The EyeHarp: A Gaze-Controlled Musical Instrument

Author: Zacharias Vamvakousis

MASTER THESIS UPF / 2011

Master in Sound and Music Computing

Master Thesis Supervisor: Rafael Ramirez
Departament of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona

UNIVERSITAT POMPEU FABRA

Abstract

In the thesis, we present the EyeHarp, a digital musical instrument based on eye tracking. The EyeHarp consists of a self-built low-cost eye-tracking device which communicates with an intuitive musical interface. The system allows the performer to produce music in real time by controlling sound settings and musical events only through eye movements.

The instrument can also be controlled using any input device that can take control of the mouse pointer, and thus it could be appropriate for many cases of people with disabilities. Following the EyeWriter' s initiative instructions, a €50 eye tracking device was constructed and used along with the provided open source code for getting the user's gaze coordinates on the screen. Having that as an input, the EyeHarp digital musical instrument was implemented in openFrameworks C++ toolkit. The user can play music in real time controlling the chords and melody of his composition through an intuitive interface based on the pie menus. On top of it, an arpeggiator machine and a step sequencer provide the opportunity of creating a harmonic and rhythmic background. Depending on the desired task, dwell time or screen button gaze selection technique is applied. A preliminary discount usability experiment is conducted having the eye gaze versus the head movements as input.

Acknowledgements

Thanks to my family for supporting me this year.
Many thanks to all my colleges from this master, for the amazing moments we shared during this year! I want to thank separately my college Emilio Molina for bringing me in contact with organizations of people with disabilities and my supervisor Dr. Rafael Ramirez for his support.

Zacharias Vamvakousis
22/6/2011

“ΜΗ ΕΝ ΠΟΛΛΟΙΣ ΟΛΙΓΑ ΛΕΓΕ, ΑΛΛ′ ΕΝ ΟΛΙΓΟΙΣ ΠΟΛΛΑ”
ΠΥΘΑΓΟΡΑΣ

“Do not say a little in many words, but a great deal in a few”
Pythagoras

Contents

Part I

Introduction

Chapter 1

Motivations

Most people would agree that expressing yourself through music is a relieving and liberating experience. Some people due to physical disability are not able to play any existing musical instrument. One case could be people with complete paralysis resulting from Amyotropic Lateral Sclerosis that can only move their eyes. Nowadays these people are able communicate using new technologies. Such examples are eye tracking and head tracking systems that provide the user with the ability to write, talk, access the internet, play computer games, draw e.t.c. Nevertheless there is no available musical instrument explicitly designed for having an eye tracking or head tracking device as the only input.

The main goal of this master thesis is to come up with the first gaze controlled musical instrument. It could also be used along with any device that can take control of the mouse pointer of a computer, such as a head tracking device. Moreover it should have similar expressive potentials to a traditional musical instrument. Having control over the melody, the chords, the expressiveness of the produced sound in real time was the first priority of the designed interface. That way such an instrument could potentially be a part of a normal music band. Interacting with other people through music or creating new musical compositions not only could improve the quality of life of people with disabilities, but also be part of a rehabilitation process.

Chapter 2

Goals

As already mentioned, the main goal was to create a gaze controlled musical instrument with similar expressive potentials to a traditional musical instrument. What is most crucial when playing a musical instrument is playing the right notes at the right moment. So, we should come up with an interface that will:

- Maximize temporal control: The performer should be able to control in real time the rhythmic, harmonic and melodic aspects of his/her composition. The gaze selection technique for playing music in real time should provide minimum response time, so the user can play melodies in-tempo.
- Minimize error rate: The buttons on the screen for playing a note should be big enough in order to reduce the possibility of playing neighbour notes, due to errors of the eye tracking system. As the accuracy of the existing eye trackers varies, the size and number of buttons on the screen should be adjustable.
- Provide real time control over more expressive aspects of the sound, like amplitude and vibrato.

Chapter 3

Thesis Outline

In the next part we present the State of the Art. The main contribution of this thesis is to come up with appropriate gaze selection techniques for playing real time melodies. In chapter 4 we survey these techniques. In chapter 5 we survey the most representative gaze controlled applications (most commonly used by people with disabilities), while in chapter 6 we survey the previous research on gaze-controlled music.

