ANTracks 2.0 - Generative Music on Multiple Multitouch Devices

Björn Wöldecke Christian Geiger
University of Applied Sciences Düsseldorf
Josef-Gockeln-Str. 9
40474 Düsseldorf, Germany
{bjoern.woeldecke, geiger}@fh-duesseldorf.de

Holger Reckter
University of Applied Sciences Harz
Friedrichstr. 57-59
38855 Wernigerode, Germany
hreckter@hs-harz.de

Florian Schulz
Berlin, Germany
florianschulz@me.com

ABSTRACT

In this paper we describe work in progress on generative music generation on multi-touch devices. Our goal is to create a musical application framework for multiple casual users that use state of the art multitouch devices. We choose the metaphor of ants moving on a hexagonal grid to interact with a pitch pattern. The set of devices used includes a custom built multitouch table and a number of iPhones to jointly create musical expressions.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces – Haptic I/O, Input Devices and strategies, Prototyping, User-centered design; I.3.6 [Computing Graphics]: Methodology and Techniques – Interaction technique

Keywords

Generative music, mobile interfaces, multitouch interaction

1. INTRODUCTION

Traditional instruments evolved over a long time to provide trained musicians with a large repertoire of interaction techniques to express themselves. Quite often the mechanical and electronic music instruments are operated with hands, feet or mouth. Traditional graphical user interfaces based on the WIMP (windows, icons, menu, pointer) paradigm reduce the human user basically to a “one finger – one eye” creature, at least from the computer’s point of view. Hence, the challenge of successful NIME design is how the designer can escape the traditional limitations of desktop interaction and how he can provide a natural interaction similar to traditional musical interfaces. As computer devices develop further we recently saw the break-through of natural interaction techniques as provided by gesture-based multitouch devices. This is mainly caused by the availability of dedicated hardware and software to the general public as demonstrated by Apple’s iPhone or Microsoft’s Windows 7. In this work we focus on a “low-floor” approach for a musical interface, which means that casual users may successfully interact with the interface and gain a positive user experience. We developed a concept for a generative music application and used the natural or reality-based interaction style, as suggested by [11]. In this paper we present an iPhone application as well as an implementation for a custom multitouch table.

2. RELATED WORK

In this section we look at selected related work in the areas of human computer interaction for NIMEs, mobile and generative music composition and table-top interaction for musical interfaces. A special topic at NIME is mobile music technology that is part of an emerging field since 2004 [13][9]. Tanaka presented the malleable music project in [19]. This approach placed participative music mixing in a mobile environment. Mobile devices are equipped with sensors and connected to a music engine that provides selected music streams. User interaction, e.g. tapping on a device remixes the sounds and creates a music stream that reflects user-related information. Georg Essl presented a number of interesting projects on mobile music generation including the newly established MoPho, a mobile phone orchestra. Recently he described the challenges of using mobile devices for music generation in [6]. The opening of the iPhone developer program to the general public led to a number of interesting musical applications for mobile devices that are available in Apple's AppStore. Starting with ports of traditional step sequencers, the use of mobile devices to control desktop synthesizers has increased. More relevant are innovative ideas that employ the new interaction techniques these devices support. Smule's Ocarina mimics the real flute instrument by rendering "holes" on the display that can be tapped simultaneously [21]. Blowing into the phone's microphone generates a tone and tilting the device creates a vibrato. It is possible to connect the device to a network so that other users can listen. Closely related is Bloom, a generative music application created by Brian Eno and Peter Chilvers. The user can generate bell-like tones that differ in pitch according to the Y-position of a user's tip on the surface. A growing colored circle (a bloom) is used to visualize a tap. This lasts as long as the mapped sound is played. The sequence of taps is stored and repeated. If the application is idle a random sequence is generated and played. Shaking the device clears the sequence. Similar work by Eno includes Trope and Air [5]. Toshio Iwai's Electroplankton [10] is a similar musical application for the Nintendo DS that uses the position of game objects for sound generation. Objects called Luminaria move around on a rectangular grid and can be controlled by switching the direction at each crossing point. The game uses touch, stylus and microphone for user interaction. The most important and most influential project of musical performance on a multitouch table is due to Martin Kaltenbrunner who developed the reacTable [12]. This instrument recognizes physical tokens placed on the table surface and generates sounds and effects according to the selected token type. Turning and moving the tokens controls selected sound parameters like pitch, volume or vibrato. Wim Fikkert presented a novel approach of a tabletop installation that provides casual users with a positive musical user experience [7]. Based on a chromatic scale multiple

