IRIS

a circular polyrhythmic music sequencer

Viljo Malmberg

Master's Thesis

Aalto University
School of Art and Design
Department of Media, 2010

Downloadable material related to this Master's Thesis can be found at:
http://www.viljomalmberg.com/iris/thesis/

Author	Viljo Malmberg
Name	Iris: A Circular Polyrhythmic Music Sequencer
Date	October 18th, 2010
School	Aalto University, School of Art and Design, Department of Media
Degree	MA in New Media
Keywords	electronic music, sequencer, polyrhythms, user interface, concept design

Abstract

Iris is a conceptual circular music sequencer, with a working touchscreen interface prototype developed as a proof of concept. It seeks for a contemporary and explorative, yet intuitive way of visualizing and manipulating musical rhythms.

The Iris interface consists of several concentric rings, representing step pattens. A playhead constantly rotates around the rings, playing turned-on steps as it comes across them. Each ring is assigned a midi note, and each active step plays a note on message.

The number of steps on each ring can range from 2 to 64. The user can change the resolution of each ring individually: as a result, during one playhead cycle, the playhead can pass 16 steps on one ring, and 12 steps on another one, as an example. This allows the user to create complex polyrhythms.

Each ring can be separately rotated in an arbitrary position with a two-finger swipe gesture. This causes the orientation of a single rhythm layer to change in relation to other layers as well as with the playhead. With minor shifts, phenomenons like swing or shuffle can also be produced.

This thesis describes the Iris concept in detail, including the design decisions on the interface and architecture. It provides a contextual framework for the production: I explore the methods of creating rhythms in different music genres. I also look at the history of sequencing and looping music, and benchmark existing products and productions related to the concept.

The outcomes of the project are reviewed from my personal perspective as a musician and designer. Initial user feedback received from the prototype is also presented. The source codes for the implementations on the iPad and Max 4 Live environment are released as a part of this thesis.

Acknowledgements

Outwardly, this thesis project has been a solo effort. In reality, I couldn't have completed it without all the invaluable help I've received in numerous ways. I would like to thank the following people for making this work possible:

Anitta and Sara, the two girls of my life, for your support, and for letting me take the time to complete this, with probably the worst timing thinkable.

My directors Antti and Markku, for comments, support and valuable ideas.

Juha, Paavo and Tuomas, for good comments, suggestions and the early testings of the interface.

Media Lab Master's Thesis Seminar and Sound Seminar attendees, for the giving constructive feedback on my presentations.

All the kind people, who have contributed to the open source community or otherwise created great software, and developed many of the tools used to create this production.

In Helsinki,
October 18th 2010,

Viljo

Contents

1. Introduction

1.1 Motivation and Background

I have approached the art of making music by two paths: As a kid, I started taking music classes and learning the very basics of playing piano and guitar, grasping a little western music theory on the go. At the same time, I had developed a growing interest in computers, and was excited about their possibilities. My first encounter with music software might have been when Jori Olkkonen's Megasound music editor for the C-64 was published in the C-lehti magazine back in 1988. I humbly typed in the source code printed on the pages, and was finally able to run the program. It felt like I suddenly had three more voices!

Soon after this, I became loosely involved with the demo scene, mainly through friends, and ended up listening quite a lot of music in demos, games and music disks. At the age of twelve, I was introduced to a genre of music software called *trackers*. They area a class of step-based sequencer applications, that organize music patterns in rows and tracks, named after the first of their kind, Karsten Obarski's Ultimate Soundtracker (Reunanen, 2010: 68-69).

Techniques for creating music with the tracker-like pattern based approach became familiar quickly, by both learning from experienced composers' pieces and also by trial and error. I learned simple tricks like fast arpeggios to imitate richer polyphonies with just few sound channels. Echo effects could also be created by copying notes from one track to another one, displacing them by a couple of steps, and adjusting their volume.

Compared to struggling with my piano lessons, "programming" music felt nice and practical, but also a bit like cheating. I think this odd way of approaching musical mark-up (in contrast to being fluent in reading sheet music) has left me with a certain 'problem solver's viewpoint: Interesting sounds and rhythms can be programmed in numbers, and developed further with an engineer's precision. Probably for the same reason, I ended up a lazy trainer and a mediocre guitar and piano player.

I look at this thesis project as a big soup mixing my various interests: performing as a electronic dance music DJ has left me with a love for turntables and their tactile approach to music. My education and daily work as a designer drive me to solve challenges in information visualization and interfaces. I have spent a good deal of my childhood in the front of a computer screen, tinkering with small projects. A strange itch appears when I get a new idea for a project – I have to get working on it right away.

1.2 Thesis Scope and Overview

The written part of this thesis addresses three main questions:

1. By what means, tools, and initial ideas, was Iris realized?
2. What kind challenges were faced during the design and implementation?
3. What possibilities are embedded in the concept of visualizing and manipulating musical patterns in a circular form?

This two-part thesis includes a production – a proof-of-concept prototype of the Iris sequencer, and this written part, which, in addition to this introdution, is divided in three:

Chapter 2 builds a contextual framework for the work. I first introduce a few milestones from the history of musical automata, that in part explain the origins of sequencing musical rhythms today. Then, I review some more recent projects and applications from the field of music and interaction design, involving sequencing of rhythms. Last, in the part "Forming Rhythm – 3 brief studies", I explore selected techniques of forming and conceptualizing complex rhythms, from viewpoints of different music genres and cultural settings.

Chapter 3 documents the creation of the Iris concept and prototype: I explain the process, from setting goals, through initial prototypes, to the improved proof of concept presented as the production part of this thesis. I also describe the elements and organization of the Iris interface and architecture. Along with the above, I open the reasoning behind my design decisions.

Chapter 4 presents the results of the project, using received feedback and self-reflection: I report the initial feedback from a small test group of electronic musicians with different backgrounds. I look at the production from a designer's and a musician's viewpoint, and review my own learning during the production of Iris. I also point possible directions for future applications and furher development, as well as paths for research.

2. Behind The Iris

2.1 Sequencing Music with Machines

What Is a (Step) Sequencer?

By a dictionary definition, sequencing simply means a process of ordering things (Cambridge Dictionary Online, 2010).

The history of mechanical devices for controlling playing of musical passages goes back all the way to the 14th century, when the first cylinder-driven bell-ringing device saw daylight. (Reuge – the history of mechanical music and music boxes, 2010). Another forerunner of the music sequencers today is the barrel organ (from the 1500s), which is driven by a barrel covered with metal spikes (Russ, 2009: 76).

A special moment in the history of electronic music, and the history of computers in general, was the first use of punch cards to control machines. The method was first implemented by the textile industry. According to the holes perforated in a card, a sewing machine mechanically performed the instructed operations. The first of its kind was a mechanical loom, invented by a French engineer Josef Marie Jacquard in 1801 (Lienhard, 1997). A Finnish reader can find a preserved Jacquard's loom from the Finnish Labour Museum Werstas, located in Tampere.

Similar punch cards soon served as an input method for Charles Babbage's seminal computational machines, such as the Analytical Engine. Some trivia included: Babbages friend and successor Ada Lovelace has later said that Babbage deeply hated music, and considered it a complete waste of his time. (Lee, 1994)

About 80 years later, the Pianolas were introduced. They utilized this same method of storing sequences of music on punch cards. Piano rolls were reproductions of actual performances, where the start and duration of the notes, as well as the playing dynamics

where stored, and later played using a special pneumatic player system. (Russ, 2009: 94)

During the 1960s and 1970s, in the era of big modular voltage-controlled analog synthesizers, a sequencer was a piece of hardware, that sent a looping sequence of notes to the synthesizer module. Majority of the devices used controlled the synthesizer with control voltages – the synthesizer determined the pitch of the played notes by the applied input voltage (Winfield, 1998). The inventor of the first such a sequencer device was the composer and inventor Raymond Scott in the mid-50s (d'Escrivan, 2007: 164). Popular bands, from Tangerine Dream to The Who, soon adopted step sequencers as tools for both live and studio use.

Figure *1: MFOS 16-Step Rotary Analog Sequencer. Courtesy of Stephen Drake.*

The generation of sequencers to follow were the CV and Gate -type sequencers. They used a simple computer to store the control voltages and note durations. A famous example of such devices is the Roland MC-8 MicroComposer., where music could be typed in as a sequence of numbers. (Russ, 2009: 192)

Roger Linn introduced his LM-1 sample based drum machine in 1979 (Russ, 2009:95). It featured a unique shuffle setting, that allowed to create an automatic swing-like feel to a step sequenced drum pattern (The Synth Museum, 2000). The Linn-style shuffle, or the "MPC Groove", also present in the Akai MPC Linn later designed, is a feature imitated even in many today's sequencer softaware (Kurasaki, 2006).

Along with the introduction of the MIDI standard in 1983, the voltage-controlled sequencers started to get out of fashion [1]. Around the same time, Roland gradually introduced its digital sequencer driven lineup of drum and bassline machines, that later defined a great deal of the electronic dance music sound palette for decades.

Today, the term *music sequencer* means a broad spectrum of tools, that can be used to record, and play back both audio and control data. Sequencers exist in both stand-

[1] *Today, many electronic musicians still utilize analog sequencers in their setups. New analog gear is also produced, like the Music From Outer Space rotary analog sequencer (see figure 1).*

alone instruments and special computer software e.g. Pro Tools, Nuendo, Cubase, Logic or Ableton Live. Tools for "old-fashioned" step sequencing exist in most of the above software packages. It can be seen as a method or tool among others – an approach that electronic music producers use to easily create machine-driven rhythms.

Probably the area where the step sequencing methods origination from the 60s and 70s are best preserved, are the different flavors of electronic dance music. Mark Butler, an American musicologist who's focus of research is largely in the area of eletronic dance music, describes the (step) sequencer as an essential function to electronic dance music production (Butler, 2006: 61-62).

Contemporary Sequencers

The "process of ordering things" can extend well beyond common music software or the 70s instruments looking like telephone switchboards. I'd like refer a couple of contemporary sequencer implementations:

Block Jam & The Tangible Sequencer

The **Block jam** is a polyrhythmic sequencer consisting of 26 small physical artifacts, called Blocks. The blocks act as an input device for a specifically designed sequencer software that interprets the arrangement of the blocks, and plays musical structures accordingly.