In part III we present the EyeHarp musical interface. In chapter 7 we describe the methodology applied while implementing the EyeHarp. In section 8.1 the EyePiano -our first approach- is described, while in section 8.2 the EyeHarp interface is presented. More specifically, we describe the controls that were designed in the EyeHarp GUI for gaze input (subsections 8.2.1, 8.2.2), the Step Sequencer layer (subsection 8.2.3), the EyeHarp layer for playing real time melodies (subsection 8.2.4) and the arpeggios generator (subsection 8.2.5).

In part IV we describe the preliminary evaluation process that was conducted and we report on the results.

Finally in part V we discuss on the conclusions and future work.

Part II

State of the Art

Chapter 4

Gaze Selection Techniques

When designing a gaze controlled interface we need to take under consideration the special characteristics of natural eye movements and design interaction techniques for them around the known characteristics of eye movements. The gaze interaction techniques that are used for selecting discrete targets, such as buttons or menu items are defined as gaze selection techniques [33].

In this section we assume that the user can use only his / her as an input for control. Alternatively other "clicking" techniques can be applied, such as clicking using the hands, the tongue, the lips, EEGs etc. Because the same modality, gaze, is used for both perception and control, a gaze-based communication system should be able to distinguish casual viewing from the desire to produce intentional commands. This way, the system can avoid the "Midas touch" problem (Jacob, 1991), wherein all objects viewed are unintentionally selected.

The following gaze selection techniques are more extensively summarized in [33] and [24].

4.1 Blinking

With blink-based selection, the user looks at a target object and blinks to select it. This method has not proven to be very popular. Deliberate blinks are judged as being unnatural (Jacob 1990)[18]. Humans blink involuntary every few seconds which would result in unintentional selections. One option which was proposed, and rejected, by Lankford (2000)[21], was to use a prolonged blink to differentiate between deliberate and involuntary blinks. Ji and Zhu (2002)[19] developed a system which uses three deliberate blinks to indicate a selection, but this is unnatural.

Especially for the case of playing melodies in real time, blinking is not an option because in order to distinguish whether we have an deliberate blinking or not, an extra delay time is introduced.

4.2 Dwell Time

This is the most popular selection method used in gaze controlled applications. A selection occurs when an object is fixated upon for a period of time exceeding a specified time-threshold. If the dwell time threshold is set too short, the user may unintentionally activate commands, while if it is too long extra response time is added.

Dwell time is used in the EyeHarp project for all the controls, apart from anything related to playing real time melodies where the Screen Button method is used.

4.3 Special Selection Area or Screen Button

Another solution to the Midas touch problem is to use a special selection area (Yamada & Fukuda, 1987[38]) or an on-screen button (Ware & Mikaelian, 1987[36]; Ohno, 1998[28]). For example, in the "quick glance" method developed by Ohno (1998), each object that could be selected was divided into two areas: command name and selection area. Selection was done by first fixating briefly on the command (to determine the name or type of the command), then confirming that a selection was required, via a brief glance at the selection area. Alternatively, a user who is experienced and knows the locations of the commands need only glance directly at the selection area associated with that command. With that method, better temporal control is expected. Not necessarily in a sense that we can "press more buttons" per second, but we can "press" each button at the desired moment. So we can better play music in-tempo. Placing each command name and selection area in slices around "pies" results to the Pie menus selection technique that was used in the EyeHarp for performing real time tasks.

4.4 Pie Menus / pEYEs

In computer interface design, a pie menu (also known as a radial menu) is a circular context menu where selection depends on direction. A pie menu is made of several "pie slices" around an inactive center [wikipedia].

Anke Huckauf et al., in 2008 introduced pEYEs [15], where pie menus were suggested for gaze control. pEYEwrite was implemented for writing with gaze input while pEYEtop was a desktop navigation interface. In figures 4.1, 4.2 we can see the implemented interfaces. In both cases a slice opens without dwell times by looking in the outer selection area of a slice. A slice can lead to another pie menu. After comparing these implementations with common gaze controlled techniques they state that: "pEYEs can be effectively and efficiently used in tasks requiring lots of selections like in typing. Furthermore, also tasks affording more orientation and navigation like a desktop can be effectively completed using pEYEs as the basic concept."

