ToCoPlay: Graphical Multi-touch Interaction for Composing and Playing Music

Sheelagh Carpendale, Sean Lynch, Miguel A. Nacenta

ToCoPlay: Graphical Multi-touch Interaction for Composing and Playing Music

Sean Lynch, Miguel A. Nacenta, Sheelagh Carpendale
University of Calgary,
Calgary, Alberta, Canada
{sglynch, miguel.nacenta, sheelagh}@ucalgary.ca

Abstract. With the advent of electronic music and computers, the human-sound interface is liberated from the specific physical constraints of traditional instruments, which means that we can design musical interfaces that provide arbitrary mappings between human actions and sound generation. This freedom has resulted in a wealth of new tools for electronic music generation that expand the limits of expression, as exemplified by projects such as Reactable and Bricktable. In this paper we present ToCoPlay, an interface that further explores the design space of collaborative, multi-touch music creation systems. ToCoPlay is unique in several respects: it allows creators to dynamically transition between the roles of composer and performer, it takes advantage of a flexible spatial mapping between a musical piece and the graphical interface elements that represent it, and it applies current and traditional interface interaction techniques for the creation of music.

Keywords: Multi-touch, collaboration, composition, music, musical instrument.

1 Introduction

Musical instruments can be considered interfaces between a musician and the acoustic space. It is, therefore, not surprising that music has been one of the main areas of application for the advances in interface science and engineering of the last few decades. The advent of new interface paradigms such as multi-touch and tangible computing have spawned a series of remarkable prototypes and commercial systems that enable new ways of creating and experiencing music, often beyond what is possible with traditional instruments. For example, new interfaces for musical expression currently allow people without music backgrounds to explore new sounds [4, 19, 21] and groups of people to perform together closely through interactive tables [14, 15, 19, 21]. Computer and interface technology has changed how music can be performed, how sound can be generated, and also how it can be composed; however, with few exceptions [12, 20, 19], composition tools do not take advantage of new interaction possibilities, and still are restricted to mouse and keyboard, sometimes with interfaces to electronic versions of traditional instruments [6, 9].

In this paper we present ToCoPlay (Touch-Compose-Play), a musical interface designed to support playing and composing music. By enabling seamless transition between performing and composing we explore facilitating musical actions that are already part of musicians' activities, since composing music requires listening and changing what is being created, and, conversely, there are many forms of music where composition is part of the performance process (e.g., jazz). Our system is based on a horizontal multi-touch surface, and is intended for people with or without musical backgrounds. During the design of this new kind of musical interface, we made design decisions to enable control, configurability, a straightforward mapping between the visual and acoustic space, and to support a minimal set of interface elements and operations.

The structure of the paper is as follows: in the next section we provide relevant background on interfaces for musical expression. Then we discuss our goals and strategies for the design of the ToCoPlay interface, followed by its detailed description. We present four examples of ToCoPlay in use, followed by a discussion of the results of our exploration, and finish with our conclusions.

2 Related Work

There exist many applications for musical expression on digital surfaces, and these vary widely in interaction methods and purpose. Most applications support a combination of the following musical activities: playing sounds, synthesizing sounds, sequencing sounds in real-time, and composing. In the following paragraphs we provide a brief summary of prior work, and how it differs from our own.

There are a large number of previous systems that focus on creating new sounds through the modification of streams or samples. Some of these are based mainly on tangibles, such as the seminal work by Jordà et al. on the Reactable [14,15], Audiopad [21], Pin & Play & Perform [27], and Condio [5]. These systems rely mostly on physical blocks that represent certain sound synthesizing operations (e.g., filters, tone generators, echo, samples) that are placed on the table and manipulated in real time to produce new sounds and rhythms. Other related work has the same approach, but is based on touch. For example, Akustich [1] is a direct-touch interface in which participants manipulate sound by performing multi-touch gestures on a table, where each gesture changes the sound and the visuals in a different way. In a similar way, performers using Sound Storm [23] stand in front of a vertical display and use touch to modify sound waves that come across the screen. The Synthesis and Control on Large Multi-touch Displays [7] demo by Perceptive Pixel provides a multi-touch knob and slider interface for the Synthesis Toolkit (a sound library for C++).

These systems resemble ToCoPlay in that they allow the real-time modification of sound by human performers, which makes them *virtual instruments*. However, they differ from ToCoPlay in several ways. First, ToCoPlay is based on standard note scales and sounds; therefore it does not focus on the generation of new sounds and

rhythms. In this respect, ToCoPlay is simpler. Second, these systems are not designed to support composition: the temporal evolution of the music is difficult to reproduce because sound is mostly dependent on the state of the interface and the specific user actions. ToCoPlay is designed to store, modify, and reproduce created sequences, which supports performance, composition, and all combinations of the two.

A second group of related systems do not support sound synthesis; instead, they focus on supporting music performance that resembles how we play traditional instruments. For example, SurfaceMusic [8] enables playing three instruments on an interactive tabletop (a drum, a string instrument, and a wind instrument) by using a specific multi-touch gesture on each (tapping, strumming, and air-pushing). The Roots [12] application generates ambient sounds when a user touches a particular region on the multi-touch horizontal table (the Bricktable [11]) and produces visually pleasing, spiraling trees. The Jam-o-drum [4] is a series of external controllers that lay on a horizontal table display, where multiple people can play sounds by tapping their own pad; the Jam-o-drum is primarily for playing games. These systems are closer to ToCoPlay in the simplicity of the basic sound elements that are combined, but they do not generally support composition any more than traditional instruments.

