiovisual artistic practice. Even if DAWs and plugins provide an incredible variety of functionality, musicians and sound artists are always seeking new ways to expand their creative and generative capabilities [1] [2]. Morphing is a well know procedure to create 2D/3D forms by the interpolation of geometrical data using different weights and parameters. In the case of music, with the exception of a few solutions to solve specific compositional problems or to make real-time transitions between tracks in videogame levels, morphing as creative remixing technique is still unexplored [3]. In this paper, using generative grammars, I will suggest some experimental morphing methods, including programmable L-systems, to generate samples, clips, and maybe even entire new songs. The result is actually intended as a starting point for further research and experimentation.

2. Morphing in 3D animation

Morphing is a well-known technique to create objects and animations. In figure 1 and figure 2 are shown some 3D models created using different morphing processes.

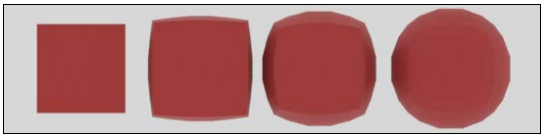


Figure 1. Morphing of a cube and a sphere with the 3DMax. Image of the author

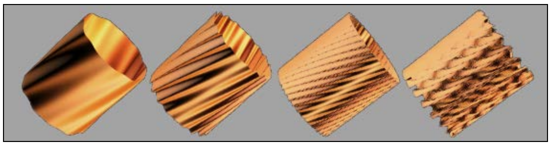


Figure 2. Using morphing to create 3D textures with generative software. Software and images of the author

3. Morphing with shape grammars and L-Systems

L-systems are recursive substitution processes that use a set of symbols and rules [4]. Symbols can represent complex audiovisual objects, geometric information and transformations like scale and rotate. L-systems are widely used to model natural forms, fractals and complex modular objects. In fact, L-systems can use different types of rules and functions to simulate the serendipity of natural forms (figure 3).

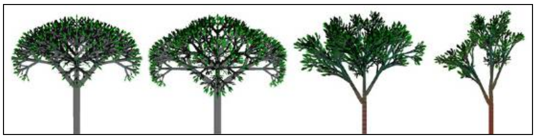


Figure 3. Tree models made with stochastic and parametric L-systems. Software and image of the author

Morphing can be easily implemented using L-systems and shape grammars in general, because the symbolic representation of objects using alphanumeric strings makes simple to interpolate and morph single objects and groups of hundreds or thousands objects as well. And it is possible to apply the morphing calculations to geometry, transformations, to single objects or groups in the scene, using different parameters and hierarchy structures. Stochastic and parametric L-Systems help to improve the music generation [5, 6] and the morphing process in many ways.

It is important to note that the interpolation is not about numbers, but rules, since L-systems' rules and strings of alphanumeric symbols share a common syntactic form that can be edited using other L-systems. Thus, the grammar of the L-system is the key to add the desired generative bias to the morphing algorithm.

Figure 4 shows the result of a morphing process using L-systems. The symbols and the rules that generate the blue spiral are interpolated with the symbols and the rules that generates the red stair like line. The morphing process creates a new grammar which rules and symbols generates the red and blue form of the right. This form combines the grammars of both parents and the morphing L-system process is controlled by rules and symbols of another L-System. A great advantage of the L-systems notation is its similarity to DNA (the molecule that holds the instructions for making all proteins), as both are strings of information that are combined in different ways.

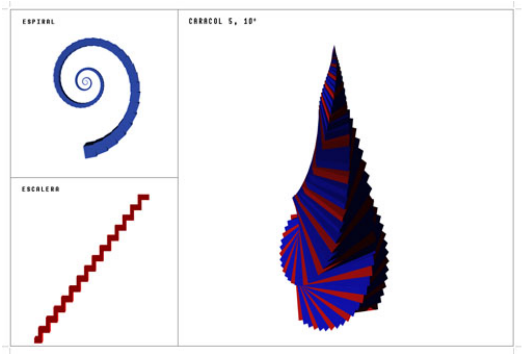


Figure 4. Morphing of 3D forms using rules interpolations. Software and image of the author

4. Musical morphing

Morphing is not usually used in music, with the exception of some specific tasks, like transitions of tracks between levels of a video game [3]. But the goal of generative musical morphing is the creation of clips, samples and entire scores (like EDM tracks) into another track or piece of music, considering morphing as a new kind of musical instrument.

The first approach is numerical, using MIDI codes for pitch, velocity and duration, and MIDI controllers to edit filters, the envelope, etcetera.