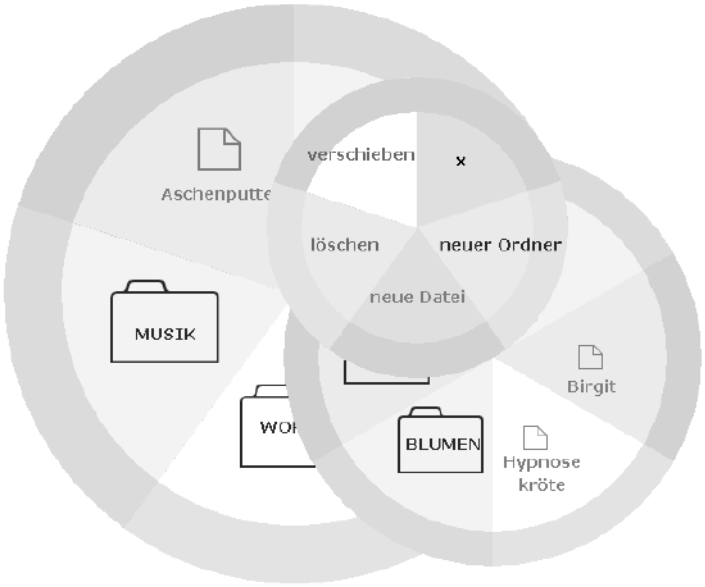

Figure 4.1: Two levels of pEYEtop with opened command pie.

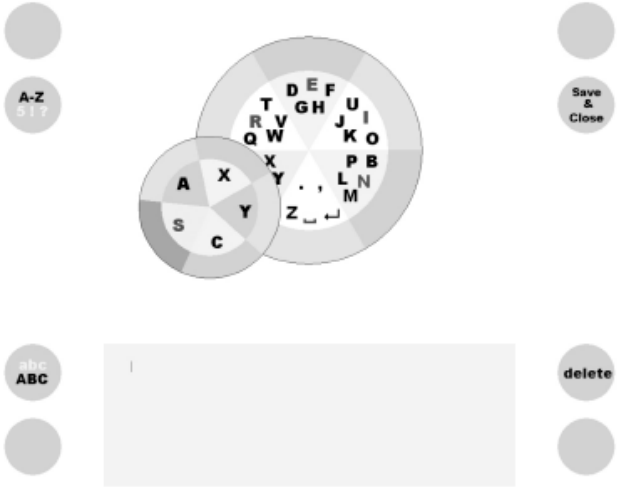

Figure 4.2: Screen layout in pEYEwrite. The central pEYE serves for typing, and on-screen buttons provide other functionalities. The text is displayed in the the lower part of the screen.

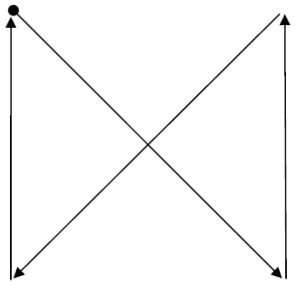

Figure 4.3: An example of a gaze gesture, starting from the top-left corner, with a gazing order of SE, N, SW, N.

4.5 Gaze gestures

In that method the user initiates a command by making a sequence of "eye strokes" in a certain order. Making a gaze gesture still requires a brief stop (fixation) between the strokes (saccades). An example of a gaze gesture can be seen in figure 4.3.

Chapter 5

Gaze-Controlled Applications

The applications related to eye tracking fall into to basic categories: diagnostic and interactive. In figure 5.1 we can see the hierarchy of eye-tracking applications as suggested by Andrew Duchowski [8].

"In its diagnostic role, the eye tracker provides objective and quantitative evidence of the user's visual and (overt) attentional processes.",
"...An interactive system must respond to or interact with the user based on the observed eye movements. Such interactive systems may fall into two application subtypes: selective and gaze-contingent. Selective systems use the point of gaze as analogous to a pointing device such as the mouse, whereas gaze-contingent systems exploit knowledge of the user's gaze to facilitate the rapid rendering of complex displays" [8].

All the gaze controlled applications fall into the interactive-selective category, and are used mostly by disabled people, providing them with the opportunity to communicate(speak, draw, access the internet, send emails, or even play computer games and interact in virtual reality environments). The EyeHarp falls into the same category as well.

In the following sections we present some representative gaze controlled applications.