There are two kinds of blocks – play and path blocks: play blocks initiate sequences that travel through other blocks, and path blocks control the route that the signal marker (called cue ball) travels. A path block can connect to the other blocks from any of it's sides, and configured to send the signal to any direction. Together the blocks form a cluster that the play travels in. More, multiple play blocks can be placed to the cluster, to create multiple overlapping rhythms (see figure 2). The block cluster is connected to the computer and sequencer through a special play block, the "master block". (Newton-Dunn, Nakano, Gibson, 2003)

Another block-based sequencer, created by Jeffrey Traer Bernstein at the Sound Lab in The Princeton University CS Department and called **The Tangible sequencer,** is also made of small boxes. These colored blocks each contain a small microcontroller and a Zigbee radio component, to be able to communicate wirelessly (in figure 3). A

Figure 2: Block Jam blocks. Picture courtesy of Sony Computer Science Laboratories

Figure 3: The Tangible Sequencer. A Youtube video screenshot. Courtesy of Jeffrey Bernstein.

software application on a PC computer is used to drag and drop sounds to the blocks. (Bernstein, 2007)

Polymachine

I haven't come across many specifically polyrhytmic step sequencers, software or hardware, but the Polymachine fits the description. It dates back to 2003, and is created by the IXI audio collective that has also given workshops at the Aalto University Media Lab. The Polymachine is a piece of software running within the Supercollider environment. It creates 1-6 step tracks, that can each consist of a variable number of steps, defined at the program's startup. Each track also has it's own tempo setting, allowing to play the variably sized sequences on top of each others with variable speeds (in figure 4). The software is available for download at: *http://www.ixi-audio.net/content/body_software_polymachine.html*

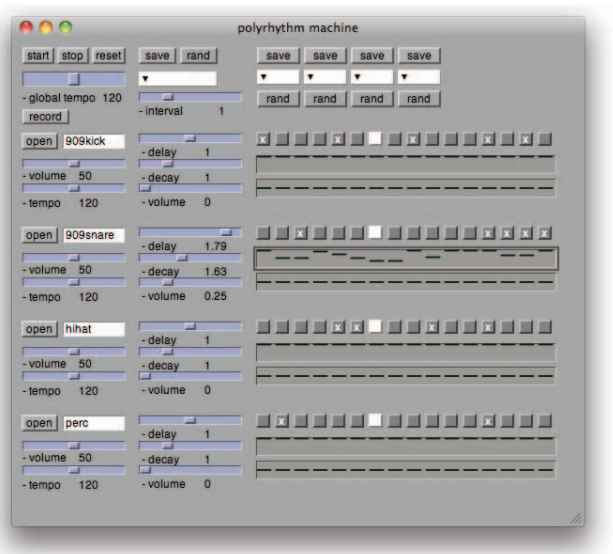

Figure 4: The Polymachine sequencer

The Loopseque

Loopseque is an interesting iPad music application I ran into while developing Iris. It looks a lot like Iris, it acts a lot like Iris – it is a circular loop sequencer, consisting of four concentric circles, each with 32 steps, which the developers, Casual Underground, call a Wheel Matrix. When in operation, the playhead rotates around the circle, playing activated steps. The user can choose of 17 specially designed samples sets to play with. Patterns can be stored for later use, as a wheel set can consist of several wheels (figure 5). (Casual Underground, 2010).

Due to the apparent visual relation between the Loopseque and this thesis production, a clarification is in place: Beyond the visualization of the step loop, there is a significant conceptual difference between Iris and Loopseque. The latter aims to be "an incredibly simple app for creating music", as stated on the loopseque website. After playing with the UI, I agree that it feels slick and intuitive to use, and one of the more usable iPad apps I have acquired and played with. However, or maybe because of this aim for simplicity, it functions very similarly to traditional drum sequencers. Although I think that loops visualized in circular form are a concept to study as such, there seem to be no new features or gestures in Loopseque, that explicitly benefit from the visualization. In other words, as a music tool, It might also work well without its circular paradigm.

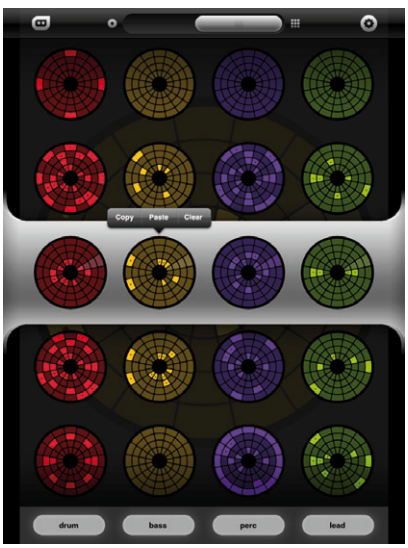

Figure 5: The Loopseque. Screenshot courtesy of Casual Underground.

In Iris, I try to enable explorativity with concurrent layers of rhythm. In the context of this work, I think of the different rhythmic layers (presented in Iris as concentric rings) as separate, trying to encourage the user to experiment with composite rhythms the layers blend into. To enable this, each ring in Iris can be of different resolution, and can be offset in time. In Iris, the visual paradigm, the use gestures and the underlying model are intertwined, and can't be easily separated from each others.

2.2 Forming Rhythm – Three Brief Studies

In this section, I take a glimpse on how the concept of rhythm is addressed in three selected music genres: traditional African music, American minimalist compositions dating back to the 60s and 70s, and modern electronic dance music.

I do not cover in-depth music theory, or give an extensive view of these musical styles in this thesis work. However, studying and listening to the genres has allowed me to find interesting connections between them, both in their history and their approach to creating rhythm. The examples provide an inspirational framework for this project, and possibly open paths for further study.

The Minimalist Techniques

Minimalism in music mostly refers to the style of selected American composers in late 60s and 70s. It shares its name and many properties with the simultaneously emerged visual arts movement (Gann, 2001).

Notable composers often referred as minimalists include Lamonte Young, Terry Riley, Steve Reich and Philip Glass – all four born less than two years apart from each others (Potter, 2000).

In an article published in American Music Center's web magazine New Music Box, composer and musicologist Steve Gann describes the key techniques utilized in early minimalist music. Among these are repetition, a steady beat, an additive process and phase-shifting (Gann, 2001).

The additive process means, that the simple rhythmic patterns evolve slowly during the song, by manipulating the length of the recurring musical phrases, or adding or layering new notes to them. Minimalist music is in many occasions referred to as Process Music.

Phase-shifting (sometimes referred to as *phasing*) can be implemented by playing two identical phrases simultaneously with minor tempo variations, so that the other phrase slowly goes out of (and back in) phase with the other (Gann, 2001).

Steve Reich, born in New York City in 1936, is considered one of the most recognized minimalist composers (Potter, 2000), and has used phasing widely in his early 70s works. In *Drumming* (1970-1971), Reich uses a tape recorder along with performing drummers, both playing the same rhythmic patterns, but with minor gradual tempo alteration, their playing thus going in and out of phase. Reich himself has said, that Drumming has been heavily influenced by his studies of West African drumming at the University of Ghana (Ross, 2006).

The Minimalist Influence

The repetitive approach of the minimalists has inspired many later artists. In his book Living Electronic Music, Simon Emmerson points out that minimalism is cited as major influence on various dance music genres. As an example example, samples from Steve Reich's Electric Counterpoint can be heard in The Orb classic, Little Fluffy Clouds (Emmerson, 2007: 68).

Another direct example is the Reich Remixed album, released in 1999, featuring recognized electronic dance music artists such as Coldcut, Ken Ishii and DJ Spooky. There's also a more recent release from year 2006, including some additional tracks.

Summary

The connection between the minimalist composing methods and modern day electronic dance music, many times created using step sequencers, serves as a source of inspiration for my work. Many electronic dance music genres utilize the same methodology and compositional tools: repetitive patterns and additive process, where more layers are gradually added to the song. In the process of developing Iris, I found it inspiring to extract some of the methods used by composers in the 60s and 70s, and examine how Iris could implement them again, creating a new offering to a step-sequencing electronic musician's toolkit.

And last, I love to play with the idea, that an interface that looks more or less like Sol Lewitt's painting *Whirls and Twirls*, is used to create rhythms with the same basic concepts of those in Reich's works (see the Massachusettes Museum of Contemporary Art website: http://www.massmoca.org/lewitt/walldrawing.php?id=1152).

Sub-Saharan Dance Drumming

The music and dance of Sub-saharan Africa, form an important part of the African diaspora: the legacy of African culture spread across the world. Many rhythmically rich Caribbean and Latin American music genres, such as rumba and salsa and other clave-based genres, are influenced by the musical roots of the African slaves. (Mundolo, 2010)

In his 1962 book "African Music in Ghana", J.H. Kwabena Nketia makes a distinction of two different rhythmic forms in Ghanaian music – *strict* and *free* rhythm. Songs with strict rhythm have a regulative beat that can be tapped to, or articulated with body movements. Free rhythm is characterized by the lack of a regulative beat – these type of rhythms are not danced to. These styles can sometimes be combined, and a "free" part can act, for example, as an intro to an otherwise strict-rhythm song. (Nketia, 1962: 64-65)

Below, the examples refer to the "strict rhythm" category, "which finds its highest expression in *dance drumming*", as stated by Willie Anku (2000).

The African perception of rhythm is strongly interconnected with culture. CK Ladzekpo (1995) describes dance drumming as an ancient institution of learning, and essential to the development of the sub-saharan human infrastructure.

The music is firmly controlled by a recurrent rhythm – an example of this is the bell pattern common in West and Central African drumming. This structure is not, however, always explicit in the music itself. It can exist solely in the performers' and audiences' minds (Anku, 2000). Once learnt, the reference grid helps the musicians and dancers to perceive the pulsation in a given piece of music (Grove Music Online). Anku presents the timeline of the bell pattern as a cycle, and notes that African music is perceived essentially as a circular concept, rather than linear (Anku, 2000).

Techniques of Building Rhythms

The technique of cross rhythm is essential to sub-saharan dance-drumming. It means the simultaneous use of contrasting rhythm patterns in the same metric context. Below, these patterns are referred to as Beat Schemes. CK Ladzekpo lists important techniques of cross-rhythms in Anlo Ewe rhythm organization (Ladzekpo, 1995):

Main Beat: The rhythm pattern that one uses as a focal point, and recognizes as the dominant beat scheme, or the pulse of the rhythm, is considered the main beat. The main beat can occur both in duple or triple pulse structures. Of the layered beat schemes, the

Main Beat is the most moderate, thus the most convenient as a focal point.

Composite Rhythm: When the contrasting main and secondary beat schemes are laid on top of each others, and kept repeating unchanged, they can be perceived as a composite rhythm. As the rhythm is interpreted as a single repeating entity by the listener, an effect of a static state is formed, in contrast to a constantly alternating pattern, which creates an effect of movement, or vitality.

A technique that Ladzekpo refers to as the **Technique of Polyrhythm**, means breaking the established composite rhythm by time-shifting the secondary beat scheme, so that it starts at a different moment in time, compared to the other beat scheme. The temporal shift can be of any size within the time span. This produces new variant textures to the rhythm.

Willie Anku points, that in contrast to Western art music, where the meter is externalized with the use of time signatures, in African music, there are ethnically perceived norms of beat perception. A superficially identical rhythm is perceived in a different way by people with different ethnic and cultural background. The Ewe, Yoruba and Bemba seem to find a regulative beat at a different position of the same rhythm (Anku, 2000).

A Representation of Real Life

In West African cultural understanding, the elements of dance drumming represent actual real-life phenomenons: The main regulative beat represents one's purpose in life, while a secondary beat represents obstacles and challenges that one comes across. The Anlo Ewe believe that the goals in life must be strong enough to be able to match the contrasting obstacles. Most sub-saharan ethnic groups share these same general concepts. (Ladzekpo, 1995)

The idea of a steady purpose, accented with obstacles, is exciting: It translates to me as a "creative chaos", where one can find inspiration by observing and interpreting the constantly altering surroundings. While listening to a complex rhythm, rather than theorizing, one can let the mind to relax and search for new answers and interpretations.

Layers of Electronic Dance Music

Electronic Dance Music (EDM) is a huge subject as such, and in musical styles, is split into countless sub-genres, each with their own rhythmic qualities. This said, I briefly introduce a function that seems to exist across most EDM genres: the layered approach to rhythm.

