MusiKalscope: A Graphical Musical Instrument

Sidney Fels
The University of British Columbia

Kazushi Nishimoto and Kenji Mase
ATR Media Integration & Communications Research Laboratories

A new multimedia system for musical and graphical expression called MusiKalscope contains subsystems that support computer-assisted jazz improvisation (RhyMe) and kaleidoscopic imagery (the Iamascope). The Graphical Musical Instrument Interface (GMII) connects these subsystems and lets performers create images and music as they play virtual drums. A functional map matches the images' mood to the music played.

Imagine a system where a performer creates art by playing a trumpet with one hand and painting with the other simultaneously. Intuitively, creating art this way, while possible, seems fairly impractical when it comes to controllability and mastery for effective expression. However, we believe that most multimedia art-generating systems urge performers to do just such things. We contend that for an artist to create good artwork with individual media such as paint or music proves difficult enough. To create them simultaneously without adequate support is nearly impossible.

We believe that using several support mechanisms to reduce the performer's cognitive load proves essential for simultaneous production in a multimedia computer-supported art system (though excessive support should be avoided). The supporting method shouldn't obstruct performers' creativity and should leave enough room for their expression. In this sense, we believe that multimedia art systems—which automatically control several media completely—don't suit artists very well. To create a good multimedia art-generating system, we suggest supporting a minimum "good enough" quality as well as allowing room for performers' creativity in all media. In this article, we introduce MusiKalscope,[1] which represents such a system.

To use MusiKalscope, you enter a dark space and pick up the virtual drumsticks. You see a beautiful kaleidoscope image of yourself on the huge video screen in front of you while jazz music starts to play in the background. As you strike the virtual drum pads in front of you, different tones selected by the RhyMe subsystem accompany the background music. At the same time, the kaleidoscopic image onscreen responds to you. When you play tension-like sounds, the image turns bluer. It returns to its original state (with no additional color mixed in) when you play chord tones. As you move and play, a beautiful mixture of imagery and music engulfs you. Figure 1 shows an example of the MusiKalscope system.

In MusiKalscope, each action the performer executes controls all media. Thus, even if one of the actions targets a specific media, such as producing music, the other media will follow according to the action's function for the specified medium. MusiKalscope does this by defining a mechanism for the performer's actions to adjust a piece of music's "color" rather than the melody while a song plays. It also assigns the same action to control the "color" of the computer graphic image. At the same time, the performer's movement and dance and the music's rhythm during performance translate directly to the aesthetic feeling provided by the visual imagery.

Other systems such as Brush de Samba, Cindy,[3] DanceSpace,[4] and Muse[5] relate to MusiKalscope. Brush de Samba relates most closely to our work—it generates music and computer graphics simultaneously. In Brush de Samba, a performer draws a picture using a drawing pad. Music generates automatically based on the pen position data. The performers generally focus on the image they're drawing rather than the music being generated. Hence the system is biased for graphical input, making it difficult to control the music quality in the system.

In contrast, Cindy is musically biased. In this system, performers play music, and an animated character named Cindy dances along to the music played. The music style controls the dance style by changing musical parameters such as the number of notes played, playing a single tone or chord, and the beat.

In DanceSpace, a video camera captures the performer's dance. The performer's movement is mapped so that the dancer's hands and feet con

trol virtual musical instruments, and the dancer's head height controls the music's pitch. At the same time, the dancer's motion creates and controls computer graphics. A colored outline of the dancer's body is represented successively. With DanceSpace, various music styles can be played. However, the sounds generated continuously ascend or descend, significantly impacting the music's quality.

Finally, Muse provides a musical interface to control a computer-generated character. Muse interprets the music played as an emotional context displayed by the Muse character. Tosa and Nakatsu[5] describe the musical grammar elsewhere. Interestingly, the computer graphics coordinates with the musical sounds. However, the grammar limits the musical scope when using Muse.

In creating MusiKalscope, we hoped to achieve three main goals as follows:

1. Maintain a good balance between the quality of computer graphics and music generated.
2. Let novices easily achieve a reasonable quality of music and imagery.
3. Impose no performance ceiling so that with more training, enhanced expression becomes possible.

Overview of MusiKalscope

MusiKalscope consists of three subsystems: the Graphical Musical Instrument Interface (GMII), RhyMe, and the Iamascope as shown in Figure 2. The GMII provides the input interface between the performer and the two other subsystems. The GMII uses two main input devices: a video camera and a Polhemus Fastrak. The video camera input feeds directly to the Iamascope for the imagery. The Iamascope's output displays on a large video screen (170 inches) in front of the performers so that they can see the imagery they're producing.

The Polhemus Fastrak data is analyzed to implement a virtual drum interface to RhyMe and the Iamascope. The name of the virtual drum pad the user strikes passes to RhyMe (see the section on the RhyMe subsystem). RhyMe then plays the particular note appropriate for that drum pad at that moment during the song (according to jazz analysis of the song being performed). The output from RhyMe plays on speakers mounted on the sides of the large video screen.

RhyMe also dictates whether the virtual drum pad struck corresponds to a chord tone or a tension tone. Currently, RhyMe determines this mapping a priori and embeds it in the GMII, which sends the appropriate signal to change the Iamascope's appearance. When the performer plays a tension note the Iamascope image becomes bluer. When the performer plays a chord note, the Iamascope returns to its normal color. The faster a user plays the virtual drum pad, the brighter the Iamascope image becomes. All changes in color and brightness to the Iamascope image will gradually return to the "normal" state if the user stops playing.

