Lose control, gain influence - Concepts for Metacontrol

Alberto de Campo
Berlin University of the Arts
decampo@udk-berlin.de

ABSTRACT

The ideas explored here are based on questioning some common assumptions in the usual conceptual models of hybrid (NIME-style) instrument design, and thinking and playing through the implications of these alternate strategies in theory and implementation.

They include: varying the mappings between controller input and changing them on the fly in performance, e.g. by gradually entangling or disentangling process parameters; recording instrument parameter state presets and control data (gesture) loops, and reusing them as flexible performance material; and creating networks of cross-influence between gestural input from multiple human players, other gestural sources, and multiple sound generating processes, which can be modified as part of the performance. In effect, this can be described as 'Lose Control, Gain Influence' (LCGI): gracefully relinquishing full control of the processes involved, in order to gain higher-order forms influence on their behavior.

These heuristics may lead both to finding non-obvious but interesting mapping strategies which can be built into more traditionally well-controlled instruments, and to new concepts for playing single-person instruments or multi-player instrument ensembles based on networks of influence. Many of these ideas can be played with using software libraries written in SuperCollider.

1. INTRODUCTION

1.1 The standard model

Simplifying greatly, one can say that the consensus model of NIME instrument design typically consists of (a) a human performer (b) her actions being sensed and converted to gestural data streams; (c) these streams being made meaningful by mapping them onto the synthesis parameters of (d) a generating process.

Following acoustic instrument design, creators of such instruments tend to spend much time on designing sophisticated sound processes and even more time on creating very finely tuned mappings, which ideally are both directly gratifying and fun to play when trying a new instrument, and allow for years of acquiring sophistication playing with them, i.e. refinability and eventually forms of virtuosity. Wessel and Wright argue strongly for this concept [1] (see Figure 1); a strong long-term example is Michel Waisvisz' The Hands, which has remained the same basic instrument concept for a very long time, in several incarnations, with long code/feature freezes to allow exploration of each particular version (personal communication, 2004).

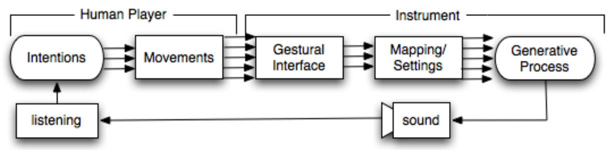

Figure 1. Model after Wessel and Wright.

1.2 Background and related work

Mapping strategies have been extensively studied in electronic music [2, 3, 4, 5], as has gestural control [6, 7, 1]. Research at STEIM has led Joel Ryan to deep considerations on electronic instrument design [8], and Michel Waisvisz to writing a near-manifesto on the artistic aspects of the design process [9]. Marc Leman has made the role of embodiment central for his perspective on music cognition [10]; and David Wessel and Newton Armstrong have extensively explored the idea of enaction in music [11, 12].

Joel Chadabe [13], considered conventional mapping as useful only for electronic instruments modeled on acoustic ones, and less so for interactive instruments. Citing Xenakis' image of the composer sailing through sonic space, he explicitly discusses performers' choices along a continuum of in/determinism of instrument behavior. His concludes that "[t]he primary benefit of an electronic instrument for a professional performer, which is that it extends the performer's capabilities in interesting, creative, and complex ways, requires an intermediary mechanism between gestural control and sound variable", and this paper tries to extend this in one particular direction.

One interesting extension of the standard model is the idea of navigating within (usually two- or three-dimensional) spaces of (multi-dimensional) parameter snapshots by means of maps and interpolation [14, 15]; this makes materialized experience with an instrument (the snapshots of parameter states) available as a performance resource. More recently, the notions of live coding [16] and just-in-time programming [17] have changed the possibility space for musical performance, and led to new definitions of what constitutes an instrument [18].

The MnM toolbox includes several interesting forms of dimensionality reduction, such as Principal Component Analysis [19]. Further related work concerns Neural Network approaches [20, 21], and manifold interfaces [22, 23].

The most directly related body of earlier work is by Dahlstedt and Nilsson[24, 25]: They have designed sophisticated control models for single-player/single-process instruments, using dynamic mappings and playing around sweets spots in parameter space as proposed here (and many other interesting features), and they played numerous performances with these instruments for several years.

Important sources of artistic inspiration for the proposed approach include David Tudor, especially his Neural Synthesis project [26]; Jessica Rylan's idiosyncratic instrument designs [27]; and Peter Blasser's instrument design work and underlying concepts [28].

