TOWARD UNDERSTANDING HUMAN-COMPUTER INTERACTION IN COMPOSING THE INSTRUMENT

Rebecca Fiebrink[1], Daniel Trueman[2], Cameron Britt[2], Michelle Nagai[2], Konrad Kaczmarek[2], Michael Early[2], MR Daniel[2], Anne Hege[2], Perry Cook[1,2]

Princeton University
[1]Department of Computer Science, [2]Department of Music

ABSTRACT

A weekly seminar consisting of seven composers and one computer scientist was convened for the purpose of exploring questions surrounding how technology can support aspects of the computer music composition process. The composers were introduced to an existing interactive software system for creating new musical interfaces and compositions, which they used throughout the seminar. The group engaged in a user-centered design process to critically evaluate and improve this software. Through documentation of the experience and analysis of composers' responses to a questionnaire following the seminar, we achieved a richer understanding of how technology can support composers' modes of working and goals in the process of computer music interface design and composition. This work also resulted in an improved compositional software system and progress toward several new musical compositions and instruments.

1. INTRODUCTION

Computer music performance interfaces in which performer action is not coupled directly to sound reaction have been termed "composed instruments," as the specification of how the sound and performer interact is a creative and intentional act of composition [13]. Studying interaction in computer music therefore frequently focuses on broadly understanding and characterizing the practical and musical consequences of (already-)composed instruments' mediation of the performer-sound relationship.

Our interest, however, lies in the *process* by which composers design new instruments and performance-time interactions: the process of composing the instrument. How do composers engage with software tools to create new musical instruments and compositions? What information and interactions are valuable to them in this process? How do they formulate, revise, and achieve their creative goals? How might we create and improve tools for making this process more efficient or more satisfactory?

To begin to explore questions like these, we convened a weekly seminar of seven composers and one computer scientist. The composers, whose degree of computer programming expertise varied widely, were introduced to an existing software system, which they employed throughout the seminar to create new musical interfaces and compositions. The group engaged in a user-centered design process to critically evaluate and collaboratively improve the software. The experience was documented through notes of each meeting, emails on the seminar mailing list, and a written questionnaire that solicited each composer's critical reflections after ten weeks of meetings. Outcomes of this work include an improved compositional and improvisational software system, modified and greatly expanded using suggestions and observations from the seminar; and progress towards several new musical pieces and instruments.

In this paper, we draw on the experiences of the seven composers to come to an enriched understanding of how technology can support composers' modes of working and goals in the process of computer music interface design and composition. In particular, we observed the following interface affordances to be important to the compositional process: supporting iterative exploration, privileging the gesture/sound relationship, providing access to both surprise and control, facilitating creation of complex systems, inviting play, and accommodating users with diverse backgrounds, skills, and goals. Our experiences underscore the mutuality of influence between the human and computer in the act of computer music composition, highlight how the choice of computer instrument mapping strategy or algorithm presents important compositional interaction implications, and raise the question of how different human-computer interaction requirements might arise from compositional goals different from those primarily observed in this study.

2. BACKGROUND

2.1. Technology and the composition process

The study of computational support for creativity is a growing focus of research in human-computer interaction (HCI), which investigates questions of how technology can foster innovation and expression in domains from the arts to software design [14]. For example, recent work by Tsandilas et al. [15] employs an HCI approach in studying

a new digital pen-based score annotation tool for music composition. The authors studied five composers interacting with the tool, and they involved the composers in a participatory design process to iteratively improve the tool. Such work explicitly studying technological support of the process of musical composition is rare.

On the other hand, much more work focuses broadly on the topic of human-computer interaction in computer music performance. Indeed, in comparison to work with acoustic instruments, interaction in computer music offers an explosion of new possibilities regarding all aspects of the performer-sound interaction, as the relationship of gesture to sound must be intentionally composed [13].

One perspective for studying and supporting performance interaction involves he notion of a composed instrument incorporating a composed *mapping*—a function from input gesture to output sound that encodes the "essence" of an interface [9],[10]. Although the mapping paradigm of gestural interaction limits interactive possibilities to those most similar to an acoustic instrument [2], mapping provides a useful starting point for discussion and systems building. Several software systems including [1],[3],[18] have been constructed to support the process of creating (more-or-less) general-purpose mappings from gesture to sound.