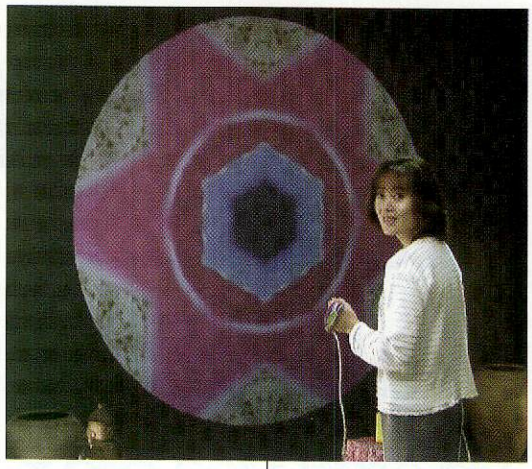


Figure 1. Captured image of a person having fun in MusiKalscope. The video camera is at the bottom of the 170-inch screen.

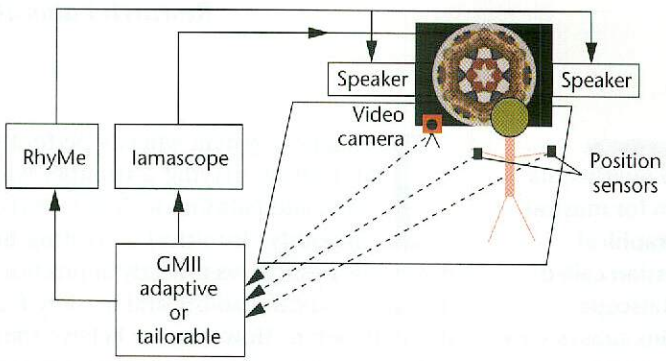


Figure 2. Block diagram of MusiKalscope. The video camera and position sensors capture the performer's (shown on the right) motions. The video image serves as input to an electronic kaleidoscope, which outputs a kaleidoscopic image on the large screen in front the performer. The position sensors create a virtual drum. Striking the virtual drum pads produces different notes that accompany the jazz background music. The RhyMe subsystem selects which notes to play—according to jazz theory—when a performer plays the virtual drums.

The Iamascope subsystem

Kaleidoscopes have captured imaginations all over the world since Sir David Brewster first invented them in 1816. The Iamascope, an interactive kaleidoscope, uses computer video and graphics technology. In the Iamascope, the performer becomes the object inside the kaleidoscope and sees the kaleidoscopic image on a large (170-inch) screen in real time. The Iamascope offers an

How a Kaleidoscope Works

You can make a simple kaleidoscope from two mirrors, some black paper, and objects to view with the kaleidoscope. Place the two mirrors on edge and use the black paper to form the other side of the long triangle as shown in Figure A. The two mirrors will reflect whatever you put at the other end, forming an image of a circle of reflected copies of the objects in it. If you adjust the mirrors so that they're an even fraction of 360 degrees, say 30 degrees, the reflections made with the two mirrors will line up to give a beautiful symmetric image as shown in Figure B. For a 30-degree mirror angle, you get 12-fold symmetry. Other angles give other symmetries. If you use three mirrors, the image appears to spread out forever. Depending on the relative angles of the mirrors to each other, you can see different symmetries. The simplest arrangement is an equilateral triangle. Thousands of variations of kaleidoscopes exist, many of which Baker[8] discusses in the book *Kaleidoscope Renaissance*.

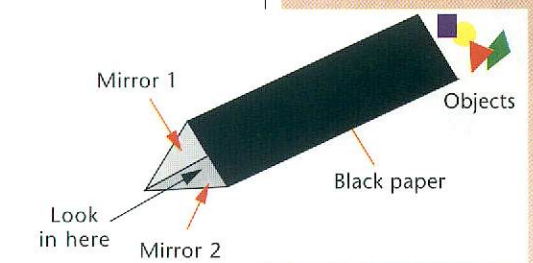


Figure A. How to make a simple two-mirror kaleidoscope.

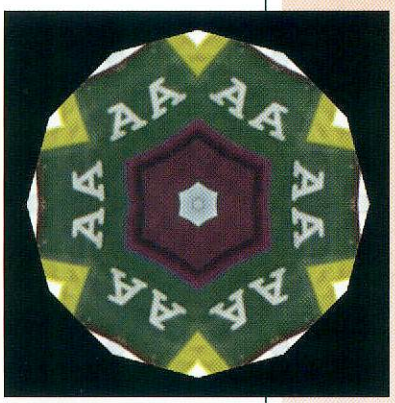


Figure B. Example image from a two-mirror kaleidoscope using 12-fold symmetry.

example of using computer technology to develop art forms. As such, it doesn't enhance another device's functionality. Rather, it provides a rich, aesthetic visual experience for the performer using it and for people watching the performance.

The Iamascope is more than a mirror-based kaleidoscope (see the sidebar "How a Kaleidoscope Works") put in front of a video camera, since a computer can generate more extensive reflections such as asymmetric reflections and different tiling patterns than mirrors. Figures 3, 4, and 5 show some of the possible images the Iamascope can produce. In Figure 3, the Iamascope image uses two mirrors, while three mirrors are simulated in Figure 4. Figure 5 shows a visually interesting pattern. Here the three-mirror version wraps around a spinning sphere. While visually appealing, the

Figure 3. Snapshot of an image created with the two-mirror kaleidoscope.