users can create musical expressions by drawing lines on a Touch Table. Similar to our project, this work was inspired by Reactogon [17], a tangible interface that allows defining a step sequence on a chromatic scale using physical objects. The Xenakis project [3] is another interesting example of generative music composition on a multitouch table. It is a multi-user instrument that allows to stochastically create music using a tangible interface similar to the reacTable.

In our application we combined randomized music generation based on a very simple algorithmic behavior with multitouch user input on small and large interactive surfaces.

3. GENERATIVE MUSIC ON A MULTI-TOUCH SURFACE

The presented work is a significant refinement and extension of ANTracks [22], so we call it ANTracks 2.0. Our interface should provide a positive user experience for the casual user who should be able to create harmonic expressions with an intuitive and minimal user interface. Thus, we decided to use computer-generated 'generative music' similar to related projects (see section 2). In our application the user defines starting notes of pitch sequences and the system computes the next note of each sequence according to a simple algorithm. To illustrate this for the casual player we used the metaphor of ants that move on a hexagonal grid.

3.1 Musical Ants

In real live, ants have a colony from where they fan out to seek food and bring it back home. To transfer this metaphor to the music domain, some simplifications have to be conducted, so that the generated musical result is not too chaotic.

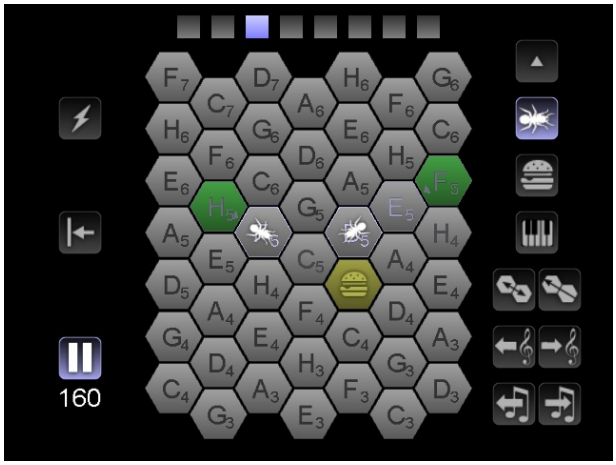

Figure 1. ANTracks 2.0 screenshot

In our algorithm, each colony is home and starting point of only one single ant, each – so the number of colonies is always equal to the number of ants. When launching the simulation, the ants start moving in the direction that is configured for their respective home field. Every field of the hexagonal grid has such a direction information telling an ant where to go next. If no direction is defined, an ant keeps its previous direction for the next step. The hexagonal grid is limited to the size of the screen. So there must be rules to define what happens, when an ant reaches the border. We decided to let the ants "reflect" from the border like a ball bouncing from a wall. It is also possible to configure another behavior, where the ants choose a random direction. This produces more variations in the paths across the grid. Each field is like a key on a piano and has an assigned pitch that is used to produce a sound when that field is triggered. There are predefined pitch patterns for the whole grid that can be loaded, modified and saved. These patterns are detailed in section 3.5. Every ant has its own rhythm that is initialized to a random pattern when the ant is created. A rhythm is a list of duration pairs (t, w), where t is the duration an ant should trigger the occupied field and w is the duration to wait for stepping to the next field. A step is not necessarily attended by a move. So, if an ant does not want to move, because there is no direction telling where to go next, it is possible that the same field will be triggered more than once. To produce repeating melodic patterns, there is a global counter representing the time how long the ants live. If this counter reaches the maximum live time, all ants die and new ants are created in the colonies. Besides the colony fields, it is possible to define food fields. If an ant discovers such a field, it takes the shortest path home and then tries to find the food field again, running back to the field where it expects to find the food. If the user has removed the food element in the meanwhile and the ant encounters an empty field, it continues in the direction of its last move. Food elements can be used to induce temporary variations into melodic paths.