A third group of interfaces enable composition. Traditional applications such as Garage Band [9], and Cubase [6] are single user, rely on traditional notation, and are meant to be used from a desktop PC. The MusicTable [24] allows people to place cards that represent pitch and instrument on a table. The cards are detected by the system and played in sequence from left to right. BlockJam [19] lets people combine attachable active physical blocks that have a button to generate a sequence of sounds. The rules of how the sound is produced depend on the type of block and its current state. The resulting sound cycles through the blocks and visual feedback is provided through a monitor. InstantCity [13] is an art music automata that plays different ambient sounds depending on how blocks are distributed on the table. Xenakis [2] is a non-deterministic system that uses tangible blocks to generate music according to probability distributions based on the distance between blocks and their similarity. In Noteput [20] participants place musical note-shaped physical blocks onto a digital horizontal surface which are played left to right according to their position, creating a physical staff. Spaces [12] is an application for composing ambient sounds on a multi-touch table, where participants touch a region of a table to make that region a warmer or cooler colour, with matching sound. Stereotronic Multi-Synth Orchestra [25] is a Microsoft Surface [17] application in which participants place notes into pre-defined concentric rings, and notes are triggered when the rotating element of each concentric ring goes over that particular note.

These systems resemble ToCoPlay in their ability to easily set up temporal sequences of sounds, but their design does not facilitate the performative component. ToCoPlay, similarly to all of these systems, allows setting up music and then triggering its performance without much human intervention; however, ToCoPlay focuses on supporting seamless transition between configuring the music, changing it, and performing or improvising with the available components.

In summary, ToCoPlay situates itself as being both a composition tool and an instrument, but not a sound synthesizer. ToCoPlay's goal is different from the goals of systems mentioned above, since we set out to achieve seamless transitions between the playing and composing activities. It also has several features that distinguish it from previous systems, most notably a flexible and precise spatial mapping (contrary to the mostly linear ones described above) and its configurability to create configurations that adapt to the person and the situation.

3 Design Goal and Strategies

As illustrated by the related work section, there is considerable activity in the development of software with musical capabilities. However, most of these focus on strengthening either the performance potential or concentrate on supporting musical composition. Our goal is to, as part of a musical software interface, investigate the integration of these activities: composition and performance. Thus, we intend to both keep interactions as simple possible and to explore ways of using simple interactions to create powerful methods for introducing complexity. In ToCoPlay, we offer an interface that can be played as an instrument, but that also invites composition. To achieve these goals we apply four main design strategies: enabling control, enabling configurability, creating a formative visual-acoustic mapping, and providing a minimal set of interface objects and operations.

3.1 Enabling Control

For ToCoPlay we opted for a discrete paradigm, which is more closely related to how a piano is played (by pressing keys) than to the continuous model used by most modern synthesizers, which are commonly controlled by turning knobs and adjusting continuous parameters. In ToCoPlay individual sounds are made by pressing keys. Complex sounds are developed through grouping, sequencing, and nesting. Grouping, sequencing and nesting are all interactively created and visually explicit.

3.2 Enabling Configurability

The physical characteristics of traditional physical instruments, such as the arrangement and shape of keys, have evolved into their current forms through many small mutations that often took centuries. Current electronic interfaces have not yet enjoyed the popularity or the time to evolve in this way. For ToCoPlay we introduce a model in which the location of keys and their grouping is fully configurable in real time. It is reconfigurable during the design of the sounds or melodies and it is reconfigurable during play. These freedoms can enhance the creation of a personalized interface that fits one's own hands and, thus, is easier to play. Through reconfigurability it can also be adapted for different people and compositions; for

example, one can group notes together so that they can be played without moving one's hand (e.g., by placing them directly under the fingers when they are in a natural posture), or one can move a certain set of notes closer when they are to be used.

3.3 Visual-acoustic Mapping

For most people composing music requires at least two basic elements: a way to try out the sounds, melodies, and harmonies as they are being created, and a way to record how these are produced (including their temporal relationships), for later performance. Many of the current music generation systems that we describe in Section 2 are inadequate for composition, not only because they lack features to record how interaction must take place to reproduce a certain sound, but also because the continuous nature of the interaction makes it difficult to create a notation that accurately and effectively relates the almost infinite variety of gestures that can take place in a multi-parametric continuous space to the way they sound.

Therefore we designed a visual-acoustic mapping that provides a straightforward relationship between what is displayed on screen and what is going to be reproduced, so that actions on the interface can be interpreted as changes in the music. Note that this is not a visualization of the sound but a spatialisation of the relationships between sounds. One such possible mapping is traditional music notation; however, this notation did not evolve to facilitate real-time interaction, is not generally accessible to novices and, most importantly, it imposes a rigid spatial mapping of time (along the staff, moving horizontally through time and then down through the scale) that directly contradicts our configurability goal. In other words, using a staff metaphor for our interactive system would not allow us to make an interface that is easily playable; it would encumber collaboration (e.g., due to its implicit orientation), and would not allow us to group elements according to other groupings that are not strictly temporal. Most of these arguments apply as well to other existing sequencing and composing applications (e.g., [2]). Again, we keep this spatial/visual relationship simple. Sequencing is specified through explicit directed links, where the spatial distance is mapped to temporal spacing.