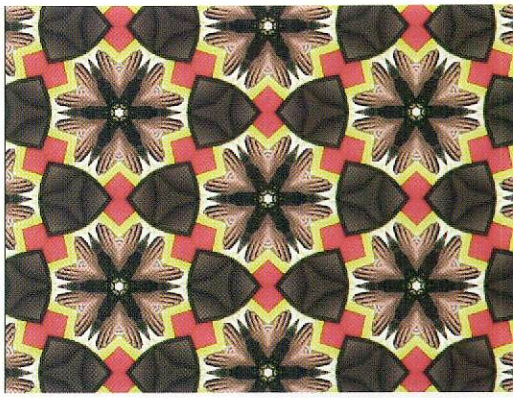


Figure 4. Snapshot of the three-mirror kaleidoscope image. Notice the performer's hand in the image.

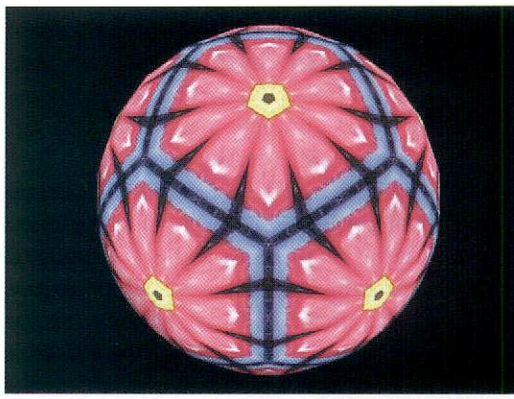


Figure 5. Snapshot of the three-mirror kaleidoscope wrapped around a sphere. The sphere's revolutions per minute can be adjusted.

sphere image isn't as easy to control compared t the nonmoving surfaces in the two- and three mirror Iamascope versions.

From the perspective of balanced, lower bound ed quality multimedia, the Iamascope possesses on

mportant quality—for the most part, anything laced inside it will look beautiful. The symmetries f the mirror reflections appeal to many people. hus, as part of a multimedia system, the lamascope rovides a reasonable minimum level of aesthetic uality even when used by novices. However, as erformers learn to move their bodies to manipuate the kaleidoscopic image, they can achieve reater forms of expression. Hence the lamascope rovides an upward pathway for achieving highly killed forms of expression through visual imagery.

Figure 6 shows a block diagram of the umascope. For input, the lamascope uses a single ideo camera connected to a video board with a rain to texture memory. Output appears on a ideo monitor. In our current implementation, the ideo image from the camera goes into texture nemory. Then, the subsystem selects the appropriate part of the video image (currently a "pie" ice but also referred to as a segment) to form the riginal image (O). The subsystem reflects the original image, creating the desired reflections (O'). In he two-mirrored version, the subsystem draws a nultipolygonal circle and applies the appropriate extures (original or reflected) alternately. The necssary reflections are simulated with texture hardware providing frame rates of 30 frames per second. his frame rate provides low-latency, high-andwidth control of the kaleidoscopic image, supporting a sense of intimacy with the lamascope.

The lamascope used in MusiKalscope is based n the reflections found in a two-mirrored kaleioscope. However, we used a pie slice (segment) rom the original video image instead of a trianular slice typical of two-mirror kaleidoscopes. hus, if the slice's arc angle represents an even nteger divisor of 360 degrees, a circular image orms using the alternation of the original image nd its mirrored reflection (as shown in Figure 6). he even integer multiple arc angle is required so hat alternating the original image with its reflecion will fill the circle exactly. For example, if we se a 30-degree pie slice, then 12 segments will ake up the circular image. The odd segments ill have the original image, and the even ones ill contain the mirror reflection. Three aspects of his method provide a beautiful effect:

1. Because the segments fill the circle exactly and a reflected image is always paired with the original slice, the boundaries of each segment will line up without perceivable discontinuity.

2. Since we use a pie slice as the original, a singularity exists at the image's center. This singularity lets viewers perceive the user's movement relative to the center and circle's outside edge.