Multiple Layers

The composition style of electronic dance music involves creating several independent layers of sound to create complete rhythmic patterns (Keller, 2003: 4). The layers can be distinct in pitch, timbre and meter (Butler 2006: 117). The layers can interact in the compositions, and by doing so, make the composition rhythmically more interesting than the sum of its parts. Butler makes an example of the beginning passage of *Chemical Brothers* song *Piku* (found on album Dig Your Own Hole, 1997). He states:

> *"Several different metrical layers are present: a sixteenth-note layer suggested by the composite rhythm, and eighth-note layer corresponding directly to the synth part, and two different quarter-note layers (one suggested by durational accents in the snare drum, and the other by melodic high points in the synth part). These last two layers, however, are not aligned. Furthermore, the passage demonstrates at least two different kinds of ambiguity. First, because the absence of layers moving at a rate slower than the quarter note, the metrical organization of the excerpt is undetermined. Although the sixteenth-note and eighth-note layers tell us that the beat division is duple, the passage does not provide any information about metrical type; each quarter-note layer may be grouped in a variety of ways. Second, because the excerpt occurs before the meter has been clearly established, it is unclear which of the two quarter-note layers is referential. Rather than superimposing a "dissonant" layer on top of a previously established set of "consonant" metrical layers, the creators of the music present two relatively equal layers at once. As a result, listeners are free to choose either layer as regulative, and to change their minds as the passage unfolds." (Butler 2006: 117-118)*

After reading this, I did a re-listening of one of my teenage favorite albums. Indeed, the beginning of the song let me turn the rhythm loop around a few times in my head,

until the regulative beat eventually took over.

"Messing with the listener's head" with metrical dissonance is of course not unique to electronic dance music, as Butler notes in another paper, published in The Society of Music Theory journal. He makes examples of a couple of rock songs, like *Close to The Edge* by *Yes*.

Butler uses the word *ambiguity* to describe the situations where the dominant metrical division is not, or stops to be, self-evident in the song. He also introduces a term called "turning the beat around", which I have many times heard used by DJs. This method or phenomenon involves adding or removing a rhythmically dominant pattern, that acts in contrast with the other rhythmic layers, allowing another pattern to dominate, and lead to a reinterpretation of the rhythm in the lister's mind. (Butler 2006: 141-152).

Layered Passages of Different Length

In his Master's Thesis, Robert Keller gives an example of contrasting rhythms occurring. "*…Meet His Maker"* by *DJ Shadow* (MCA Records 2002) contains a guitar passage that demonstrates the phasing of different rhythmic layers. Starting from approx. 0:45, the song is composited of three layers, two of them in duple time, and one in triple time: the guitar plays a phrase of 12 beats (which can be heard as triples), the percussion plays a phrase of 8 beats, as the drum loop repeats every 4 beats. The layers thus go out of phase, and reach each others every 24 beats (the lowest number that can be divided by all 12, 8 and 4). (Keller, 2003: 26-28)

Summary

The examples above show, that complex rhythmic implementations do exist in electronic dance music, despite it's reputation of being rhythmically monotonous. Being heavily influenced by the DJ culture, which in many cases seems to stay at a distance from academic discourse, even this brief exploration feels refreshing. The rhythm of EDM doesn't have to be listened through only a strict 4/4 time signature and a genre label (although 4/4 probably was configured to the producer's sequencer when creating the piece).

3. Designing Iris

3.1 The Iris Concept

Iris is the goddess of the rainbow; she is the daughter of Electra and Thaumas, and serves as the messenger messenger for Zeus and Hera. [1]

Iris [ee'-ris]
noun 1. Iris, the rainbow. [2]

Iris is a conceptual music sequencer. It's stands firmly on the foundations of classic step sequencers, but builds on them by offering a novel way to work with rhythms. This is done partly through the interface solution, partly by the underlying design.

Using Iris, the user creates rhythms by operating a rainbow-colored wheel displayed on a touchscreen. The wheel consists of concentric circles, each divided into a number of segments, or steps. Rhythms are created by tapping or swiping the steps with a finger, toggling them on or off. Each circle represent a single note, each step is an instruction to play this note at a specified time. A playhead rotates around the circles, playing the active steps as it passes them.

The resolution of each ring can set to anything between 2 and 64. As a result, during one playhead cycle, the playhead can pass 16 steps on one ring, and 12 steps on another one, as an example. This allows the user to lay duple, triple or perhaps quintuple-based patters on top of each others. The length of the played notes correspond the resolution – on a ring divided to a 16-step grid, 16th notes are played.

Each of the rings can be rotated to a different angle, offsetting its steps in relation

---

[1] *Theoi Project, http://www.theoi.com/Pontios/Iris.html*

[2] *Velazquez® Spanish and English Dictionary*

to the other rings. Depending on the dominancy of the offset rhythm layer, this might lead to interesting textural changes in the composite rhythm, or even to a completely new interpretation of the rhythm by the listener. This concept relates strongly to the Technique of Polyrhythm in Ghanaian dance drumming, mentioned in chapter 2.2.

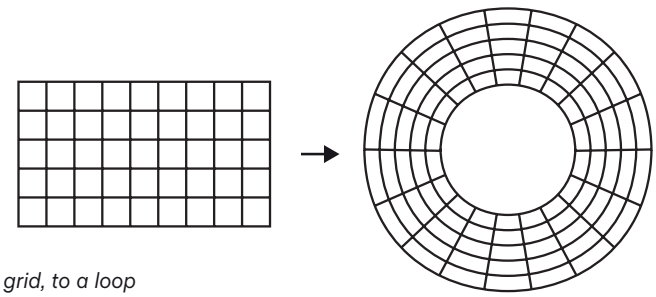

Figure 6: From a grid, to a loop

To phrase it in a very technical way, Iris transforms a 2-dimensional pattern of steps from a Cartesian coordinate system into a polar one (see figure 6). This creates a visualization representing a closed loop, and since the interface is designed for editing a loop of steps, the solution seems natural – visualizing recursive temporal data in a circular form is something most of us are used to interpret in form of clock faces. Thus, the solution fits to Norman's definition of natural mapping of controls (Norman 2002: 23). In the book Practical Design With Data, author Brian Suda, while problematizing circular visualizations in general, also notes that they suit well for representing time (Suda 2010: 153-154).

The transformation effectively hides the visual cues for the start and end of the pattern. While layers of the rhythmic texture can be offset, they retain their visual form. There is no need to set new start and end points for a loop – the loop is visualized just as it was and is – a repeating pattern. In a way, it seems less restraining to think of loops without start and end points; by a dictionary definition, loops are "endless".

A circular visualization as a representation of musical time is not a new idea as such, and there are quite a few implementations of ring-shaped interfaces for music sequencers (see section 2.1 for a couple of examples). What I consider particularly interesting, are the possibilities that emerge from the visualization method. Iris combines the visualization with gestures, creating tools that didn't exist in the traditional step sequencers. Displacing the rhythmic layers may help breaking a pattern in the users' traditional workflow, and help to explore new interesting results in a playful way.

3.2 Creation Process

The basic idea for creating a circular sequencer first emerged around a year ago, in the fall of 2009. It came as an after wave of a few other music instrument concepts I had thought of and sketched. I was seeking for a Master's Thesis project, but none of the previous ideas seemed to fit into a realistic time or resource frame. The new idea seemed intriguing, and of a proportion that could be brought to a tangible form mostly by stretching my own skills.

With a working title "The Wheel", the project launched as a concept for an interface for composing rhythms in a live performance setting: a step sequencer with a DJ turntable like visual approach controls – with benefits.

From Performing, Towards Experimenting

In the beginning, I approached the concept from an electronic dance music producer's viewpoint. This shows in some of the features in the first presented concept (see figure 7):

- An ability to scratch the wheel similarly to a record on a DJ turntable – utilizing a realtime audio loop buffer, recorded from the audio output of the sound system.
- A shuffle or swing function, similar to the ones found in popular sequencers and groove boxes (see the chapter 2.1, page 15).
- Various multitouch gestures for manipulating the wheel content as fast as possible

As I was already somewhat proficient with the Adobe Flash environment, an early prototype (operated with a mouse) was developed quickly. The first presentation of the concept, including an early demo, was held at Media Lab Helsinki in December 2009, and a bit further taken prototype was demoed for the Media Lab Sound Seminar attendees in January 2010 (shown in figure 8).

Both the interface and the sequencer were built with Flash: Users could modify steps on the rings, and turn the rings to different offsets. An open source graphics library called Degrafa was used in the production. I developed the sequencer using the open

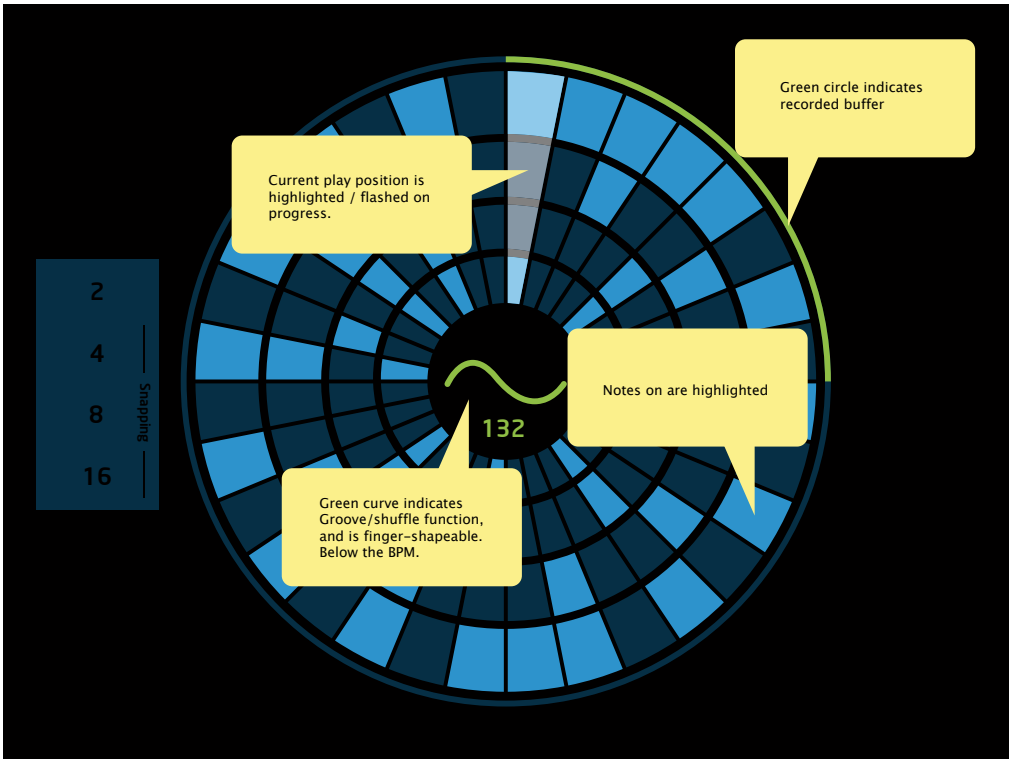

Figure 7: An early presentation mockup of The Wheel, a predecessor of Iris, 2009.

source *StandingWave* Actionscript library (used, for example, in the *Noteflight* online music notation service). The sounds used in the prototype were StandingWave's demo sounds, that sounded mostly like a minimalist sine wave "piano". Only some of the planned features were implemented at this time: creating ring sets of variable resolution, toggling the steps on and off, an ability to offset the rings and a tempo control.

