Instrumentalizing Synthesis Models

Lonce Wyse
Communications and New Media
National University of Singapore
www.anclab.org
lonce.wyse@nus.edu.sg

Nguyen Dinh Duy
School of Computing
National University of Singapore
di.dy211@gmail.com

ABSTRACT
An important part of building interactive sound models is designing the interface and control strategy. The multidimensional structure of the gestures natural for a musical or physical interface may have little obvious relationship to the parameters that a sound synthesis algorithm exposes for control. A common situation arises when there is a nonlinear synthesis technique for which a traditional instrumental interface with quasi-independent control of pitch and expression is desired. This paper presents a semi-automatic meta-modeling tool called the Instrumentalizer for embedding arbitrary synthesis algorithms in a control structure that exposes traditional instrument controls for pitch and expression.

Keywords
Musical interface, parameter mapping, expressive control.

1. INTRODUCTION
There are some characteristic features of affordances that are common to many traditional instruments used in music where notes are the primary form-bearing element. First and most obvious is the existence of a way to control the course pitch value. On a guitar it would be the choice of string and fret, on a wind instrument it is the fingering pattern. In addition, there is generally a finer control of pitch available through a variety of means such as stretching strings or adjusting mouth pressure on a reed. Finally, there are usually a few means of timbral control such as the location along a string used for plucking or bowing, or the combination of embouchure and air pressure applied to a wind instrument.

It is also typical of musical instruments that the timbral and pitch dimensions are not orthogonal is the space of controls. We see examples of one control mapping to multiple perceptual attributes in almost all instruments (e.g. the piano) where along with pitch, the spectral envelope of the sound changes as a function of the configuration of the primary pitch interface. Similarly, many of the traditional instrument interfaces associated with timbral control such as embouchure also affect pitch. In fact, the mappings between control and perceptual attributes are generally many-to-many. The particular relationship between pitch and dimensions of timbre under interface variation is a defining characteristic of individual instruments.

Synthetic sound models are not constrained by the physics and mechanics that give rise to such characteristic features of traditional musical instruments. A large class of existing linear synthesis models completely separate timbral from pitch control which is uncharacteristic of most physical sounds sources. On the other hand, there are also plenty of nonlinear models with relationships between pitch and timbre "built in", and which provide no simple parametric means of independently controlling pitch or timbre in desired or intuitive ways. Examples include the Chua oscillator [5] or a model of stiff string vibration whose equations provide no musically convenient pitch of frequency parameter. In order to domesticate these models for use in a traditional instrument form, we need to a) define the range of sounds the instruments will make from the range of possibilities defined by the synthesis algorithm, b) provide the performance interface with controls for course and fine pitch, and c) design the desired relationship between pitch and timbre that give a musical instrument much of its character and definition.

Here we present a computational tool consisting of a method and a graphical interface for working with almost any sound synthesis algorithm to create a new parameterization with characteristics typical of traditional musical instruments. The method utilizes aspects of "perceptual feature based synthesis methods" that analyze the output of a synthesis model and automatically map controls to synthesis parameters (e.g. Horner et al. [3]) and explicit mapping and interpolation (e.g. Bowler et al. [2]). In the two-layer mapping framework developed by Wanderley, Schnell, and Rovan [10], our instrument model parameter mapping corresponds to the "inner layer" wrapping the sound model, and is designed to facilitate the development of a second gesture-centric outer mapping layer from a physical instrument or system generating traditional musical control data (Figure 1).

2. MAPPING TECHNIQUE
Our goal is to reparameterize a synthesis algorithm to produce a new sound model that exposes the following low-dimensional parameter set that can be used as part of an expressive instrument in traditional tonal music contexts:

- chromatic pitch,
- pitch bend,
- expression (synthesis model dependent),
- transient behavior,
- gain.

The result is a new model that embeds the original synthesis model, mapping the above parameters in a generally many-to-many fashion [9] to the underlying synthesis model parameters.

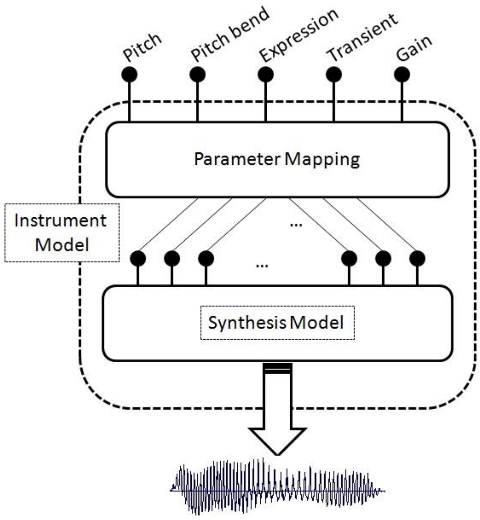

Figure 1. The Instrumentalizeer wraps a sound synthesis model to produce a new model with control parameters typical of traditional instruments. It supports the creation of the parameter mapping layer for musically navigating a subset of the space defined by the synthesis model parameters.