3. The pie slice permits using different visual scales from the image.

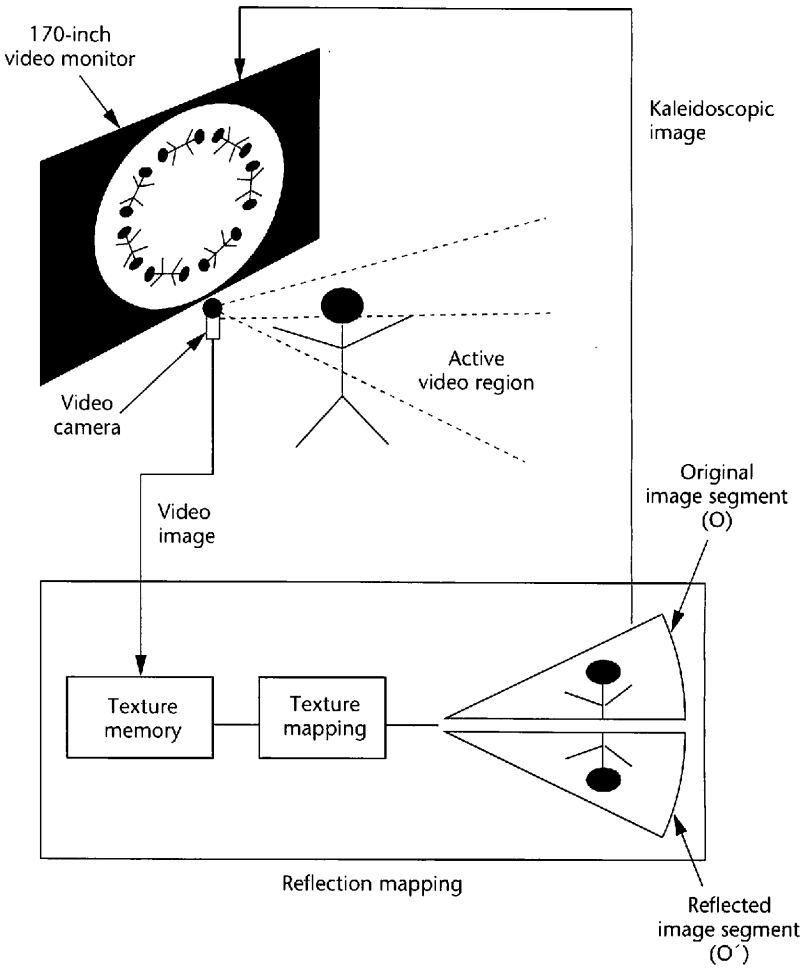


Figure 6. Block diagram of the lamascope. The subsystem runs at 30 fps.

The slice's outer edge captures a large area of the video image; toward the image's center it captures only a small area for the reflections. Placing objects close to the slice's center makes it difficult to recognize them in the kaleidoscope image, allowing for more abstract forms of expression. (Likewise, the user can move closer or farther from the video camera to get different scaling effects.)

The lamascope also features additional controls that MusiKalscope exploits: controls for image background color mixing, image brightness, arc angle, slice angle rotation speed, and so on. MusiKalscope uses the image background color mixing and brightness controls to match the lamascope image's mood with improvisational sounds the performer makes using RhyMe.

What Goes on During Jazz Improvisation

During improvisational jazz performance, players need to recognize many musical qualities to know which notes to play next to express their mood and feeling. First, as they play the piece, they can figure out the current key as it changes throughout the piece from the musical context. Then, knowing the current key, when they play a chord they can categorize it (and its absolute pitches) as a chord relative to the current tonic (key).

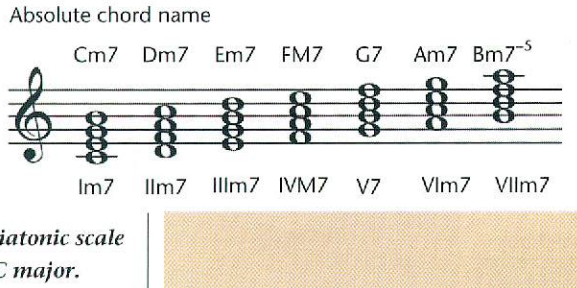


Figure C. Diatonic scale chords for C major.

The relative chords are called diatonic scale chords. For example, if we're playing and we hear an E-minor 7 chord, we can determine its diatonic scale chord name because we know what key we're in. If we're in C major, E is the III note and the diatonic scale chord is III-minor 7 as shown in Figure C. However, if the current key is G major, then E is the VI note and the diatonic scale chord is VI-minor 7. Figure C shows the chords and the name of each chord in the diatonic scale chords in C major.

Determining the key of a piece at any point in time can be difficult because it changes. The problem is that any particular chord can come from different keys. However, the dominant seventh chord is nearly unique, thus we can determine the tonic of the current key when we hear it. However, it doesn't provide enough information to determine whether the key is major or minor. For this, we need to hear a tonic chord to figure it out. Luckily, the tonic chord usually follows a dominant seventh chord. For the other chords, we have to consider more of the surrounding chords to determine the key. For example, we can look for common sequences of chords.

Once we know the diatonic scale chord and key, we need to know the available note scale to choose an appropriate note to play. A note scale represents a sequence of notes such as major and minor. Jazz uses many note scales (some are listed in Figure D). Each chord in the diatonic scale chords has been empirically determined to correspond to a certain note scale or scales. This determination leads to a theoretical analysis of jazz. Once we know the note scale, we can construct phrases from the available notes. Each note in the chosen note scale imparts a specific feeling on the piece depending on its position in the scale. For example, the sixth note of a scale often gives a sense of tension. Some notes aren't in the scale—we call these out-of-scale notes—and would sound bad if played. Other notes should be avoided even though they're in the scale. These notes have no fixed rules. Figure D shows the note scales and the notes of each diatonic scale chord in the key of C major. The notes in parenthesis should be avoided. The feeling (or function) of each note in a note scale is only roughly consistent, and the exceptions must be understood for each note scale.

Continuing our example, we know that Em7 will be played and the key is C major. From this, we determined that the diatonic scale chord is III-minor 7 and know that the scale name is E Phrygian. We now can choose which note to play. The sixth note in this note scale is a tension note. So if we play a C, we'll produce a tension feeling. Similarly, throughout the rest of the piece we can choose which note to play depending on our mood and the mood of the piece.

Chord	Scale name	Notes of scale
IM7	Ionian	
IIm7	Dorian	
IIIm7	Phrygian	
IVM7	Lydian	
V7	Mixo-Lydian	
VIm7	Aeolian	
VIIm7-5	Locrian	

Figure D. Correspondence between diatonic scale chords and available note scales in C major. Notes in parentheses should be avoided.