3.4 Minimal Objects and Operations

Making an interface accessible to novices requires simplicity in the number and complexity of object types in the interface and their operations. We set out to create an interface that was based on a minimalistic approach. Unlike many existing musical interfaces that have large lists of objects, filters, types of connections and parameters, our interface needs only a few objects and a few rules about the ways the objects can relate to each other.

It might seem counter intuitive to restrict the design of the system to a small set of primitives in order to enhance expressivity and encourage playfulness. However,

there is a wealth of experience in other fields that shows that minimalistic systems can lead to complex and expressive results; for example, a von Neumann machine, the Game of Life [10], SWARM intelligence [3], or the game of Go.

4 The ToCoPlay Interface

The interface of ToCoPlay is composed of five atomic elements: keys, key fountains, containers, links, and dummy keys. These elements can be combined to enable the properties outlined by the design section above. The following subsections provide a detailed description of each of the atomic elements, their interactions, and how their design, interactions and combinations support our design objectives.

4.1 Keys

The base atomic element in ToCoPlay is the key, a group of which are shown in Figure 1. A key is a circular interface element that plays a specific note of a specific instrument when touched. The note assigned to a key is one of the twelve pitches of the Chromatic Scale, which are represented visually by a change in value of the key's color; that is, lower notes are represented by darker colors than higher notes. The hue of the key indicates the instrument (or timbre) of the note. At this moment, two synthetic instruments, a sine and a square wave generator, are represented by pink and yellow key hues respectively.

Figure 1. Keys in ToCoPlay, representing the C Chromatic Scale

Keys are circular to facilitate being easily touched while having a small footprint, which allows many keys to be visible simultaneously on the interface. Note that, unlike with most traditional interfaces, ToCoPlay can have many keys for the same note which, as we will see, supports the generative nature of the system. The one-to-one mapping between the colour of the keys and the sound that they emit is designed to enable a direct relationship between the visual and acoustic spaces.

Tapping a key instantly plays the corresponding note and makes it transparent, establishing a clear connection between the visual and the acoustic feedback of the system. A key can be moved and repositioned anywhere across the interface. Repositioning keys allows the musician to arrange them in ways that support playing the interface as a customizable instrument; for example, the keys necessary to play different leit-motifs of a piece can be placed so that they correspond to the tips of the fingers when the hand is in different areas of the table. The mobility of keys on the interface also enables clustering of notes in different visual layouts which, as described under 'links', further supports the compositional aspects of the interface. Keys can also be flicked around in the interface to quickly move them away from an

area of interest, or to remove them from the interface if flicked out of the visible area of the interactive table.

4.2 Key Fountains

New keys are created by dragging them out of *key fountains*. Key fountains are represented by a large circle surrounded by the smaller, circular keys (see Figure 3). The keys are arranged around the circle clockwise from C to B in half-tone intervals (12 notes). When a key is removed from a fountain, another copy appears in its place, which enables the creation of an unlimited number of keys of each type, as shown in Figure 2.

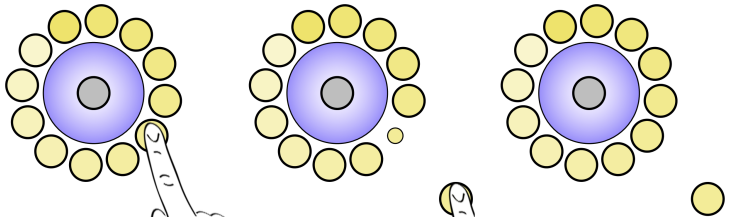

Figure 2. A key is removed from a key fountain, and is replaced by another key.

The interface contains a key fountain for each instrument (timbre). Although fountains initially appear at the center of the application, they can be moved around by dragging them from the central circle to make space for other kinds of interaction. It is also possible to rotate and scale fountains through the standard two-point RST gesture [18] to accommodate the needs of people located in different areas of the table. Consistent with the key interface elements, a fountain can also be flicked away out of the screen, which clears it from memory. Note fountains can be duplicated by double tapping on their central circle, again to accommodate the needs of multiple people around the table.

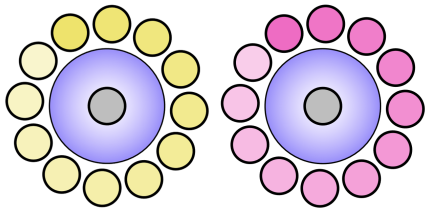

Figure 3. The two different key fountains, each representing a different timbre. Each key fountain consists of a large circle surrounded by keys, representing the notes in the chromatic scale, from C (darkest) to B (lightest)

4.3 Containers

Containers, which are used to hold elements, are regions defined by free-form touch-traces on the table (see Figure 4). Containers are created by starting a touch-trace on an unoccupied place on the table and closing off the trace. A line indicates the outline

of a container while it is being drawn, allowing for the precise definition of regions. When the finger is released and the container is completed, it appears as a translucent blue shape, and it automatically contains the objects that the trace enclosed.

