Personalization, Expressivity, and Learnability of an Implicit Mapping Strategy for Physical Interfaces

David J. Merrill and Joseph A. Paradiso
Responsive Environments Group, MIT Media Laboratory, Cambridge, Mass. USA
{dmerrill, joep}@media.mit.edu

ABSTRACT
We present a new model for configuring the connections between user input and system output in a physical interface with diverse sensor degrees of freedom across several input modalities. Our system allows a user to demonstrate input gestures and manipulations directly to the system, teaching it the desired mappings by example. We developed a musical control application in which user-defined gestures and user-assigned manipulations trigger and modify sounds. The effectiveness of our system was tested by experimentally comparing our user-definable system to a similar, pre-configured version. The results suggest that users prefer to actively configure a physical interface to having expertly-configured presets. In addition, we propose our model as a more general mapping discovery tool for physical interface designers.

Keywords
Physical interaction, sensors, affordance mapping, gestural bookmark, electronic music controllers, demonstrational interfaces, experiment.

INTRODUCTION
Physically embodied interfaces to computers have experienced great advances from developments in industrial design, sensing, miniaturization, and data transmission. Today's plentitude of cheap, compact sensors easily allows many different modalities of user input, both implicit and explicit, to be instrumented [18]. Correspondingly, the space of possible interaction techniques, application areas, and interface metaphors is growing in exciting new directions. However, along with this progress has come the complex problem of finding a compelling mapping from an increasingly open-ended user input space to the range of possible system behavior [9]. We have developed a system called FlexiGesture that attempts to address this challenge with a configuration-by-example method for mapping the affordances of a physical interface.

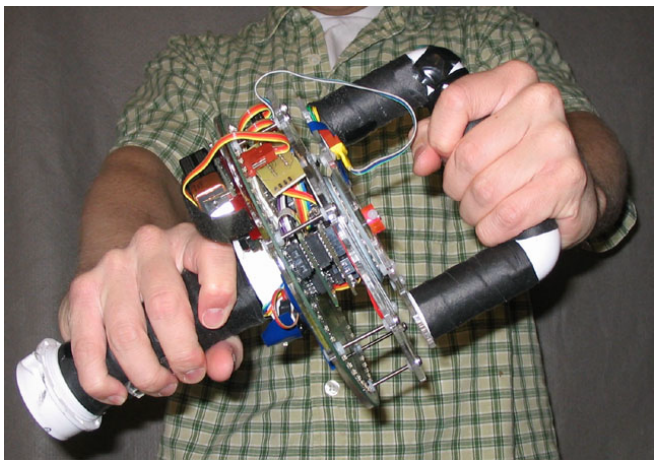

Figure 1: FlexiGesture in use

The application area of music control requires the system to have a high degree of temporal responsiveness, making it an interesting and useful testbed for other areas of human-computer interaction. The basic task that our system addresses is common to many systems that involve a physical interface: mapping a number of continuous and discrete input affordances to compelling and desirable system output. Our approach could also be applied to other tasks like video editing, 3-d modeling, or restoring manual dexterity to a person with a physical handicap. These tasks are mentioned because they feature a potentially arbitrary connection between input and output that could benefit from personalization. New electronic musical devices often offer little physical-gesture-to-sound intuition, and can feature a huge input control space. The large input space, coupled with an equally large output space can make effective mapping of sound-to-input difficult, time consuming, and often hand-tuned to individual performers. The most straightforward way to map is to borrow behavior from existing acoustic instruments. Bernd Schoner built a system that learned the associations between the sensed input state of a violin and the acoustic output in low-level, data-driven manner in order to re-synthesize violin sounds from new user input [21]. Other attempts to map physical motion to synthesized sound in a flexible way have included the use of neural networks to map hand or full-body gestures onto synthesizer parameters [14] [10] [5].

Whereas these systems borrow from an existing gestural language (as in the case of Schoner's violin) or focus solely on semi-interactively training the system to map a small number of continuous inputs onto system parameters, FlexiGesture explores the immediate, temporally-based assignment of novel gestures to sound. Designers of consumer interfaces often accomplish the mapping task by creating a "common case" default configuration that attempts to accommodate the largest number of users while annoying the fewest. The common case interface is often created by a domain expert, or informed by design heuristics or "think aloud" trials with sample users [19]. Most consumers use device presets without ever bothering to personalize them. Due to the generic way that these configurations are created, they can be non-optimal to many individual users.

By providing a teach-by-example way to configure a physical interface, we seek to improve a user's level of personalization and engagement with the system. In addition to providing a fluid and natural two-handed interaction and a number of different input degrees-of-freedom, our system adds new ways to configure the interface. These include (1) allowing the example-based creation of open-ended "gestural bookmarks" with which the user can trigger sounds to be played; and (2) example-based assignment of continuous input affordances to sound-modulation (effect) parameters. Furthermore, we are interested in the relationship that develops between an instrument and performer when the device is able to learn along with the player.