3.2 Multitouch Interaction

We decided to use multitouch interaction techniques as an intuitive and natural way to interact with the application. For this, we defined a small set of multitouch capable widgets.

Figure 2. ANTracks 2.0 on the multitouch table Flux [8]

On the top row of the screen (see figure 1) is the display of the lifetime counter. The blue square goes through all grey squares counting the beat with the currently set tempo (in the screenshot, this is 160 bpm). To change the tempo, the play button can be used as a dial, when tapping on the button and dragging the finger outside. The angle of rotation then determines the tempo change. There are also two other buttons on the left side to reset the counter and to clear all fields. Most of the buttons on the right side of the screen affect the editing mode. This mode controls how the application interprets multitouch gestures on the hexagon grid:

Piano mode: This is the default mode when launching the application. In this mode, tapping on a field triggers a sound with the pitch associated to that field. The behavior resembles a piano. It is also possible to drag the finger across several adjacent fields to produce pitch sequences. This is especially interesting when using the Melodic Table (see section 3.5), because the "pianist" can play almost all possible melodies within the selected diatonic scale without releasing the finger.

Colony configuration mode: A tap on a field defines a colony, the starting point of an ant. A swipe is used to set the field's direction. Another tap on an existing colony field will erase that colony and its associated ant.

Direction configuration mode: This mode is similar to the colony configuration mode. The only difference is, that no colony is created when doing a swipe. Only the direction of the field is modified.

Food configuration mode: When this mode is active, a tap on a field places a food element. Another tap on an existing element erases it.

Pitch assignment swap mode: In this mode, dragging a field onto another one swaps the associated pitches.

Pitch assignment shift mode: In this mode, dragging a field in one of the six directions shifts all assigned pitches one field in that direction. The last pitch before the border is shifted to the other end of the line.

There are are also functions to load and save pitch patterns and map patterns. Pitch patterns tell the application the pitch of each field in the hexagon grid. Map patterns are all other field configurations telling the application the direction information of the fields and where the colonies and food elements are placed. When loading a pattern, a new screen with a selection list of all known patterns appears.

3.3 Multiple Multitouch Interaction

The long-term objective of this work is to develop a framework for multiple multitouch devices that are connected and/or connectable with each other to open up a new dimension of collaborative multitouch interaction. A first step towards this goal is the implementation of one single application running on different multitouch devices and to synchronize the musical parameters. At this early stage of development, the application works on a multitouch table and on the iPhone while showing the same widget layout. This made it possible to distribute touch events between the devices to synchronize them. The next step will be the extension of the synchronization mechanism to allow for different user interfaces. This way, every device can show its own view of the parameters. Currently, the application does not adapt to the different capabilities of the used multitouch devices. Ostensibly, there are different display sizes which, in the simplest case, necessitate different user interface layouts and interaction techniques. Because of its small size, a mobile phone application is best suited for single user interaction. This makes it possible to use simpler multitouch mechanisms, because there can be no conflicts between different users trying to interact on the same surface. In case of the iPhone, there are additional sensors like an accelerometer which could be used to control sound parameters like pitch bend. In contrast to a mobile device, the immobility of a table does not allow for such functionality. So, there must be additional widgets or interaction techniques on the surface to control these parameters. An advantage of a multitouch table compared to a mobile device is that the user does not have to hold the device in her hand. This allows the use of the second available hand to improve the multitouch experience employing two-handed interaction techniques. Furthermore, with the large screen space more than one user can interact with the application at a time. The larger surface of a table also offers the potential for more GUI elements to be displayed and controlled simultaneously.