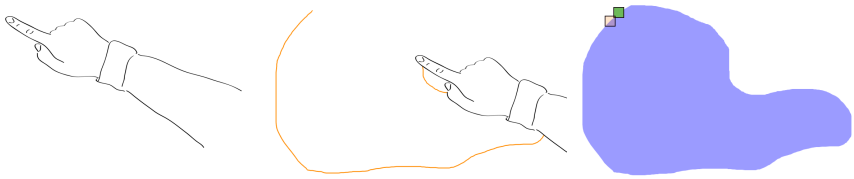

Figure 4. To trace a container start in an unoccupied space and create a closed form.

Containers can group heterogeneous collections of any interface objects except key fountains (see Figure 5). Containers can also contain other containers, which supports the generative nature of composition by allowing nesting. They enable the individuals to work with their own pieces of a composition, or of an instrument, and then to combine them at a later point in time.

Similarly to key fountains, containers can be moved, scaled or deleted (i.e., they can be dragged, flicked or manipulated with two-finger RST). Objects such as keys and other containers that the container currently holds stay with the container as the container is moved or rotated. This allows musicians to manage the screen area efficiently by manipulating meaningful groupings of objects and supports socially-based use of the space [22]. An object (e.g., a key) can be added to a container by moving it from outside the container to inside the container, or by moving the container so that it fully contains the object.

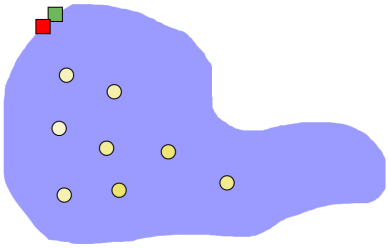

Figure 5. A container with keys placed inside.

Containers also have two buttons. The container and all of the elements and structure it contains can be copied or duplicated by tapping on its green button. The copy of the container appears at a small offset from the original (see Figure 6). The copy also oscillates for a brief period of time to highlight that it is a new object. Next to the copy button, there is a red button that is used to control the lock state of the container. A container can be in one of two states: locked or unlocked. An unlocked container can be moved, resized, or rotated and also allows its contents to be moved or removed. When locked, the container prevents accidental movement or transformation of its contents and itself, keeping all elements in place. When unlocked, containers and their contents can be manipulated to find different internal

and external configurations. If a container is unlocked, an element can be moved out of a container; once it is fully outside of a container, and is released, it no longer belongs to that parent container. The opacity of the red lock button reflects the current state of a given container: opaque red means locked, translucent means unlocked. Containers can be deleted in one of two ways: either by throwing them off screen like keys or key fountains, or by a gesture that crosses the container boundary twice.

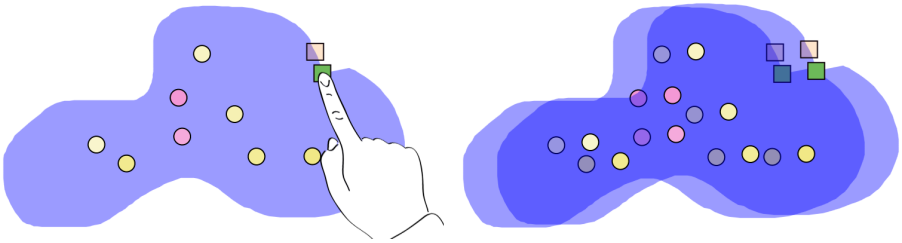

Figure 6. Pressing the container's green button creates a duplicate at a slight offset.

Together, duplication and repositioning provide the freedom to design personalized performance arrangements. Container and key groupings can become 'instruments' that can be shaped according to the specific anatomical or musical needs of the performer (e.g., according to the hand shape) and placed by the musician in different areas as required by the performance (e.g., storing instruments in distant regions when not required and bringing them close when being used). Simultaneously, the grouping of keys supported by containers, combined with nesting, enables the natural organization of musical compositions in different hierarchies such as themes, choruses, bridges, sections, etc. Container manipulation also supports reducing and organizing containers in space according to their current relevance (shown in Figure 7).

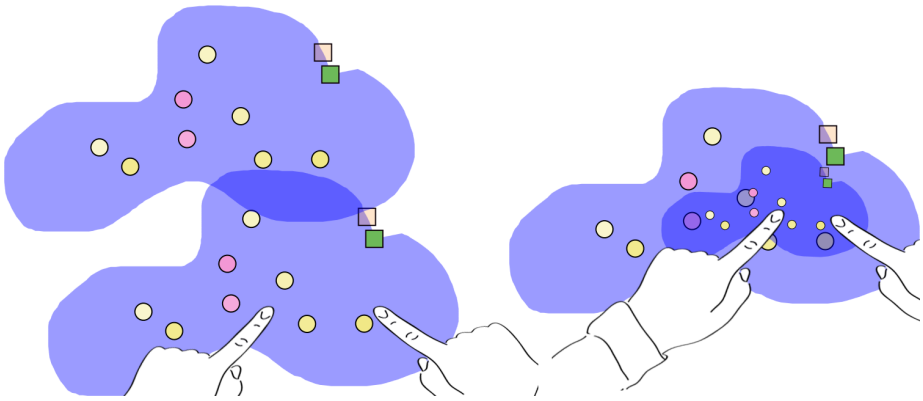

Figure 7. A container is rotated, scaled and translated to fit inside of another container.