Finally, the flexibility the SuperCollider language affords to performance, and implementing a number of concepts in SuperCollider quarks (extension libraries) has helped the author explore earlier incarnations of the ideas synthesized into a larger concept here [29, 30, 31] .

1.3 Possible deviations

Playing advocatus diaboli, one can argue that the standard model ignores what should strike us as the most obvious advantage of hybrid instruments: their body can be changed very quickly and substantially. This includes their software components and their complex states, with a wide diversity of prepared alternatives. While this violates traditional notions of instrument identity, it does open a huge possibility space which is well worth extended exploration. In fact, one can devise a contrasting concept: In the chain human > physical device > gestural data > mapping > output process, it is precisely the mapping process that is the easiest to expose to change during performance. This is certainly useful for experimenting with new instrument sketches, and given the risk affinity of dedicated improvisers, also in concert situations. As a performer, one can choose to put oneself into situations where a performance system will surprise not only the audience, but also oneself. In other words, we argue that in concert situations it may be preferable to lose control if this loss can be made meaningful by gaining influence. One line of reasoning here is that thinking of technical mental models distracts performers from listening to the music they create in the moment; relying on listening to gain intuitive understanding for how the instrument responds to one's actions will tend to create deeper immersion in the flow.

The MetaKtl software library for SuperCollider (which I have been developing) explores these notions in practice, and allows experimentation with widely varying degrees of losing control of and gaining influence on performance setups and hybrid instruments.

1.4 Simplifications

Note that in order to expose the problems of interest as clearly as possible, a number of simplifications underlie the following discussion:

- the instrument model is intentionally kept very simple
- some control paths can and should be mapped directly, bypassing the mediating layers proposed here
- discussion is intentionally limited to continuous numerical parameters

A conceptual and technical simplification is that in many of the software components described below, parameters are normalised to bipolar range [-1, 1], as this simplifies the notion of networks of influence compared to unipolar range: e.g. multiply by 0.5 will reduce influence of that signal on whatever it influences. This derives from analog modular systems where anything can go anywhere, and may generally be a useful approach in software as well, as its use in Waag's KeyWorx software shows.

2. PROPOSITIONS AND EXAMPLES

The following sections make propositions toward influence-oriented mapping strategies, and discuss them with examples that are part of the MetaKtl software library.

2.1 Mappings can be performance options

The first candidate for LCGI (Lose Control Gain Influence) is the actual numerical mapping in a technically trivial instrument. In the textbook case, each stream coming from one controller element sets one parameter of the sound process. The obvious items that can be tweaked here are the ranges of the sound control parameters, and which controller goes to which element(s). Experts often advise novices to introduce correlations between parameters, as this happens in physical instruments; the classic example being brightness rising along with loudness. This is a first step toward LCGI.

2.1.1 Influx

The Influx class extends this idea of correlations between parameters in several directions: Any number of named control parameters can be mapped onto any number of named process parameters, by having a matrix of weights for the amount of influence of each control parameter on each process parameter.

A trivial case would be map input x only to output a, y on b, and z on c (in effect, an identity matrix):

Table 1. 1-to-1 mapping of interface parameters to process parameters.

–	x	y	z
a	1	0	0
b	0	1	0
c	0	0	1

Influx provides several ways to create and modify such weight mappings, one can:

- design influence mappings by hand and store them as presets, such as the one above
- vary influence mappings by entanglement, i.e. adding or subtracting small random offsets to each weight
- vary influence mappings by disentanglement, i.e. by crossfading toward a known weight preset.
- create interpolated influence mappings by blending existing ones.
- modify weights of interest by code or GUI.

All of these leave instrument identity largely intact, as both the physical interaction device and its range of affordances and the sound process and its parametric possibility space remain the same. What changes is: with every new set of weights, a different subspace of the overall state becomes accessible, allowing one to find different sweet areas; more entangled mappings create more complex changes even with simple movements, which may be more interesting to play; having less technical understanding of the mapping may create better listening attention and better immersion in the playing process.

2.2 Memory on different time scales is useful

Performers develop their practice by acquiring varied performance experiences over time, and referring back to them in many subtle ways. Computers open up new options here: They allow keeping technically recorded memories of aspects of performances, such as recording sound behaviors of interest as process parameter states: *Presets*, and recording performance gestures as controller data: *Event Loops*. Both may be collected from earlier performances, or come from within a current one, and both are very interesting materials for reuse and modification in performance.