As Drummond [4] writes, "Interactive music systems are of course not 'found objects'… Interactive systems require interaction to realize the compositional structures and potentials encoded in the system." Composers' writings offer valuable insights into the their practice in the process of *composing* the instrument [8]. A recent *Organised Sound* special issue [12] provides some excellent, broader discussion of the intersection of interaction and composition, including an analysis informed by the role of embodied cognition in interactive system design [16], and by an interrogation of the definition and scope of interaction in music [4]. In our work, we seek to make sense of the observed human-computer interactions of a group of composers engaging in creative practice, and in the process improve a specific software tool. Our collaborative, practice-based approach to studying instrument composition provides a complementary perspective to this past work.

2.2. Generative mappings and the Wekinator

Hunt and Wanderley [9] classify mapping strategies into *explicit* strategies that explicitly define the input-to-output relationships and *generative* strategies in which an algorithm such as a neural network generates the mapping function. At a high level, neural networks and other supervised learning algorithms generate a mapping function using a training dataset consisting of multiple (gesture, sound parameter) pairs. In training, the algorithm uses the training set to construct the mapping function, which is a model of the relationship between gestures and sounds in the training set. This model can then produce a sound in response to any new gesture. In reality, of course, the learning algorithm and model do not work on the gesture and sound themselves, but on the gesture as digitally represented by a vector of features (often gestural sensor values, such as accelerometer outputs or webcam pixel values, which may themselves be processed with smoothing, color tracking, etc.), and on the sound via numerical parameters to some synthesis algorithm or compositional system.

The Wekinator is an existing open-source software "meta-instrument… which allows musicians, composers, and new instrument designers to interactively train and modify many standard machine learning algorithms in real-time" [7]. Like ESCHER [18], it is a modular system that allows users to experimentally generate and use mappings between arbitrary input gestural controllers and arbitrary synthesis systems. However, a primary intention of the system is to support real-time creation and exploration of mappings, and to do so using a supervised learning paradigm: the user may iteratively create training examples of (gestural feature, sound parameter) pairs, train the algorithms, evaluate the resulting models via hands-on experimentation and objective accuracy metrics, modify the training examples and algorithms, and repeat. It differs most from other machine learning mapping tools such as [1] and [3] in this emphasis on real-time interaction within a single, general-purpose interface.

The Wekinator provides several built-in modules for extracting features from gestural inputs, including plug-and-play support for human interface (HID) devices. Users may also employ either ChucK or Open Sound Control-enabled environments to implement their own feature extractors and/or to use the parameters output by the trained models. The Wekinator supports the creation of composed instruments in the strict sense of [13], but it also allows for more complex performance-time interactions (e.g., parameters may affect an intelligent process, or the mapping may be dynamically modified at performance time, as in [7]). In this paper, we avoid a strict differentiation between composition, instrument building, and improvisation, and we use the term "mapping" simply to denote a function computed on input "features" to produce output "parameters" in real-time, without further assumptions regarding the source of the inputs or roles of the outputs within an interactive system.

3. METHODOLOGY

We held a seminar consisting of ten weekly, 2–3 hour meetings of 7 composers and 1 computer scientist (the organizer and first author). The Wekinator software was a focal point of the meetings and a foundation for the collaborative study: composers were asked to learn to use it, employ it however they found most interesting as a compositional tool, and collaborate with the computer

scientist in a user-centered design process [17] to critique, improve, and test iterative revisions of the Wekinator. The participation of the composers was not otherwise constrained, and they were not obligated to produce a composition, instrument, or other artifact, nor were they graded or otherwise compensated.

Participation was open and advertised to all composers at our institution. Of the 7 who chose to participate, 1 was a faculty member and 6 were graduate students; 4 were male, 3 were female. Their self-assessed programming expertise ranged from 1 (very low) to 4 (somewhat high) on a 5-point Likert-style scale (mean: 2.86). Six rated their prior machine learning knowledge as "very low", and the seventh rated it as "somewhat low." None had used machine learning or generative mapping strategies before in their compositions, for any purpose, though most had experience designing and using explicit mapping strategies.