After the demos, one of my two instructors, Antti Ikonen, suggested that I should examine selected 60s and 70s minimalist compositions, including ones by Steve Reich. Research and listenings of minimalist works, along with selected readings on African music and polyrhythms, gave me a new perspective to my core concept, and also helped me to frame the thesis subject more clearly. There had been multiple design questions existing in the concept, and I had to make a choice on which of them to address. The solution now seems self-evident: before jumping into creating an optimized end-product with novel features for a specific use, I should first study how the new approach of ma-

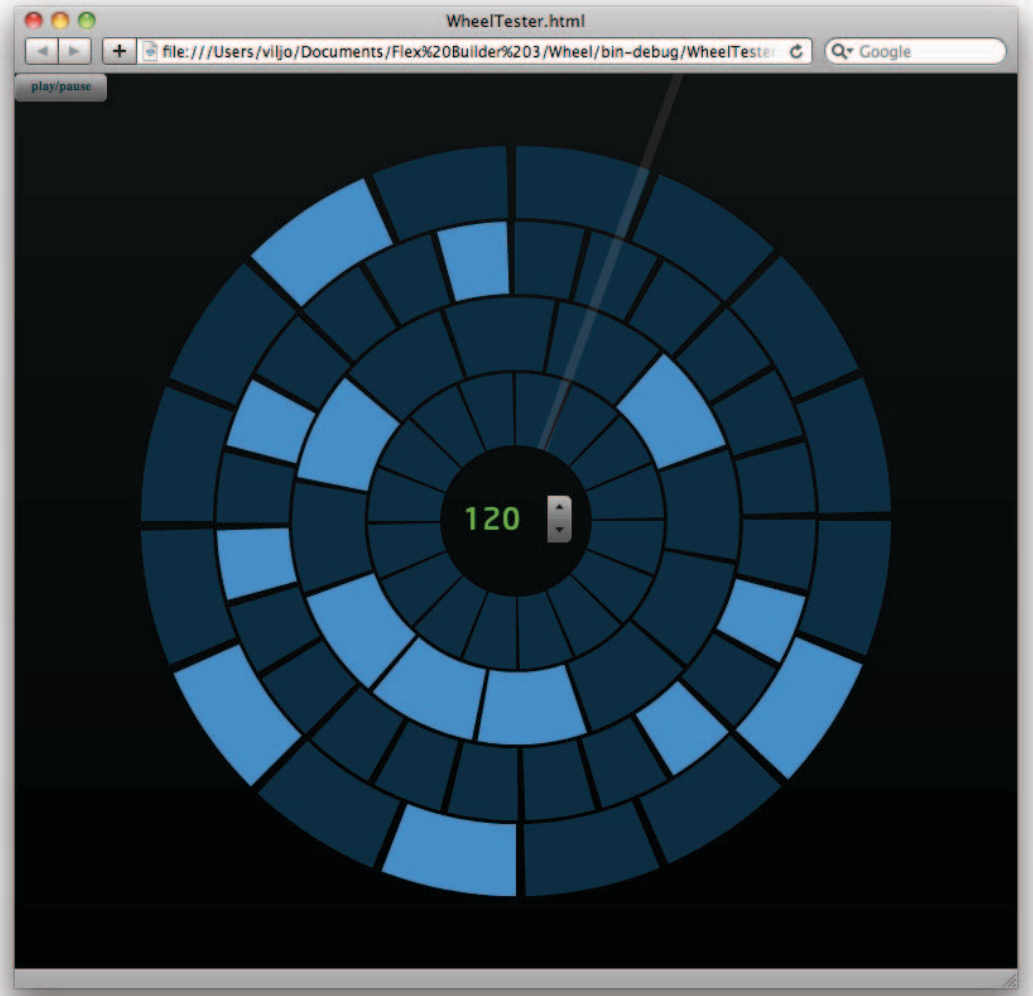

Figure 8: A screenshot of The Wheel flash prototype, 2009.

nipulating rhythms with the rotating concentric circles paradigm is experienced in general.

Redefining Goals

When I shifted my focus from developing a production-ready tool to creating an experimental interface for experimenting with rhythms, I still wanted to have a definition of a target audience to develop for, as well as a set of qualities to proof my coming prototype against.

The target audience for Iris are electronic musicians in general, interested in finding new tools for experimenting with rhythm. The tool can be used both as a part of a composing process, as well as in real-time performance use.

As electronic musicians are often accustomed to a certain kind of setup, the choice of tools used will further narrow the target audience. If modular enough, the end product will appeal to a larger audience, when it can be used together with various soft- and hardware combinations.

Based on the previous research on the example genres, polyrhythms and metric dissonance are apparent in various musical styles, and the sequencer could potentially be utilized to create rhythms across musical genres.

After framing the production's target audience, I defined a set of design goals for my prototype:

An inspirational Tool with New Features

The design should allow the user to create and edit rhythm patterns more complex than with a basic step sequencer. It should be inspiring, fun to use and encourage to musical experimentation. The concept does not aim to directly replace an existing solution, but rather offer an alternative approach to composing rhythms.

Easy and Fast to Use

The natural mapping of toggle buttons to the sequencer's patterns on/off states makes a step sequencer very straight-forward and intuitive to use, and I should try to cherish this apparent simplicity in the design. Basic tasks should be completed as easily as possible, allowing the user to focus on making music. I would like to give the user a feeling that while operating Iris, he/she is rather playing an instrument than programming a pattern. In all, the interface should be easy to approach, and have a gentle learning curve, allowing the new user to quickly step aboard.

Designed for The Task

The design should take account varying use environments, ranging from live venues to a studio. Potential challenges, such as inconsistent lighting or otherwise noisy environments, should be addressed.

An Integrable Tool

The implementation should be modular and easily integrated into an existing electronic musician's toolkit. In addition to providing connectivity with the other tools, Iris should preferably run on hardware and software platforms already existing in (or typical to) the musician's setup. The design's form factor should also allow maximum portability and easy placement.

Moving to Modular

The second prototype, developed during spring 2010, consisted of a Flash-based interface and an ad hoc sequencer unit programmed in Max/MSP. Splitting the design in modules served specific purposes:

My prototype Flash sequencer wasn't quite precise enough to play the polyrhythms involving fine temporal adjustments, due to its use of the Actionscript Timer class. While time-precise Flash music applications do exist (see Andre Michelle's and Joa Ebert's Popforge library or the AudioTool for examples), fixing this issue would have required a lot of extra work. Also, using an internal sound engine in the prototype was limiting the experimenting, as I couldn't integrate it into my existing setup. To allow testing the interface in different environments and use cases, I wanted to search for a modular solution.

As a result, I created a custom step sequencer. It connected to a Flash client running on the same computer, using a network socket connection. Developing with Max allowed to output generic midi data from the sequencer, and thus connecting it to most available soft- and hardware music tools. The development process required additional learning, since I had very limited prior hands-on experience with the environment. To my benefit, I had previously created small pieces of software using Quartz Composer, which shares its visual programming language paradigm with Max.

"Your iPad Has Arrived."

The original concept involved a touchscreen as a medium for the user interface, but I had not yet found a viable solution for it. At the time, I was looking into several alternatives: consumer touchscreen PC's or even custom developed FTIR solutions. Quick experimenting with the all-in-one computers however revealed their touch response was pretty slowly. Again, a custom projector-camera-installation would have required a lot more work and testing than I was prepared to at the time.

In April 2010, Apple released the iPad device. With new enthusiasm, I returned to the drawing board an started thinking of the new device as a promising platform for the Iris concept. On a luckily-timed family trip to the US in May 2010, I managed to buy a unit for testing.

A few things supported selecting the iPad as a prototyping and development platform:

- It was reported to have a very responsive touchscreen (Albert, 2010), with multitouch capabilities.
- The small form factor is ideal for an external, portable controller.
- Wireless networking is available out of the box.
- Apple has developed rich libraries with a lot of tools for manipulating graphics and audio.

Cons can be pointed out, too:

- The prototype is, to some extent, locked within the Apple's development environment, and is not directly portable to other environments.
- Objective-C, the primary programming language used for development on iOS (formerly known as iPhone OS) wasn't easy for me to approach, partly due to its Smalltalk-like syntax. In contrast to this, the XCode IDE used for development helped to ease the steep learning curve.

A lot of electronic musicians had previously found the iPhone as an instrument or controller in their setups, and a great amount of applications had already been published for the platform – Hexler's TouchOSC to name one. It was expected that the developers and musicians would also quickly jump onboard the iPad train. In this light, learning to develop on the new platform would likely turn beneficial in future projects. So during the summer of 2010, I slowly grasped some basics of Objective-C and the development environment.

Turning to Max 4 Live

I had previously introduced myself to Ableton Live. Live is a software application for music composition, songwriting, recording, production, remixing and live performance (Ableton AG., 2010), and it has gained popularity particularly in the electronic dance music scene. Working with short phrases, or loops, is an integral part of Live's paradigm

of creating music.

In November 2009, Ableton released the Max 4 Live environment with Cycling 74, the creators of Max/MSP. Max 4 Live allows running Max patches inside Live, creating new plugin devices to the system.

As the iPad has become available, many electronic musicians have adopted it as a controller for their Ableton Live setups (Denver Ableton User Group, 2010) using software such as TouchAble, Griid (Scarth, 2010) or TouchOsc. In case Iris is developed beyond the scope of this thesis, Live would provide a potential environment to host it, allowing the end users to easily integrate it into their existing setups.

Running the existing sequencer on Max 4 Live required only minor modifications on the patch. Integration to the Live environment also offered some additional benefits: the sequencer was almost automatically integrated with the music software's transport. Live also offered new interface elements to build the user interface for the sequencer settings (See the description of the Iris architecture in section 3.4).

3.3 The User Interface

The interface is split into two parts: the *touchscreen interface* running on iPad, and the *sequencer unit* 's interface in the Ableton Live effect chain. I have tried to maintain a separation of interface functions between these two modules: the preferences and settings are, for most part, handled inside Live. This feels convenient, considering that Iris can serve as part of a larger ensemble built around the Live environment. After the setup, the manipulation of the rhythm layers is done through the touchscreen interface. I have tried to maintain a consistent look and feel across these two domains, while designing in respect to the possibilities and restrains of the mediums.

Inside Ableton Live

The sequencer unit's interface lives inside the Ableton Live track view. A 169 pixels tall container (seen in figure 9) at the bottom of the screen represents a daisy chain of different interconnected devices on a single track. When developing in Max/MSP, a patch can include a presentation view, revealing only the necessary elements to control the patch. Inslde Max 4 Live, this presentation is shown in the track view. As a Max 4 Live MIDI Effect, Iris is placed as the first item on a track, sending MIDI signals to the devices following it.

The Sequencer Unit

I have tried to provide additional cues for the user, by adding visual connections between the controls. This aims to help the user to understand the underlying event flow (seen in figure 10).

Figure 9: A typical Ableton Live workspace (version 8.1)

As a feature provided by the Ableton Live environment, all controls can be mapped to an external MIDI controls. This means, that any of the editable elements in the sequencer unit's interface, can also be controlled with another device, such as a MIDI keyboard.