In this paper we present an experimental study that compares our configuration-by-example interface to a pre-configured adaptation of the same physical interface. The results suggest a better personalization experience and enhanced sense of expressiveness with the configuration-by-example interface than with a pre-defined configuration. We then examine the gestures that users in our study associated with different sounds, finding trends that suggest connections between the particular sounds and the forms of the gestures that people chose to create.

RELATED WORK

There have been many research projects surrounding the idea of programming by example [7] [15] [16] [22]. Sometimes referred to as demonstrational interfaces, these systems have been applied to tasks such as graphical manipulation, text editing and formatting, file management, user interface creation and general programming. Whereas these systems have produced important knowledge about mostly sequential, symbolic tasks involving a traditional GUI, our system applies the demonstrational interface paradigm to a multi-degree-of-freedom physical controller and a real-time task.

In the domain of audio synthesis and control, several existing interfaces have a "learn mode" in which a MIDI [13] controller can be associated with a particular parameterized feature of the system. For instance, the Lexicon MPX110 [11] can be put into a state in which twiddling one of the knobs on the front panel then sending a MIDI controller message to the unit will create mapping between the particular MIDI controller and the knob's effect. Abelton's "Live" software [1] has a similar feature to connect a MIDI controller to a software parameter. The continuous control component of our system takes this style of mapping a step further by finding the input sensor with the maximal variance (even when several inputs may be varying) and by capturing the direction of the deflection. Our gestural mapping component pushes the idea even further by allowing any physical spatial motion to be associated with a sound. This gives a performer the freedom to execute dramatic gestures that also function to trigger their desired sounds.

Interface designers have long been interested in recognizing the personalized handwriting or speech of individual users, and these applications typically include a training portion. With FlexiGesture, we apply this style of individualized configuration to a novel and multi-degree-of-freedom physical interface in a way that is quickly configurable and re-configurable. The resulting corpus of data collected makes it a useful tool to study user tendencies and configuration preferences for a physical interface (see Discussion, below).

DESIGN

The FlexiGesture device consists of a small acrylic and PVC device with two taped handgrips oriented orthogonally to each other, mounted above and below a springed, return-to-center rotating carriage. Also below the rotating carriage are the device's embedded sensors and communication electronics, as shown in Figure 2. The electronics are based on the Stack sensing platform [3]. The input modalities provided by the onboard sensors of FlexiGesture include 3-degree-of-freedom (DOF) acceleration, 3-DOF rotation, 4-DOF squeezing, 2-DOF bending, and 1-DOF twisting (see the Future Work section for recently added input modalities). Acceleration is sensed with two dual-axis accelerometers, and rotation with three single-axis gyros. Continuous button squeezing is sensed with force-sensitive resistors, bend is sensed with four back-to-back flex sensors (two per DOF), and a potentiometer senses the rotation of the upper carriage. Analog signal-conditioning circuitry pre-processes the sensor outputs, and the resulting voltages are sampled by a microcontroller at a full-system rate of 180Hz. This data is transmitted serially to a computer running Windows XP. This computer is running an application written in Java that interprets the incoming data stream and sends sound control commands in the form of OpenSoundControl-formatted datagram packets [17] over a network connection to a second computer running Pure-Data [20]. The Pure-Data patch generates the sounds and effects that the user hears. For more technical documentation and discussion, see [12].

Spatial gestures, consisting of acceleration and rotation, can be associated with particular sounds, and are used to trigger their associated sounds in play mode (see the "play mode" section below for details). The continuous inputs to the system (force, bend and twist) can be mapped onto sound-manipulation parameters, and can thus be used to "sculpt" any currently playing sound.

Rationale

Our overall goal is to make a multi-DOF physical interface that a user can configure by "showing" the device their desired connection between input gesture and system behavior. The system immediately learns to respond to arbitrary, multi-DOF input in a flexible and personalized way. The device's physical form was designed to be small and lightweight enough that it can be comfortably grasped in one or two hands. The input affordances were chosen to present a wide range of possible modes of interaction as possible in a device of this size. The decision to support two-handed manipulation was informed by the dexterity with which people manipulate hand-held tools and materials in other creative domains like woodworking or sculpting. Two-handed interfaces are likewise common in musical controllers, both electronic and acoustic. Two-handed manipulation tasks generally have a preferred division of labor between the left and right hands [8], and in order to support the left or right-handedness of the user, the device was made as symmetric as possible. FlexiGesture was intentionally made to look distinct from existing musical instruments, in order to avoid invoking familiar and routine modes of musical interaction.

