mWorlds: Novel Human Interaction with Virtual Worlds

Guy Garnett
School of Music,
Institute for Advanced Computing Applications and Technologies
University of Illinois
garnett@illinois.edu

Robert E. McGrath
National Center for Supercomputing Applications
University of Illlinois, Urbana-Champaign
mcgrath@ncsa.uiuc.edu

Mary Pietrowicz
Institute for Advanced Computing Applications and Technologies
NCSA, and School of Music
University of Illinois
maryp@ncsa.uiuc.edu

ABSTRACT

Virtual worlds, 3D simulations of real or imagined worlds, are far richer and more dynamic than most 2D computer applications. Extended and augmented realities, and cyberphysical systems, which integrate an experience between both the physical and virtual worlds, provide even more possibilities. We believe this richness cries out for a more expressive, more powerful, more dynamic human control interface. To effect a paradigm change toward traditional human-computer interaction, we are investigating high performance interfaces modeled after the techniques of musicians and other performing artists coupled to machine learning techniques to drive the virtual environment. We approach the problem by extracting structured information from the actions of performing artists, translating that information into an appropriate control language and applying it to high-performance interactions with virtual worlds. To make this work possible, we are creating an integrated framework to simplify the use and exploration of machine learning techniques to mediate between the performer-generated data and the multidimensional virtual representation. Our framework facilitates employing automated learning and data mining techniques to extract features and relationships from multiple streams of data (audio, motion capture, etc.), to discover meaningful performative "gestures", and to provide mappings between multiple semantic domains.

Categories and Subject Descriptors

H.5.1 Multimedia Information Systems, H.5.2 User Interfaces, H.5.5 Sound and Music Computing, I.2.6 Learning, J.5 Arts and Humanities

General Terms

Design, Experimentation, Human Factors, Machine Learning.

Keywords

3D worlds, Interactive performance.

1. INTRODUCTION

Musicians, dancers, and other performing artists have at their disposal years of painstakingly learned, precise and practiced physical skills, aural perceptions, and cognitive and physical awareness and control. We look to the performing arts for models and inspiration to develop radical new approaches to interaction between humans and virtual environments.

Our research brings the creative practice, insights, and skills of artistic performance into the development of novel high-performance human-computer interfaces for virtual worlds. The embodied creativity of artists—initially musicians—has rarely been tapped for the understanding it can provide to control outside of the musical medium itself. Understanding and exploiting the precision and flexibility of performing musicians in controlling their instruments, we believe, may lead to a paradigm shift in approaches to human-computer interfaces more generally. At the same time, it has the potential to significantly impact the arts community itself, a traditionally technologically underserved community: we hope to open up entirely new areas of computer-mediated expression for artistic exploration. This project makes some high risk postulates: first, that performing artists can quickly apply and adapt their learned performance skills to a suitably configured computer interface to effect complex and nuanced control over high density information presented in a multimodal computer interface, and, second, the kind of full body movement such performers are adept at will yield new insights in dynamic interactions with information rich computing systems.

Our methodology brings together musicians who are also computer scientists, and computer scientists who are also musicians. We believe the refined and highly practiced skills of these creative performers can help us develop and design new, transformative interfaces—perhaps entirely new approaches—to control the complex software systems associated with virtual worlds, and in the process, create an extended virtual-physical reality with new creative possibilities for performers. While we focus at first on performing musicians, we believe some our results will generalize to other trained movement experts including dancers, actors, but also athletes and astronauts.

Our approach to virtual world building, unlike the common commercial and standard game approach, does not draw a distinct boundary between creating the world and "performing" or "playing" in it. Our approach to novel interfaces allows for example, a violinist to use his playing as a navigation device in the virtual world, but also to use the same subtle nuanced performance parameters as control parameters for creating and manipulating 3D models in the virtual world.

Our research group has been developing mWorlds [7], an open framework for collaborative, distributed creation and use of such virtual worlds. mWorlds allows us to draw on the skills of creative artists and translate them into a powerful yet natural control of an interactive virtual world. mWorlds has been used experimentally and elements of it have been used in public musical performances (e.g., [18]) and in a museum installation [19].

In this paper, we describe the prototype system that allows for high-level integration of multiple control streams and flexible and varied processing paradigms (including signal processing and data mining techniques), translation and mapping of control and analyzed signals into control of features of a virtual world. mWorlds, makes use of a decentralized, peer-to-peer (P2P) network model, described below. The collection of distributed objects can include signal generation and processing elements, analysis elements, 3D modeling and rendering, manipulation, and control elements, and will be managed by a virtual network of cooperating object managers, subscription services, and other agents on top of a peer-to-peer network. A communication manager will deliver updates to applications and services that subscribe to particular information.

1.1 An Example Scenario

Virtual worlds, such as Second Life, show some of the rich possibilities afforded by networked virtual worlds. However, current environments are very limited in the kind and degree of user interactions available: they are usually restricted to keyboard and mouse interfaces. For example, if you wished to play music in such an environment, you are typically limited to uploading an audio stream, with little or no ability to alter the stream interactively.