All meetings involved group discussion and time for individuals to experiment with the Wekinator on their own laptops, during which they shared their creations and solicited feedback and help from the group. The computer scientist verbally solicited feature requests and bug reports, and invited group discussion of possible improvements. Each week, she implemented the requested functionality into the Wekinator (as was feasible) and distributed an updated version to the composers before the following meeting, the beginning of which was spent demonstrating the updated software. In all, 8 such design iterations were completed.

For each meeting, we recorded text minutes of the group's activities, discussion topics, and specific questions, problem reports, and feature requests regarding the software. The group communicated via an e-mail list, and composers sometimes e-mailed and met with the organizer directly. After ten meetings, each composer completed an individual questionnaire soliciting reflective feedback and demographic information using 20 multi-part free response and Likert-style questions. These artifacts documenting the experience of the seminar participants form the basis of the following analysis.

4. OBSERVATIONS

4.1. Composition of instruments and gestural control interfaces

After learning how to use the Wekinator, the composers chose to primarily employ it to create and explore new gestural control systems in which neural networks were employed to map from gestural features to sound synthesis parameters. With few exceptions, they focused on creating systems in which gestural state continuously drove real-valued parameters of sound synthesis algorithms, as opposed to systems incorporating recognition of and response to discrete gesture events or categories. Input devices used include the laptop accelerometer, the Mad Catz Real World Golf Game controller, custom-built microcontroller and sensor systems, Wacom tablets, and other HID devices. Composers also experimented with a variety of synthesis environments and algorithms in both ChucK and Max/MSP. The most-often used synthesis algorithms were two physical models of imaginary instruments, each with over 10 parameters, most of which had complex relationships with each other and the perceived sound. The next most popular methods were granular sampling and synthesis.

4.2. Hands-on exploration and experimentation

Shneiderman [14] discusses the importance of exploration of alternatives in the creative process across many disciplines; in fact, his first design principle for creativity support tools is "Support exploratory search." In our observations, composition and creation were indeed focused around interacting with the training set and training process to perform explorations of alternative mappings. One composer describes a typical approach: "I started by selecting exactly which variables I wanted to control on the synth side, and then created a few (two or three) presets that resulted in interesting sounds. I then trained the network various different ways to try and get the most expressive mapping of the input features." In the questionnaire, all composers discussed the importance of exploring and evaluating a variety of sounds, gestures and mappings. When asked how they evaluated whether they liked a mapping, everyone replied that they played with it in real-time, experimenting with making various gestures while listening to the sonic response. Furthermore, composers iteratively explored by training the network, evaluating the mapping by playing with it, modifying some property of the system, re-training and re-evaluating. The most useful methods that composers found for improving mappings were adding more examples to an existing training set and then retraining in order to augment or refine the current mapping (6 had tried; mean usefulness score 2.3 out of 3), deleting the training set and building a new training set from scratch (5 tried; mean 2.8), changing the input features or controller (5 tried; mean 2.4), and changing the synthesis method (4 tried; mean 2.4). Composers rarely or never added training data manually or modified the neural network algorithm, although these tasks were possible in the GUI.

4.3. What do composers value?

In this section, we draw on composers' feature requests, complaints, group discussions, and questionnaire responses to highlight recurring themes surrounding the interaction paradigms and interface qualities that composers most valued in the process of creating new instruments using the Wekinator. We also describe improvements made to the Wekinator through the user-centered design process.

4.3.1. Privileging the gesture-sound relationship via physicality and abstraction

When asked what they liked about creating mappings with the Wekinator and how they felt it was useful to their compositional practice, a common thread throughout composers' responses was the privileging of the relationship between gesture and sound, both in composition and performance. One obvious way in which the Wekinator emphasizes physicality is the fact that the user generates training examples for the neural network by actually performing gestures in real-time. Training examples can be created one gesture at a time: the user sets the desired sound parameters in the GUI (Wekinator records these as the target output values for the training examples, and sends them to the synthesizer module producing sound), then the user instructs the Wekinator to record the vector of gestural input features generated as he/she performs the gesture. The user may also employ a dynamic "play-along" process for creating training examples [6], in which the Wekinator follows a user-defined "score" to update the synthesis parameters over time: the user gestures along with the dynamic sounds, while the system continuously records each synchronous gestural feature vector and parameters pair as a new training example.