The RhyMe subsystem

Typically, in real-time multimedia art creation that uses music, the performer can't concentrate on playing music only. However, even if a well-trained musician concentrates on only playing music, it's still difficult to generate good music. Therefore, in many cases of multimedia art cr ation, the musical medium may not have parti ularly good quality.

Our strategy to avoid this problem in mult media artwork is to apply structure to a perfo mance artificially. In MusiKalscope we use

omputer to control the performance's structure, hereby applying our own definition of music. We all this "computer-supported improvisation" and leveloped a subsystem called RhyMe to provide this upport. Working from this philosophy, we restrict he type and structure of music that can be created luring a performance. However, we're rewarded by llowing the performer easier access to musically ppealing performance within these confines.

In our research we focused on supporting jazz mprovisation, in particular "bebop." Improvisaion is a critical component of jazz. For unaccusomed listeners, jazz improvisation performances nay sound like random compositions. However, f studied, these performances can be analyzed as onforming to some well-defined jazz improvisaional theory. Well-trained jazz players often anayze the song playing in their head and compose mprovisation pieces by using notes dictated by a articular jazz theory. By applying the theory, the iece can have a jazzy atmosphere and the player an express the "color" of a song derived from its hord progression structure. Of course, this analyis may not be performed consciously.

Playing jazz improvisation based on theory oses two problems. First, it's difficult for people o master the theory. Second, it's also difficult to eflect the knowledge of the piece during a realime performance even if you've mastered the heory. To overcome the first difficulty, several ystems have been developed that analyze songs utomatically using a knowledge base of particular jazz theories.[2,4,7,8] These systems can show users heoretically determined notes in a song that correspond to different jazz effects. By referring to the esults of the analysis, performers can improvise sing theoretically correct notes. However, since azz music has a complicated structure, it's still difficult for nonprofessionals to improvise even when referring to the analysis' results. To make natters worse, it's even more difficult to use this ype of system in a multimedia art performance where attention may need to shift to other media epresentations in real time.

To overcome the difficulties encountered in hese systems during live performance, we mployed the following method in RhyMe. First, he subsystem automatically analyzes a song ffline based on a particular jazz theory. Then, durng performance—based on the analyzed results—he subsystem assigns only notes available in the ote scale to playing positions of the musical nstrument used in the performance. (See the next ection for the mapping between notes and posi-

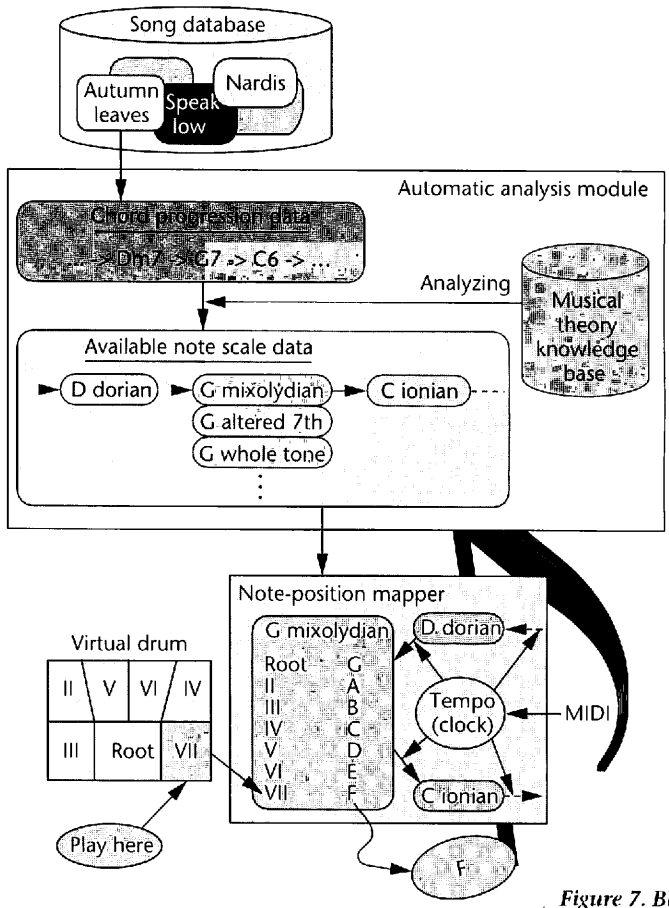


Figure 7. Block diagram of the RhyMe subsystem.

tions in the virtual drum set, and see the sidebar "What Goes on During Jazz Improvisation" for a discussion of note scales.) The jazz piece plays continuously while the performer joins in with improvisation assisted by RhyMe.

Figure 7 shows a block diagram of the RhyMe subsystem. RhyMe consists of a song database, an automatic analysis module, and a note-position mapper. The song database stores chord progression data of several musical pieces. Sample data of a piece looks like

...(Dm7, 2) (G7, 2) (C6, 4)...

Each entry consists of the chord name and the number of beats. The above example corresponds to the following score: D-minor 7th for 2 beats, G dominant 7th for 2 beats, and then C 6th for 4 beats. RhyMe uses data from the currently selected song as input to the analysis module.