Using our system, on the other hand, a highly trained, professional violinist (or other instrumentalist) is able to interact, navigate and project herself into a shared, virtual environment using her performative gestures, and the resultant audio signals, alone. Through a transparent violin-to-virtual-world interface, the musician will control her virtual presentation, eventually including the design and "editing" of virtual objects, virtual navigation, and all other aspects of the virtual world.

Using a combination of audio spectrum analysis, machine learning and data-mining techniques, video motion-tracking and gesture sensors, a rich picture of the violinist's performance gesture data can be realized. These data streams then function as control inputs to a synthetic world. Through careful shaping of a musical improvisation, or careful "performance" of a particular "score," the violinist can propel a virtual avatar through virtual landscapes. With subtle variations in musical timbre the avatar can be driven to mold virtual terrain, create a virtual sculpture, or even "sing" a tune – coming full circle from acoustical musical input to a new electronic musical output now projected in a shared virtual space.

For example, our prototypes have already explored mapping sets of pitches played on the violin to locations in 3D virtual space: playing notes from pitch set A moves you toward location A′, playing notes from pitch set B moves you toward location B′. Relying on the existing skills of musicians—in this case pitch memory, pitch production, melodic improvisation with a collection of pitches, etc.—allows us immediately to begin addressing the heart of the problem: discovering and implementing dynamic mappings between the signals generated by the music interface and the rich possibilities of the virtual world. These interactions enable not only a new realm of aesthetic expressions and experiences as mediated by technology, but also a new model for navigation, creation, and modification of the properties of the augmented reality.

2. Comparison to Other Work

mWorlds will provide an immersive experience in extended virtual reality, and allow us to develop and explore novel approaches to human-computer interaction using the learned techniques of musicians and dancers. Such a system becomes an extension of the performers in it, and the performance will include the virtual reality for both audience presentation and as performer feedback. Using machine learning components, the system will adapt to feedback from performers, change its operation as needed, and provide the change feedback to performers, who, in turn, adapt their performance. This system goes beyond simple 1:1 mappings of inputs to actions.

Existing virtual world systems fall short of this vision in a number of ways:

- Most systems have limited mouse-window-keyboard interfaces, sometimes including a game controller.

- Most systems are limited to a 1:1 mapping of a single input action to a corresponding single resultant action in the virtual world, and do not support many-to-many mappings.
- Most systems do not provide a true augmented reality experience; they simply provide limited controls to operations in the virtual world.
- The controls are static and inflexible (users can't change system behavior, or even modify the mappings of events to actions during runtime).
- No feedback loop between the user/performer and the system exists; responses to actions are fixed.
- Most systems were not designed for use in the creative arts.

Mass market virtual worlds, such as Second Life [16] and World of Warcraft [1] have complex and rich environments, which a user can manipulate in a large variety of ways, but the control interfaces are geared toward low bandwidth mouse-based pointers and alphanumeric keyboards rendered to 2D or 3D graphical displays.

Newer games, such as Guitar Hero [22] explicitly, though primitively, model musician interactions within a game, yet the controller used is an extremely "dumbed down" toy version of a guitar and the game has minimal, if any, virtual world attributes. Wii games are more immersive, but as with Guitar Hero, their interactions are limited to simple game controllers—though in this case including accelerometers—and more-or-less 1:1 relationships between user input and system response.

Traditional motion capture systems are also often used in a similarly limited way: making direct one-to-one correspondences between points on the body of a performer and corresponding points on the body of a virtual character. We build on the work of Dyaberi et al. [5, 6] but rather than treating this as a pattern matching problem we explore the possibility of mapping directly from the characterized and analyzed parameters into the control space. Work on interactive environments by Camurri et al. [3] also shares some relationship to our project. They use gesture capture systems to interact, one-on-one, with a "multimodal" environment, which can include robotic agents and various 2D display devices, whereas our target environment is a multiuser, shared, 3D virtual environment. Goudeseune's thesis [9] provides a more general approach to solving some of these complex mapping problems that we will adapt in order to create mappings into the virtual environment rather than, as in his case, into audio synthesis parameters. Work by Peiper, et al, [12] has used motion capture techniques that focus on mapping sensors on a violin to analyze the motions of the violinist, but do not map them onto any non-audio feedback system.

In most work with audio or physical gestures, there is no high-level representation of the movement or sound at all; there is merely a one-to-one correspondence between a motion or sound and a resultant action. Our approach is to extract higher-level features, and to combine features from many sources. Thus we seek to recognize meaningful gestures in sound, movement, and other actions. Together these form a composite input of derived gestures that are repeatable for trained musicians and dancers.

Well-established projects at NCSA such as SEASR [10] and open source projects (Weka [21], RapidMiner [14]) all provide machine learning and data mining capabilities useful for recognizing gestures and processing complex inputs. Issues not addressed by these environments include real-time performance, suitability for use with streaming data, adaptive run-time behavior modification, post-processing, and event generation and other system-level integration required for application in a distributed virtual world.