Several people contrasted their previous experiences creating instruments that used explicit mappings coded in Max/MSP or Chuck, and remarked on how this affected composition for them. One wrote, "I find that I've been so overwhelmingly trained to think of these things in one-to-one ways (uh, I want tilting forward to control volume, and, uh, tilting left/right to, uh, control.... pitch, yeah that's it) that I basically want to retrain myself NOT to think that way anymore, and rather to privilege physical interactions with the interface and sonic sculpting on the synthesis end, and ask the Wekinator to connect them for me."

Composers frequently discussed the Wekinator's abstraction of the mapping functions as being useful in focusing their attention on sound and gesture, and contrasted this to explicit mapping methods that drew their attention and effort to refining the details of the mapping functions that had minor or unpredictable relationships to the sound and movement. One wrote, "By thinking about the mapping mechanism… as a closed 'black box', Wekinator allowed me to explore the parameter space in a more intuitive and natural way. In the past when I have really focused on trying to create an expressive mapping between inputs and outputs, I ended up spending so much time and energy on the mappings themselves that they started to eclipse the actually sonic result."

Several people also commented on a prioritization of the gesture-sound relationship in the evaluation and performance of mappings after they were created. In discussing how they evaluate a given mapping, all users emphasized the importance of playing with it and getting a "feel" for it. This "feel" had particular implications for performance and composition: "They (the performers) can see a relationship between my control gestures and the sounds. For dancers, at least the ones I work with, this is really important as they want to have a sense of the musician being in it with them, 'getting' the movement, rather than just playing back canned sound that have no relationship to the movement of the moment." One person reflected on the type of composition suggested by his use with the software: "I could imagine using the Wekinator for a piece where the instrument itself was the focal point, possibly some kind of concerto, or where there is an element of theatrics to the performance. I think that the gestural relationships that the Wekinator creates between the controller and the sound producer (synth) would be very interesting to highlight in a piece."

While users in general had positive experiences with this aspect of the Wekinator compared to previous mapping strategies, group discussion led to several software improvements to focus the training interaction even more tightly around the sound/gesture relationship. In particular, the play-along functionality was improved so that scores could be created and modified within the GUI, allowing the user to iteratively compose and audition dynamic sonic/physical gestures. We are currently implementing user-controlled temporal interpolation *between* parameter settings in a play-along score, so that play-along gestures can also respond to the temporal aspect of transitions in the sonic landscape. Additionally, we have added GUI support for computing "meta-features" from gestural inputs, so that the performer can incorporate speed and acceleration of gesture along with position as first-order means of controlling sound.

4.3.2. Reducing barriers to speed and ease of exploration

Many composers expressed frustration at the difficulty and slowness of exploring mappings using previous tools; one composer who has been building new musical interfaces for over 10 years wrote, "Building these kinds of instruments requires an enormous amount of experimentation and play. There are simply way too many combinations of features and parameters to manually think about trying – too many decisions to make – and too many combinations that are useless. It's a process that invariably takes way too much time; the ratio of time spent to satisfactory musical experiences is super high." In contrast, composers frequently referenced the speed and ease of creating and exploring mappings using the Wekinator: "As I work mostly with improvisation, I found Wekinator's ability to quickly map a number of input features to a potentially different number of output parameters very useful."

Group-initiated improvements to the Wekinator also often focused on streamlining the physical and cognitive processes of mapping creation. For example, foot pedal

support was introduced to control the training process without hands. Composers using synthesis patches in Max/MSP indicated a disruption in workflow due to context switching from one application to another; as a result, we have augmented the OSC bindings between Wekinator GUI and the synthesis module to propagate state changes between the two environments.