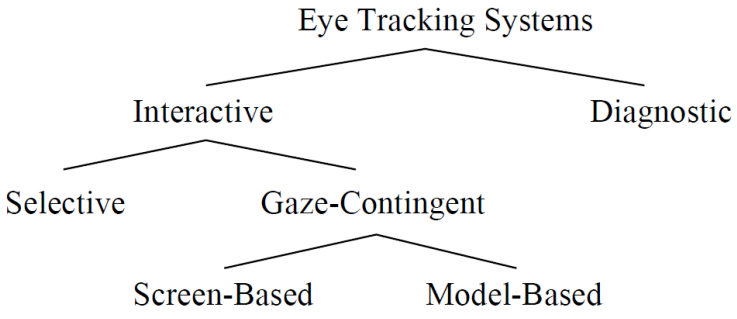

Figure 5.1: Hierarchy of Eye-Tracking Applications

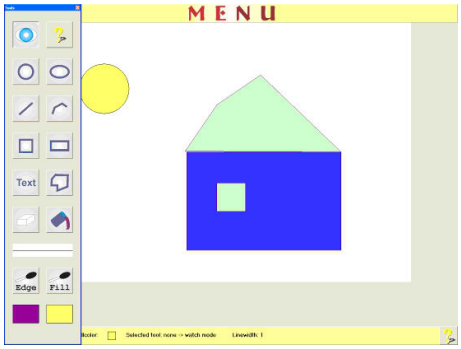

Figure 5.2: EyeArt: a drawing program that can be controlled by eye movements. EyeArt includes tools for drawing basic shapes, tools for changing color and line width, on-screen keyboard, as well as an inbuilt help and options dialog that can all be controlled by gaze alone.

5.1 Drawing with eye movements

The most straightforward way of drawing with eye movements is "Free-eye drawing" proposed by Tchalenko, J. (2001): a thick line with varying colors follows the user's eye movements. With that method it is impossible to draw fine shapes, like curves. The gaze can only move by saccadic movements that can only define straight lines.

The solution to that problem is to provide the user with specific tools for drawing basic shapes (line, circle, square, etc.), tools for changing color and line width, while dwell time is commonly applied to face the Midas touch problem (see figure 5.2).

Such applications are the EyeWriter [37], EyeDraw [12] and EyeArt [25].

5.2 Text entry with eye movements

Direct gaze pointing The most common way to use gaze for text entry is direct pointing by looking at the desired letter (figure 5.3). Dwell time is then most commonly used method for selecting each letter. Keys and controls can be organized hierarchically in menus and sub-menus to save space, and special techniques such as automatic word prediction can be used to speed up the text entry process.

Eye Switches Some people may have difficulties in fixating because of their physical condition or state of health. They cannot keep their gaze still for the time needed to focus. Voluntary eye blinks or winks can be used as binary switches. For text entry, blinks are usually combined with a scanning technique, with letters organized into a matrix. The system moves the focus automatically by scanning the alphabet matrix line by line. The highlighted line is selected by an eye blink. Then the individual letters on the selected line are scanned through and again the user blinks when the desired letter is highlighted.

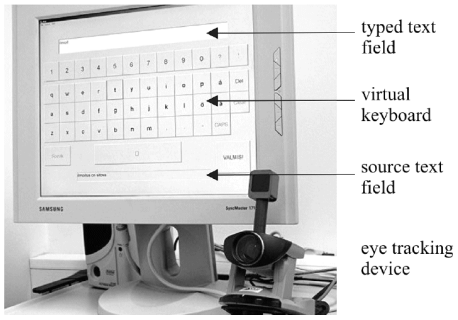

Figure 5.3: Qwerty on-screen keyboard with dwell time

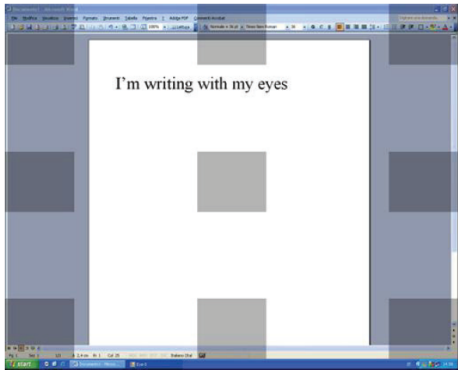

Figure 5.4: Eye-S with hot spots shown

Discrete Gaze Gestures Another interesting approach is that of gaze gesture for eye-typing. Porta and Turina (2008)[30] developed Eye-S to take advantage of gaze gestures for both text entry and control of computer applications. Nine on-screen areas act as hot spots for the start and end points of the eye strokes. The user enters a character or command by glancing at the hot spots in the order specified for the desired gesture (see figure 5.4). A similar system is EyeWrite by Wobbrock et al. (2008). [37]

As already mentioned pie menus can provide a text input method without any dwell time as well. In pEYEwrite (see figure 4.2) only two saccade eye movements are required to type a letter.