Another component of our work is derived from extensive research in virtual reality (VR). VR systems, such as the CAVE [4] use position sensors and head tracking to display and interact in 3D. A good overview of this area is provided by Sherman and Craig [17]. Some of our previous work [8] is related to this: in the Virtual Score, a CAVE application, we created mappings from position and orientation sensors to create an editable, immersive 3D VR 'score.' We could then maneuver within this score by a series of arm and hand gestures derived from musical conducting gestures (waving a baton in time, for example) while the gestures moved the user through the score and simultaneously realized a performance of the score by generating real-time digital audio that correlated with the gestures and the score indications.

2.1 The mWorlds Framework

The mWorlds framework is designed as a decentralized, peer-to-peer system, in which hardware devices and software modules can be dynamically configured to build up and tear down specific applications. This requires a flexible network layer, in which virtual networks can be overlaid on the evolving network, hosts, and software services.

A complete mWorlds implementation will manage one or more collections of objects distributed across a virtual network of cooperating communication managers, subscription services, and other agents that form a peer-to-peer network. Each communication manager is responsible for delivering updates to applications and services that subscribe to particular information.

Careful management of events and streams is required when integrating physical and virtual worlds into an extended reality. Much of the raw data from the real world presents itself as streaming inputs (for example, audio, video, and position/orientation of control devices). The virtual world also has streaming sources such as position, orientation, and acceleration of objects in the space. Furthermore, streaming inputs coexist with discrete events (which could be interpreted as "sparse streams"), such as virtual object creation and destruction, motion detection, and musical motive and gesture detection.

Our ambitious goal presents the following challenges:

1) Event and stream synchronization,
2) Single-scan algorithms for analysis,
3) Data and stream description,
4) Stream summarization and knowledge representation from multiple time perspectives (current, recent, distant, etc.),
5) High-performance requirements to process and respond to streams,
6) Representation of system context and its impact on event interpretation,
7) Stream storage considerations,
8) The need to change analytical techniques over time in response to the changing state of the world, and
9) Definition of appropriate system actions in response to our interpretation of the state of the world.

The example scenario suggests the complex challenges presented by these ambitious goals, as well as the eclectic mix of analytics that might be applied. The mWorlds software provides a flexible framework which enables the use of many different analytic components in an integrated system. Many of the individual components we need to realize the above scenarios have already been developed, some by us and some by other researchers.

A peer-to-peer architecture is flexible and can be used at many scales. However, many aspects of an interactive 3D virtual world are highly sensitive to latency and many operations may involve large amounts of data (e.g., high resolution graphics or real time video), so there will be a need for careful performance tuning and caching of the distributed services.

The decoupled architecture is implemented through an open message passing scheme, similar to the spirit of Open Sound Control [23] and Max/MSP [13], extended to larger networks and a larger variety of equipment. The abstract message passing system is augmented by a subscription service which also manages peer registration and routing.

3. A Decentralized Distributed System

The mWorlds system is a decentralized system with several major components, including:

- mEntityStore – generic store
  - mObjectStore—manages objects of a virtual world
  - mEventStore—manages streams of data and logical events
- Stream Management
- Computational Simulations (e.g. physics)
- Computational Analytics (e.g., machine learning to detect events in data streams)
- Interactive I/O
- Interactive 3D graphics

These components communicate through the Communication Manager, which manages subscriptions and notification (see Section 3.3). The Communication Manager uses a Peer-to-Peer message passing system, built on open software.

The system has a persistent state represented by an abstract mEntityStore. The mEntityStore supports entity management, entity identification and typing, property and descriptor management, storage, change monitoring, and continuous query. This shared software layer allows us to address distributed computing requirements and peer-to-peer functions in a common way for streams, events, and objects.

Two subclasses of mEntityStore provide specialized management for “Objects” (mObjectStore, section 3.1) and “Events” (mEventStore, section 3.2). The loose coupling between object and event representation and the rest of the system also allows us to optimize performance of the display in the virtual world, and it gives us low-level control of timing that we will require to address synchronization issues in the future.

Interactive applications subscribe to objects and events from one or more mObjectStores and mEventStores (section 3.4). The communication manager provides data to generate and update an interactive 3D scene.

3.1 Virtual World: the mObjectStore

The mObjectStore provides a persistent store for collections of objects of a virtual world, along with interfaces to update and access the objects state. The mObjectStore is a distributed service; each instance may manage some of the objects of the virtual world, possibly including replicates and cached copies.

Though it is designed to be independent of any particular renderer, the mObjectStore has a generic data model that includes elements that are analogous to scene graphs used in graphics applications (e.g., X3D [2]), to geometry meshes, and other graphics-related elements. The object data includes arbitrary properties, including, when relevant, geometry and appearance data.

The mObjectStore manages updates to the virtual world, whether from manual interactive “editing” or simulations (such as physics simulations) or another source. The mObjectStore produces a continuous log of all changes to the objects managed by the store. The Communication Manager matches these changes to subscriptions, and dispatches requested data to any subscribers.

Interactive applications subscribe to data from a group of objects (e.g., within a given distance) from one or more mObjectStores. Computational components may also update the objects in the mObjectStore. For example, as a physics simulation computes new positions of an object in a virtual world, the updates are sent to one or more mObjectStores, which in turn update subscribers.