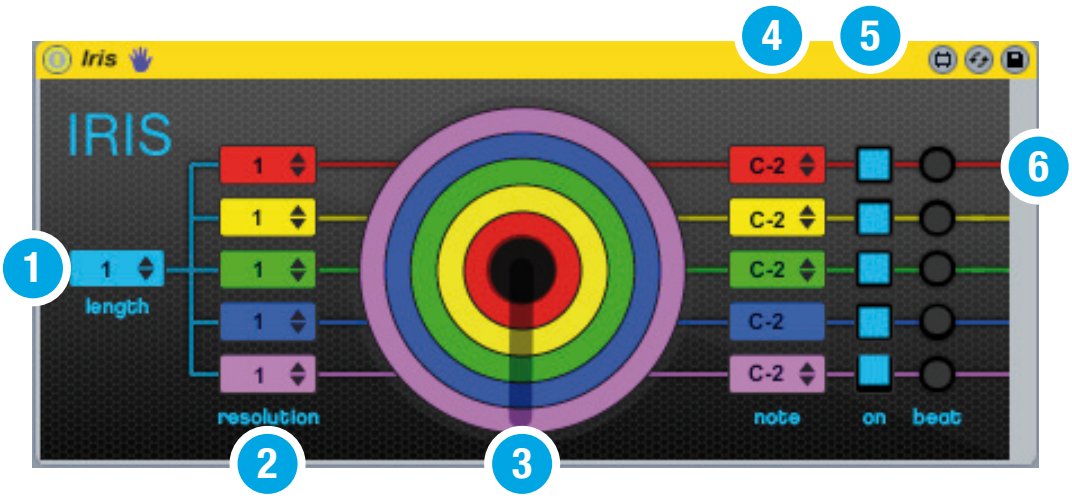

Figure 10: The Iris interface inside the Ableton Live track view

1. Loop length (in measures)

The user can define the loop length in measures, based on the tempo setting in Ableton Live's transport.

2. Pattern resolutions

The resolution of each circle can be can be set here, ranging from 1 to 64.

3. Loop position indicator

Indicates the position of currently playing loop. It's visualization, in part, tries to help associating the other controls to the colored rings. The indicator is actually implemented using a standard Max dial control in an inverse orientation, stripped of all other visual elements but the needle, and displayed on a custom background.

4. Note assignments

The note assignment for each ring can be defined with the correspondingly colored control.

5. Ring activation toggles

These controls act as channel on/off toggles. They control whether the MIDI messages of a specific ring are passed through.

6. Beat indicators

Played notes are indicated with a dedicated flashing led for each ring.

The Touchscreen Interface

Interface Elements

For describing the interface elements, I utilize a categorization of shape, size, color, orientation, texture and position, suggested by Cooper et al. (2007: 291):

Shape and Orientation

The interface is laid in a circular form: it consists of five concentric circles, divided into a number of segments acting as toggleable steps. In addition to creating a naturally mapped visual representation of a loop (discussed earlier in the Iris Concept section), a possible argument for the circular positioning of controls can be argumented using Fitts's Law, dating back to 1950s:

According to this law, "The time required to move to a target is a function of the target size, and distance to the target." (Lidwell, Holden, Butler, 2003: 82). Presuming that the user is likely to create the patterns a layer at the time, moving linearly towards the end of the pattern, the looping allows moving to edit the next layer seamlessly, without the need to move back and search for the start of the next pattern (this is a comparison to a typical step sequencer layout, with steps laid in a rectangular grid).

This argument would however require further study, since the size of the steps in rings vary in size, depending on the resolution and placement of the concentric ring in question: The circular layout causes the steps in the inner circles to have a smaller area than the ones on the outer ones. During the first demos of the concept in December 2010, an audience member asked whether the user should be able to freely organize the concentric rings to fight this problem. Such a feature is not yet implemented in the prototype.

Styling individual Steps

The steps are shaped as small arcs – segments of rings, divided with a small gutter. In the early prototypes, all steps were filled with color, the inactive steps left dim. In the final prototype, I changed the inactive steps to have a bright colored border with no fill. In viewing conditions with distracting reflections, having the step bounds presented in high contrast seems more legible (shown in figure 11).

Figure 11: The Iris Prototype running on the iPad, 2010.

Ring Colors

Each ring in the wheel is of distinct color. This design choice has to objectives: it helps to distinguish the rings as separate entities (layers of rhythm), and also helps the user to associate the rings with possibly different sounds. Cooper et al (2007: 293-294) note that visual properties such as color should be used for grouping interface elements together to create clear hierarchy.

When prototyping the device, I found colors to be helpful when trying to remember which ring was assigned a specific sound. As an examle, it seemed easier to associate the snare drum sound to the yellow ring, rather that the "second inner ring".

All selected ring colors are bright and saturated. The hues are selected almost evenly across the visible spectrum for maximum differentation. The number of rings, limited by the screen estate, narrows the color palette to five easily recognizable and primary and secondary hues (Feisner, 2000: 11). The colors are also selected to be easily named by anyone, and thus communicated: A 1969 study by Berlin and Kay, as referenced by Ware (2004: 112), reports that red, yellow, green, blue, brown and purple are the ones most likely to exist in different languages. Ware also points out that the basic colors are good candidates for coding data (Ware, 2004: 112).

I tested the color choices against the two most common forms of color blindness, protanopia and deuteranopia (Ware, 2004: 99-100). A simple test was done by proofing the ring colors with Adobe Photoshop's color proofing profiles. The rainbow-like scale presented potential problems: Adjacent rings seemed hard to distinguish, and combined unintentional color groupings, contrasting with the original idea of color labeling. Since I wanted to preserve the visual idea of the rainbow colored interface, I addressed the problem with small adjustments to hue and value of each ring's colors. This made the rings distinguishable in color, seen through the proofing profiles. Measuring the actual effects of the quick fix requires further study. Using other visual cues, such as textures, could also help to solve the issue.

The ring colors are not imperative in terms of the interface's general usability, but can improve the overall experience. I would like to further experiment if recognizing the layers quickly by color adds to the use efficiency.

Prototyping and Performance Considerations

Beyond usability and aesthetics, there are couple of more aspects to cover: When prototyping and developing novel interface concepts with no direct real-world analogy, it

makes sense to design them using simple geometric forms, minimal contours and a restricted color palette (Cooper et al., 2007:308). Seeking for the simplest possible solution communicating the message clearly, i.e. sustaining a low signal-to-noise ratio, is a good general guideline to all design (Lidwell, Holden, Butler, 2007: 182).

One might also assume that, when an interface is prototyped using a simple visualization, and later found to be effective, there's no need to redesign it. This could be one explanation behind the aesthetics of simplicity of some popular touchscreen controllers such as the Jazzmutant Lemur.

Simple visualizations are also fast to render. This is essential, when one wishes to implement very responsive interfaces: The processor load caused by screen redrawing should be minimized. The optimization pays off best in situations where processing power is limited – such as in mobile gadgets like the iPad.

Physical Orientation

With the Iris prototype, the iPad is intended to be used in the portrait mode, although the circular visualization works rather similarly viewed from any direction. Keeping the iPad in hand in the portrait mode puts less weight to the fingertips. Also, if placed on a stand, the tablet reservers less estate on the desktop.

How to Interact with The Iris UI

The design of the interactions and gestures try to meet the goals defined in the previous section 3.2:

- The interactions should feel natural, and the learning of the gestures should be easy.
- The interface should courage for experimentation. Iris is a composing tool for exploring new rhythms and creating ideas.

Modifying Patterns: Tapping and Painting

A single tap on a step turns it on or off. Basic editing is rather straightforward, and derived from a typical step sequencer interface (shown in figure 12).

A swipe with a single finger paints multiple steps on or off, depending on the swipe direction: a clockwise swipe paints on, a counterclockwise off. When designing this, there were at least two other possible modes of choice, derived from simple boolean logic: negating each step involved in the swipe, or deciding the result of the whole swipe