The second approach is with L-systems, since grammars make easy, using Mole's terminology [7], to represent "sonic objects" (pitch, loudness, duration) as well as "sonic cells" (measures or phrases) and to edit and interpolate their parameters. For optimal results, the two approaches can be combined. In the following section the fundamentals of generative music morphing will be explained.

4.1 Basic morphing interpolations

In any case, morphing can be done in three ways (figure 5). The first is to interpolate pitch and duration of the notes of score 1 with the notes of score 2, using MIDI code; I called this method "average mode" or "chemical reaction mode" (figure 6). The second is to alternate tones of 1 score with the tones of the second score; I called this method "alternate mode". It is possible to alternate single notes or group of notes in any order that you may declare in the rules. In figure 6 is shown some cases of average mode morphing interpolations.

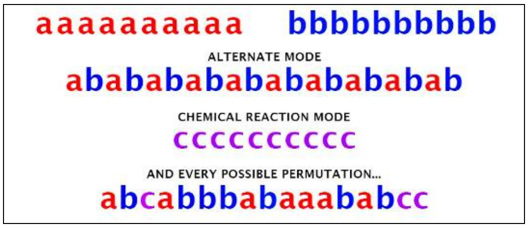


Figure 5. The musical morphing basic algorithm modes.

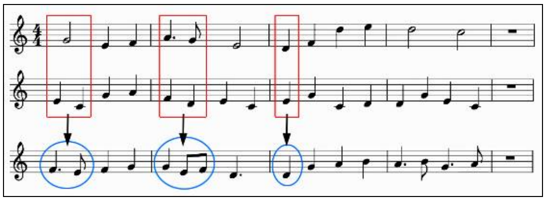


Figure 6. Morphing of pitches and durations with average mode. For instance, the G of the first measure (pitch 67) is added to the E (64) of the second score, resulting in F (65). The same with durations. Note that there are different valid solutions, even leaving the pitch unchanged.

The morphing process can be repeated many times using different parameters and re-morphing the morphing results, for instance, to create accompaniment or more voices. A sort of emergent melody and harmony may appear, as in the fourth sample provided with this paper.

Parents samples Average and alternate morphing Multiple morphing

Arpeggio.wav Glissando.wav

MorphGA1.wav MorphGA2.wav

GenerativeMorph01.wav

4.2 Advanced morphing interpolations

Theoretically, the morphing calculations are straightforward, but in the real world it is necessary to consider a lot of exceptions that make the process a lot more complicated than it initially appeared.

4.2.1 Pitch and rhythm calculations

In the first place, the average of pitches and durations is not always mathematically correct. In fact, we must take into consideration that MIDI numbers are integers and other constraints such as scales and rhythm. This is easily understood in the following example:

Pitch 1 = 60 (C)
Pitch 2 = 65 (F)
Morph pitch = (60 + 65)/2 = 64, 62 or 63?
63 is not in the scale of C major.

The same happens with durations, if you want to respect the signature and rhythm. Thus, the morphing calculation parameters depend on many possible factors: harmony, counterpoint, movement or the actual chord progression. Eventually, computation could consider the neighborhood of any particular note. In this way the morphing process evolves like a cellular automata.

4.2.2 Score calculations

Now, the morphing process can be computed easily if scores match their number of tones, measures and length. But this only happens if you create your scores from scratch and in the right way (matching the abovementioned numbers). Using scores of other composers, we have to consider differences in notes, durations, scales and length.

Different problems arise considering morphing transformations of sonic cells (Moles, 1968), like measures, periods, phrases, genres (EDM, Gregorian chants, jazz…) and the score musical form (Iterative and reverting types, strophic types, etcetera). This can be very complicated if we are trying to morph music of different origins and traditions.

Another problem is the existence of other factors like portamento and expression, and also sound design. Here synthesizer come into hand. We can morph musical instruments, envelopes and filters using MIDI messages. Many synthesizers provide enough MIDI implementation to do this in real time. This is also possible using DAW and digital synthesizers, like Puredata in Ableton Live.

And finally, there is the problem of morphing different voices, for instance, with piano scores. The main difficulty is to match the morphing of the main voice (for instance, the right hand score) with the accompaniment (the left hand, or another instrument). This must take into account the morphing of the first voice or right hand.

Hearing the following samples clearly confirms the difficulty of this task.