Continuous Gaze Gestures Finally another very unique and approach is eye-typing by continuous pointing gestures. David J. Ward in his PHD thesis (2001)[35] presented Dasher (see figure 5.5) :

> "Dasher is a text-entry interface driven by continuous two-dimensional gestures, delivered, for example, via a mouse, touch screen, or eyetracker; the user writes by steering through a continuously expanding two-dimensional world containing alternative continuations of the text, arranged alphabetically. Dasher uses a language model to predict which letters might come next and makes those letters easier to

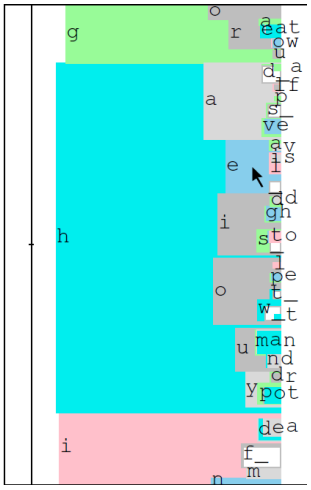

Figure 5.5: Screenshot of Dasher when the user begins writing hello. Here, the user has zoomed in on the portion of the shelf containing messages beginning with g, h, and i. Following the letter h, the language model makes the letters a, e, i, o, u, and y easier to write by giving them more space.

> write. The language model can be trained on example documents in almost any language, and adapts to the user's language as she writes."[23]

A more extensive survey of eye-typing techniques can be found in the PHD thesis of Päivi Majaranta with title "Text Entry by Eye Gaze" (2009). [24].

5.3 Other dedicated eye-controlled applications and commercial eye control systems

In addition to eye typing and eye drawing, there are several dedicated eye-controlled applications, such as e-mail, Internet browsing ([6] ,[26]), accessing online libraries ([22]), games ([7], [16], [31]), and interaction with online virtual communities ([3], [34]). An extensive survey of gaze controlled games can be found in [17]. Some of the applications, such as games and Internet browsing, are included in many of the commercial eye control systems targeted at people with disabilities. An up-to-date list of eye tracking systems used as assistive devices, can be found online at:
http://www.cogain.org/eyetrackers/ (accessed on 22 September 2011)

Chapter 6

Previous Research in Gaze-Controlled Music

6.1 Leon Theremin gaze control over the timbre of the Theremin

As Albert Glinsky write in his book: *Theremin: ether music and espionage*[10]:

> "...mere shifts in the performer' s glance could trigger changes in its timbre. At a distance of about six to ten feet in front of the performer, lenses arranged across a strip were trained on one of the player' s eyes. Behind each lens, a concealed photoelectric cell was attached to its own tone generator. As the player' s eyeball rotated, staring in turn at different lenses, the corresponding photocell behind each one recognized the gaze of the pupil and switched on the associated timbre. Glancing from lens to lends produced a variety of contrasting tone color, while the hands were free to regulate volume and pitch. The inventor even devised a system to compensate for the swaying of the head: the monitored region of the eyeball was defined by x and y coordinates that met at the corner of the tear duct, following its motions as the head shifted."

It is impressive to consider that Leon Theremin contacted all that research working with pure electronics, while he also devises a system to compensate for the swaying of the head.

6.2 Andrea Polli, 1997

The first system using eye tracking devices to produce music in real time was proposed by Andrea Polli in 1997 [29]. Polli developed a system which allowed performers to access a grid of nine words spoken by a single human voice by making saccadic movements to nine different directions. After trying different artistic implementations Polli concluded that improvising with the eye-tracking instrument could produce the same feeling for the performer as improvisation with a traditional instrument [29].

In 2001 she performed "Intuitive Ocusonics", a system for sound performance using eye tracking instruments to be performed live. Instruments were played using distinct eye movements. Polli's compositions responded to video images of the eye, not specifically the pupil center which are parsed and processed twelve times per second using the STEIM's BigEye software (www.steim.org). With this technology it was impossible to calibrate the pupils position to the computer screen coordinates, thus the user does not have precise control of the system.