4.4 Links

The elements described thus far support playing notes, grouping keys so that they form 'instruments' of arbitrary shapes, sizes, and orientations, and distributing these groupings in meaningful layouts over the table. However, the nature of the interaction with these elements does not enable composition or playback, since each note still has to be directly played by the musician. In order to support composition it is necessary to provide a mechanism that allows sequencing of notes at different times and intervals. We provide this mechanism through our link interface element.

Links are one-directional connections from one key to another that denote a sequence in how the notes of the corresponding keys are played (see Figure 8). Links are created by tracing a line that connects two keys when the original key is inside a locked container. Providing that the original key is within a locked container, any two keys can be connected, including keys belonging to different containers, keys that do not belong to any container or even keys that belong to contained sub-containers. Links can be deleted by tracing a line that crosses the link's representation within a locked container or in blank space.

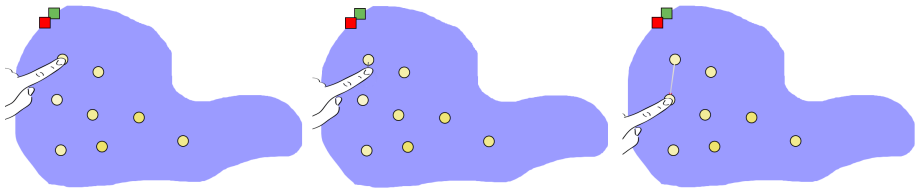

Figure 8. A key, inside a locked container, is pressed and then connected to another key, forming a link.

If a key is tapped and it has an outgoing link to another key, the other keys will automatically be played when the original key has finished playing its note. If the subsequent keys have links, the subsequent notes will be played and so on, until the chain ends, thus providing a way to sequence music events. The direction of a link is indicated by an arrow at the end of the link, indicating which key is the initial key and which is the subsequent key. The length of the link determines the timing between the tapping or signal reception of the initial key and the start of the note of the subsequent key. The initial key's duration depends also on the duration of the following link, since its note will play until the next note in this particular chain starts to play. Timings created with links are not fully continuous and instead snap to multiples of 125ms.

A key can have any number of incoming or outgoing links; every time that a link sends a signal to a key, its entire set of outgoing links will fire signals to the corresponding keys. This branching enables creating chords and counterpoint melodies –common elements of music in most cultures– as well as cyclic patterns and recursive music as we will show in the examples of Section 7.

4.5 Dummy Keys

Dummy keys are special keys that do not play any sound by themselves, but serve as proxies to activate other keys. A dummy key's incoming link length is ignored. By placing a dummy key that connects distant keys A and B (in that sequence) close to B, the note of key A will be played for a time proportional to the distance between the dummy note and key B instead of by the distance between keys A and B (see Figure 9). This allows for flexibility in the connection and placing of groups of keys that need to be played close in time but need to remain in non-adjacent locations. Similarly, a combination of dummy keys can be used to create instruments with keys located in anatomically convenient locations, as we will show in Section 6.1. Dummy keys can be created by dragging them from the center of a key fountain, and their inner colour is grey.

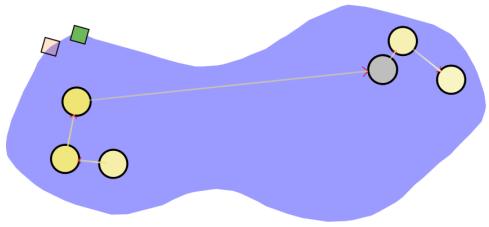

Figure 9. A dummy key connects two keys at a large distance. As a result, the key with the outgoing link will have a shorter distance to the outgoing link of the dummy key.

5 Implementation

We implemented ToCoPlay to work on the Microsoft Surface™, and used Microsoft's WPF™ libraries for Surface development. As a result, every visual object in ToCoPlay receives events when it is touched. For sound, we used Microsoft's MediaPlayer framework, and triggered wav samples. The wav samples were generated to have two distinct timbres: that of a chromatic scale with sinusoidal tones, and of a chromatic scale with square tones. We used a simple tone generator application to create these, and loaded multiple instances of each file so as to have the ability to play a sample more than once at any point in time. For the visual elements of ToCoPlay, prototypes were built using Microsoft Expression Blend™, and the final versions were built using a combination of Microsoft's C# and XAML.

6 Use Scenarios

6.1 Chord and Scale Instrument

This use case scenario illustrates the configurability of the system through the use of shaped containers, key layouts, and dummy keys. It corresponds to Example 1 in the video figure. The video shows how to build two instruments. The first one is a simple instrument with the five keys of a blues pentatonic scale, which are arranged in space to correspond to the tips of the fingers of the performer. Keys are made larger to make them easier to play, and adjusted for angle of comfort for the performer.

The second is an ad-hoc chord instrument that can play three basic chords of a simplified blues progression (C7, F7, and G7). Each chord is formed by four keys, which are activated by a dummy key simultaneously. Each of the chord's dummy keys are activated (with no delay) by a dummy key located within their respective containers, to facilitate playing with the left hand (see Figure 10 and video figure). The performer shown on the video (one of the authors) does not have any piano experience.