The mapping technique used to define the new parametric dimensions uses a familiar concept of interpolation between points in the space of the synthesis parameters, sometimes referred to as morphing because of the sonic transition that occurs during a parametric traversal between the end points[7]. If for a particular synthesis model, sound A can be specified by a vector of parameters P_A, and sound B can be specified by a vector of parameters P_B, then a "morph" of the sound is generated between A and B by interpolating the parameter sets between points P_A and P_B.

2.1 Defining Pitch Contour

To define the behaviour of the instrument under the target instrument parameterization, the designer first chooses two points from anywhere within the parameter space of the synthesis algorithm that produces pitches at the endpoints of the range desired for the new instrument. There will in general be many different parameter settings that produce a given pitch. If the synthesis algorithm exposes a large number of parameters the search space, as well as the subspaces of settings that produce the desired pitches, could be vast. Automating this part of the process would be possible, but would require additional model-dependent constraints in order to choose between all possible configurations yielding the same pitch. Furthermore, the choice of the these endpoints is the central design decision in the system, since it determines one of the defining characteristic of musical instruments as discussed earlier – how timbre changes with pitch. "Timbre" here means any possible perceptual attribute other than pitch (eg harmonic content, texture, stridence, hollowness, noisiness, etc).

Assuming that pitch changes smoothly and monotonically along a line connecting the endpoints P_A and P_B, then locations can be found for all the pitches of interest, (e.g. a chromatic scale) along the line. How this is done is the topic of the next section. Of course, the smooth and monotonic pitch change conditions do not hold between any two points in the entire parameter space for all synthesis algorithms, but they do hold in some regions of parameter space for a substantial number of them, including many nonlinear algorithms commonly used for sound synthesis such as stiff string models and chaotic dynamical systems. These are the types of synthesis algorithms we are particularly interested in instrumentalizing.

2.2 Identifying Pitches

Once the endpoints of a contour that will define the pitch (and timbral) range of the new instrument have been identified, the next task is to identify the pitch values along the line between the endpoints. For our stated traditional instrument goals, this means finding the points along the line that generate sounds corresponding to each of chromatic notes between the end values. The current system supports several methods for this task.

The method of choice uses automatic pitch assignments. First the line connecting the endpoints is divided into 10 steps per semitone separating the endpoints. An automatic pitch detection algorithm presented by Maher and Beauchamp [4] is then used to assign pitches to each of these points. Because the endpoint pitch values were identified manually and small steps are taken along the line connecting the endpoints, at each step we can constrain the automatic pitch detector to search and produce the best match it finds within a small neighbourhood of the previous pitch. This reduces the risk of octave and harmonic errors commonly produced by pitch detection algorithms.

Smoothness and monotonicity requirements are then checked for the data set and the designer is warned if they are not met. If they are met, then the data are used as the starting points for a finer search for the points which the automatic pitch algorithm assigns values corresponding to the chromatic pitch values between the endpoints. We currently use .06%, approximately a tenth of a semitone, as the threshold for determining a match.

The results of the automatic pitch assignments vary depending upon the sounds produced by the synthesis model and the pitch detection algorithm used. Of course, pitch is a psychophysical phenomenon that only sometimes corresponds to the fundamental frequency of a signal if one exists. Often synthesis algorithms produce sounds that are inharmonic or too noisy for an automatic pitch detector to work reliably at all. Because the automatic method can sometimes produce unusable results, the user can also choose to simply divide the pitch contour evenly into the number of segments corresponding to the number of chromatic steps between the endpoint pitches.

In both cases the parameter values automatically identified for the pitches can serve as a starting point that can then be refined manually by the model designer. The result of this process is a set of points along a straight line, the "pitch morphing line", through the parameter space of the synthesis model that represent a chromatic scale.

2.3 Expression

For the expression parameter, we seek a systematic way of varying the sound quality without necessarily having to change the pitch. Again making some mild local pitch continuity assumptions, we expect to be able to move off from any point along the pitch morphing line along an isopitch contour - a curve of constant pitch that intersects the morphing line at that pitch. As we move along an isopitch contour, the corresponding parameter changes will in general produce timbral changes in the sound produced by the synthesis algorithm. These timbral changes that occur along isopitch contours are what the "expression" parameter in the "instrumelntalized" model will control.