4.3.3. Providing access to surprise and discovery

Composers strongly emphasized the creative benefits of being surprised by the sounds and by the sonic-physical gestures that resulted from the generated mappings. For example: "There is the potential of creating very expressive instruments using mappings from the Wekinator. In traditional circumstances (before the Wekinator) programmers might try to constrain the values in such a way that they're always getting the results they wanted for whatever piece the instrument was being designed. Nothing wrong with that. But with Wekinator it's possible to get the controller into an unanticipated configuration and get an unexpected sound. Such a sound might not have a place in the piece the composer is working on, but might if that instrument is used in another piece." Another wrote, "There is simply no way I would be able to manually create the mappings that the Wekinator comes up with; being able to playfully explore a space that I've roughly mapped out, but that the Wekinator has provided the detail for, is inspiring."

In order to allow users to better make use of the new sounds they discovered, we added a "parameter palette" to the Wekinator. Now, at any time while playing with a mapping, a user can take a snapshot of the current synthesis parameters and add it to this palette. The user can later view the palette and set the synthesis parameters from any of the recorded snapshots, making it possible to add a new training example matching a gesture to this sound. Or, the user can play through the list of snapshots as a score for play-along learning, and optionally edit the ordering and durations of the snapshots in the score.

4.3.4. Balancing discovery with control and constraint

The ability to constrain and control the mapping functions was also very influential in composers' evaluations of the system's usefulness, and it drove many of their suggestions for improvement. Several composers developed a practice with the Wekinator in which they consciously manipulated the training process to create mappings with an appropriate balance of predictability and surprise. One common strategy was to decide on sonic and gestural boundaries of the compositional space (e.g., minimum and maximum parameter values and controller positions), then to create initial training examples that paired extreme synthesis parameter values with extreme gestures. After training the network, users played with the system to explore the sounds they found in between and outside the boundaries of these "anchor" examples. Several users employed the addition of training examples as a means to enforce predictability: "I try to add additional points in between the extremes, where I know what I want to be most predictable."

When some composers desired to exercise a finer degree of control over the behavior of individual parameters in response to gesture, while leaving the mapping of features to other parameters unchanged, we added checkboxes to the GUI to indicate whether the training examples currently being created were "active" for each parameter. By using one independent neural network per parameter, each of which might be trained on different sets, we were able to dynamically turn each training example "on or off" for each parameter.

It was also important to several composers that they be able to restrict which features of the gestural inputs affect each output parameter, for example controlling perceptually separable parameters such as pitch and timbre using physically separable inputs, (see [11]) or even distinct input controllers. The Wekinator was therefore augmented with GUI support for the choice, configuration, and dynamic modification of feature-to-parameter relationships. This allowed for experimentation with one-to-one, one-to-many, many-to-one, strategies characterized by [10] in the context of explicit mappings.

Despite the above modifications, most composers found the Wekinator to offer unsatisfactory degrees of control for certain tasks. One in particular had simple but quite specific ideas about the nature of mappings that he wanted to create, for example assigning a linear relationship between a control gesture and vibrato speed of a physical model. He found the process of attempting to teach the Wekinator this relationship through examples frustrating and not wholly successful. An admitted "control freak," he resigned himself to mappings that weren't "completely 'correctly' mapped according to my original plans… that doesn't seem to be the strength of this." Most composers indicated that, in general, they would choose other mapping strategies and tools to create very simple mappings that they knew they could explicitly code in a few lines, or for reliable triggers such as those easily created with MIDI foot pedals.

4.3.5. Supporting creation of useful, complex mappings

Several composers wrote that they valued the ease with which complex mappings could be created, and noted parallels between the degree of complexity and difficulty in Wekinator-created mappings and those in acoustic instruments. For example, "there seems to be a happy medium where there is linear, logical control, and more intuitive, unpredictable control [using the mapping]… For example when I play an acoustic string instrument, say a cello, I can predictably produce pitch by placing my finger at a certain point on the string. However this is still

dependent on a number of other more subtle features (similar to things that have been incorporated into the blotar model [the synthesis algorithm being used]) like bow pressure, placement of the bow on the string, the angle of the bow, the movement of the finger holding down the string. The interactions are not necessarily 'linear' or obvious. I think the combination of the two is what makes an acoustic instrument exciting, and the same seems true here." Another composer wrote, "Like any good instrument, acoustic or electronic, when the Wekinator is trained well it provided enough 'difficulty' in playing that it really does engage the performer. Once the training is over, and you really start to explore, it becomes a process of finding new sounds and spaces that the mapping has created and then trying to include those into your vocabulary of performing with the instrument."