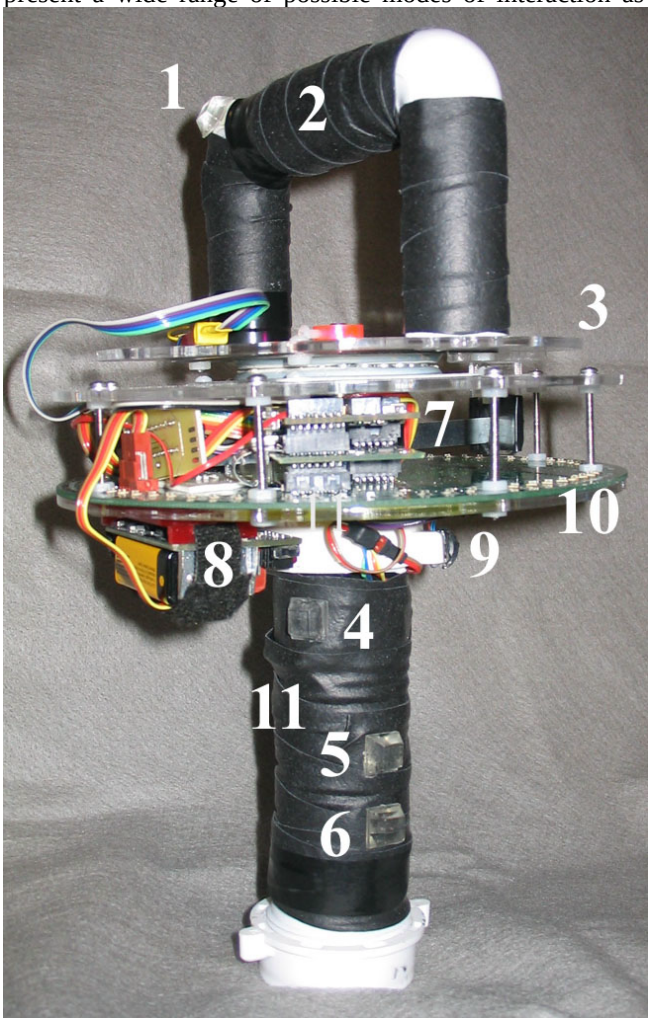

Figure 2: The physical device. Numbers indicate the following: (1) upper thumb button, (2) upper squeezable handle (3) rotating carriage (4) trigger button (5) lower handle button A (6) lower handle button B (7) inner carriage electronics (8) battery (9) toggle button

Gestural Mapping

The efficient creation of mappings from input gesture to output behavior happens in the gestural mapping mode of the Java program. FlexiGesture begins as a blank slate, unmapped, with no pre-configured connections between input gesture and sound. The user can explore the set of possible sounds either by clicking through a list on the computer screen or by pressing a toggle button up and down with their thumb. Pressing the toggle button up plays the previous sound in the list, and pressing down plays the next sound. After hearing a sound that they would like to use in play mode (see the "Play Mode" section), they can assign that sound to a gesture by performing the desired gesture while squeezing a "trigger" button with their index finger. This action causes the system to record the gesture, and creates a connection between the gesture and sound. In the current interface, the user can create up to ten of these "gestural bookmarks" that are then immediately usable in play mode.

Since the device is constantly in motion, the use of the trigger button allows the system to know when the user is intentionally executing a gesture. A ring of LED's mounted beneath the rotating carriage lights up whenever the trigger button is squeezed, giving visual feedback to the user.

Continuous Control Assignment

The continuous inputs of FlexiGesture (twist, bend and squeeze) can be mapped one-to-one onto a number of sound manipulation effects. These effects are: volume level, tremolo speed, pitch-shift, ring-modulation frequency, and the center frequency of a sweeping band-pass filter. The user can enter assignment mode for a desired effect by clicking on a button in the GUI. In assignment mode for a particular effect, the program sweeps the effect back and forth slowly through its dynamic range and monitors the windowed variance on each of the continuous inputs (a rate of 1Hz was found to be usable for this sweep). The user can create a mapping by following the sweep of the effect with pressure on a continuous button, bending of the lower handle, or twisting of the upper carriage.

At each extrema of the effect's sweep, the system notes which continuous sensor is currently experiencing the most

windowed variance, and records the current value of that particular sensor (steady-state and excited-state variances of each sensor are calibrated manually beforehand). Consecutive readings are considered consistent when the same sensor is experiencing the maximal amount of variance, and the instantaneous value of the sensor has crossed the midpoint of its range. For instance, if the twist affordance is being mapped, the system would decide that a pair of consistent readings had been found when the user has twisted the carriage from one end of its travel range to the other, reaching the endpoints of this twisting motion when the effect's oscillation was also near its extrema. Since the measurement of the sensor value is taken at the effect's extrema, the user's sweep can be up to nearly 90 degrees out of phase with the effect, and the mapping will still be created correctly. This allows the assignment process to be forgiving in case the user is not following along precisely with the effect's sweep.

The system notes the instantaneous value of the sensor when creating a mapping in order to produce correct polarity – that is, it remembers which end of the sensor's range should correspond to which end of the effect's range, based on how the user's pressure or twist had been following along with the effect's sweep. At the moment that the system notices three consecutive consistent extrema (three in a row was found to be a good compromise: it feels fairly immediate while avoiding the creation of spurious mappings that tend to occur with a shorter consistency requirement), the mapping is created from the given sensor to the given effect and the effect becomes active. The user can instantly manipulate the effect with the associated affordance. This immediate feedback enables the user to quickly understand how the mapping feels when in use, an experience that sometimes causes them to realize that they don't like it, in which case they will quickly re-map the effect.