3.2 Events and Streams: the mEventStore

The infrastructure we are designing will process events and streams from multiple sources, summarize and store knowledge extracted from streams at different time perspectives, support pluggable, dynamic, networks for fuzzy stream analysis, and provide dynamic mapping of actions.

The mEventStore is similar to the mObjectStore, managing a collection of streams of events. As in the case of the mObjectStore, components subscribe to receive events. Data may be collected from many sources, including sensors such as accelerometers, video, and audio. This data is converted into streams, which may trigger events or be subjected to further analysis. With this system, for example, it is possible to extract musical motives and gestures from live audio, extract light and color palettes from the extended reality, and respond with dynamically mapped changes in virtual world textures, physics engine operations, visual displays, and audio processing.

Beyond the common entity layer for events and objects, events of all types maintain origination information, event-specific description and typing, and source identification and description. Streams require even further support. They are continuous, ordered, and time-sensitive, so they include support for monitoring the stream life cycle, managing the components (ordered bundles of information that pass through them), describing content and type of stream data, stream sequencing, and basic storage. Composite events go beyond the level of simple streams and discrete events: they are event collections that allow a software agent to communicate with arbitrary groups of streams and discrete events.

An mWorlds software agent or client may create, send, and query for events and event properties. A continuous query of an event stream is a filtered input stream for the client. We have used this functionality in a visualization that translates regional motion of people in the physical world to regional motion of waves in a body of water in the virtual world. To accomplish this, a video capture client analyzed differences in activity in the physical space and transmitted location and difference information over a stream. A second client monitored the stream for changes, and mapped location and difference information to a region in the body of water and the strength of the resulting water waves, respectively.

3.3 Subscriptions and Message Delivery: Communication Manager

The components of the system communicate through message passing to coordinate and maintain a consistent state across the distributed objects. A fundamental goal of the Communication Manager (CM) is to route messages to the intended consumers, and to deliver the messages with an appropriate quality of service.

The messages include state changes of the objects (e.g., position), sensor readings (including audio and video streams), and events generated by programs. The messages are heterogeneous, and various messages must meet different constraints. For example, position updates from a sensor are typically small messages (such as three floats), but may come at a relatively rapid rate (100s of times per second), yet the data semantics often allows lost messages to be ignored, rather than resent, in favor of more current data. On the other hand, audio and video require more bandwidth, usually must meet latency constraints, and may be less tolerant of dropped or out of order messages.

The subscription service is a distributed registry that allows instances of the CM to request certain data, e.g., position updates for a specific set of objects. The subscription service performs the initial matchmaking, locating sources that may provide the requested data.

The communication manager will cooperate with brokers and agents that can optimize network use for particular cases, through a combination of strategies, such as:

- Data specific quality of service
- Collocating tightly coupled data and services
- Progressive transmission, compression
- Optimized routing schemes (e.g., a virtual routing tree)

3.4 Interactive 3D Graphics

The virtual world will be used through a variety of interactive 3D graphical environments, built using open source tool kits such as OGRE [11], blender [15], and Max [13]. These components provide a view into the virtual world(s), through which the users can manipulate the objects, including interactive "editing".

Interactive applications subscribe to a group of objects (e.g., within a given range within the virtual world) from one or more mObjectStores. The communication manager provides data to generate interactive 3D graphics, and to update the state of the scene. The application may also subscribe to specific kinds of data and/or events, which are used to control the behavior of the virtual world.

For example, a user might use motion sensors to 'edit' a 3D shape. The motions will generate a stream of data, which will in turn generate a series of logical Events (e.g., representing the "gestures" of the user). An editor program may use these gestures to generate and modify the objects in one or more mObjectStores. The resulting changes will be delivered to any subscribers.

4. Data Analysis, Machine Learning, and Transformation

The mWorlds environment provides a flexible framework that enables the use of many different analytic components. The complex and rich data from the environment must be analyzed, transformed, and mapped into meaningful events in the system in a way that will serve many applications and artistic visions. Our approach uses a combination of off-the-shelf components, custom heuristic processors, and machine learning techniques. Each kind of component serves a specific purpose. Whenever possible and practical, we use off-the-shelf components, such as Max/MSP [13] (and Max/MSP externals) for flow control and audio processing. We developed heuristic processing modules to analyze aspects of sound or other input data that are easy to quantify, derive, or calculate. For example, determining the average intervallic distance between notes over time is a fast, efficient mathematical calculation. When we are analyzing data qualitatively, however, particularly when doing so is programmatically difficult, inefficient, or inaccurate, we use machine-learning techniques. Sound gesture recognition is an example of a task well suited for machine learning. The combined approach provides an efficient, flexible result that can deliver many–to-many mappings between the performer's input and events in the virtual world. Machine-learning techniques also provide the flexibility for runtime adjustment of system behavior without changes to the codebase. Ultimately, the system will support an element of feedback so that both system and performer can adjust their actions and responses during performance.