The automatic analysis module has a musical theory knowledge base. Currently, this knowledge base builds on the Berklee theory, a well-known jazz theory (mainly bebop style jazz). Based on

this knowledge, this module analyzes the chord progression (from the input data) and determines what kinds of scales are available at each chord given the "context" of the chord progression. The whole analysis process is complex, but the idea can be illustrated by using the example above. First the analysis module looks for a dominant 7th chord and assumes that it's a V7 diatonic scale chord. As a result, the tonic—the note corresponding to I—can be assigned. In this example, G7 is a dominant 7th chord, hence note C is assigned as the tonic. Second, the module interprets and relabels the absolute chord progression data in relative terms (using diatonic scale chords) using the assigned tonic. In this example, the chord progression is interpreted as IIm7-V7-I6. Third, the module matches the interpreted chord progression with one of the chord-progression patterns stored a priori in the module. In this example, this chord progression matches with the major II-V-I chord progression, a typical chord-progression pattern. Fourth, if the module can find a matching chord-progression pattern, the module decides the assigned tonic is correct and outputs available note scales for each chord by looking up the chord-scale map. In this example, the following correspondent scales are output: II (dorian), V (mixolydian, altered 7th, whole tone, and so on), and I (ionian). Finally, the subsystem reinterprets the relative note name as an absolute note. From these results, the available note scales of the example are determined as shown in Figure 7—that is, D dorian, G mixolydian or G altered 7th or G whole tone scale, and C ionian.

The subsystem enters these analyzed results into the note-position mapper. This module maps a scale's notes at each time point in the song onto the musical instrument's playing positions. The mapping changes while notes synchronize with the song's progression. The instrument used in MusiKalscope is a virtual drum machine (see the section "A simple virtual drum"), which has seven active zones. Notice that the root zone corresponds to the root note of the currently available note scale (for example, a G note in Figure 7), while the II and III active zones correspond to the II and III notes of the currently available note scale (in Figure 7, the A and B note, respectively). In our example, G mixolydian is the available note scale. If the performer plays the VII zone, the actual note played is an F as shown in Figure 7. However, when the available note scale changes to D dorian by the song's progression, the note played by striking the VII zone becomes C.

The whole RhyMe subsystem uses the Musicia Instrument Digital Interface (MIDI), thus, th MIDI timing clock data handles the synchroniza tion. Using this scheme, either chord tone zone (root, III, V, VII) can be played or tension not zones (II, IV and VI) corresponding to the note' relative position in the currently available not scale. The key point to remember, though, is tha RhyMe dictates the actual note played.

When using conventional musical instru ments, performers must always judge which note are theoretically correct and which note ha which color by thinking about the song's chor progression and their own theoretical knowledge However, using RhyMe, performers don't have t determine notes while playing a song. Since onl theoretically correct notes are mapped onto th musical instrument at any time in the song, th performer doesn't need to know the chord pro gression and theory. Furthermore, notes ar mapped onto adequately classified zones (that is chord tone or tension note).

We call this mapping method a "fixed-functio mapping": a note always has a certain functio depending on the context of the current chor and chord progression. For example, the third not from the root has a function to decide whether th current chord is major or minor. In RhyMe, th third note is always mapped onto the III position Therefore, performers can easily play the thir note whenever they need the sound that empha sizes the major or minor mood in the perfor mance. Thus, this method allows easy access fo performers to express color throughout the song The main result of this is that performers' cogn tive loads can be reduced drastically when creat ing music, and at the same time they can easil maintain a reasonable quality of music through out the piece. Further, the RhyMe subsystem give performers room for free composition. From thi reasoning, RhyMe proves a suitable subsystem fo generating music in a real-time multimedia art per formance system such as MusiKalscope.

The music generating methods used i DanceSpace and Brush de Samba also theoretica ly analyze songs similar to the technique used i RhyMe. As such, in these systems, performers ca play theoretically correct notes without payin conscious attention during each moment in th song. However, these systems don't offer obviou discrete playing positions to performers to contro the music. In particular, in these systems, per formers control only successive ascending o descending phrases. Therefore, it's very difficul

for performers to compose phrases they want to play. The RhyMe subsystem provides discrete playing positions that can be played using instruments like keyboards, virtual drums, or other MIDI instruments. Thus, performers can easily choose notes with specific functions and compose phrases freely by combining the functions.

GMII: Bringing the pieces together

The GMII in MusiKalscope links the two subsystems RhyMe and the Iamascope. The GMII provides a set of virtual drums (see the next section) the performer uses to play with RhyMe and control some parameters of the Iamascope image. With RhyMe, a song's chord progression determines its global color. However, the performer can play different tones with different velocities to control local (in time) color—for example, playing a chord tone, playing a tension tone, or playing strongly or weakly. For MusiKalscope, we mapped the available scale's chord tone and tension notes onto different virtual drum pad zones. Therefore, players can easily choose to play a chord tone or a tension note to control the song's local color. The virtual drum set provided by the GMII provides seven customizable active zones corresponding to either chord tone or tension notes (each corresponding to an offset from the root tone note). Figure 8 shows the default configuration of the active zones. The functions associated with each zone are activated when performers make a down-up motion of their hands in the zone.