Play Mode

The user can enter play mode by clicking on a button in the GUI. Play mode enables all of the continuous effect mappings, and allows the user to trigger and sustain sounds. Triggering a sound works the same way as training a gesture-to-sound mapping: the user executes the physical spatial gesture with the device, while simultaneously depressing the "trigger" button. The Java software uses a dynamic time-warping (DTW) algorithm based on dynamic programming [2] to find the optimally matching gesture from the set of example gestures. DTW was chosen as the gesture-recognition algorithm for FlexiGesture because it provides decent recognition rates in polynomial time, while allowing for model training with as few as one example. Most importantly, DTW can operate on data vectors with different lengths.

Figure 3: A user executes a gesture (left) and finesses a continuous effect via the twisting affordance (right)

A fundamental usability goal for the system was that it should allow a user to create a gesture-to-sound association easily, and without excessive training. This goal suggested that a gesture-classification scheme should be used that could classify as robustly as possible without a large number of examples of each gesture. Statistical pattern-recognition methods tend to require significant amounts of training data before they become useful classifiers, and for this reason they were considered less suitable. When the DTW algorithm determines the best match, a sound-triggering message is sent across the network to the Pure-Data patch, which triggers the playing of the given sound.

To sustain the most recent sound, the user presses the thumb toggle button up. Multiple sounds can be sustained simultaneously, and pressing the toggle button down stops only the most recently sustained sound. Pressing the toggle button directly in towards the device immediately stops *all* sustained sounds. The continuous-input-to-effect mappings are all enabled during play mode, and are applied to all currently playing sounds. The user can revisit either of the mapping-creation modes at any time, removing and creating gestural bookmarks or effect mappings.

The thumb toggle button gestures and the use of the trigger button to create and execute gestural bookmarks are the only pre-scripted gestures in the system. The particular set of physical affordances present in the device, as well as the separation between training mode and play mode, represent a small set of inherent assumptions about, and constraints on the interaction. In order to serve the overall goal of creating a largely open-ended system for gesture and continuous control assignment and play, this small set of pre-scripted gestures and assumptions were built in. To create a completely assumption-free interface would cross more deeply into learning theory and artificial intelligence, and would be interesting future work.

The overall design begins to define a model for configuring the behavior of any physical interface. In training mode, the triggering of a generic system event can be associated with a unique physical gesture by performing the gesture directly after manually invoking the event. The use of temporal coincidence to create an association between a signal and a desired behavior is a concept borrowed from behavioral psychology. Our concept of a physical gesture generalizes to a segment taken from any collection of

arbitrary sensor data streams. These data could be inertial, capacitive, optical, or even mouse-based or the output from a speech recognizer. FlexiGesture also suggests a model for quickly associating any continuous parameter of a system with a continuous physical input affordance by prompting the user to follow along with a slowly oscillating effect in a consistent manner. Our strategy for configuring continuous affordances allows the system to discover the user's desired mapping in a way that also captures its polarity. The system shows these methods for the music-control task as a concrete example of a broad class of applications that employ a physical controller that provides a number of diverse sensor data streams, and arbitrarily-mapped system outputs. The task could have equivalently been the control of dramatic stage lighting, an action-game character, an improvisational graphics display or a mobile phone.

EXPERIMENT

We conducted an experiment to attempt to understand the costs and benefits resulting from the configuration-by-example interaction for the physical device, compared to a pre-configured device. Because FlexiGesture allows participants to create their own arbitrary physical gestures and continuous-affordance-to-effect mappings, we expected participants to experience a greater feeling of personalization, and a preference towards the personally-configured device for creative tasks. By the same token, we expected that the extra effort (both physical and cognitive), required in creating custom mappings for the device may make it less easy to learn initially. However, we expect the benefits of personalization to be quite compelling, and the design goal was that users indicate a preference for the personally-configurable device; our experiment attempts to investigate these expectations.

Task

For this experiment, we created a play/performance task. Participants were asked to explore the system, triggering, sustaining and modifying sounds for as long as they wished. Since the application domain for the current device is music, a minimally structured task was chosen in order to allow for creative expression, while still allowing us to collect participants' subjective impressions of the system as a whole as well as data about their triggering and manipulation behavior.

A small pilot study was run, which prompted us to change two things about the system and experimental setup. The device is equipped with bend sensors inside the lower handle, which were meant to provide another two continuous degrees of freedom for the controller. However, since the same handle also had pressure sensors mounted on its outside, it was impossible to bend the handle without creating an even greater disturbance on the pressure sensors. Although with sufficient analysis, these parameters could be adequately decoupled, for this experiment, the bend-sensors were not used. We also learned through user feedback in the pilot that a video would be a consistent way to introduce the device and its affordances to participants in the study. So for our study, we created two introductory videos. Participants watched the appropriate video before each section of the experiment.