In order to demonstrate our approach, we developed a reference application that processes audio data from a live musical performance, recognizes qualitative sound gestures in the music, transforms the sound gestures into a multi-parametric event stream, and responds visually in the virtual world. The virtual world in this application becomes an extension of the performer and the musician's instrument: a novel human-computer interface.

Our reference application detects the degree of disjunction and separation in the notes played, which we refer to as its degree of "pointillism." Highly pointillistic sound has a disjoint quality and features maximum difference in timbre, loudness, and pitch. The listener may perceive pointillistic sound as being "pointy" or "spiky," so, for demonstration purposes, we translated this quality into the degree of spiky effects in the visual, virtual world. The greater the degree of pointillism, the more "pointy" or "spiky" our primary object appears in the virtual world (in this case, an icosahedron) appears. In addition, we analyzed the distribution of pitch classes (the twelve notes in the musical chromatic scale), and transformed this into activity at one of the twelve vertices on the icosahedron. When a musician plays something, the icosahedron changes shape to reflect the relative prevalence of each pitch class. If the musician plays only the note 'C,' for example, the vertex corresponding to the note 'C' on the icosahedron will extrude outward. When other notes are included, their corresponding vertices will extrude outward in proportion to the relative number of times they occur. Our reference application is a relatively simple test case, but one that we could explore in detail and demonstrate our concepts effectively.

The following sections describe each stage of processing, from data input and analysis, to transformation and mapping in the virtual world. Figure 1 below provides a pictorial overview of the process. We also discuss the flute as a novel human-computer interface into the augmented reality, with aesthetic considerations for performance. Finally, we discuss future directions for our work.

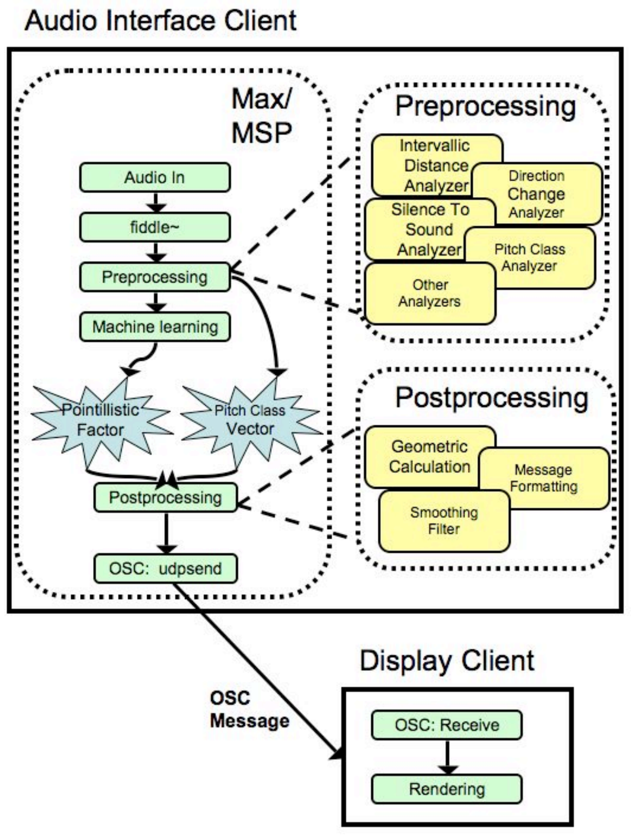

Figure 1: Reference Application Overview

4.1 Data Analysis and Real-Time Heuristic Processing

As shown in Figure 1, the reference application uses the Max/MSP application for flow control and off the shelf audio processing. It accepts live audio input from the built-in analog-to-digital converter in Max, and routes the stream to “fiddle~”, an external Max/MSP application for pitch tracking [24] .

A suite of real-time analysis modules accepts the output stream from fiddle~, pushes it through a sliding time window, and continuously processes the data within the context of the window. The primary heuristic modules for the reference application include

- A pitch class analyzer,
- An intervallic distance analyzer,
- A silence-to-sound ratio calculator, and
- A direction change analyzer.

The pitch class analyzer calculates the relative frequency of occurrence of each pitch class within the current time window. The intervallic distance analyzer calculates the average distance between notes. The silence-to-sound ratio calculator detects the degree of connection between notes, and the direction change analyzer detects the amount of changes in pitch direction (an aspect of melodic continuity or discontinuity). Each of these analysis modules produces an independent output stream, which will be used in later stages of processing.

A limited static analysis based on heuristics only is possible. However, a heuristics-only approach requires software changes for each new feature or functional adjustment. It also does not provide a path to achieving run-time feedback and adjustment of behavior. Heuristic analysis is static and inflexible when used alone.

Heuristic analysis techniques, however, have advantages that enhance system performance when they are used as preprocessors to machine learning techniques. First, preprocessing extracts relevant, measurable features from sound, allowing the machine learning techniques downstream to focus on the fuzzy, qualitative aspects of music and the detection of higher-level sound gestures. Preprocessing also generalizes the input processing for many sound sources, so that differences in timbre, for example, do not throw off the learning algorithm. Finally, a layer of preprocessing enables a wide range of higher-level processing. Intervallic distance measurements, for example, will be useful for detecting audio “sweep” gestures as well as pointillistic gestures.