So far, Presets and Event Loops have been explored in more detail within the context described; one can easily imagine other notions of memorable aspects of performance that could be made storable and available in similar ways.

2.2.1 Presets

While playing with a hybrid instrument of the simple type described, one may hit sweet spots in parameter space; here it is useful to have very quick ways of saving these locations as a trajectory of timed snapshots. One can then return to them as reference points, and play relative to this known location. Influx allows shifting its output values such that the center of the influence space will correspond to the preset; movements away from center will diverge from it. As one can also scale control space size on the fly, one can zoom in on the area around the preset for more subtle variations of a parameter combination of interest. This idea has also been proposed by Dahlstedt[24, 25].

To support this line of thought, the ProxyPreset family of classes (in the SuperCollider library JITLibExtensions) allow blending presets, morphing between them by hand, or by automated crossfades, and various forms of creating random variations of existing presets. They also handle saving and loading presets to and from disk.

2.2.2 Event Loops

Conceptually, EventLoops can record any control data events as lists of key-value pairs, which contain absolute and relative event time, and any other named values describing the event technically and semantically. This generality allows capturing a great variety of event streams occurring in time, from recording algorithmically generated streams to capturing performance data coming from input devices, or from external sources.

The EventLoop class (in the MetaKtl quark) allows many playback modifications of these loops: they are scalable in tempo; quantizable to a given tempo and phase; segments can be selected from it; playback direction and a factor for scrambling local event order can be set. An EventLoop also keeps multiple recorded EventLists and can switch between them on the fly.

In performance, an EventLoop can e.g. be used to replace a live input stream (e.g. realtime-acquired HID data), then the loop can be reshaped while playing. It allows polyphonic layering by letting it loop and setting it on automutating mode so it keeps slowly shifting with each repetition. For loops of note-like events, it could easily be extended to allow algorithmic sequencing of multiple recorded lists.

The control data variant, KtlLoop, also allows on-the-fly rescaling of numerical control data in the recorded lists. The gesture can be scaled to larger or smaller ranges, and shifted by an offset in the parameter space. All these modification can quickly be accessed in performance, and the opportunities they create are quite distinct from audio loops.

2.3 Gaining influence may be worth losing control

From a second-order cybernetics point of view, systems need observers to exist, and observer neutrality and objectivity are not considered very useful [32]. In fact, radical constructivism [33, 34] would even contend that in order to understand a system, interacting heavily with it is the best option for an observer who thus becomes part of the system observed. From that perspective, observing one's own influence on a system seems clearly more interesting than aiming to exert full control.

The nodes in a network of influences should proceed quite politely about exchanging information, and then deciding what to make of them: Every node should be able to listen to input from many nodes, evaluate how much trust to put in which suggestion (from which source), and then decide what to do with the influences it received. And of course, every sending node should be allowed to try to influence many listening nodes simultaneously.

2.3.1 InfluxMix

InfluxMix implements this behavior by accepting influences from any number of sources, and storing them independently as e.g. ('y': ('srcA': 0.1, 'srcB': 0.2), 'x': ('srcA': 0.5, 'srcB': -0.3)), meaning for parameter y, srcA suggest a value of 0.1, srcB suggests 0.2, and similar for parameter x. Each source gets a trust factor (1 by default) which may be modified later to give that source more or less influence. The influences are weighted, summed, and scaled