4.3.6. An invitation to play

"Play" was a common word in composers' descriptions of how they interacted with the Wekinator and what they liked about it. "Play" and "playability" were important themes related to prioritizing gestural interaction, and in positively characterizing the tone of the interaction as collaborative, informal, and exploratory. According to one composer, "It feels like design software that lends itself to the imagining of 'playful' instrument to me. By 'playful' I mean both whimsical as well as embodied and outside the box." According to another, "The way that the Wekinator becomes like a toy and collaborator makes it very appealing."

4.3.7. Attention to accessibility in a creative paradigm

The broad range of composers' proficiency in computer programming and use of the command line presented initial challenges. The initial version of the Wekinator required the system configuration (e.g., which synthesis module would be used) to be specified within a ChucK code file, which was then run using the OS X Terminal. These requirements effectively barred the Wekinator's "invitation to play" from extending to the composers who were not comfortable experimenting with these tools. As a result, we created an optional subsystem within the Wekinator GUI for configuring and running ChucK.

As all composers gained experience using the software, group and individual discussions revealed more subtle discrepancies in ways of thinking and speaking about technology and its role in composition, which have important implications for the usefulness of the software to diverse groups of users. The least subtle and most problematic of these surrounded technical language embedded in the GUI used by the computer scientist. The term "feature," for example, is commonly used in machine learning as we have used it in this paper. However, "feature" can mean an aspect or even "selling point" of a piece of software, or a perceived aspect of a sound or composition. Multiple composers expressed frustration at confusing "features" and "parameters," and this confusion was sometimes a barrier to interacting with the system and understanding its potential uses. While most composers focused their creative interaction with the system on modifying the training set, one composer expressed a disconnect between this approach to composition and her work: "I just don't ever think 'spread sheet' or 'data set' when I'm creating work and this may be because my work is usually simultaneously conceptual and narrative."

4.4. Improved software and new compositions

The participation of diverse composers led to practical improvements of the Wekinator, many of which had not been considered by the computer scientist. The updated version of the software is available for public download at http://code.google.com/p/wekinator/. We believe the Wekinator offers composers a novel and useful tool: reflecting on their experiences with the software after ten weeks, composers strongly indicated that the Wekinator allowed them to create more expressive mappings than other techniques (mean 4.5 on 5-point Likert scale) and that the Wekinator allowed them to create mappings more easily (mean 4.67). As of this time, one composer from the seminar has composed and recorded a complete work that incorporates the Wekinator in two of the performance interfaces, and several others are employing it in ongoing work on compositions and instruments.

5. DISCUSSION AND FUTURE DIRECTIONS

5.1. Training the user: Mutual influence and interaction

Nearly all composers characterized their interaction with the Wekinator as more complex than an exercising of control over technology; the ways the software challenged and influenced them were important aspects of their experience as users. They often relied on the randomness and complexity offered by the learning algorithms to discover the existence of unknown and often inspiring locations in the sonic space of a synthesis algorithm and the sonic-gesture space of a mapping. Even though the Wekinator user is nominally in charge of the supervised learning "training" process, it is apparent that interaction with the system likewise trains the user. Through using the system, composers learned about what interaction strategies produced mappings that they liked ("At first I tried to single out each output parameter individually and train the system one at a time, although ultimately I found that it proved to be most useful when I didn't try to micro-manage the inputs/outputs, and just gave it a snapshot of all the features and then let the network sort itself out."), and they learned that certain mapping strategies (such as enforcing a linear relationship between features and parameters) were hard to satisfactorily produce. As a

result, their goals and subsequent interaction with the system often changed course.