4.2 Gesture Recognition and Machine Learning Techniques

Musical gestures are collections of sounds that tend to be heard as a single unit, and which may be given meaning. Some typical sound gestures include

- Sweeps (rapid movement in a single general direction)
- Periodicities (repetition of melody or rhythm, rapid alternations such as trills and tremolos, regular variation in pitch or loudness within a note, such as vibrato, and regular patterns of change in speed, loudness, or timbral quality)
- Transition gestures (e.g., transition from nonpitched, airy, or noisy sound to a sound with a definite pitch)

Gestures tend to be subjective and qualitative, present to greater or lesser degree. If “ideal” sweeps are rapid and unidirectional, sweeps that move more slowly than the ideal are still sweeps, but they are “less strong” on the continuum of sweeps. Gestures may also combine, as in periodic, or repeated sweeps, and they may be embedded within other gestures (e.g., trills within sweeps), making segmentation difficult. These qualities mean that boundaries are fuzzy, and that gesture recognition should not be a binary function.

Machine learning techniques are well-suited here because they easily support classification schemes across a continuum (from “not a sweep” to “strong ideal sweep”). Classification algorithms may output statistical confidence (70% likely to be a member of class A, 30% likely to be a member of class B, and 0% likely to be a member of class C). The reference application treats pointillism this way and classifies all music on a continuum of bins ranging from “very smooth” to “smooth” to “mixed” to “pointillistic” to “very pointillistic”. The algorithm just has to “box” the appropriate spot on the continuum, and misclassifications are minimized.

Machine learning techniques also support our future work in that it is possible to “retrain” an algorithm during runtime. A performer can reinforce or discourage the actions of the machine learning algorithm to produce desired results.

We trained a set of Weka classification models [21] to recognize and categorize the degree of pointillism in live music. Each classification model used the outputs from preprocessing (streams of intervallic distance, silence, and pitch direction).

Providing pointillism training for our reference application was difficult at first. Just as classification of qualitative characteristics can be subjective, selection of training data sets may also be subjective. We considered using improvised music and music from the literature. Improvised music, however, had a wide variance (i.e., it was difficult to keep some aspects constant while varying just the degree of pointillism), and it therefore had mixed classification results.

Using music from the literature was also problematic because a wide selection was required for optimal training. Evaluating, playing, and recording excerpts from this category was time consuming, and it was difficult to find “ideal” examples (most music just isn’t written this way). Furthermore, limited choices exist for the most pointillistic category. Instead, we trained the models in scalar and intervallic technique exercises for the flute. These exercises (drawn heavily from Taffanel and Gaubert [20]) are intended to be played with a variety of articulations. We simply extended them with quantified ratios of connected vs. separated notes, direction changes, and intervallic content. Then, we drew statistical boundaries across the training sets. The least pointillistic training sets had the smallest amount of separated notes and direction changes, and the smallest average intervallic distance between notes, while the most pointillistic training sets had articulated notes with silence separating the notes, many direction changes, and wide intervallic content.

In the course of creating the reference application, we developed a library of tools that enabled Max/MSP applications to train Weka machine learning algorithms and interact with the resulting models in real time. Some of these capabilities include the collection of training data into input fragments, the assembling of training data fragments into a single training file, the management of multiple models within Max/MSP applications, the invocation of trained models with data from live input streams, and the interpretation of output from machine learning models.

In addition to our reference application, we have used automated learning techniques to successfully process sound streams to detect transposition within a musical set class. While this is a simple problem, easily achieved by other methods, it allowed us to test our framework and verify results. We used a sliding window technique both for training and processing, and achieved greater than 90-95% accuracy with the Weka "Decorate," "BayesNet," and "LMT" classifiers, and with a "Multilayer Perceptron" algorithm. These classifiers handled transitional passages particularly well and correctly classified the gray areas with reasonable probability of belonging to more than one set class. We also demonstrated our analysis by generating an accompaniment that followed the transposition changes in live sound.

4.3 Transformation and Mapping

The raw output of heuristic or machine learning models must be transformed into something meaningful and useful within the virtual world. In our reference application, we examined the relative emphasis of all of the pitch classes played and the degree of pointillism present in a time window, and transformed these measurements into a continuous stream of geometric outputs.

The first set of postprocessing modules resolved the probabilistic classification outputs from machine learning models into a single floating point number scaled between 0-100. The second main postprocessor, the "geometric translator," combined the resulting pointillistic factor and pitch class vector into a description of a shape that is more or less spiky, depending on the degree of pointillism, with extrusions at the appropriate locations on the icosahedron corresponding to the twelve pitch classes.

As long as continuous audio input was present, the application produced a continuous stream of geometric outputs. Finally, a message formatter distilled the desired geometric description output into an OSC [23] message, and sent it to listening applications.

Our display client receives the OSC message types it has requested, and it updates the icosahedron on the display according to the geometric description in the message. From the perspective of the display client, these OSC messages are also input streams.

Instead of relying on just one-to-one mappings from simple interface devices, we have used machine learning techniques to analyze the relationships among the various streams of data extracted from the performer over time, and multi-layered mappings are the result.