The way the instrument designer creates this expressive component is by defining a second pitch morphing line in the neighbourhood of the first with endpoints that produce pitches equal to the endpoints of the first pitch line. Again, there will in general be a large number of possible pitch-matched endpoints to choose from. The choice of the particular points is a design decision determined by the desired timbral characteristics of the instrument.

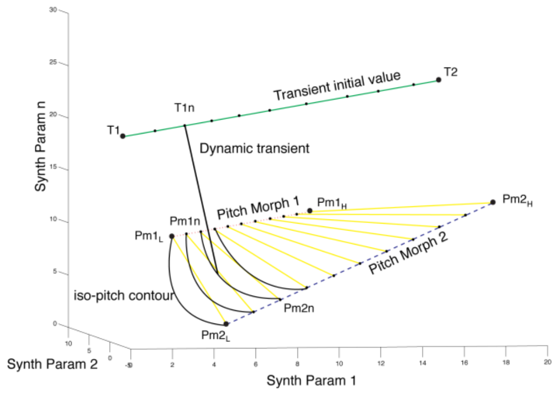

Figure 1. Two pitch morphing lines (Pitch Morph 1 and Pitch Morph2) determine a 2-dimensional submanifold (the "twisted plane") of the higher-dimensional parameter space of the synthesis model. Iso-pitch contours are curves (shown in black) are identified by searching the 2D submanifold. The Instrumentalizer also supports the definition of a line through parameter space where note transients will begin before gliding dynamically toward the pitch/expression location determined by the instrument performer.

Now two pitch lines through the possibly high-dimensional parameter space have been defined and the task of the system is to find iso-pitch contrours connecting any two corresponding pitch points from each line. The problem is that in the high-dimensional parameter space, there may be an infinite number of them.

We constrain the search to a particular two-dimensional submanifold defined by the straight lines that connect the corresponding pitch points along the two lines. Unfortunately, the straight lines connecting the two pitch morphers are unlikely to be the iso-pitch contours we seek, but our continuity assumptions guarantee that iso-pitch contours exist on any two-dimension submanifold defined by the pitch morphing lines. The "twisted plane" was chosen as a search space for its simplicity.

We divide the pitch/expression manifold into a fine grid of points and, in a way analogous to the search for pitches along the pitch line, derive a pitch for each point on the grid using the automatic pitch detection method. We then create a 2-dimensional array data structure to store the synthesizer model parameter values where one dimension corresponds to pitch and the other to expression. That is, each column (or row) corresponds to a single pitch (or expression value).

During performance, the pitch and expression parameters of the instrumentalized model then serve as floating point indexes into the 2D data structure. It is very fast to use the values to index the 4 closest stored parameter values, and then interpolate between them to get smooth changes as the pitch and expression controls change in real time.

The set of synthesis model parameters for sustained notes are thus constrained to exist and move along this two-dimensional manifold of points under the control of the new instrument pitch and expression parameters.

2.4 Transients

The first few tens of milliseconds following the onsent of a sound are commonly referred to as transients. Transients are an important characteristic feature for the perceptual identification of different instruments. They can also be manipulated to some extent by performers for a variety of expressive effects.

The technique of exploiting points and paths through synthesis algorithm parameter spaces we have developed for expressive pitch and timbral instrumental characteristics can be extended to transients. It will come as no surprise that the way the Insturmentalizer does this is to permit the designer to define a line with the choice of two endpoints in the parameter space of the underlying synthesis algorithm. The points on the transient line are used as starting points for a dynamic glide through synthesis model parameter space at the onset of notes. In the current implementation, there are no pitch values associated with points along the line. Only the new instrument's transient parameter, controlled for example by articulatory gestures, is used to determine the starting point for the parameter glide.

2.5 Performance

The result of using the Instrumentalizer design tool is a sound model that exposes instrument parameters for chromatic pitch, pitch bend, expression, transient characteristics, and gain. When a player initiates a note on a controller for the instrument model, the underlying synthesis parameter values are set to a location along the transient contour as determined by the instrument's transient parameter.

The synthesis model parameters then move along a line from the starting values to the point on the pitch/expression manifold over the course of time defined by the instrument designer. A continuum of different transients for any given note is thus determined by the performer's selection of the transient parameter.

The pitch and pitch bend parameters determine a pitch value, that together with the expression value, define a point on the 2D "twisted plane" in the synthesis model parameter space that is the target of the dynamic transient motion.