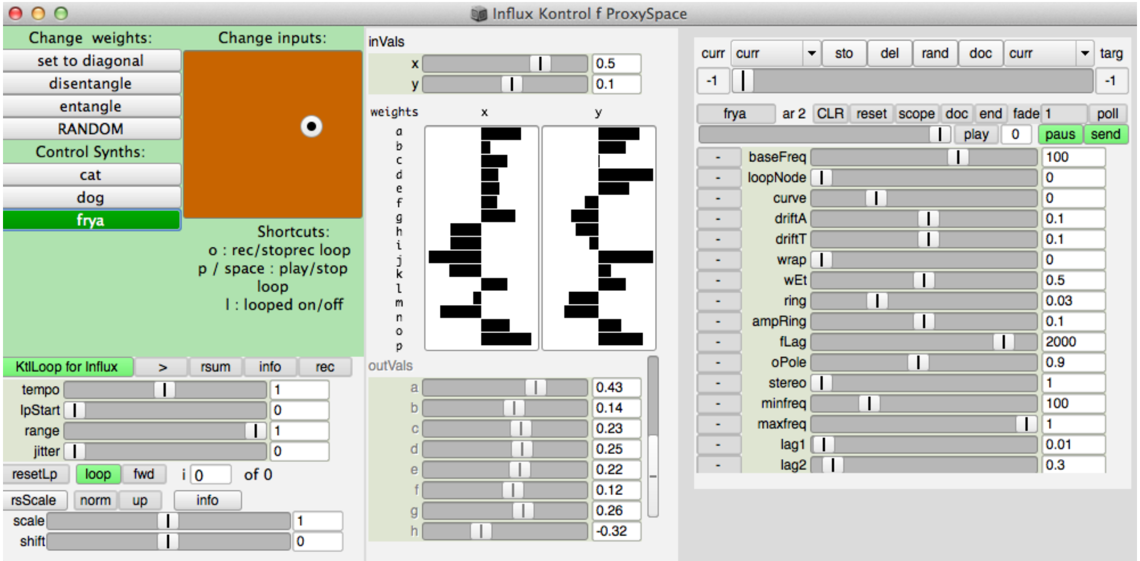

Figure 2. GUI with 2D slider + Influx, KtlLoop, and Preset controls

by a damping factor ranging from 0 (which means linear sum of all influences) to 0.5 (quasi equal power sum) to 1 (linear mean). Like Influx, the output can be shifted by a list of offsets corresponding to a preset, so it allows playing relative to known locations.

This strategy opens many possibilities: Several players could influence the same sound process together; one source can slowly move the center of the parameter space for other, more lively players. A combination of three players, three processes could play with varying degrees of source dis/entanglement. EventLoops or random sources could create small inner motion while a human player is playing with the same process at the same time.

2.3.2 *InfluxSpread*

InfluxSpread is the counterpart to InfluxMix; it handles sending influences to multiple listening destinations. Like all of its family, the configuration of these destination mappings is very flexible and can be changed very quickly in performance.

3. PRACTICE AND FUTURE WORK

3.1 Presets

The ProxyPreset classes have been used in many of the author's own projects, and in instruments made by, for and with other artists, to wide approval of its usefulness. Integrating frequent recording of presets and finding ways of fluidly playing with them is still to be sketched in code, and to explore as a performance strategy.

3.2 KtlLoop

Precursors of the KtlLoop classes have been used in the GamePad quark since 2005, and later in the KeyPlayer and other contexts. Its generalized redesign has opened a number of new possibilities, such as more fluid integration into performance instruments with fast switching between control and metacontrol levels. Similar concepts are studied by the Modality project [35].

3.3 Influx

Influx, being the newest metacontrol family member, has seen about two months of use by research colleagues, student participants and professional musicians. The example handed out to them is shown in Figure 2, and can map a 2D GUI slider to 3 synth processes with 6, 10 or 16 process parameters. With an Influx for variable mappings, a KtlLoop for gesture recording, and 3 destination processes it has a large possibility space already.

Users informally reported that the search for interesting zones is enjoyable and often pleasantly surprising, and that it is easy to find zones in the parameter space worth keeping as state snapshots. When listening to and observing people perform even with only a single 2d slider, the complex sound behavior induced by Influx is quite appealing and a far cry from telltale single slider/single parameter movements; and players appear quite absorbed in listening, so possibly the very opacity of the mapping does free players to listen more attentively to the changes their actions induce. Finally, networked multi-player influx setups introduce interesting layers of independent influence; informal experiments with such setups with student groups have been quite promising, and have been adopted by some students for their practice.

3.4 Next steps

Metacontrol as proposed here is an artistic strategy rather than a specific design alternative, so the best criterium for its relevance is simply the extent to which it gets adopted into the practice of professional performers. For this, the first step will be to create more example setups from very simple to very complex, play with them extensively in our