4.4 The Human-computer Interface and Aesthetic Considerations

We explored the range of controls possible with the flute as a human-computer interface and controller into the reference application. Pointillistic classification and pitch class tracking required a limited amount of analysis, but the relatively large range of visual effects a performer could produce was surprising. As we expand the range of controls, the performer's challenge may not be how to create a wide range of desired effects, but how to reasonably constrain the response in the virtual world.

A composer or performer working with our system is essentially working with a new instrument. The flutist is no longer playing just the flute, but an instrument that can produce combined audio and visual effects. The first task of the performer is to learn how the system responds to various kinds of sounds and playing styles. The flutist must know what is likely to occur in response to all playing techniques, and must practice evoking the new, extended responses. This step gives the performer technical fluency. The second task of the performer is to understand the impact of the combined audio and visual effect so that she can play expressively. Sound alone has one kind of impact or meaning, but the same sound combined with visual effects is a different entity with potentially different impact and meaning for an audience.

As an example, it is possible to play something in a very pointillistic style that sounds big and climatic with a correspondingly large, imposing, spiky display. It is also possible to play something in a pointillistic style that sounds light, almost with puffs of air between the notes, but which also produces a ferocious "spike-ball" on the display. The effects are very different, and the composer or performer will need to be in conscious control of the entire result.

In order for a performer to learn the technique of the new instrument, and play with expression, the system must provide logical cues that the performer can use to connect his actions with the system response. The cues must be consistent, and repeatable, because a performer cannot control the system without them. In our reference application, for example, the performer can always “grow a spike” on the icosahedron by playing a single repeated note, and the note always corresponds to the same spot on the icosahedron. This relationship is consistent and easy to interpret in performance. Having a logical relationship between the system’s cues and responses and the performer’s input will become an even more important consideration when we include a feedback loop that allows the performer to change the system’s responses.

Because of the modularity of the mWorlds framework, performers and composers can easily create different vertex/pitch mappings on the icosahedron for their pieces and playing styles. For example, if a hexachord and its complement map to opposite sides of the icosahedron, different sides of the icosahedron will animate when the performer plays over the two different pitch sets. Or, for a piece featuring many major and minor second intervals, the composer may use a layout with adjacent pitch classes having adjacent vertices. Then, the chromatic melody will appear to “crawl” over the surface of the icosahedron. Both tuned presets and real-time adjustment using a feedback loop will be useful in supporting the desires of performers and composers.

4.5 Future directions

The relationship between performer action and system action is crucial. We will continue to expand the ways that the performer can interact with the system, and the system can inform the performer about its actions. We will also explore mutual feedback mechanisms that allow the user to tune system response in real time. This strong feedback mechanism will respond to the “human in the loop,” and will take advantage of the performer’s training and background. Both performer and system can adjust their responses until a satisfactory result is achieved.

Our approach augments this with machine learning techniques and the use of structured, higher-level semantics. We think ultimately that sound gestures can be used almost as a control or interaction language, with a user-definable vocabulary and rich, contextual meaning. These language semantics will then be driven by the intentions of the performer. Similarly, we intend to characterize actions and activities in the virtual environment with related concepts, and use the semantic representations to discover relevant mappings between gesture and virtual world response.

These extensions will give us an environment for exploring aesthetics of the combined real and virtual environment as an extended musical instrument and performance environment. Attempting to use the environment to develop new creative works will, in turn, provide practical drivers for further developments.

5. Conclusions

The mWorlds project is developing and exploring new paradigms for user interaction and control of virtual environments within a specific context of performing musicians to develop and extend its control interfaces to incorporate more fully embodied human gestures through movement and sound, and allowing us to create a multimodal interface driven—uniquely—by a performing musician model.

The software framework builds on and integrates open source software from several sources, in a flexible peer-to-peer architecture.

Automated learning and data mining techniques are a critical technique for dynamic analysis of sensor data to understand the users behavior, and to map between the real world and the virtual world.

While most human interfaces to computers are low bandwidth and are designed for the lowest common denominator, our approach uses a music performance model because we believe the refined and highly practiced skills of these creative performers can help us understand, develop and design new transformative interfaces to control the complex software systems associated with virtual worlds. We are no longer in the age where computers are novel to most users and there is no longer much need to make the interfaces as easy as possible at the expense of their full power. We believe studying musicians will help us bring about a paradigm shift in computer control and give us new insights into designing high performance computing systems. Just as musicians spend a lifetime mastering their instrument and utilizing its full expressive potential, we believe for at least certain computer users, especially in the realm of virtual worlds, a lifetime of mastering new technology ought to be rewarded with transformative new capabilities. Indeed, our experience as musician/computer scientists led us to the notion of using musicians and the control interfaces they are experienced with (violins, etc) as the key to controlling the vast, almost unlimited, possibilities of virtual worlds.

Achieving our goals will require many advances and entirely new approaches to navigating, controlling, manipulating, and interacting within multiuser virtual worlds. The ongoing mWorlds project provides a scalable framework, including real time audio, machine learning, and virtual environment representation, that facilitates exploring this research area and our diverse interdisciplinary team of musicians and computer scientists provides the domain expertise.