We mapped the Iamascope's control according to the functions that RhyMe uses for the active zones. We chose the mapping a priori based on artistic license. Thus, for zones that produce tension notes according to RhyMe, the Iamascope image provides a more visual tension. For this effect, we would make the background mixing color bluer for a sense of foreboding. The more tension zones struck, the bluer the Iamascope image looks. If the performer strikes a chord zone, then the Iamascope image immediately returns to its normal, nonblue state to provide a sense of resolution. If the performer doesn't strike any key, the Iamascope image gradually returns to its normal state over time. The velocity with which the performer strikes the virtual drum controls the Iamascope image's brightness. For fast strikes, the Iamascope image lights up strongly in a flash and gradually returns to its normal brightness.

A simple virtual drum. Many different instruments can control RhyMe—any instrument with seven or more keys can be used. For MusiKalscope, performers' hands and arms must be unencumbered so that they can move freely in front of the Iamascope while simultaneously playing with RhyMe (to control the visual imagery). To do this, we used a virtual drum set along with a Polhemus 3-space Fastrak system that measures performers' hand and arm motions. The magnetically based trackers measure a receiver's six degrees of freedom (position and orientation) relative to a source. In MusiKalscope, we used two receivers—one for each hand. Performers attach the receivers to their arms using straps, allowing them to move their arms and hands freely. Down-up movements of their arms strike the virtual drum pads according to the mapping shown in Figure 8. The active zones currently lie horizontally (parallel to the floor) in front of the fixed source. We plan to attach an additional receiver to the performer's body so that the active zones are relative to the performer and not a stationary point in front of the Iamascope.

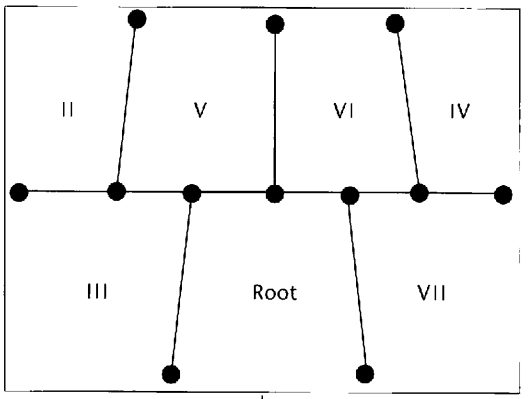


Figure 8. The virtual drum's seven active zones.

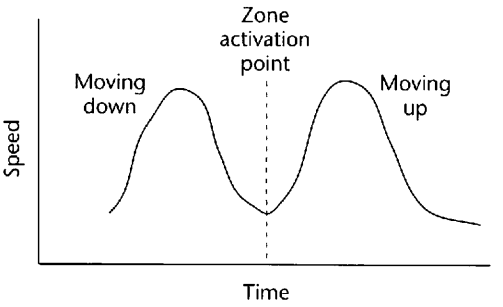


Figure 9. Idealized speed profile graph showing a performer moving a hand down and then up.

To implement the virtual drums, we must measure the velocity of the performers' arms. Figure 9 shows an idealized speed profile of a performer moving a hand down and then up. Notice that as the performer's hand moves down from a stationary position, the speed increases. Then as the performer prepares to turn the hand around, the speed decreases. As the hand rises again, the speed reaches its minimum (ideally, it should reach 0). As soon as our virtual drum system detects a change in hand direction, the zone becomes activated. (The performer must also have exceeded a minimum downward speed threshold before the change in hand direction can activate a zone.) The particular name of the zone that becomes activated passes to RhyMe and the Iamascope controls.

Adjusting the metaphor. One subtle point about this virtual drum scheme is that the change in direction doesn't correspond exactly to the drum metaphor. Often, performers strike a real drum using a ballistic downward hand motion. Contact with the drum changes the direction of the performer's hand. However, the virtual drum doesn't have any surfaces to contact. Instead, the performer's own muscles have to change the direction. Using a drum metaphor, users expect the system to respond during the ballistic downward stroke of their arm. However, with the virtual drum, activation occurs when their hand changes direction—this happens somewhat later than the ballistic stroke. This introduces a perceptual lag in the system stemming from the wrong metaphor. A closer metaphor might be a conductor's baton. We're currently investigating other sensors (such as accelerometers, infrared batons, and vision systems) and techniques that match the perceptual striking position of the virtual drum metaphor.

Discussion of MusiKalscope

We displayed an earlier version of MusiKalscope in Kyoto, Japan. During this display several hundred people used MusiKalscope, providing an opportunity to observe both performers and audiences. We observed two important points. First, novice performers tended to focus either on producing music or producing visual imagery but not both. Second, audiences often enjoyed the combined visual and musical performance, but had difficulty appreciating and understanding the relationship between the performer and the performance they were hearing and seeing.

The fact that performers attended either to producing music or playing with visual imagery didn't surprise us. In particular, MusiKalscope's main point is to let performers focus only on one aspect at a time. The complimentary medium always maintains a reasonably good quality level even if it's ignored or played poorly. However, either system responds appropriately to allow improved expression as the performer becomes skilled. From this point of view, the system succeeded.

Unfortunately, the kaleidoscopic images bear little resemblance to the performer, since only a small slice of the whole image is used in the reflections. Thus, on the one hand, the movement and changes in the image aren't obviously attributed to the performer. Once an audience member becomes a performer by stepping into MusiKalscope, the relationship is obvious and lasting. Perhaps we need to familiarize the audience with this novel ar form or explain it to them better. On the othe hand, the imagery is dynamic and beautiful, so th connection between performer and imagery isn' critical for appreciation. For the musical perfor mance, identifying the relationship between th performer's actions and the music proves difficul because background music always plays an doesn't change with the performer's actions Further, lag introduced by the virtual drums exas perated the problem (both for the audience an performer). Improving the lag situation will help However, an inherent trade-off exists between pro viding a minimum quality level of music and let ting the performer control the music, establishin a strong link between actions and sounds.