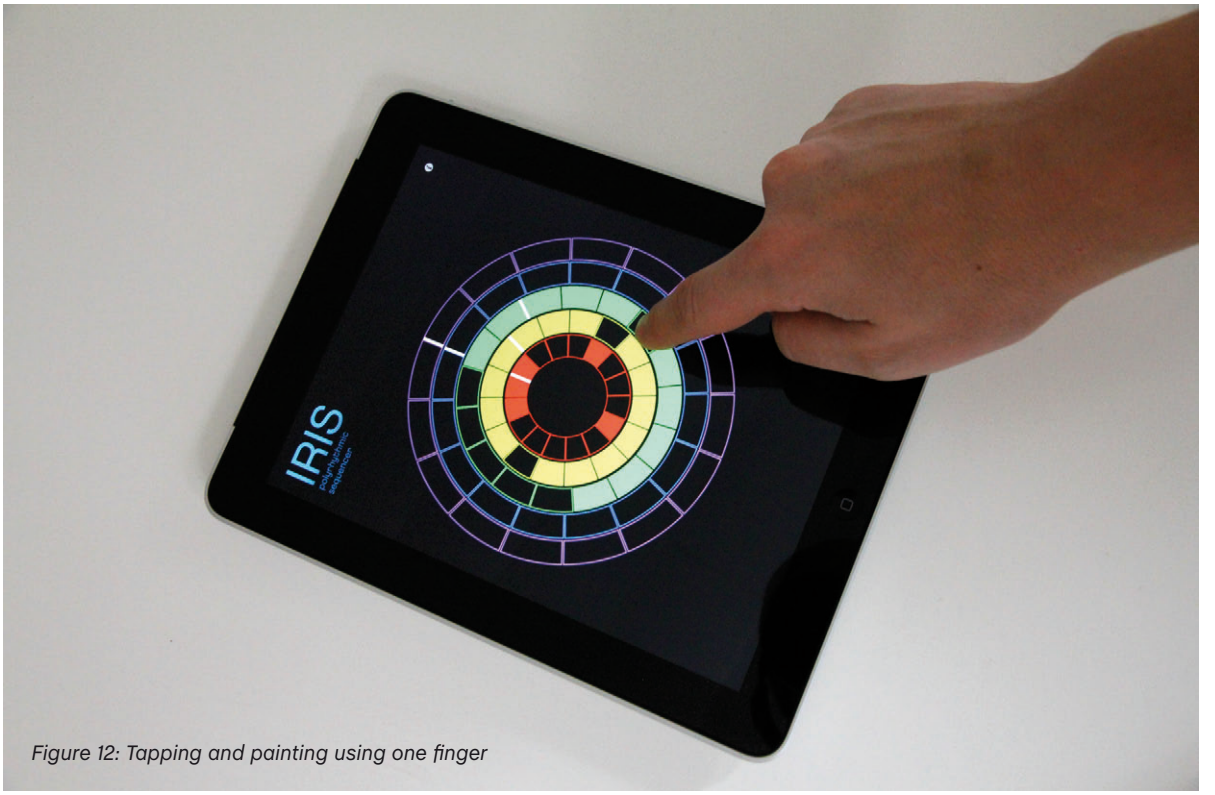

Figure 12: Tapping and painting using one finger

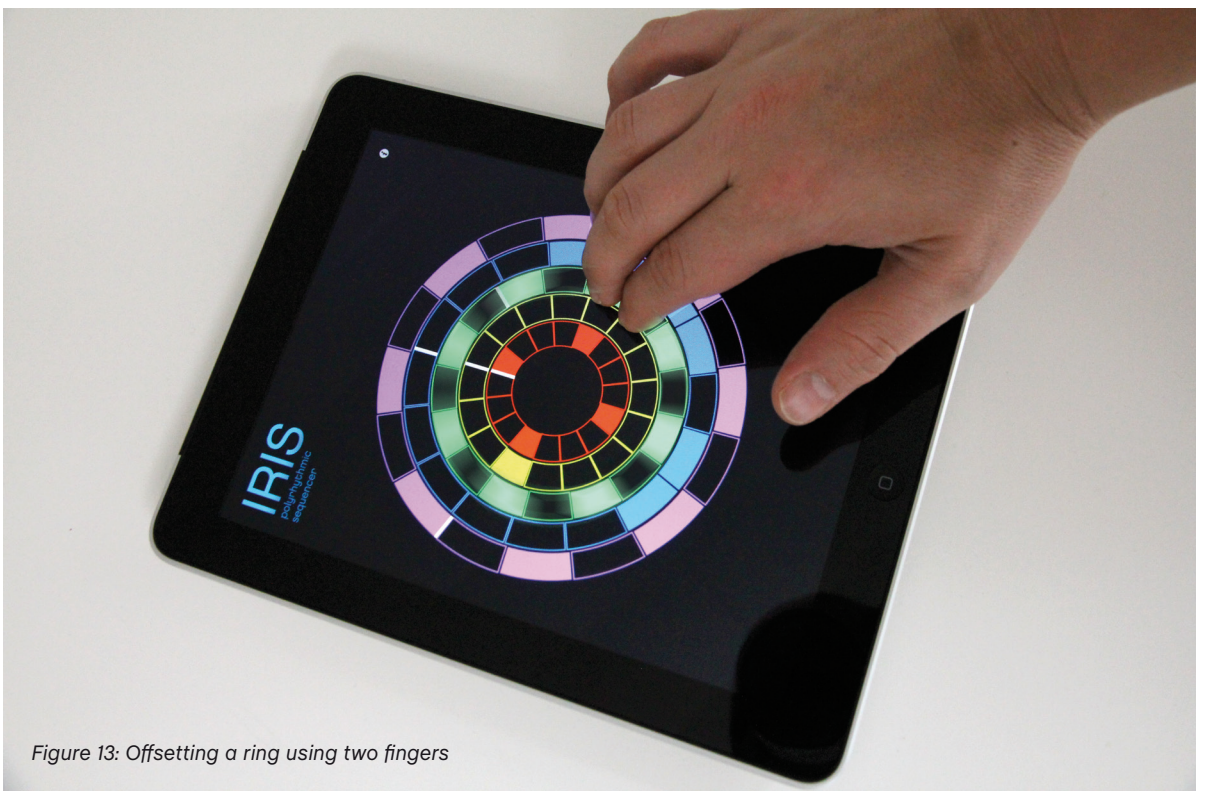

Figure 13: Offsetting a ring using two fingers

from the first step hit (if the first step is off, make sure that it, along with every step included in the swipe, gets turned on).

The choice I ended up with was for speed: Selecting the mode by swipe direction potentially allows rougher gestures, and in many occasions, doesn't require the user to hit an exact step when deleting steps.

When swiping, only one ring where the finger was on at the beginning of the swipe is affected. The finger can wander in- or outside the ring during the swipe gesture. This feature of "locking the edit to a specific ring" it was included after experimenting with the Flash-based prototype. In most use cases the feature makes good sense, presuming that the user is typically editing one layer of the rhythm (one ring) at a time.

Offsetting Rings

A ring can be turned to an arbitrary angle, using a two-finger swipe gesture (presented in figure 13). In effect, this means shifting the steps' temporal position inside the loop. This is the feature that expands Iris's application beyond a typical step sequencer – stepless moving the rhythm layers becomes possible. [1]

In many common touchscreen applications, the one finger swipe can be used to drag elements on the screen (try the Google Maps application on iOS or Android touchscreen devices as an example). The design decision to reserve the one finger swipe for painting steps was a deliberate one: while rotation of rings is of essential function in Iris's concept, I still consider the painting of steps still the primary editing tool. Two-finger scrolling is also used in many applications, including some touchscreen PCs (Saffer, 2009: 67).

---

[1] *The painting and offset gestures required to leave some space in the UI design, between the screen border and the outmost circle. The iPad's flat surface area is larger than the actual screen and touch area, and moving the swipe out of the screen area ends the gesture prematurely.*

Visual Feedback

The Iris touchscreen interface uses a visual "playhead" to inform the user of the loop progress. The playhead is visualized as a white line, drawn along the concentric circles' radius, and rotating around the center point.

Aural Feedback

Jokiniemi et al (2008, LNCS 5270) have studied cross-modal rhythm perception, and how people perceive rhythms through different senses (i.e. through hearing, vision and tactile senses). They report that when trying to recognize rhythms, unimodal auditory condition seems to provide the most accurate recognition across their tests, while visual modality seems to offer the least accurate cues for rhythmic perception.

As Iris is "just" a sequencer, it doesn't output audio as such. In theory, the output can be anything that can be controlled by MIDI messages, and possibly no audio is involved at all. While in many typical use cases, the aural feedback will be instant and synchronized with the visual feedback, this can't be taken as granted, as it depends on the users configuration.

Other Possible Feedback Channels

In the context of interfaces, the term 'haptics' stands for "science of touch" in real and virtual environments, including technologies that stimulate the user by forces, vibrations and/or motion (Robles-Da-La-Torre). Common haptics implementations include interfaces (e.g. a mobile phone with a touchscreen) that provide vibrotactile responses to the user, when interacting with interface elements.

Miranda & Wanderley (2006, p. 72) define two types of haptic music controllers: tactile simulators that provide skin sensation, and force feedback devices. Designers have included haptic feedback designs into electronic musical instruments for various reasons: to enhance expressivity and controllability, creating resemblance to the real world by haptic realism, or just for the attractivity of the interface. One of the first implementations of haptic feedback in electronic instruments is the artificial weighting of keyboard keys to resemble an actual acoustic piano (Pedrosa & MacLean, 2008).

A possible haptics implementation for Iris could work in a couple of ways: first, it could give haptic feedback when operating the interface. Second, it could provide an additional feedback channel to communicate the programmed rhythms to the user.

As a practical example of rhythmic feedback, Yamaha Clickstation metronome in-

cludes a small vibrating pad that provides the musician with a haptic que of the playing rhythm (Yamaha Corporation of America, 2010).

In controllers aimed at musical performance, such as Iris, haptic cues could aid the user in environments where noise caused by the environment disturbs the aural or visual feedback channels.

3.4 Implementation and Architecture

The implementation consists of two wirelessly interconnected parts: the sequencer and the touchscreen interface (see figure 14).

The Sequencer

The sequencer is a custom Max/MSP patch, running in the Max 4 Live environment inside Ableton Live. It can be divided into various more or less self-contained units (see picture 15):

Pattern Sequencers

The pattern sequencers are five separate subpatches, that receive the played pattern, the current loop position and it's own offset position from outside.

Each time when new position position data is received, the sequencer first compensates for the offset, and then checks if the new position value, ranging between 0 and 1, is in the area of a new element in the pattern list. In this case, a MIDI note is generated. The note length depends on the pattern length – I.e. the interval matches the ring resolution.

Metronome and Transport

The metro object in the Metronome and Transport module follows the Ableton Live transport. The current position is constantly sent to three targets:

- The pattern sequencers
- The circular visualization in the Live track view
- The touchscreen interface, via the OSC communication unit

The loop length control present in the interface, is also implemented in this unit.