Parents' samples

Alternate morphing

Bach.wav

Boccherini.wav

BoccheriniBachAI.wav

5. Developing the morphing process

Generally speaking, when experimenting with code, it is a good practice to keep things as simple as possible. In the morphing's case, a practical solution is to design the morphing process like an onion skin, in other words, using different layers and the hierarchy of Moles [5]. These are the super cell layer (genre, musical form, and structure), the sonic cell layer (periods, phrases, etc.) and the sonic object layer (pitch, velocity and durations). The top layer (genre, structure, etc.) provides values to the parameters of the inner layers, the last being pitch, velocity and duration calculations. Considering differences between scores, it is also necessary to split the morphing calculations in three steps.

The first is preprocessing, to adjust properties and values of different scores, such as the number of measures, the scale and the number of notes among others. The second step is the morphing itself, using the preprocessed data. The final step is post processing or postproduction, to adjust errors, harmony, rhythm, expression, considering different voices, or the main instrument with its accompaniment (Figure 7).

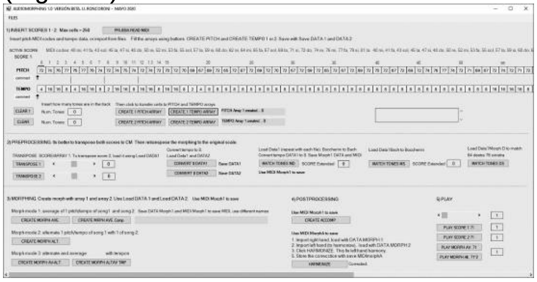


Figure 7. The MIDI morphing application layout. The first block of components lets insert the musical information of parent scores. The second block is to perform preprocessing operations, the third is the final morphing palette. Software and image of the author

6. Morphing with L-systems

The morphing calculations, as I have briefly explained, are not deterministic, since the result depends on rhythm, harmony or expression and their esthetic subjective interpretations. The point is that the software, to match individual styles, should provide the appropriate means to choose between different functions, options, parameters and values. In this sense, generative morphing could be implemented like a new musical digital instrument. But some actual limitations of L-systems' procedures make this task if not impossible, very difficult, as the next sections will explicate.

6.1 Improving L-systems algorithms

In the first place, standard L-systems do not provide enough control over the process, first if the user wants to create the score from scratch, and secondly, when it is necessary to modify the morphing interpolations and the properties of recursion, the main characteristic of L-systems. It is easier to understand one of these difficulties in the case of 3D forms. Consider the model of Figure 8: every row of the model has different quantities of bricks, and their relative positions are also different.

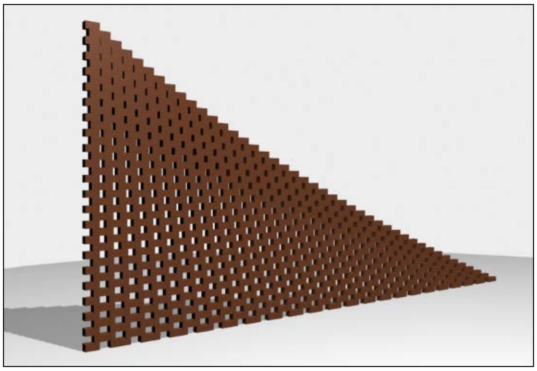


Figure 8. Easy to do with any programming language, impossible to do with a standard recursive substitution process. Software and image of the author

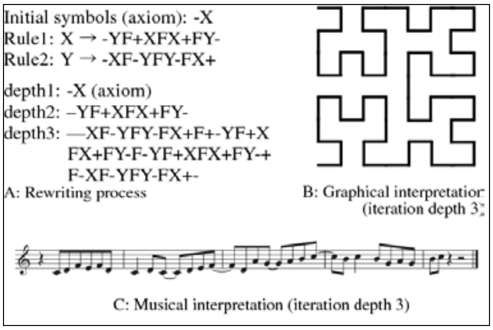
In fact, the values depend on the level of the row. In standard L-systems' grammars it will be necessary to create a rule for every brick and space for every row, which will make the grammar of the system too complex and the advantages of recursion to be lost. The same problems arise when manipulating musical information, for instance, to match velocity with duration, chords, with beats and so on.

You cannot solve this kind of problems even with timed, parametric or context L-Systems. So special programmable rules and symbols were developed and added to the standard L-System algorithms. To mention just one: *subL-systems*, which are full L-Systems (the children) inside another L-System (the parent). This way it is easy to build very complex modular objects made of objects that interact between them. A better description of these techniques is included in [9] and the software can be downloaded from http:www.digitalpoiesis.org.

6.2 Improving the interface

Now, symbols and rules can represent and compute musical and visual information simultaneously. This makes possible to morph colors with sounds, or to use images to morph music. The possibilities of creative experimentations are endless.