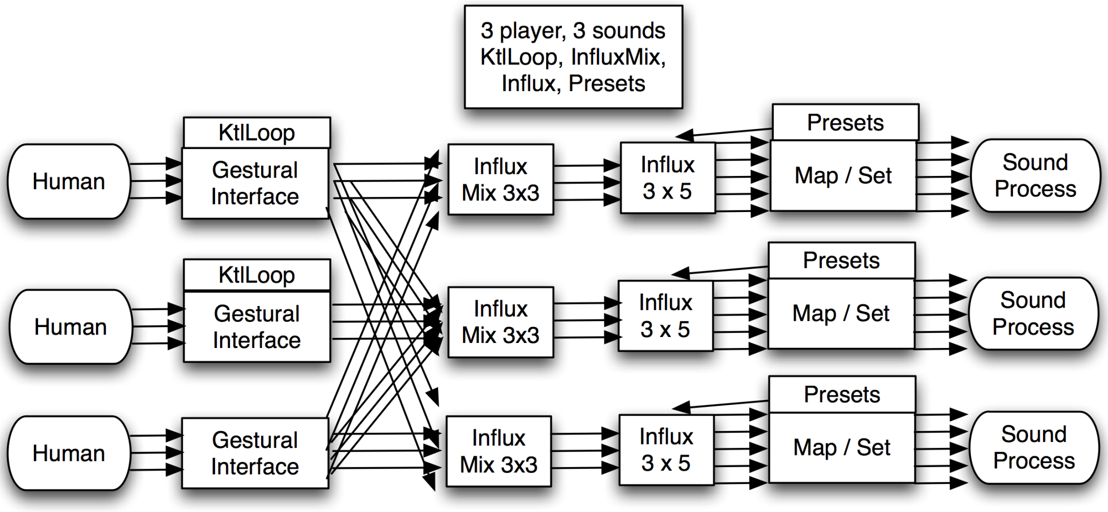

Figure 3. 3 players, 3 processes, Influx, KtlLoop, and Presets

own music projects, distribute them to our network of fellow musicians, and invite them to open feedback.

Secondly we aim to create working examples for the multi-person/single-instrument concept, in order to learn whether collaborative playing with influence-based concepts creates new playing experiences. Thirdly, multi-person multi-instrument setups remain to be explored, where networks of influence replace control on several layers. We expect that a setup of 3 players and 3 instruments as shown in Figure 3 will already allow gaining useful practical experience with and understanding of this approach.

Finally, in further work that will introduce a major conceptual step forward, response times and feedback will be generalised from an existing proof of concept to the entire system. New modules will have internal framerates at which changes in value are handled. This allows optional fade or lag times before arriving at a newly set value, intended latency before passing on responses to next inputs, and lockup-safe feedback paths in these networks of influence. One very nice variant of this has been written by associate researcher Dominik Hildebrand: an audio/control signal version of the Influx mechanisms, which allows control signal feedback paths, introducing quite complex behavior in simple network topologies.

4. ACKNOWLEDGMENTS

The author would like to thank Hannes Hoelzl for many years collaborating on these ideas, and experimentation with the implementations described here; Julian Rohrhuber for many years of collaborating on just-in-time programming and many related projects; students of Code Experiments at UdK Berlin for trying the prototypes; the members of the 3DMIN project at UdK and TU Berlin for essential discussions, and the Einstein Stiftung Berlin for supporting the 3DMIN project.

5. REFERENCES

[1] D. Wessel and M. Wright, "Problems and prospects for intimate musical control of computers," *Computer Music Journal*, vol. 26, no. 3, pp. 11–22, 2002.

[2] T. Winkler, "Making motion musical: Gesture mapping strategies for interactive computer music," in *ICMC Proceedings*, 1995, pp. 261–264.

[3] J. B. Rovan, M. M. Wanderley, S. Dubnov, and P. Depalle, "Instrumental gestural mapping strategies as expressivity determinants in computer music performance," in *Proceedings of the Kansei-The Technology of Emotion Workshop,(Genova-Italy)*. Citeseer, 1997.

[4] A. Hunt and R. Kirk, "Mapping strategies for musical performance," *Trends in Gestural Control of Music*, vol. 21, 2000.

[5] A. D. Hunt, M. Paradis, and M. Wanderley, "The importance of parameter mapping in electronic instrument design," *Journal of New Music Research*, vol. 32, no. 4, pp. 429–440, December 2003.

[6] C. Cadoz, M. M. Wanderley *et al.*, "Gesture-music," *Trends in gestural control of music*, vol. 12, p. 101, 2000.

[7] M. M. Wanderley, "Gestural control of music," in *International Workshop Human Supervision and Control in Engineering and Music*, 2001, pp. 632–644.

[8] J. Ryan, "Some Remarks on Musical Instrument Design at STEIM," *Contemporary Music Review*, vol. 6, no. 1, pp. 3–17. Also available online: *http://www.steim.org/steim/texts.phtml?id=3*, 1991.

[9] M. Waisvisz, "Trends in gestural control of music, chapter round table," *Ircam-Centre Pompidou*, 2000.

[10] M. Leman, *Embodied Music Cognition and Mediation Technology*. Cambridge, MA, USA: MIT Press, 2008.