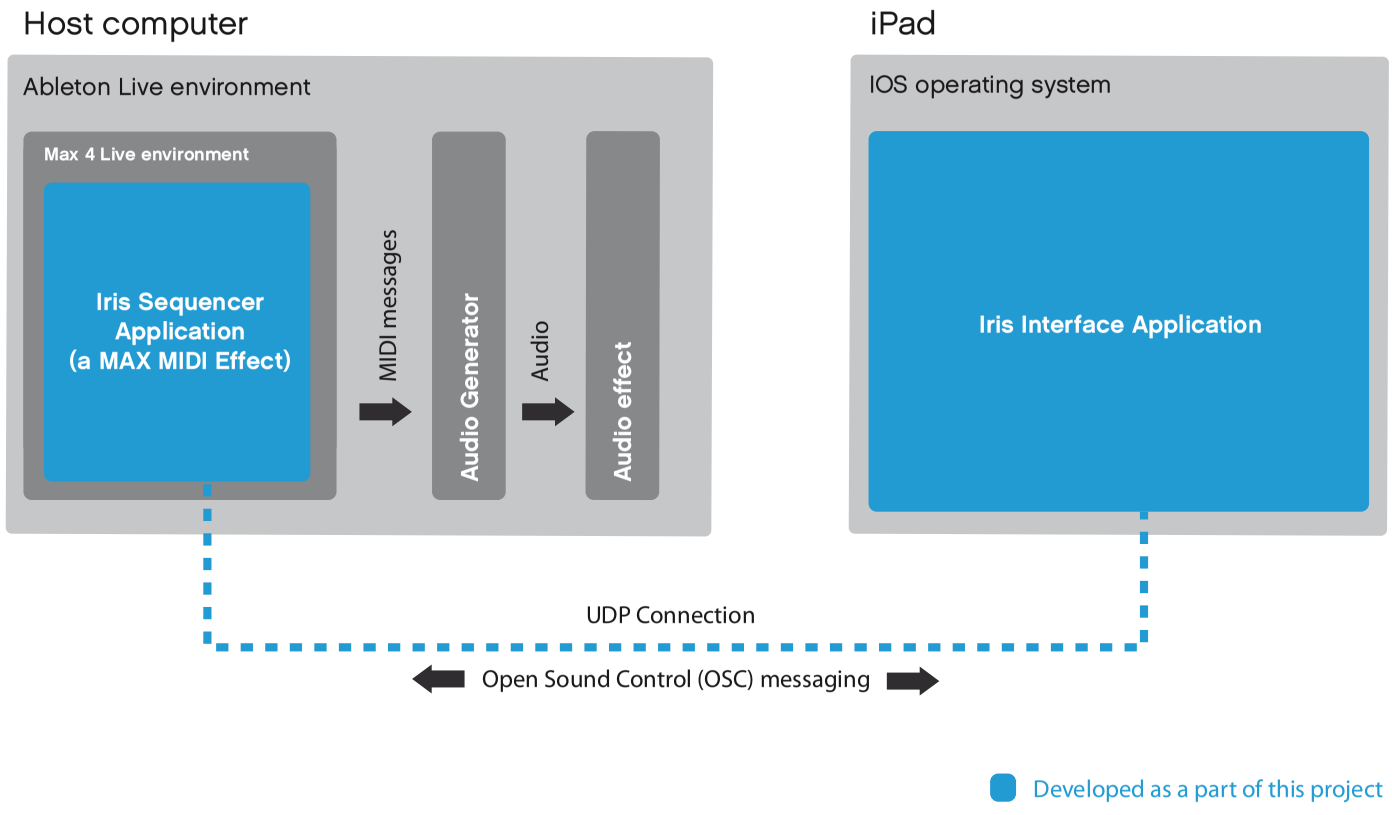

Figure 14: The Iris prototype architecture

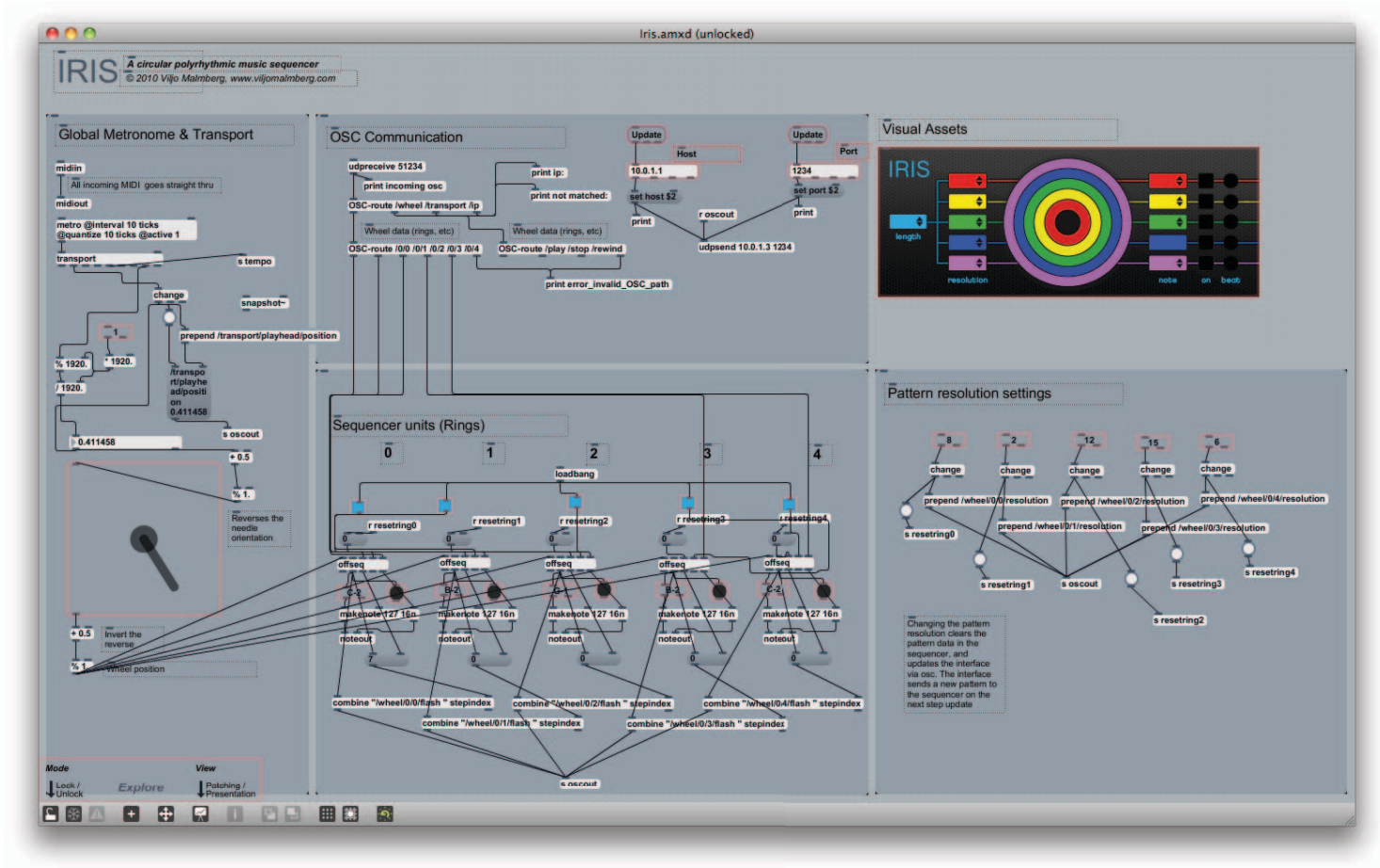

Figure 15: The Iris Max 4 Live patch (18 October 2010)

OSC Communication Unit

The OSC communication unit is responsible for sending and receiving messages in Open Sound Control format, through an UDP network connection. A receiver object picks up messages created by the sequencers and the transport.

Incoming messages from the network are routed to the responsible units, utilizing the OSC-route object from the CNMAT Max/MSP/Jitter package, available for download at the University of Berkeley's website: http://cnmat.berkeley.edu/patch/4029.

The Touchscreen Interface

The Iris interface application runs on the iPad, on top of the Apple iOS platform. It is developed using Objective-C and the XCode development environment.

When Iris is started by tapping the application icon (in figure 16), a splash screen welcomes the user (shown in figure 17). It stays on the screen for 3 seconds, after which the main interface is revealed.

Application Structure

The prototype implementation consists of 4 main classes:

Figure 16: The Iris application icon

The Main Controller Class (IrisViewController)

This class creates the network connections, and takes care of sending and receiving the OSC messages. It passes the commands received from the sequencer, to be displayed by the Main view. It also receivers the notifications sent from the rings, and messages them to the sequencer using OSC.

Main View

The Main view class is the visualization of the whole wheel consisting of concentric rings. All touch recognition is handled here: The Main view recognizes whether a finger has touched the display, and reacts to events such as moving one or two fingers on it. Manipulated rings and steps are seleceted based on the touches' angle and radius measured from the center of the wheel.

Figure 17: The Iris touchscreen interface splash screen

Ring Views

In the implementation, five instances of the Ring class are created, each taking care of drawing the ring's steps on screen.

Playhead View

The playhead is visualized as a simple line along the circles' radius, reaching from the inner border from the innermost circle, up to the outer border of the outmost circle. The Main view requests the playhead to update, based on messages received by IrisViewController.

Frameworks and Tools

The OSC communication is implemented using the open source VVOSC library. It can be found at Google Code: http://code.google.com/p/vvopensource/

Existing Issues in The Implementation

While the touchscreen interface implementation now works at a proof-of-concept level, some problems still exist at the time of writing: The network connection between the iPad and the computer running the sequencer is not particularly reliable. The iPad seems to drop the wireless connection every now and then, displaying the system's network selection dialog. Receiving messages from the sequencer on the iPad, seems to be more unreliable than receiving the iPad's messages on the sequencer. These issues unfortunately affect the overall use experience of the Iris prototype, and I intend to address them as soon as possible.

Developing the touchscreen interface on the iPad has been the most challenging single task in this project. I have been able to build on my existing programming experience, but learning a new language and a set of frameworks is still a long journey – I am in the early steps of it.

4. Feedback and Conclusions

4.1 Initial Feedback

In the final stages of development, I invited three selected people over to play and comment on the current "nightly" development versions of Iris. The tone of the sessions was informal, and they included free discussions on the subject, while the invitees got to experiment with the prototype.

All three subjects have diverse backgrounds in creating electronic music: one is a performing electronic dance music artist, one is a freelance hip hop producer, and one is a sound technology student from the Sibelius Academy, also working as a producer on freelance basis. They all share basic knowledge of Ableton Live as a production environment, and the play multiple instruments.

Some of the sessions included instant development along with the testing – new ideas of fixes for bugs were created on the go. The approach closely resembled a software development method called pair programming, where the person acting as the "driver" is operating the computer and modifying the program code, while the other person is observing, pointing out problems and suggesting ideas (Williams, Kessler, Cunningham, Jeffries, 2000: 20).

Suggestions and New Ideas

Concepts that were created or further crystallized during these sessions, and merged into the design presented in this thesis, include:

- Each ring now has a small starting point marker, indicating the original start point of the loop.
- Instead of just launching note on events, Iris now plays notes with off messages, the intervals corresponding to the ring's resolution.

A snapping feature was requested, to quickly make exact adjustments of the rings' positions, as well as a tool to reset the positions of the rings back to their starting point.

A feature which two of the test subjects brought to discussion, was the ability of creating simultaneously playing loops of different lengths. Automated movement of separate rings (closely resembling Reich's phasing method) were suggested as well. Related to this, simple physics implementation involving friction (one could push a ring to speed, and it would eventually slow down to stop), was proposed too.

During these sessions, we also tried mapping a MIDI keyboard controller to Iris's note selectors. The melody for the rhythm created in Iris could be manipulated in real-time when the sequence was playing. This kind of mapping is a built-in feature in Live, and in part strengthens my view of Live as a viable prototyping platform for Iris.

Creative routing of MIDI signals was also experimented with: The output from Iris could be picked up with several channels in Live, allowing easy connectivity to multiple instruments at the same time. Combined with filtering of the MIDI data, each of Iris's rings can be assigned to a different instrument chain inside Live.

Challenges During Demo Sessions

The foremost challenge during the sessions was the unreliable OSC networking. Operating the interface sometimes didn't affect the sequence, leaving the user in disappointment. The wireless network connection on the iPad dropped every now and then, and a part of the sent or received messages were lost. The playhead visualization on the iPad interface was also missing from the demo sessions, due to the poor messaging from the host to the touch interface. An bug, that was later fixed, caused a small sector of the Iris interface to become unresponsive to taps, while the swipe and rotation gestures remained working.

General Feedback

In these sessions, I received positive feedback on the concept in general. Iris was seen as a potential tool for experimentation and creating rhythms. One user noted that "it seems to offer a new way of thinking rhythms". Another user said that "the circular sequencing is just great fun. I am intrigued by the fact that the loop is visualized, as a loop".

The need for a "main beat" was emphasized: Especially when creating dance music related rhythms, two users were distracted when the dominant beat got displaced. The other of them preferred a working method where he left in place two layers containing the dominating pulse, while dragging the other layers around, creating moving textures. A question was raised, whether Iris should be thought as a complete ensemble, or a part of a setup with possibly other sequencers or rhythm creating elements.

A test user who performs electronic dance music live, noted, that the interface encourages him to create "African style" or "Fela Kuti"-like beats, and it helps him to do so quickly and intuitively. He also said that he could well think of using the interface in a live dance music setup.

Also interesting was, that the users seemed to map instruments to the circles in varying ways. One user started creating rhythms from the innermost circle, first placing a kick drum sound, while another user started from the outmost circle, placing a kick drum there.

Several output instruments were tested with Iris: different drum machines, analog modeling synthesizers, an organ-like sound generator and different simulated idiophone sounds. Playing with vibraphone- or marimba-like sounds lead one test user to make references to Steve Reich's Music for 18 Musicians, or Indonesian gamelan music. Combinations of Iris with an accompanying drum machine, and Iris with a analog bassline synthesizer were also experimented with.

4.2 Conclusions and Future Scenarios

Creating Iris was an educative process. It forced me to challenge my own views, and do research to better understand where my ideas originate from.

The project has allowed me to assess my working methods: It has been rewarding to challenge myself in many fields: visual design, concept design and technical design, in the specific context of electronic music. However, many pitfalls and extra work could have been avoided by tighter collaboration with other people. While I want to preserve the interdisciplinary approach, I'd like to shift my ways of working towards communicating and exchanging ideas more during the process.

By developing Iris, I have gained many new tools for realizing future concepts: I have strengthened my programming skills in Max/MSP, and obtained basic knowledge for working with the iOS environment. Above all, I have had to work intensively on an idea, and shape it into a form that can be clearly explained and communicated. Grasping the process of clarifying a concept, chopping off the unnecessary and establishing simplicity, is a core skill for a designer to learn and develop in.