3.4 Implementation

As we want to run the application on multiple multitouch devices running different operating systems, we paid attention to develop a portable codebase for the whole functionality. This was also to reduce code redundancy and error-proneness. Considering these requirements, we decided to write the application in C++ and use OpenGL / OpenGL ES to display the graphics. For the communication of sound events, we provide a back-end based on Open Sound Control (OSC) [15], which is a simple UDP-based network protocol specialized for musical applications. Currently we have ports for Linux, MacOS X and iPhoneOS. For the OSC protocol, we used the oscpack library [16] which worked seamlessly on all of them. As these operating systems are UNIX compatible, there were only few modifications necessary. The main differences are between the iPhone and the other two. For Linux and MacOS X, we use the GLUT library to create an OpenGL window and the TUIO protocol [20], which is based on OSC, to receive multitouch events. On the iPhone we had to implement a small GUI layer in Objective-C for this purpose. We also wanted to provide the possibility to run the different multitouch devices standalone. Because of this, the iPhone port was a little bit more complicated. The limited multitasking capabilities did not allow to start a software synthesizer as a second task. Therefore, we implemented a very simple sample-based sound generator. Furthermore, we implemented MIDI back-ends for better operating system integration. On Linux we make use of the Advanced Linux Sound Architecture (ALSA) [1] to output MIDI messages. For MacOS X the MIDI messages are delivered through CoreMIDI [14]. An advantage of a direct MIDI connection is a slightly better latency, while the OSC back-end is useful to provide network transparency.

Figure 3. ANTracks 2.0 running on the iPhone

3.5 Pitch Patterns on a Hexagonal Grid

Each field of the hexagonal grid has a pitch. A pitch pattern is a configuration of pitches for the whole grid. When testing different pitch patterns on the hexagonal grid and listening to the results, one important question comes up: Which one is the best pattern? If the pattern is too simple like putting seven octaves on top of each other with the C on the left side and the B on the right side, the musical results quickly become very boring. The reason is that in such a constellation only a small set of melodic intervals can emerge. In the above example, there will only be the intervals of minor and major second, octave, as well as minor and major seventh – very dissonant intervals, except the octave. Maybe there are better patterns which allow for more useful intervals. Indeed,

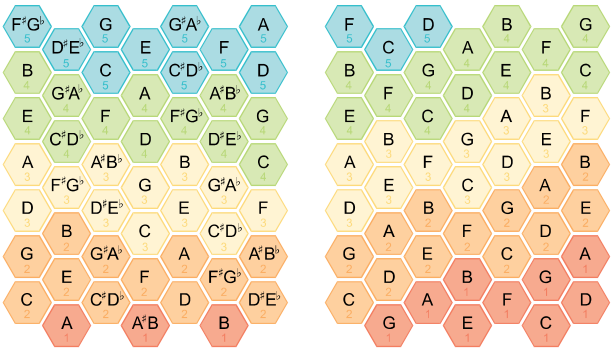

Figure 4. Harmonic Table and Melodic Table

there are such patterns. One prominent example is the *Harmonic Table* (see figure 4) used on C-Thru Music's AXiS MIDI controllers [4] or the Reactogon [17]. This pattern is optimized for mapping chords. When selecting C-E-G – the C major chord – the pitches are arranged in a triangle with adjacent fields in the north and north-east of the root note. The Harmonic Table has one interesting property: When displacing the same triangle to another pitch, e.g. G-B-D, this is still a major chord. The same applies to the minor, diminished and augmented chords. Even seventh and extended chords have individual shapes that can be displaced to every other pitch without losing their characteristics. When using the Harmonic Table for ANTracks 2.0, the result is not very satisfying and sounds odd. The reason is, that an ant goes along a straight line in some direction for several steps and there are only three different melodic intervals that can be produced this way: minor third, major third and major fifth. When creating tonal music – which is the one most people are accustomed to – a melody primarily consists of the pitches selected from one single diatonic scale. When an ant goes to the north-east, only major thirds are played, which means the ant repeatedly leaves the diatonic scale. When there is a change in direction, it is even worse. The resulting melodies sound arbitrary to tonally trained ears. A possible solution could be to find a pattern that does not allow an ant to leave the diatonic scale. One such pattern is the stack of octaves mentioned above. But it has the disadvantage of causing dissonant intervals. We found another pattern, which we call the *Melodic Table* (see figure 4), that has the same properties but increases the number of possible intervals. The Melodic Table permits every melodic interval of a diatonic scale except the seventh, by going in one of twelve directions – to one of the six neighbors or along one of the six edges between two neighbor hexagons. The added directions are acceptable, because the leap is not too far and does not skip another field. There is not only one single Melodic Table. By transposing the C major Melodic Table (raising the pitch of each field in the grid), it is possible to generate one table for every diatonic scale starting on one of the pitches of the twelve-tone scale. The 7x7 grids shown in figure 4 have one major shortcoming: They have borders. This means, the number of possible melodic intervals are limited on the border fields. Because of its structure, the Melodic Table can be extended to the left and right revealing a repeated pattern. To eliminate the left and right borders, the grid could be mapped onto a cylinder to geometrically reflect the cyclical nature of the pattern. We are planning to extend the existing application to make use of this possibility.