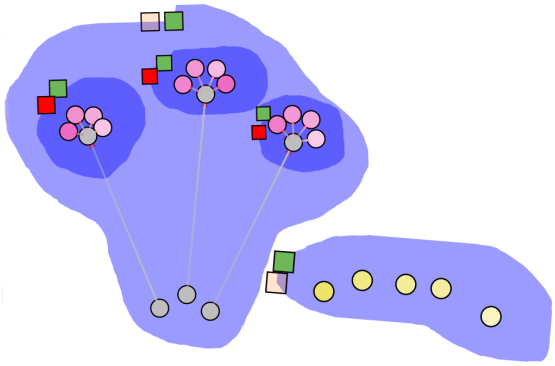

Figure 10. A chord instrument (left) and a scale instrument (right).

6.2 Harmonious Loops

This scenario illustrates how simple musical loops can produce a more complex musical texture through straightforward copying of a container, and the real-time rearrangement of the keys. In this example we part from a single loop, then copy it, modify the copy's loop timing slightly, and play both loops simultaneously. Because both loops are similar but not identical in length, the harmonies being played change continuously, in a sequence that repeats itself on a period much longer than the duration of either loop, creating a complex musical texture out of two simple elements.

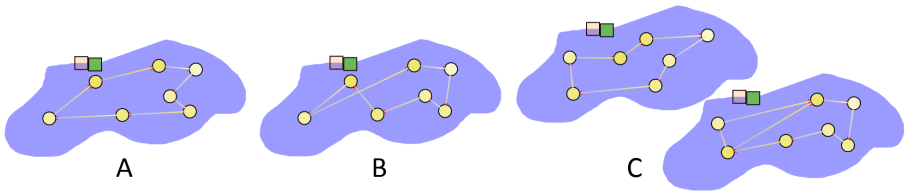

Figure 11. A) A simple loop; B) A slightly modified copy of the loop in A); A) and B) played together will form a complex musical texture; C) Both loops can be modified in real time to further evolve the texture created.

6.3 Canon

In music, a Canon is a musical form that is based on the repetition of the same melody by several instruments or voices, starting at different points in time. Because each instrument or singer has a delay with respect to the others, different notes sound simultaneously, resulting in chords. This example is a more sophisticated evolution of the harmonic loops, with a more sophisticated output. The triangular structure on top (see Figure 12) triggers the start of a full cycle of the melody (bottom), at times that are delayed by exactly one third of the length of the full melody. The cycle ends when one of the links of the triangular loop is cut (see video figure, Example 3).

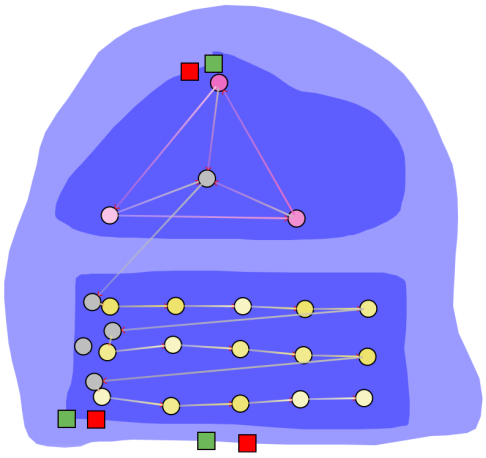

Figure 12. A Canon composition

6.4 Making Music with Shapes

This scenario takes advantage of the flexibility of our visual-acoustic mapping and the configurability of the interface to show how visually creative configurations can be created. In this case, two participants are collaborating in creating music to illustrate

Antoine de Saint-Exupéry's famous story from the Little Prince. Figure 13 and the video figure shows how two participants can create shapes, modify them, populate them with music, and then combine them to generate a small sound composition. This example illustrates how ToCoPlay can also be playful, and suitable not only for serious composition, but also for musical games. The creation of shapes to embed musical themes might also prove useful to improve recall and identification of musical sections when the piece is complex, even if not used with visual-aesthetical goals.

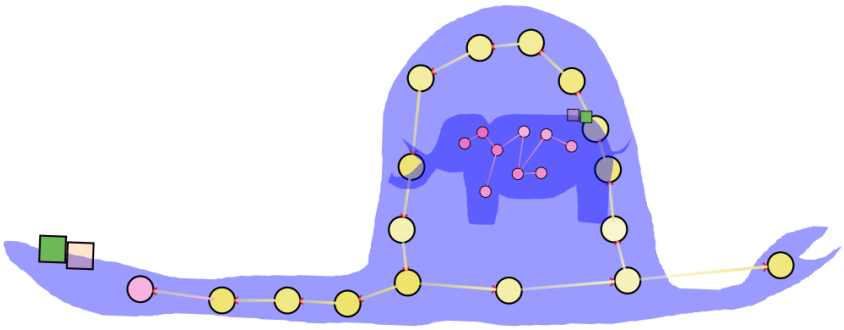

Figure 13. Two shapes are created with connected links inside: an elephant inside a snake; or is it a hat?