Future of Iris

At the end of the thesis project, the Iris concept is taken to a tangible form – it can now be examined and experimented with. In a possible further study, I'd like to find out whether the new process of editing rhythms could somehow "enhance" the musicians' beat perception and teach them to recognize rhythmic patterns and the relations of the elements of composite rhythms in a new way. Currently, I like to use it as a starting point for creating rhythms, as it helps me to break my traditional patterns and finding new results.

Iris could possibly be utilized as a learning tool for musicians, helping to get familiar with complicated polyrhythms. Drummer Heikki Malmberg (no relation to the author) suggests a sequencer as a tool for this purpose (Malmberg, 2009).

This project started with an idea of a performer's tool, and soon morphed into a study on the circular interface concept itself. How Iris would work as a musical instrument for a performing electronic music artist, remains as an open question for further study. Some design decisions already taken, like the choice of used tools and the modular approach, could support this development.

Iris has also already spawned new ideas that could well advance as independent projects. One of the ideas include extracting the circular representation from the sequencing context, and use the circles for realtime audio and midi recording. The possibility of creating loops of different lengths, i.e. using multiple playheads to play loops, could also be a potential "spinoff" project. This idea came out repeatedly in the feedback sessions.

The Iris source code is released as a part of this thesis work. It is far from production quality, but reveals the architectural decisions, and is available to build upon. All downloadable materials can be found at *http://www.viljomalmberg.com/iris/thesis/*

Summary

For closing words, I'd like to borrow a quote from Master Drummer CK Ladzekpo:

> *"In the cultural understanding, the technique of polyrhythm simply asserts the highly unpredictable occurrences of obstacles in human life. They occur without a warning. It reinforces the need for the development of a strong and productive purpose built on a foundation of adequate preparation for life.*
>
> *…*
>
> *Blocking off a beat scheme to ease the hostility between opposing beat schemes of unfamiliar rhythmic contrast was often severely punished as my avoidance of the real challenges of life. A rare guidance in the proper management of opposing beat schemes of a rhythmic contrast was usually in form of a large dose of philosophy such as: to solve a problem, you must convert obstacles into stepping stones." (Ladzekpo, 1995)*

I find a relationship between this quote, and my thesis work and learning process as a designer: This project pushed me to face obstacles, many related to my own lack of knowledge and perspective. Tackling these obstacles has paid off as learning and a broader view. For me, alongside with the realized concept, the process is a major personal outcome of this work.

5. References

Books and Articles

Ableton AG, 2010. Live 8. [Online] Available at: <http://www.ableton.com/live-8> [Accessed 1 October 2010]

Albert, T., 2010. 10 biggest surprises (good and bad) about the iPad. [Online] Available at: <http://www.toddalbert.com/wp/10-biggest-surprises-good-and-bad-about-the-ipad/> [Accessed 17 October 2010]

Anku, w., 2000. Circles and Time: A Theory of Structural Organization of Rhythm in African Music. Music Theory Online, [online] Available at: <http://mto.societymusictheory.org/issues/mto.00.6.1/mto.00.6.1.anku_frames.html> [Accessed 20 September 2010].

Bernstein, J., 2007. Tangible Sequencer. [Online] Available at: <http://www.tangiblesequencer.com/> [Accessed 10 October 2010]

Butler, M., 2006. Unlocking the groove: rhythm, meter and musical design in electronic dance music. Indiana University Press.

Cambridge Dictionary Online, 2010. [online] Available at: < http://dictionary.cambridge.org/dictionary/british/sequencing> [Accessed 17 October 2010]

Casual Underground, 2010. Loopseque | See Music, touch music, be music. [Online] Available at: <http://www.loopseque.com/> [Accessed 10 October 2010]

Cooper, A. Reimann, R. And Cronin, D., 2007. About Face 3: The Essentials of Interaction Design. Whiley Publishing Inc.

Denver Ableton User Group, 2010. Season 2, episode 1 – New Meet Up. [Online] Available at: <http://abletondenver.com/season-2-episode-1-new-meet-up/> [Accessed 17 October 2010]

Emmerson, S., 2007. Living Electronic Music. Ashgate Publishing Limited.

d'Escrivan, J., 2007. Electronic music and the moving image. In: N. Collins and J. d'Escriván, eds. 2007. The Cambridge Companion to Electronic Music. Cambridge University Press.

Feisner, E., 2001. Colour: How to Use Color in Art and Design. Laurence King Publishing.

Gann, K., 2001. Minimal Music, Maximal Impact. New Music Box, [online] Available at: <http://newmusicbox.org/page.nmbx?id=31tp01> [Accessed 1 Oct 2010].

Grove Music Online, 2010. Africa. Oxford Music Online. [Online] Available at: <http://www.oxfordmusiconline.com:80/subscriber/article/grove/music/00268> [Accessed 22 September 2010]

Jokiniemi, M. Raisamo, R. Lylykangas, J. and Surakka, V., 2008. Crossmodal Rhythm Perception. In: A. Pirhonen and S. Brewster, eds. 2009. HAID 2008, LNCS 5270, pp. 111-119.

Keller, R., 2003. Mapping the Soundscape: Rhythm and Formal Structure in Electronic Dance Music. [Online] Available at: <http://etd.lib.fsu.edu/theses/available/etd-12112003-214228/> [Accessed 8 October 2010]

Kurasaki, K., 2006. MPC Groove Template Tutorial. [Online] Available at: <http://www.peff.com/journal/2006/01/28/mpc-groove-template-tutorial/> [Accessed 15 October 2010]

Ladzekpo, CK., 1995. Foundation Course in African Dance-Drumming. [Online] Available at: <http://home.comcast.net/~dzinyaladzekpo/Foundation.html>

Lee, J., 1994: Charles Babbage. [online] Available at: <http://ei.cs.vt.edu/~history/Babbage.html>

Lienhard, J., 1997. Engines of Our Ingenuity No. 1145: Jacquard and Babbage.

[online] Available at: <http://www.uh.edu/engines/epi1145.htm> [Accessed 15 September 2010]

Malmberg, H., 2009. Superimposed Subdivisions (Polyrhythm Hell). Online Drummer. [Online] Available at: <http://onlinedrummer.com/pdf.php?Id=135> [Accessed 3 October 2010]

Mundolo, S., 2010. African Music Culture. [Online] Available at: <http://www.africanmusicblog.com/tag/sub-saharan-african-music> [Accessed 1 October 2010]

Newton-Dunn, H. Nakano, H. Gibson, J., 2003. Block Jam: A Tangible Interface for Interactive Music. Journal of New Music Research, Volume 32, Issue 4 December 2003, pp. 383-393. [Online] Available at: <http://www.informaworld.com/smpp/content~db=all~content=a714018082> [Accessed 17 October 2010]

Nketia, J., 1960. African Music in Ghana. The Arts Council of Ghana.

Norman, D., 2002. The Design of Everyday Things. Basic Books.

Potter, K., 2000. Four Musical Minimalists. Cambridge University Press.

Reuge – the history of mechanical music and music boxes, 2010. [online] Available at: < http://www.reuge.com/static-content/about-reuge/the-history-of-mechanical-music-and-music-boxes.html> [Accessed 17 October 2010]

Reunanen, M., 2010. Computer Demos—What Makes Them Tick?, [online] Available at: <http://www.kameli.net/demoresearch2/reunanen-licthesis.pdf> [Accessed 2 August 2010]

Robles-De-La-Torre, G. What is Haptics? International Society of Haptics. [Online] Available at: <http://www.isfh.org/haptics.html#fr> [Accessed 2 October 2010]

Ross, A., 2006. An article on Steve Reich. [Online] Available at: <http://www.stevereich.com/> [Accessed 15 September 2010]

Russ, M., 2009: Sound Synthesis and Sampling, 3rd ed. Elsevier Ltd.

Saffer, D., 2008. Designing Gestural Interfaces: Touchscreens and Interactive Devices. O'Reilly Books.

Scarth, G., 2010. Griid vs touchAble: iPad Ableton Live controllers head to head. Musicradar.com. [Online] Available at: <http://www.musicradar.com/news/tech/griid-vs-touchable-ipad-ableton-live-controllers-head-to-head-275330>

Suda, B., 2010. A Practical Guide to Designing with Data. [e-book] Five Simple Steps. Available through: Five Simple Steps <http://www.fivesimplesteps.com> [Accessed 6 Oct 2010].

The Synth Museum, 2000. Linn: LM-1. [Online] Available at: <http://www.synthmuseum.com/linn/linlm101.html> [Accessed 1 September 2010]

Ware, C., 2004. Information Visualization: Perception for Design. Morgan Kaufmann Publishers.

Williams, L. Kessler, R.R. Cunningham, W. Jeffries, R., 2000. Strengthening the case for pair programming. IEEE Software, August 2000. [Online] Available at: <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=854064>

Winfield, A., 1998: Studio Technology – The MIDI Sequencer. [online] Available at: < <http://www.ias.uwe.ac.uk/~a-winfie/teach2003/st_mseq.htm> [Accessed 15 October 2010]

Yamaha Corporation of America, 2010. CLST-100 ClickStation. [Online] Available at: <http://www.yamaha.com/yamahavgn/CDA/ContentDetail/ModelSeriesDetail.html?CNTID=23870&CTID=551162> [Accessed 15 September 2010]

Photographs and illustrations

Figure 1: Drake, S. , 2009: MFOS 16-Step Rotary Analog Sequencer. [photograph] Available at: <http://commons.wikimedia.org/wiki/File:MFOS_16-Step_Rotary_Analog_Sequencer,_and_Fishpants.jpg>

Figure 2: Sony Computer Science Laboratories, 2003: Tangible blocks of Block Jam. [Photograph] Available at: <http://www.sonycsl.co.jp/IL/projects/blockjam/index.html>

Figure 3: Bernstein, J., 2007:The Tangible Sequencer. A Youtube video screenshot. [Photograph] Available at: <http://www.youtube.com/watch?v=Jm39v_MdnzM&feature=related>

Figure 4: Malmberg, V., 2010: The Polymachine sequencer screenshot [Screenshot].

Figure 5: Casual Underground, 2010: The Loopseque. [Screenshot] Available at: <http://www.loopseque.com>

Figure 6: Malmberg, V., 2010: Looping a grid. [Illustration]

Figure 7: Malmberg, V., 2010: An early presentation mockup of The Wheel, a predecessor of Iris, 2009. [Illustration]

Figure 8: Malmberg, V., 2010: A screenshot of The Wheel flash prototype, 2009 [Screenshot].

Figure 9: Malmberg, V., 2010: A typical Ableton Live workspace (version 8.1). [Screenshot]

Figure 10: Malmberg, V., 2010: The Iris interface inside the Ableton Live track view. [Screenshot]

Figure 11: Malmberg, V., 2010: The Iris Prototype running on the iPad, 2010. [Photograph]

Figure 12: Malmberg, V., 2010: Tapping and painting using one finger. [Photograph]

Figure 13: Malmberg, V., 2010: Offsetting a ring using two fingers. [Photograph]

Figure 14: Malmberg, V., 2010: The Iris prototype architecture. [Illustration]

Figure 15: Malmberg, V., 2010: The Iris Max 4 Live patch (18 October 2010) [Screenshot]

Figure 16: Malmberg, V., 2010: The Iris application icon. [Illustration]

Figure 17: Malmberg, V., 2010: The Iris touchscreen interface splash screen [Photograph]