Experimental Design

Participants performed our task under two different conditions, which we designed to be as similar as possible in physical setup. The design was within-participant, meaning that every participant experienced both conditions. The order of the conditions was randomized, with half of the participants experiencing the presets condition before the configuration-by-example condition, and the other half experiencing the configuration-by-example condition first.

Presets condition: To represent the majority of pre-configured interfaces, we created a "common-case" pre-configured condition. The pre-configured condition had three mappings from physical gesture to sound, and three continuous-control-to-effect mappings. The creation of the configuration was based on observation of people using the device during development, and was also informed by the authors' experience with the device. We expected this condition to be easier to learn to use, but less compelling in terms of expressivity and engaging-ness.

Configuration-by-example condition: To test our configuration-by-example model, we created a condition in which the device began without any pre-configured mappings. Participants were asked to configure as many gesture-to-sound and continuous-control-to-effect mappings as they wanted. Though we expected it to be more difficult to learn initially, we anticipated that this condition would be more engaging, and so we expected to see a preference for it in terms of personalization and interest in future play.

Participants

We used 25 participants, 15 male and 10 female, recruited with emails to lists around the MIT campus. Sessions lasted 40-55 minutes; each participant was given a $5 gift certificate for a local ice cream shop.

Procedure

Participants were presented with the device and the PC software that allowed them to switch between modes. For each condition, we explained to the participant that they would watch the introductory video, then interact with the device until they were told to move on to the next task. Our system recorded the gestures and mappings that they created in the configuration-by-example condition, and also recorded time-stamped entries in a logfile when a new gesture was triggered in either mode. At the end of each condition, participants were asked to report subjective impressions about the device in a short survey. After completing both conditions, participants completed a survey comparing both conditions, and were encouraged to

give open-ended feedback about the system and their experience as a whole.

Results

The surveys asked subjective questions about the system related to expressivity, personalization, and ease of use. Participants selected their answers on a 7-point Likert scale. We analyzed the data with a repeated measures test. Order of condition presentation was ruled out as a confounding factor, as no significant order-related differences were found.

Self-Report Results: Our questionnaire asked participants how personalized they felt the system was, and how appropriate they considered the level of personalization (see Figure 4 for a full summary). With a two-tailed t-test, we found a significant overall effect of condition for both questions, indicating a greater amount of personalization and appropriateness in the configuration-by-example condition. We also found a marginally significant result indicating an increased potential for expressiveness for the same condition ($t(24) = -1.66$, $p = .110$). The questionnaire also asked about ease-of-learning for the two conditions, showing a significant overall effect of condition indicating that the pre-configured system was easier to learn, both in terms of triggering sounds with physical gestures and manipulating effects with the continuous inputs. The final survey asked users to explicitly choose between the two conditions along a number dimensions, including expressivity, novelty, ease of use, and interest in performance (see Figure 5). Participants indicated a preference for the configuration-by-example interface along all of these dimensions except ease of use.

In the open-ended response section, many users suggested that they would have benefited from a longer amount of time to use the system. Additionally, some requested that triggered sounds start playing before the gesture was completed, as one would expect from a musical controller employing elaborate gestures and operating in a non-percussive mode. Our trigger-and-modify model does introduce a certain disconnect between the sound trigger and continuous control, whereas an acoustic musical instrument is more seamless (e.g., even a drum responds to velocity and hit position, and a violin heavily combines both). This disconnect is a similar characteristic of many electronic music controllers, but could be explored in the future with different classification scheme such as continuous hidden-markov-models, or explicit design to "re-couple" the input gestures and the synthesis algorithms [6]. Our dynamic-time-warping scheme typically completed in 50-500 milliseconds, depending on the number and length of gestures the participant had trained. This running time produced a noticeable latency in triggering sounds, while the continuous manipulation ran in real-time. We discuss ways to address this latency in Future Work.

The accuracy of the classification varied based on how different a participant's physical gestures were. In advance testing, the system was able to classify novel gestures into one of 10 classes with 98% overall accuracy. In practice, classification accuracy was slightly lower, and varied from person to person depending on how individually unique the participant made their gestures.

Usage Statistics Results: A number of usage statistics were collected during the task in addition to the survey data. The first statistic pulled from the implicit data was the length of the trained gestures versus the length of the sounds that they were configured to trigger. These processed gesture lengths were significantly correlated with the lengths of the sounds, $r(10) = .73$, $p = .017$.