[11] D. Wessel, “An Enactive Approach to Computer Music Performance,” in *Proc. of 'Rencontres Musicales Pluridisciplinaires'*, GRAME, Ed., Lyon, France, 2006. [Online]. Available: www.grame.fr/RMPD/RMPD2006/

[12] N. Armstrong, “An enactive approach to digital musical instrument design,” Ph.D. dissertation, Princeton University, 2006.

[13] J. Chadabe, “The limitations of mapping as a structural descriptive in electronic instruments,” in *Proceedings of the 2002 conference on New interfaces for musical expression*. National University of Singapore, 2002, pp. 1–5.

[14] J. Ryan, “Musical visualization - prospecting for new performance and composition tools for electronic music,” http://jr.home.xs4all.nl/MuViz.htm.

[15] R. Bencina, “The metasurface: applying natural neighbour interpolation to two-to-many mapping,” in *Proceedings of the 2005 conference on New interfaces for musical expression*. National University of Singapore, 2005, pp. 101–104.

[16] N. Collins, A. McLean, J. Rohrhuber, and A. Ward, “Live coding in laptop performance,” *Organised Sound*, vol. 8, no. 03, pp. 321–330, 2003.

[17] J. Rohrhuber, A. de Campo, and R. Wieser, “Algorithms Today - Notes on Language Design for Just In Time Programming,” in *Proceedings of the ICMC 2005, Barcelona*, 2005.

[18] T. Magnusson, “Designing constraints: Composing and performing with digital musical systems,” *Computer Music Journal*, vol. 34, no. 4, pp. 62–73, 2010.

[19] F. Bevilacqua, R. Müller, and N. Schnell, “Mnm: a max/msp mapping toolbox,” in *Proceedings of the 2005 conference on New interfaces for musical expression*. National University of Singapore, 2005, pp. 85–88.

[20] P. Modler, “Neural networks for mapping hand gestures to sound synthesis parameters,” *Trends in Gestural Control of Music, Ircam*, 2000.

[21] S. S. Fels and G. E. Hinton, “Glove-talkii-a neural-network interface which maps gestures to parallel formant speech synthesizer controls,” *Neural Networks, IEEE Transactions on*, vol. 9, no. 1, pp. 205–212, 1998.

[22] I. Choi, “A manifold interface for kinesthetic notation in high-dimensional systems,” *Trends in Gestural Control of Music. Battier and Wanderley, eds*, 2000.

[23] I. Choi, R. Bargar, and C. Goudeseune, “A manifold interface for a high dimensional control space,” in *Proceedings of the 1995 International Computer Music Conference*, 1995, pp. 385–392.

[24] P. Dahlstedt, “Dynamic mapping strategies for expressive synthesis performance and improvisation,” in *Computer Music Modeling and Retrieval. Genesis of Meaning in Sound and Music*. Springer, 2009, pp. 227–242.

[25] P. Dahlstedt and P. A. Nilsson, “Free flight in parameter space: A dynamic mapping strategy for expressive free impro,” in *Applications of Evolutionary Computing*. Springer, 2008, pp. 479–484.

[26] D. Tudor, “Neural synthesis - album notes,” http://www.lovely.com/albumnotes/notes1602.html, 1993.

[27] J. Rylan, “Flower electronics,” http://www.flowerelectronics.com.

[28] P. Blasser, “Ciat-lonbarde, home of post-70s electronics,” http://ciat-lonbarde.net.

[29] J. McCartney, “SuperCollider3,” http://supercollider.sourceforge.net, sourceforge, 2003-2007. [Online]. Available: http://supercollider.sourceforge.net

[30] ——, “Rethinking the computer music language: Supercollider,” *Computer Music Journal*, vol. 26, no. 4, pp. 61–68, 2002.

[31] C. Sattinger, D. Stowell *et al.*, “SuperCollider Quarks,” http://quarks.sourceforge.net, sourceforge, 2003-2014. [Online]. Available: http://supercollider.sourceforge.net

[32] H. v. Foerster, *Understanding understanding: essays on cybernetics and cognition*. Springer, 2003.

[33] E. v. Glasersfeld, “An introduction to radical constructivism,” *The invented reality*, pp. 17–40, 1984.

[34] H. v. Foerster and E. v. Glasersfeld, “Wie wir uns erfinden: Eine autobiographie des radikalen konstruktivismus,” *Carl-Auer-Systeme Verlag: Heidelberg*, 1999.

[35] J. Carey and M. e. a. Baalman, “Modality,” http://modality.bek.no.