3. IMPLEMENTATION

The system is written in Java and functions as a Netbeans [6] wizard that generates the code for the new instrument that "wraps" the underlying synthesis model exposing only the new canonical instrument parameters. The synthesis models we have been using are also written in Java using the ASound library [11], though the technique does not depend on ASound capabilities in any way beyond using the API to play sounds, change parameters, and retrieve the generated audio for analysis.

The Instrumentalizer provides a graphical interface for exploring the underlying synthesis model, choosing the values that define the contours, controlling the operation of the pitch detector, generating tones for matching and identifying pitches for the sounds produced by the synthesis algorithm, and for the manual intervention that is sometimes necessary to tune the automated components such as, for example, by choosing the pitch detection algorithm.

4. SUMMARY AND FUTURE WORK

We have developed a technique for the design of instrumental interfaces to a wide range of sound synthesis models. Designers specify just four points in the parameter space of the synthesis model which define key instrumental characteristics such as how timbre changes with pitch and what timrbral dimensions are traversed for expressive variation. Two additional points are specified by designers to define a range of transient characteristics that can be controlled during performance.

A wide variety of sound models are amenable to the instrumentalization described here. Many instruments with substantially different characteristics are generally possible for any given sound model. Nonetheless, the process of constructing a new instrument exploits only the structure of the sound model itself to create the various expressive capabilities.

We are currently expanding the capabilities of the modeling tool with other generic components for traditional musical controls over arbitrary synthesis models in several ways. Currently the new instrument pitch parameter is linear in pitch, but other functions could also be interesting. For example, a "musical" choice might be a rounded staircase function that dwells around scale notes with level segments, and moves more quickly between notes [1]. Of course, this kind of mapping could also be left to another outer layer of mapping from controller to musical parameters.

The current Instrumentalizer is presented under the assumption that pitch will be the primary form-bearing dimension of the music played by the instrument. However, there is no reason why some other perceptual attribute could not be used instead of pitch, and the subspace identification and iso-attribute contours located in a way similar to what has been done here. Compared to the original sound synthesis model, the Instrumentalization process would still tend to reduce the number of parameters, restrict the space of sound traversed, orient the behaviour to a perceptual attribute rather than a synthesis parameter, and thereby add definition and expressivity to the resulting instrument.

5. ACKNOWLEDGMENTS

This work is supported by the Singapore-MIT GAMBIT program of the Media Development Authority in Singapore.

6. REFERENCES

[1] Arfib, D., Couturier, J.M., Kessous, L., Verfaille, V., Strategies of mapping between gesture control parameters and synthesis models parameters using perceptual spaces, *Organised Sound* 7(2), Cambridge University Press, 2002, 135-152.

[2] Bowler, I., Purvis, A., Mannin, P., and Bailey, N. On mapping N articulation onto M synthesizer-control parameters, in Proceedings of the International Computer Music Conference 1990.

[3] Horner, A., Beauchamp, J., and Haken, L. Machine Tongues XVI: Genetic algorithms and their application to FM matching synthesis, Computer Music Journal. (1993). 17(3).

[4] Maher, R., and Beauchamp, J., Frequency tracking of solo and duet passages using a harmonic two-way mismatch procedure, in *The Journal of the Acoustical Society of America*, 92:4, 1992, p.2429.

[5] Mayer-Kress, G. Choi, I. Weber, N. Barrger, R. Hubler, A. (1993). Musical signals from Chua's circuit in *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 40(10), 688-695

[6] Netbeans Integrated Development Environment http://www.netbeans.org.

[7] Pendharkar, C. Gurevich, M., and Wyse, L. Parameterized morphing as a mapping technique for sound synthesis. *Proc. of the 9th Int. Conf. on Digital Audio Effect.* Montreal, Canada. 2006, 1-6.

[8] Rodet, X. Models of musical instruments from Chua's circuit with time delay, in *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*. (1993). 40(10), 696-701

[9] Rovan, J. B., Wanderley, M. M., Dubnov, S., and Depalle, P. Instrumental gestural mapping strategies as expressivity determinants in computer music performance, in A. Camurri (ed.) *Kansei, The Technology of Emotion. Proc. of the AIMI Int. Workshop*, Genoa: Associazione di Informatica Musicale Italiana. 1997, pp. 68-73.

[10] M. Wanderley, N. Schnell, and J. Rovan, Escher - Modeling and Performing Composed Instruments in Real Time. In Proc. IEEE International Conference on Systems, Man and Cybernetics, 1998, 1040-1044.

[11] Wyse, L. A Sound Modeling and Synthesis System Designed for Maximum Usability, *Proceedings of the International Computer Music Conference*, Singapore, 2003.