7 Discussion

7.1 Continuous versus Discrete Control

Designing music interfaces based on multi-touch interactive surfaces provides strong advantages; for example, a horizontal surface can enable collaboration, the input and output space coincide and are fully configurable, and a virtually infinite number of parameters and levels can be controlled through touch manipulations, traces and gestures. These manipulations are often of a continuous nature; for example, in the Reactable the rotation of an element is often mapped to a continuous parameter such as pitch or volume. Our system intentionally diverts from this model and instead offers a small discretized set of alternatives: a key can only play one of twelve notes, and time intervals are multiples of 125ms.

We fully recognize the importance of continuous control, especially for the manipulation of a complex, continuous control space such as sound synthesis. However, our design is more discrete because our goals are different from those of synthesizer and computer instruments. We prefer instead to make the results of actions easier to control and predict, and therefore easier to learn for people without a background in music. In exchange for this simplification of interaction, we are

reducing how parameters that can be manipulated through our system; for example, it is not possible yet to produce vibrato or tremolo (small variations in pitch or volume) with ToCoPlay.

7.2 Homomorphic Relation between Space and Sound

Many of the existing synthesizers and computer-based instruments described in the related work, as well as most VJing interfaces (e.g., [26]) provide visuals that depend mostly on the sound that is generated at that moment (or shortly before) and/or the input being provided. Since our main goal to support seamless transitions between performing and composing, we provide a tool that supports the broader view of the piece required by composers. We achieve this through our straightforward mapping that relates the visuals of the interface (the position, colour and lengths of the links) to sound. In particular, the mapping between the length of links and the duration and sequence of notes was of great help and impact when creating the examples described above, and it was often understood in demonstrations before it was explained. This relationship between the visual and the sound spaces is also useful in enabling shifts from composition to performance, since looking at the interface could allow performers to intervene and perform modification of a composed element in real time.

Although it would have been possible to create one-to-one relationships between the spatial and the acoustic spaces, we preferred to allow some freedom. Unlike traditional musical notations, which are generally isomorphic, our spatialisation of time allows several patterns to generate the same sequence of notes; for example, by changing the angle between keys without changing their distance or using dummy keys. This makes space for layouts that are not exclusively following musical requirements, but also accommodate interface considerations (e.g., collaboration, shape of the hand as in Examples 1 and 4), and aesthetic or storytelling possibilities, as shown by Example 4. We believe the visual aesthetic elements can be complementary and important for performing and composing, especially for people without musical background and for children; in fact, notations that cross media boundaries in art are not uncommon (e.g., Apollinaire's concrete poetry).

7.3 Few Building Blocks Can Go a Long Way

Our current version of ToCoPlay is certainly limited in several aspects, and does not provide many of the expressive channels that other music applications offer. Nevertheless, we found our decision to keep the interface to a few elements and rules useful and rewarding. We believe that the small set of elements will make the interface easier to learn, and we showed through our examples how initially simple combinations of a few elements can result in fairly complex sound textures (e.g., Example 2).

This kind of approach is particularly suited to a multi-touch interface, and generally difficult to apply to tangible interfaces such as the Reactable [15] and Audiopad [21], which would require many physical elements and can be quite expensive. In multi-touch systems have advantages in that we are free to replicate elements until the screen is full. However, touch detection and touch feedback do not achieve the same tactile and haptic quality of tangible interfaces.

7.4 Limitations

ToCoPlay is an exploration of musical interfaces and, as such, has helped us uncover new challenges and presented new difficulties. Probably most obvious is the need to improve the quality and quantity of samples for instruments to improve the quality of the sound. The sinusoidal- and square-wave generators used as initial instruments were adequate for our initial explorations, but we are aware that new, richer sounds can substantially help increase the musical value of the interface. Adding an external sound generation engine (possibly on a different machine) can help improve the quality of sound and, at the same time, solve some of the problems derived from dealing with an interface that has many elements and momentarily freezes for some operations such as large container creation.

Similarly, we have contemplated adding new controllable parameters to increase the expressivity of the system. For example, changing the volume (dynamics) of individual notes or containers can be valuable for composing. We also do not currently provide a method of creating pauses and delays in a composition. Future changes in this direction must, however, be weighed against the increase of interface complexity that contradicts our strategy of maintaining a simple set of basic elements and interactions.

Some of the aspects of our visual-acoustic mappings also present space for improvement. For example, we have noticed that the mapping from the HSV colour space of keys to their note's pitch is difficult to read and compare, especially if notes are distant from each other. We are already considering other kinds of mappings based on symbols and non-linear mappings of color to pitch, and mappings that would facilitate the inclusion of more than one octave.

Another issue with our spatial mapping is the ability to read complex scores of music. We are interested in finding out whether spatial relationships could become problematic (e.g., with a lot of containers embedded within one another) and how the interface can be modified for large scale compositions.

Finally, our current evaluation of the system has been constrained to just a few people from our department. Although reactions have been positive, we intend to validate our design by placing it into the real world and gather more impressions to produce a new, better version.