Question Topic	Presets Mean	User-config. Mean	Significance
How much "personalization" did you feel that this system offered to you? (1) no personalization…(7) very much personalization	3.36	4.68	$t(24) = -5.28$, $p < .001$
Please rate your feelings about the level of personalization you experienced. (1) far too little personalization…(7) far too much personalization	3.24	3.80	$t(24) = -2.34$, $p = .028$
How expressive did you feel that you could be in using this system? (1) not at all expressive…(7) extremely expressive	3.80	4.36	$t(24) = -1.66$, $p = .110$
How easy was it for you to learn to trigger the sounds that you wanted? (1) not easy at all…(7) extremely easy	5.52	5.0	$t(24) = 2.0$, $p = .056$
How easy was it for you to learn to manipulate the effects? (1) not easy at all…(7) extremely easy	5.84	5.12	$t(24) = 2.90$, $p = .008$

Figure 4: Findings related to personalization, expressivity, and ease of learning

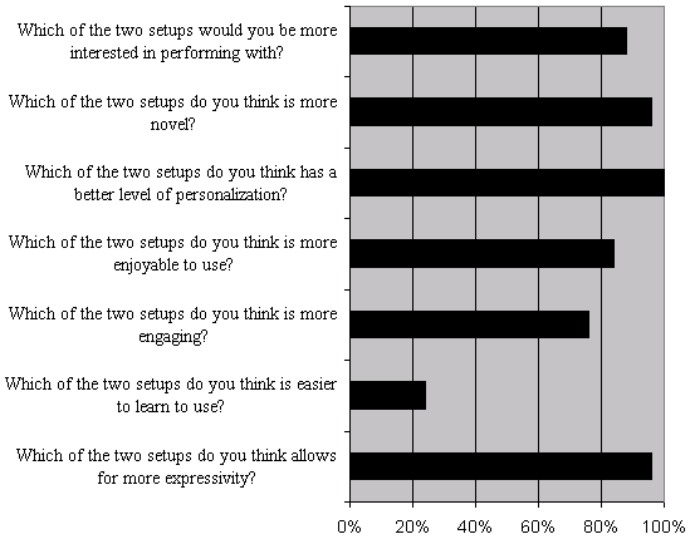

Figure 5: Percentage of participants who chose the configuration-by-example system.

	Acceleration	Rotation
turntable	16.659	21.079
tom (drum)	17.053	16.725
sleigh bell	11.802	13.537
vibra-slap	12.060	11.635
clapping	12.027	7.751
cymbal	15.953	14.843
drone1	10.749	8.369
drone2	2.612	3.095
drone3	12.132	10.061
noise	2.952	6.691

Figure 6: Average gestural acceleration and rotational energy per sound (measured as the mean of scaled, per-subject, per-sensor variances)

	Number that used it	Used it with positive mapping
ring modulation	14	71%
volume	15	33%
band-pass filter	14	36%
tremolo	19	47%
pitch-shift	22	91%

Figure 7: Effects available, usage, and polarity observed

We also analyzed the inertial features of the users' gestures. The primary measure we have analyzed is the per-sensor root-mean-squared (RMS) energy present in the gestures. To preprocess the inertial data, the average energy was computed, per-sensor per-participant, and each gesture's energy profile was then normalized by these values. Figure 6 summarizes the average energy from the accelerometers and gyros from gestures associated with the 10 sounds.

Finally, we collected data about the kinds of continuous-control-to-effect mappings that users created. Figure 7 shows two things: The first is the number of participants (out of 25 total) that created a mapping for the particular effect, which gives an indication of the effect's popularity. The second item shown is the percentage of the participants that trained an input-DOF for the effect that created a "positive" mapping to the effect. The polarity of a mapping (positive vs. negative) refers to how the range of the sensor is mapped to the range of the effect. A positive mapping associates increased pressure on an FSR or clockwise rotation of the carriage with an increased level of the effect.

Discussion

Manufacturers of interfaces, musical and otherwise, have long provided preset configurations in order to allow users to have an easy initial experience with their product "out of the box" without necessarily having to configure the device. Our results suggest that an interface that supports configuration-by-example, although possibly more difficult to learn initially, features a greater sense of personalization and expressivity, even when compared to a carefully crafted default configuration. We interpret this result to suggest that a learning curve in a physical interface for event-triggering and continuous control may not necessarily be undesirable, and in fact may in some circumstances enhance the connection between user and device. In the musical domain, this finding is likely related to the sense of satisfaction felt when a learner achieves a new level of mastery of a traditional musical instrument. Our configuration-by-example interface was considered more compelling than the pre-configured interface along a broad range of dimensions, including expressivity, engaging-ness, enjoyability, personalization, novelty and interest in performance with the system (Fig. 4). This preference for a interface that may be more difficult to learn indicates a difference between systems built to support expressive interaction and "walk up and use" systems like information kiosks that are built so that users can effectively use them without any training. Users may be more tolerant of – and perhaps appreciate more – some amount of learning on a device supporting an expressive task. By the same token, expressive, static musical instruments take years to learn properly. With FlexiGesture we've begun to investigate what can happen when an interface can begin to learn along with the player.

The usage statistics that we collected suggest several interesting consistencies about the way that people used FlexiGesture. The correlation between the lengths of the sounds and the lengths of the gestures that subjects created

for them indicates that people tend to associate longer gestures with longer sounds, and vice versa. Interface designers looking to map a number of input gestures onto system behavior could apply this result by picking longer gestures for longer responses, and shorter gestures for shorter responses.