Viewing the Wekinator and other technologies for supporting composition as meta-instruments suggests a way to reconnect our analysis with the large body of work discussing interaction in performance, via considering interactions with the composition software through the lens of interactive performance paradigms and metaphors proposed by in the literature. Chadabe's conversational metaphor for interaction [2] seems to be particularly appropriate to the interaction style we observed with the Wekinator. Software that supports other interaction models, such as his "sailing a ship in stormy waters," [4] suggests yet another space of compositional styles and outcomes.

5.2. Compositional interaction and mapping strategies

In light of existing work categorizing mapping strategies as generative or explicit, many of our observations regarding interaction with the Wekinator appear at first glance to be rooted in the Wekinator's nature as a generative mapping creation system, in which the desired shape of the mapping is implicitly encoded in the training examples. Generating mappings from training examples inherently supports an embodied and enactive [19] approach to the mapping specification processes, in that gestures can be used to create the training examples. The specification of mapping via example also enables many low-level details to be abstracted away, speeding up the mapping exploration process, facilitating use by non-programmers, and freeing the composer to focus on the creative process. The use of neural networks, which are capable of introducing great nonlinearities into the learned functions, also contributes to serendipitous discovery and efficient production of complex mappings. Overall, we conclude that many qualities of the Wekinator that are most helpful in compositional interaction are well supported by properties of the generative mapping strategy. However, a user who has a specific notion of the mathematical relationship between gestural features and sound parameters may be frustrated by the difficulty and inefficiency of implicitly encoding this mapping via training examples; it is therefore also obvious that the suitability of example-driven mapping generation is contingent on users' compositional goals.

On closer examination, one can imagine interfaces that do not enforce a clear dichotomy between example-driving and function definition-driven creation, supported by underlying algorithms that are difficult to mathematically categorize as purely generative or explicit. For example, simple linear or polynomial regression algorithms could be embedded in an interface that allows the user to dynamically switch between supplying examples and explicitly editing equations.

5.3. Exploratory and objective-oriented activities

Our last point of discussion arises out of observations of how composers did *not* use the Wekinator, namely in the construction of gesture event detection or classification systems. Supervised learning has a long history of application to gesture recognition problems (e.g. [5]), and the Wekinator includes several common and powerful classification algorithms that could be used for these purposes. One composer attempted to use the system in this manner (for drum stroke recognition) for several weeks at the beginning of the seminar, but he ultimately chose to focus instead on creating continuous timbral mappings with other controllers.

One explanation for this observation is that the Wekinator currently offers better support for exploratory, open-ended mapping creation than it does for creating models that solve an objective problem. In particular, an objective-oriented approach prioritizes predictability heavily and discovery not at all: a user is probably not interested in being surprised by a drum stroke labeler that produces the wrong label. Additional infrastructure to enforce predictability by facilitating the training of more accurate models might therefore be appropriate, for example a graphical interface to perform fine-grained edits of the training data to annotate onsets or remove noise, or interfaces for refining the selection and post-processing of the features. Work along the lines of supporting feature selection and algorithm parameter selection might take inspiration from Weka itself [20], the very popular graphical toolkit for supporting machine learning experimentation in more non-realtime domains. Integration of more standard features used in audio and gesture analysis in music information retrieval might also be helpful.

However, some composers tempered this proposal with an alternative explanation: they had not yet had the chance to learn as much about the Wekinator's functionality supporting gesture classification as they would like. Clearly, further work might be done to study composers interacting with the software to accomplish more objective-oriented tasks, so we can better understand interaction requirements and user learning curve in this problem space.

6. CONCLUSIONS

Usefulness and accessibility in a music composition system may be very broadly defined and difficult to measure. As we have discussed, qualities of interaction in the process of composition have both striking and subtle consequences for the type of composition produced and the composer's satisfaction with the process. These consequences stem not just from limits on the space of compositions that are technologically feasible using an interface, but on the compositional possibilities that are

suggested to the user and how the software best enables the composer to navigate through and refine these possibilities, as expressed through the GUI, underlying algorithms, and even the language and metaphors embedded in a system.