But, to take advantage of these possibilities, is needed a solution to cope with the complexity of information inside L-systems [10, 11] (figure 9). In fact, a full system must control and manipulate strings of thousands and thousands of symbols. In the case of music, this is too demanding, since every single piece of information counts for the overall beauty.


Initial symbols (axiom): -X
Rule1: X → -YF+XFX+FY-
Rule2: Y → -XF-YFY-FX+
depth1: -X (axiom)
depth2: –YF+XFX+FY-
depth3: —XF-YFY-FX+F+-YF+X
FX+FY-F-YF+XFX+FY-+
F-XF-YFY-FX+-
A: Rewriting process
B: Graphical interpretation (iteration depth 3)
C: Musical interpretation (iteration depth 3)
Figure 9. Musical L-Systems notation. Tsubasa and Kurosawa (2012)

For instance, the L-system musical notation should provide a decent interactive visualization of the score L-system symbolic representation, such as the position of the symbol in the scale and its tempo. The interface must provide interactive commands to read, play and edit the information of sonic objects (compare Figure 9 with figure 10).

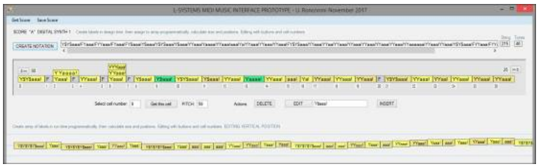


Figure 10. Sonic buttons can be clicked to read or play the pitch. Size represents duration, colors help to spot, for instance, tonic instances, pauses or strong and weak beats. The vertical position visualizes the ascending or descending movements. Software and image of the author

7. Musical morphing as a new instrument

The idea is to help the user create new scores with morphing algorithms in real time. The workflow should run in this way: first, the preprocessing computations. Then the instrument performs the morphing process using the input values of the user; this serves as a starting point. In the second step the user can edit and arrange the resulting score in real time, changing options and values on the fly. Controls are provided to edit the full score or any particular section, measure or note, for instance, to change the pitch, duration or both, for the entire measure, strong or weak beats.

By changing weights and other parameters, the user can create the music morphing step by step, in a natural and flexible way. In the final step the user can arrange the score to fix rhythm, harmony, tempos, since the options may give inconsistent tones that need to be fixed.

The editing and remixing section of the application will change the display accordingly to the actual task in progress (preprocessing, editing, or post processing). The playback section of the interface, to facilitate the production workflow, provides controls to play or loop the MIDI file like DAWS usually do, but adding options especially designed with the generative morphing in mind: loop changing modes automatically, loop changing only strong or weak beats or loop changing some specific part of the score. In any case the interface design process will be incremental, since useful options are discovered creating music and updating the layout on the go. Figure 11 and 12 show the L-systems score generator and the prototype of the morphing application.

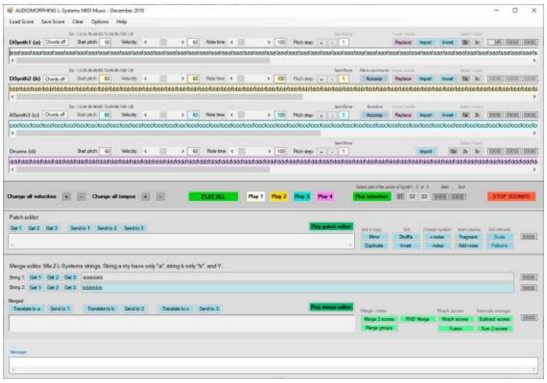


Figure 11. L-Systems generator interface. Scores can be saved as MIDI files and processed in the morphing module. Software of the author

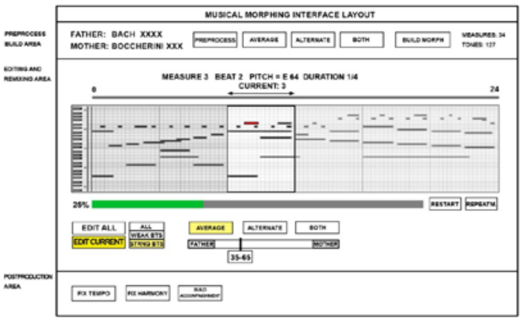


Fig. 12. Interface prototype for the morphing instrument. Design of the author

8. Conclusions

In this paper were described the basic concepts of musical morphing and instrument prototypes. The research is in its beginnings, there is clearly a lot of experimentation and work left to do that hopefully will be presented in the following conferences. So far, the following conclusion will share some insights and discussion topics that the research suggested:

a) The generative morphing process is an experimental instrument that will not always deliver beautiful music (experimental music in general is usually tough to hear…) , but it certainly lets extract interesting and unique clips, samples and patches to be used in standard remixing processes with DAWs. In this sense, it could be used as a sample generator plugin that expand the possibility of DAWs, somehow exhausted even considering the huge amount of products in the market [].