Finally, one frequent comment from perform ers was that too much lag existed between the vir tual drum strike and the sound produced. A discussed previously, this is probably due to th incorrect metaphor for the sensor system we use We're working to improve this situation.

Future directions

As mentioned earlier, we had the followin three goals in mind when we developed MusiKalscope:

1. Obtain good balance between the quality o computer graphics and the music generated.

2. Let novices achieve a reasonable quality o music and imagery easily.

3. Allow an upward path for skill acquisition s experts can become more expressive wit MusiKalscope.

We achieved these goals mostly by combinin two systems that each have a reasonable mini mum quality even if completely ignored. Th RhyMe subsystem will continue to play pleasan music regardless of user input. It can never pla particularly bad sounding music (in its genre), bu with user input can sound even better. Likewise the Iamascope, like a standard kaleidoscope, gen erally produces beautiful images no matter wha goes into it. However, the Iamascope allows per formers the freedom to express themselves b moving inside it. Through movement and imagi nation, the performers control the image's expres siveness. Thus, both subsystems are balanced wit respect to quality.

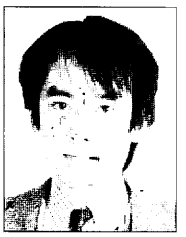
We're improving three main areas to furthe

accomplish the goals set out. First, we want to make the system easier to use skillfully. Currently, even though novice users produce reasonable quality music and images, the progression to express themselves well is not as fast as we would like. The main difficulty stems from the awkward input device of the virtual drum set. We plan to modify this device either by using different sensors or improving the recognition techniques. Second, we currently have only a small selection of songs analyzed for use in RhyMe. Some songs suit accompaniment with visual imagery better than others. We're planning on increasing the available selections. Third, it's an iterative process determining which aspects of the Iamascope best fit a particular music genre. We're investigating different ways to control the Iamascope to enhance the feeling it provides along with matching the particular "color" of the music performed. **MM**

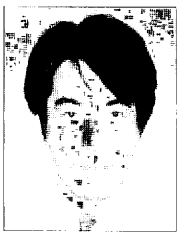
References

1. S.S. Fels, K. Nishimoto, and K. Mase, "MusiKalscope: A Graphical Musical Instrument," *Proc. IEEE Int'l Conf. on Multimedia Computing and Systems (ICMCS 97)*, IEEE CS Press, Los Alamitos, Calif., June 1997, pp. 55-62.
2. A. Kotani and P. Maes, "An Environment for Musical Collaboration between Agents and Users," *Proc. Lifelike Computer Characters*, Sept. 1995, Snowbird, Utah, p. 54.
3. M. Goto and Y. Muraoka, "A Virtual Dancer 'Cindy'—Interactive Performance of a Music-Controlled CG Dancer," *Proc. Lifelike Computer Characters*, Snowbird, Utah, Oct. 1996, p. 65.
4. F. Sparacino, *Choreographing Media for Interactive Virtual Environments*, master's thesis, Media Arts and Sciences, Massachusetts Institute of Technology, Cambridge, Mass., 1996.
5. N. Tosa and R. Nakatsu, Life-Like Communication Agent—Emotion Sensing Character 'MIC' and Feeling Session Character 'Muse'," *Proc. Int'l Conf. on Multimedia Computing and Systems*, IEEE CS Press, Los Alamitos, Calif., June 1996, pp. 12-19.
6. C. Baker, *Kaleidoscope Renaissance*, Beechcliff Books, Annapolis, Md., 1993.
7. K. Hirata, "Towards Formalizing Jazz Piano Knowledge with a Deductive Object-Oriented Approach," *Proc. of Artificial Intelligence and Music*, Int'l Joint Conf. Artificial Intelligence (IJCAI), Montreal, Aug. 1995, pp. 77-80.
8. D. Horowitz, "Representing Musical Knowledge in a Jazz Improvisation System," *Proc. Artificial Intelligence and Music*, Int'l Joint Conf. Artificial Intelligence (IJCAI), Montreal, Aug.1995, pp. 16-23.

Sidney Fels is an assistant professor in the Department of Electrical and Computer Engineering at the University of British Columbia. His research interests include adaptive interfaces, neural networks, intelligent agents, speech synthesis, and computer-based artwork. He received his MS and PhD degrees in computer science from the University of Toronto in 1990 and 1994, respectively. He received a BASc in electrical engineering from the University of Waterloo in 1988.



Kazushi Nishimoto is a visiting researcher with Department 2, ATR Media Integration & Communications Research Laboratories in Kyoto, Japan. His research interests include agent technologies to enhance human creativity. He received a BS and MS in mechanical engineering from Kyoto University, Japan in 1985 and 1987, respectively.



Kenji Mase is the head of Department 2, ATR Media Integration & Communications Research Laboratories in Kyoto, Japan. His research interests include computer graphics and computer vision and their applications for human-machine interfaces. He received a BS in electrical engineering and an MS and PhD in information Engineering from Nagoya University, Japan in 1979, 1981, and 1992, respectively.

Readers may contact Fels at the Dept. of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC V6T 1Z4, e-mail ssfels@ece.ubc.ca.