4. DISCUSSION

In this paper we presented a work in progress implementation of a novel musical application that works on multiple multitouch devices and different operating systems. This is intended as a foundation for further research to develop a collaborative musical interaction utilizing mobile and stationary multitouch devices. More advanced multitouch interactions for the presented application are currently under development. Furthermore, we work on improving the concept of the *Melodic Table*. The current progress of the project will be showcased at [2].

5. REFERENCES

[1] Advanced Linux Sound Architecture. www.alsa-project.org

[2] ANTracks web site. http://www.antracks.de/

[3] Bischof, M., Conradi, B., Lachenmaier, P., Linde, K., Meier, M., Pötzl, P., and André, E.: XENAKIS: Combining tangible interaction with probability-based musical composition. In Proc. of 2nd Int. Conf. on Tangible and Embedded interaction – TEI '08, ACM, New York, 2008.

[4] C-Thru Music: AXiS-64 MIDI controller. http://www.c-thru-music.com/cgi/?page=prod_axis-64

[5] Eno, B.: Generative Music. http://generativemusic.com

[6] Essl., G., Wang, G., Rohs, M.: Developments and Challenges turning Mobile Phones into Generic Music Performance Platforms, In Proc. of Mobile Music Workshop, Vienna, 2008

[7] Fikkert, W., Hakvoort, M., van der Vet, P., and Nijholt, A., FeelSound: Collaborative Composing of Acoustic Music. In: The 5th Advances in Computer Entertainment Technology Conf. (ACE '09), Athens, Greece, pp. 294-297, 2009

[8] Flux – Fully Liberating User eXperience.http://milab.org/projects/flux/

[9] Gaye, L., Holmquist, L. E., Behrendt, F., Tanaka, A.: Mobile Music Technology: Report on an Emerging Community. NIME 2006, Paris, France

[10] Iwai, Toshio: Elektroplankton. http://en.wikipedia.org/wiki/Electroplankton

[11] Jacob, R.J.K., Girouard, A., Hirshfield, L.M., Horn, M.S. Shaer, O. Solovey, E.T., Zigelbaum, J. Reality-Based Interaction: A Framework for Post-WIMP Interfaces. Proc. ACM CHI 2008 Conference, pp. 201-210, ACM Press

[12] Kaltenbrunner, M., Jordà, S., Geiger, G., and Alonso, M.: The reacTable: A collaborative musical instrument. Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WET ICE, 2006, pp. 406-411.

[13] Kirisits, N., Behrendt, F., Gaye, L., Tanaka, A.: Creative Interactions - The Mobile Music Workshop 04-08, Vienna, 2008, http://www.mobilemusicworkshop.org/

[14] MacOS X Reference Library. http://developer.apple.com/mac/library/navigation/index.html

[15] Open Sound Control. http://opensoundcontrol.org/spec-1_0

[16] oscpack. http://www.audiomulch.com/~rossb/code/oscpack/

[17] Reactogon. www.youtube.com/watch?v=AklKy2NDpqs

[18] Schulz, F.: ANTracks - generative mobile music composition. Bachelor Thesis (in German). Hochschule Harz, 2009

[19] Tanaka, A., Tokui, N., and Momeni, A. Facilitating Collective Musical Creativity. In Proc. of ACM Multimedia, 2005

[20] TUIO protocol. http://tuio.org

[21] Wang, Ge: Designing Smule's Ocarina: The iPhone's Magic Flute. Proc. of the Int. Conf. NIME, 2009, Pittsburg, USA

[22] Geiger, C., Schulz, F., Reckter, H.: ANTracks - Generative Mobile Music Composition. Advances in Entertainment Computing, ACE Conf., ACM Press, 2009, Athens, Greece