Even more interesting than the correlation between the lengths of sounds and gestures is a closer look at the features of the gestures being associated with the various sounds. Figure 6 suggests several interesting patterns. First, the "turntable" gestures contain much more energy in rotational than in acceleration. By the same token, the "clapping" gestures contain more energy in acceleration than in rotation. These findings are consistent with the types of motions that are used to produce these sounds under traditional musical situations: turntable scratching is an activity that involves rotation, while a clapping motion features more acceleration. The gestures associated with the "drone" sounds tend to have a more equal combination of acceleration and rotation, which may be explained by the observation that the physical motion normally associated with creating those sounds is less defined. For instance, a drone could be produced by a bowed instrument like a violin or hurdy-gurdy, or a wind instrument like a trumpet or ottu. Since the sensors are always off-axis to some degree based on how the user holds the device, we never see a complete absence of either acceleration or rotation.

These patterns suggest that people bring experience from their lifetime of manipulating the physical world to the current open-ended gesture creation task. This observation reinforces the notion that a designer can rely on metaphors from users' experiences when constructing a physical interface. Turning this around, a device like FlexiGesture could be used to explore people's innate gestural associations for sounds and other system behaviors, even with sounds and behaviors that produce no clear intuitions from the physical world. These could be sounds synthesized algorithmically in ways that bear no direct resemblance to any acoustic instrument, or editing tasks that only exist in software. The possibility to explore users' gestural intuitions is how FlexiGesture provides an exciting model for creating new mappings, capturing data about how people intuitively connect input to output in a large space of possibilities. A more systematic characterization of users' gestural intuitions about sounds and other synthesized stimuli would be interesting further work, and would produce an important dataset for interface designers.

Looking at Figure 7 we notice that most of the effects are skewed towards either a positive or negative mapping. The consistency seen here may not be surprising, but it suggests that people are predisposed to create continuous mappings in certain ways. For instance, the fact that 91% of participants who used pitch shift created a positive mapping illustrates that the natural intuition for applying a pitch shift is that increased squeeze should produce more of the effect. In the other direction we see that squeezing tended to be associated with decreasing volume, not necessarily what intuition would predict. A designer could use these trends, and the data from the other effects, to understand better how to create continuous-affordance-to-effect mappings for a physical interface. Similarly interesting and potentially useful patterns show up in the mapping of ring-modulation and the band-pass filter.

The statistics we captured from the configuration-by-example illustrate the usefulness of FlexiGesture as a tool for user interface designers. With the explosion in mapping possibilities for new physical interfaces, data captured in this manner can be important way for designers to understand how people would naturally associate gesture and sound. For interfaces where it is not practical to allow users to configure-by-example, the type of information gathered with FlexiGesture and systems like it could produce a useful and structured way to create default mappings.

CONCLUSIONS

FlexiGesture shows how users can create mappings from input gesture to system behavior in a multi-degree-of-freedom physical interface by example in a way that suggests a more satisfactory user experience than with preset expertly-designed mappings. We presented our system design, interaction techniques, design rationale, and an experiment. Our mapping and music creation task is representative of a larger set of physical-gesture-to-control-mapping creation tasks for which this type of mapping strategy should be useful, such as game control, video and photo editing, 3D sculpting, or assistive devices for handicapped people. We demonstrate how a system using a physical input device can be taught by example to recognize open-ended symbolic gestures for event-triggering and to quickly associate continuous inputs with continuous modulation parameters to produce a more personalized and satisfying user experience. Moreover, we suggest that data collected from this type of system could be useful as a tool to inform the design and mapping of new physical interfaces.

FUTURE WORK

It would be interesting to add more input degrees-of-freedom to the device, to give the user a wider and more open-ended range of possibilities to define gestures and continuous mappings. Some work in this direction has been started already; primarily, an electric field sensing device was constructed that permits the system to measure the distance between FlexiGesture and 4 separate receive-electrodes. These 4 channels are sampled at the same rate as the other sensors, and could be used as inputs to gesture training, effect manipulation or both. Tilt is another feature that can be measured from the existing accelerometers, and it could also be used as a continuous input, as well as for triggers. Moreover, as the current system only uses the inertial inputs (acceleration and rotation) for gesture

training and recognition, it would be interesting to allow *all* of the inputs to be part of trigger gestures. Promoting all of the input degrees of freedom to participate in the triggering of gestures would necessarily increase the computational load though, slowing the current system down and could decrease recognition accuracy. Some pre-filtering or on-the-fly data reduction (such as only feeding into the classification algorithm the data from the dominant sensors whose mean variance exceeded a given threshold) could potentially mitigate recognition speed and/or accuracy problems.