b) Morphing can be done without using L-systems, but the DNA metaphor makes the process simpler to design and implement, and more “generative” [12]. The modularity of L-systems grammars facilitates collaborative creative workflows. For instance, in the grammar can be combined rules of different authors, resulting in a sort of surrealist *exquisite corpse*.

c) The interface design is essential for morphing but also, but for generative art in general, because it makes the process transparent (rules and symbols are always in sight) and feasible for real time editing. The drawback is that the interface design and implementation need a lot of work, even more than the required by the morphing algorithm itself.

d) In this sense, generative grammars and morphing are instances of computational creativity that let discover and analyze in practice many concepts about creativity and Artificial Intelligence as originally posed by Boden [13]. For instance, generative versus combinatorial creativity. In comparison to neural networks technologies like GAN, generative grammars have the advantage of a better transparency and intelligibility [14] of the running processes.

e) From the artistic and educational point of view, it is very interesting to combine L-systems with and etnomathematics, like the digital musical yupana I presented in the last conference [15]. Musical morphing can be used to develop digital interculturality [16], since the morphing is about rules, and rules can embed natural computation and traditional creative techniques.

Bibliography

[1] Gunkel, D. 2016. *Of Remixology. Ethics and Aesthetics after Remix*. Cambridge: MIT Press.

[2] Carnovalini, F., and A. Rodà. 2020. “Computational Creativity and Music Generation Systems: An Introduction to the State of the Art”. *Frontiers in Artificial*

Intelligence 3: 14.

[3] Wooller, R. and Brown, A. (2005). “Investigating Morphing Algorithms for Generative Music”. Innocent, T. (Ed.) *Proceedings of Third Iteration.* Centre for Electronic Media Art, Australia, 189-198.

[4] Prusinkiewicz, P. 1986. “Score Generation with L-systems.” *Proceedings of the International Computer Music Conference*, pp 455- 457.

[5] Manousakis, S. 2009. "Non-Standard Sound Synthesis with L-Systems." *Leonardo Music Journal* 19: 85–94. Project MUSE muse.jhu.edu/article/363706.

[6] Rodrigues, A., E. Costa, A. Cardoso, P. Machado, and T. Cruz. (2016). *Evolving L-Systems with Musical Notes*. 9596. DOI: 10.1007/978-3-319-31008-4_13.

[7] Worth, P. and S. Stepney, S. 2005. “Growing Music: musical interpretations of L-Systems.” *EvoMUSART workshop*. EuroGP 2005, Lausanne, Switzerland, March 2005, 545-550.

[8] Moles, A. (1968). *Information Theory and Esthetic Perception*. Urbana: University of Illinois Press.

[9] Roncoroni, U. and Crousse, V. (2016). “Programming shape grammars and string substitution rewriting systems for generative art”. In Soddu, C. y E. Colabella (Ed.) *XIX Generative Art Conference Proceedings*. Rome: Argenia.

[10] Tsubasa, T and Kurosawa, K. (2012). *Automatic Melodic Grammar Generation for Polyphonic Music Using a Classifier System*. 9th Sound and Music Computing Conference (SMC2012), Copenhagen, Denmark.

[11] Tfirn, M. (2012). *The Musical Mapping of L-Systems*. Master thesis. Wesleyan University (Middletown, Connecticut).

[12] Karwaszewska, M. (2019). “Intermedial Score – Structural Filiations in the Context of Music-Literature Relations as well as Musical and Visual Relations”. In Soddu, C. y E. Colabella (Ed.) *XXII Generative Art Conference Proceedings*. Rome: Argenia.

[13] Boden, M. (2004). *The Creative Mind*. Nueva York: Routledge.

[14] Colton, S. (2008). “Creativity Versus the Perception of Creativity in Computational Systems”. *AAAI spring symposium: creative intelligent systems*, 2008 - aaai.org.

[15] Roncoroni, U. (2019). “Using the Inka's Calculator for Generative Art”. In Soddu, C. y E. Colabella (Ed.) *XXII Generative Art Conference Proceedings*. Rome: Argenia.

[16] Varma, R. (2006). “Making computer science minority friendly”. *Communications of the ACM*, 49(2) 129-134.