8 Conclusion

This paper presents ToCoPlay, a multi-touch tabletop musical interface that allows multiple people to create music. The main goal of ToCoPlay is to provide an interface that enables the performance of music while also encouraging composition. We applied four main strategies to design a system that allows seamless transitions between the performance and composition activities of participants: enabling control, enabling configurability, creating a simple visual-acoustic mapping, and relying on a minimal set of interface elements and operations. We also provide a set of four examples that demonstrate different aspects of the operation of the interface and its expressive capabilities. Finally, we discuss the implications of our design decisions and how they make ToCoPlay unique.

9 Acknowledgements

We would like to thank Uta Hinrichs for her valuable assistance creating the first version of the video. We are also grateful for comments and discussion from Lawrence Fyfe and many other members of the Interactions Lab and the Innovis group. This work was supported in part by Canada's Natural Science and Engineering Research Council (NSERC), Canadian Foundations for Innovation (CFI), Alberta's Informatics Circle of Research Excellence (iCORE) and by SMART Technologies Inc.

References

1. Akustich. http://modin.yuri.at/tangibles/data/akustisch.mp4. Last accessed January 2011.
2. Bischof, M., Conradi, B., Lachenmaier, P., Linde, K., Meier, M., Pötzl, André, E., 2008. Xenakis: combining tangible interaction with probability-based musical composition. In *Proceedings of TEI '08*. ACM, pp. 121-124.
3. Bonabeau, E., Dorigo, M., Theraulaz, G. 1999. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press.
4. Blaine, T., Perkis,T. 2000. The Jam-O-Drum interactive music system: a study in interaction design. In *Proceedings of DIS '00*, ACM, pp. 165-173
5. Condio. http://media.aau.dk/~gtal05/condioProject.html. Last accessed January 2011
6. Cubase. http://www.steinberg.net/en/products/cubase/cubase6_start.html. Last accessed January 2011
7. Davidson, P.L., Han, J.L. 2006. Synthesis and control on large scale multi-touch sensing displays. In *Proceedings of NIME '06*, IRCAM, pp. 216-219.
8. Fyfe, L., Lynch, S., Hull, C., Carpendale, S. 2010. SurfaceMusic: Mapping Virtual Touch-based Instruments to Physical Models. In *Proceedings NIME '10,* pp. 360-363.
9. Garage Band. http://www.apple.com/ilife/garageband/. Last accessed January 2011.
10. Gardner, M., On cellular automata, self-replication, the Garden of Eden and the game life. 1971. *Scientific American*, 1971 v.224, n.4. pp. 112-117.

11. Hochenbaum, J., Vallis, O., Bricktable: A Musical Tangible Multi-Touch Interface. In *Proceedings of Berlin Open Conference 09*, Berlin, Germany, 2009.
12. Hochenbaum, J., Vallis, O., Diakopolous, D., Murphy, J., Kapur, A. 2010. Designing Expressive Musical Interfaces for Tabletop Surfaces. In *Proceedings of NIME '10,* pp. 315-318
13. Hauert, S., Reichmuth, D., Instant City. http://www.instantcity.ch. Last accessed January 2011.
14. Jordà. S. Sonigraphical instruments: from FMOL to the reacTable. 2003. *Proceedings of NIME'03*, pp. 70-76.
15. Jordà, S., Geiger, G., Alonso, M., Kaltenbrunner, M. 2007. The reacTable: exploring the synergy between live music performance and tabletop tangible interfaces. In *Proceedings of TEI '07*. ACM, pp. 139-146.
16. Max/MSP. http://cycling74.com/. Last accessed January 2011.
17. Microsoft Corporation. Microsoft Surface. http://www.microsoft.com/surface/. Last accessed January 2011.
18. Krueger, M.W., Gionfriddo, T., Hinrichsen, K. 1985. VIDEOPLACE - An Artificial Reality, In *Proceedings of CHI'85*, ACM, pp. 35 - 40.
19. Newton-Dunn, H., Nakano, H., Gibson, J. 2003. Block jam: a tangible interface for interactive music. In *Proceedings of NIME '03*. pp. 170-177.
20. Noteput. http://www.jonasheuer.de/index.php/noteput/. Last accessed January 2009
21. Patten, J., Recht, B., Ishii, H. 2002. Audiopad: a tag-based interface for musical performance. In *Proceedings of NIME '02.* pp. 1-6.
22. Scott, S.D., Carpendale, M.S., Inkpen, K.M. 2004. Territoriality in collaborative tabletop workspaces. In *Proceedings of CSCW '04*. ACM, pp. 294-303.
23. Sound Storm. http://subcycle.org/. Last accessed January 2011
24. Stavness, I., Gluck, J., Vilhan, L., Fels, S. 2005 The MUSICtable: a map-based ubiquitous system for social interaction with a digital music collection. In *Proceedings of ICEC '05*, Springer, pp. 291-302.
25. Stereotronic Multi-synth Orchestra. http://www.fashionbuddha.com/. Last accessed January 2011.
26. Taylor, S., Izadi, S., Kirk, D., Harper, R., Garcia-Mendoza, A. 2009. Turning the tables: an interactive surface for vjing. In *Proceedings CHI '09*. ACM, pp. 1251-1254.
27. Villar, N., T. Lindsay, A., Gellersen, H.. 2005. Pin & Play & Perform: a rearrangeable interface for musical composition and performance. In *Proceedings of NIME '05*, pp 188-191.