Whether or not we can achieve breakthroughs in designing and conceptualizing user interfaces to virtual worlds, we are certain to open up entirely new areas for enhanced creative exploration.

We believe that, just as musicians spend a lifetime mastering their instrument and utilizing its full expressive potential, mastering technology ought to be rewarded with transformative new capabilities within virtual worlds. Solutions to the computing problems associated with these interfaces will extend new creative and commercial possibilities to broad new communities, not only in the performing arts, but it will also lead to new insights applicable to high-dimensional, multimodal interfaces to complex simulations and cyberphysical systems.

6. ACKNOWLEDGMENTS

Special thanks to the Institute for Advanced Computing Applications and Technologies (IACAT) and the National Center for Supercomputing Applications (NCSA) for supporting our collaborative work.

7. REFERENCES

[1] Blizzard.com, *World of Warcraft Community Site*, (2007), http://www.worldofwarcraft.com/index.xml.

[2] Brutzman, D. and L. Daly., *X3D : extensible 3D graphics for Web authors*, Boston, Elsevier/Morgan Kaufmann. 2007.

[3] Camurri, A. and P. Ferrentino, *Interactive Environments for Music and Multimeda.* Multimedia Systems, *7*, 1 (January 1999) 32-47.

[4] CAVE, *Cave Automatic Virtual Environment*, (2009), http://en.wikipedia.org/wiki/Cave_Automatic_Virtual_Environment.

[5] Dyaberi, V., H. Sundaram, T. Rikakis, and J. James. *The Computational Extraction of Spatio-Temporal Formal Structures in the Interactive Dance Work `22'.* In: *Asilomar Conference on Signals, Systems and Computers, 2006. ACSSC '06*. (2006).

[6] Dyaberi, V., H. Sundaram, T. Rikakis, and J. James. *The computational extraction of temporal formal structures in the interactive dance work '22'.* In: *14th annual ACM international conference on Multimedia*. (2006) 748 - 751.

[7] Garnett, G., R. McGrath, and R. Campbell, *Virtual Worlds: Infrastructure for Large-scale Collaboration*, (2007), http://www.culturalcomputing.uiuc.edu/mWorlds.pdf.

[8] Garnett, G.E., K. Choi, T. Johnson, and V. Subramanian. *VirtualScore: exploring music in an immersive virtual environment.* In: *Symp. on Sensing and Input for Media-Centric Systems (SIMS)*. (2002) 19-23.

[9] Goudeseune, C., *Composing with Parameters for Synthetic Instruments*. Music. D.M.A., Urbana, University of Illinois, Urbana-Champaign. 2001.

[10] Llorà, X., B. Ács, L.S. Auvil, B. Capitanu, M.E. Welgey, and D.E. Goldberg. *Meandre: Semantic-Driven Data-Intensive Flows in the Clouds*. In: *4th IEEE International Conference on e-Science* (2008).

[11] OGRE3D, *OGRE 3D: Open Source Graphics Engine*, (2007), http://www.ogre3d.org/.

[12] Peiper, C., D. Warden, and G. Garnett. *An interface for real-time classification of articulations produced by violin bowing* In: *Conference on New Instruments for Musical Expression, NIME* (2003) 192-196.

[13] Puckette, M., *Max at Seventeen.* Comput. Music J., *26*, 4 (Dec. 2002) 31-43.

[14] rapid-i, *RapidMiner*, (2009), http://rapid-i.com/content/blogcategory/38/69/.

[15] Roosendaal, T. and S. Selleri, *The Official Blender 2.3 Guide: Free 3D Creation Suite for Modeling, Animation, and Rendering* San Francisco, No Starch Press. 2005.

[16] Rymaszewski, M., W.J. Au, M. Wallace, C. Winters, C. Ondrejka, and B. Batstone-Cunningham, *Second Life: The Official Guide*, Indianapolis, Wiley. 2007.

[17] Sherman, W.R. and A.B. Craig, *Understanding Virtual Reality: Interface Application and Design*, San Francisco, Morgan Kaufmann. 2003.

[18] Smith, B. and G. Garnett, *MusiVerse*, International Computer Music Conference, 29 August, 2007.

[19] Smith, B.D., *Musiverse*, CANVAS (Collaborative Advanced Navigation Virtual Art Studio), Krannert Art Museum, April 4 through June 1, 2008.

[20] Taffanel, P. and P. Gaubert, *17 Big Daily Finger Excercises for the Flute*. 1958.

[21] Weka, *Weka Software for Data Mining and Machine Learning*, (2009), http://www.cs.waikato.ac.nz/ml/weka/.

[22] Wixon, D., *Guitar Hero: the inspirational story of an "overnight" success.* interactions, *14*, 3 (2007) 16-17.

[23] Wright, M., *Open Sound Control: an enabling technology for musical networking.* Organised Sound, *10*, 3 (2005/12/01 2005) 193-200.

[24] Puckette, M., T. Apel, and D. Zicarelli. *Real-time audio analysis tools for Pd and MSP.* In: *ICMC.* (1998)