Classification latency was mentioned in the Results section. In addition to experimenting with different classification schemes to increase speed, another approach would be to impose a fixed latency that would be predictable to a performer. If for instance, each classification took exactly 500 milliseconds, the player would learn to anticipate this delay in the same way that a drummer learns to begin swinging their drumstick in advance of the desired moment of impact. Haptic or tactile feedback during the classification computation could give the player a sort of "physical progress bar" along the way, which might further relieve perceptual discomfort associated with latency.

In order to support the navigation of more (e.g., tens or hundreds of) possible output sounds, a less linear sound-space navigation method would be necessary. This navigation could utilize a tilting or 3-dimensional position-based interaction, and could allow a more random-access traversal of a perceptual or parametrically defined output space.

Finally, it would be interesting to verify the trends suggested in this paper with similar studies involving different physical controllers and application domains.

ACKNOWLEDGEMENTS

We thank Stacy Morris, Ari Benbasat, Mark Feldmeier and the rest of the Responsive Environments group, James Patten, Jacob Eisenstein, Win Burleson and Mike Norton for their generous assistance with this project. This work was supported by the Things That Think Consortium and the sponsors of the MIT Media Lab.

REFERENCES

1. Abelton Live website. Available at http://www.ableton.com/
2. Bellman, R.E. Dynamic Programming, Princeton University Press, Princeton, New Jersey, USA, 1957.
3. Benbasat, A.Y., Morris, S.J, and Paradiso, J.A. A Wireless Modular Sensor Architecture and its Application in On-Shoe Gait Analysis, in *Proceedings of the 2003 IEEE International Conference on Sensors*. (2003) Volume 2, 1086-1091.
4. Clynes, Manfred. Sentics: the Touch of the Emotions. New York: Doubleday and Company, 1977.
5. Cont, A., Coduys, T., and Henry, C. Real-time Gesture Mapping in Pd Environment using Neural Networks, in the *Proceedings of the International Conference on New Interfaces for Musical Expression.* (Hamamatsu Japan, June 2004), Shizuoka University of Art and Culture, Faculty of Design, 39-42.
6. Cook, P. Remutualizing the Instrument: Co-Design of Synthesis Algorithms and Controllers, in *Proceedings of the Stockholm Music and Acoustics Conference* (Stockholm Sweden, August 2003), 677-680.
7. Cypher, A. EAGER: Programming Repetitive Tasks By Example, in *CHI '91 Conference Proceedings* (New Orleans LA, April 27-May 2 1991), ACM Press, 33-39.
8. Hinckley, K., Pausch, R., Proffitt, D., Patten, J., and Kassell, N. Cooperative Bimanual Action, in Proceedings of CHI '97 (Atlanta GA, March 1997), ACM Press, 27-34.
9. Hunt, A., M. Wanderley, and R. Kirk. Towards a Model for Instrumental Mapping in Expert Musical Interaction, in *Proceedings of the 2000 International Computer Music Conference* (San Francisco CA, August 2000), International Computer Music Association, 209-212.
10. Lee, M., A. Freed, and D. Wessel. 1991. Real-Time Neural Network Processing of Gestural and Acoustic Signals, in *Proceedings of the 1991 International Computer Music Conference* (Montreal Canada, 1991), International Computer Music Association, 277-280.
11. Lexicon MPX110 website. Available at http://www.lexiconpro.com/mpx110/index.asp
12. Merrill, D. FlexiGesture: A sensor-rich real-time adaptive gesture and affordance learning platform for electronic music control (May 2004), S.M. Thesis, MIT.
13. MIDI Manufacturers Association. Available at http://www.midi.org.
14. Modler, P. Neural Networks for Mapping Gestures to Sound Synthesis, in *Trends in Gestural Control of Music* (2000), Centre Pompidou, 301-313.
15. Myers, B. A. Creating User Interfaces by Demonstration. Academic Press, Boston, 1988.
16. Olsen, D. R. Jr., and Dance, J. R. Macros by Example in a Graphical UIMS, in *Computer Graphics and Applications* (January 1988) 8, 1, IEEE, 68-78.
17. OSC Website. Available at http://www.opensoundcontrol.org/
18. Paradiso, J. Hsiao, K., Benbasat, A., and Teegarden, Z. Design and Implementation of Expressive Footwear, in *IBM Systems Journal* (October 2000), Volume 39, Nos. 3 & 4, IBM, 511-529.
19. Pirhonen, A., Brewster, S. & Holguin, C. Gestural and audio metaphors as a means of control for mobile devices, in *Proceedings of CHI '02* (Minneapolis MN, April 2002), ACM Press, 291-298.
20. Pure Data website. Available at http://crca.ucsd.edu/~msp/software.html
21. Schoner, B. Cooper, C. and Gershenfeld, N. Cluster-Weighted Sampling for Synthesis and Cross-Synthesis of Violin Family Instruments, in *Journal of New Music Research (1999)*, Vol 28(2), Swets & Zeitlinger, 81-89.
22. Smith, D.C. Pygmalion, A Computer Program to Model and Stimulate Creative Thought. Birkhauser Verlag, Basel, Switzerland, 1977.