Our experience applying a human-computer interaction approach to investigating how technology can support computer music composition has highlighted these challenges but also enriched our understanding of the new musical instrument design process in spite of them. We have observed and discussed several properties of interaction that are valued by composers engaging in this process, and we have drawn on our experiences to suggest new ways of framing interaction, mapping strategies, and compositional goals. Our collaboration has also led to improvements in the Wekinator software, which several composers continue to employ in their work.

7. ACKNOWLEDGEMENTS

This material is based upon work supported under a National Science Foundation Graduate Research Fellowship. This work is also supported by a John D. and Catherine T. MacArthur Foundation Digital Media and Learning grant.

8. REFERENCES

[1] Bevilacqua, F., R. Müller, and N. Schnell. "MnM: A Max/MSP mapping toolbox," *Proc. NIME*, 2005, pp. 85–88.

[2] Chadabe, J. "The limitations of mapping as a structural descriptive in electronic musical instruments," *Proc. NIME*, 2002.

[3] Cont, A., T. Coduys, and C. Henry, "Real-time gesture mapping in Pd environment using neural networks," *Proc. NIME,* 2004, pp. 39–42.

[4] Drummond, J. "Understanding interactive systems", *Organised Sound*, vol. 14, no. 2, pp. 124–33, 2009.

[5] Fels, S. S. and G. E. Hinton. "Glove-Talk: A neural network interface between a data-glove and a speech synthesizer," *IEEE Trans. on Neural Networks*, vol. 4, 1993.

[6] Fiebrink, R., P. R. Cook, and D. Trueman. "Play-along mapping of musical controllers," *Proc. ICMC,* 2009.

[7] Fiebrink, R., D. Trueman, and P. R. Cook. "A meta-instrument for interactive, on-the-fly machine learning," *Proc. NIME*, 2009.

[8] Hahn, T., and C. Bahn. "Pikapika—The collaborative composition of an interactive sonic character," *Organised sound* vol. 7, no. 3, pp. 229–238, 2003.

[9] Hunt, A., and M. M. Wanderley, "Mapping performer parameters to synthesis engines," *Organised Sound*, vol. 7, pp. 97–108, 2002.

[10] Hunt, A., M. M. Wanderley, and M. Paradis. "The importance of parameter mapping in electronic instrument design," *Proc. NIME*, 2002.

[11] Jacob, R., L. Sibert, D. McFarlane, and M. Mullen, Jr. "Integrality and separability of input devices," *Proc. ACM TOCHI*, vol. 1., no. 1., pp. 3–26, 1994.

[12] Paine, G., Ed. *Organised Sound,* vol. 14, no. 2**,** pp 121–123, 2009.

[13] Schnell, N., and M. Battier. "Introducing composed instruments, technical and musicological implications," *Proc. NIME*, 2002.

[14] Shneiderman, B. "Creativity support tools: Accelerating discovery and innovation." *Comm. ACM* vol. 50, no. 12, pp. 20–32, Dec. 2007.

[15] Tsandilas, T., Letondal, C., and Mackay, W. E. 2009. "Musink: Composing music through augmented drawing," *Proc. ACM CHI* 2009, pp. 819–828.

[16] Van Nort, D. "Instrumental listening: Sonic gesture as design principle," *Organised sound*, vol. 14, no. 2, 2009, pp. 177–187.

[17] Vredenburg, K., J. Mao, P. W. Smith, and T. Carey. "A survey of user-centered design practice," *Proc. SIGCHI* 2002, pp. 471–478.

[18] Wanderley, M. M., N. Schnell, and J. Rovan. "ESCHER: Modeling and performing composed instruments in real-time," *IEEE Conf. on Systems, Man, and Cybernetics*, 1998, vol. 2, pp. 1080–1084.

[19] Wessel, D. "An Enactive Approach to Computer Music Performance," *Le Feedback dans la Creation Musical*, Y. Orlarey, Ed., Lyon, France: Studio Gramme, 2006, pp. 93–98.

[20] Witten, I., and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2005.