6.3 EyeMusic

Hornof et al [13] proposed a system based on a commercial eye tracking system, the LC Technologies Eyegaze System, which provides accurate gazepoint data using the standard pupil-center corneal-reflection technique. In the system, the coordinates of the user' s gaze are sent to MAX/MSP for generating sound. They study both the case of using fixation detection algorithms for choosing an object and the raw data form the eye tracker. When trying to implement an eye-piano they report that the musicians that tried the system preferred to work with the raw data instead of a dispersion-based fixation-detection for playing the notes. A velocity-based fixation-detection algorithm is proposed instead. After trying that interface on a user for almost two weeks, one hour per day of practice, they reported that: "he was slightly able to improve his ability to move to the intended piano keys and (b) he was not at all able to improve his rhythmic accuracy", so they abandoned the idea of the eye-piano. Instead, they consider designing more interactive tools using Storyboarding. The performer moves an eye-controlled cursor around on the screen, and makes the cursor come into direct visual contact with other visual objects on the screen, producing a visual and sonic reaction. The user interacts with objects that appear on the screen, through a series of interaction sequences (like a scenario).

Hornof and Vessey in a recent technical report evaluate four different methods for converting real-time eye movement data into control signals (two fixation based and two saccade-based methods). They conduct an experiment comparing the musicians' ability to use each method to trigger sounds at precise times, and examined how quickly musicians are able to move their eyes to produce correctly-timed, evenly-paced rhythms. The results indicate that fixation based eye-control algorithms provide better timing control than saccade based algorithms, and that people have a fundamental performance limitation for tapping out eye-controlled rhythms that lies somewhere between two and four beats per second [14]. Hornof claims in [13] that velocity-based (as opposed to dispersion-based) fixation-detection algorithms work better for rhythmic control with the eyes. Fixation-detection algorithms typically employ a minimum fixation duration of 100 ms which would impose an upper bound of ten eye-taps per second.

6.4 Oculog

Kim et al. [20] present a low cost eye-tracking system with innovative characteristics, called Oculog. For selecting objects, blink detection is implemented. The data from the eye tracking device are mapped to PureData for generating and interacting with four sequences. In their

system the performer's field of vision is divided into four discrete quadrants. The direction of eye movement detected by the Oculog camera software is encoded as a combination of horizontal position (pitch) and vertical position (velocity): pitch 0 is produced by looking to the extreme left, note number 127 to the extreme right; velocity 0 is produced by looking down, velocity 127 by looking up. Assigned to each quadrant is a real-time tone generator. Each tone generator is driven by a cyclic sequence. Oculog also detects torsional movement of the eye, but this is not mapped to any control feature. The authors claim that eye tracking systems are appropriate for micro-tonal tuning.

6.5 Previous Installation designed for people with disabilities

Adam Boulanger, in his Phd thesis (2010) [5], "...demonstrates the design and implementation of devices that structure music interaction from the neural basis of rehabilitation. At the conclusion of this research, it is possible to envision an area where users are empowered during scientifically based creative tasks to compose neurological change." In his approach, the Hyperscore [9] Composition Program is used along with appropriate implemented control devices designed for each of the users -including eye trackers-. Hyperscore is a very high level composition tool.

Another case where eye tracking was used as the only an input by a disabled person to control music was led by the University of East London's SMARTlab team in 2009, where "MyTobii commercial eye tracker is being used in a world premiere of a live, eye-controlled music and dance session."[1]. In this case James Brosnan "having cerebral palsy, being a wheelchair user and being unable to speak" ... "controls the computer, called My Tobii, using his eyes, and jams using a software programme called Grid 2, designed by UK company, Sensory Software. A sensor follows his eye-movement, via which he picks out pre-recorded sequences of music." "The Grid 2 allows people with limited or unclear speech to use a computer as a voice output communication aid, using symbols or text to build sentences." [1]. Not much information were found about this certain installation.

Finally, a first version of the EyeHarp gaze controlled musical instrument was presented at the Sound and Music Computing conference in Padova on July, 2011.[32]

[1] http://www.tobii.com/en/assistive-technology/north-america/news-events/us-press-releases-tobii-ati/music-at-the-blink-of-an-eye/ (accessed on 1/9/2011)

Part III

The EyeHarp

Chapter 7

Methodology

One of the reasons why there are not many applications explicitly designed for eye-trackers is that eye trackers used to be extremely expensive. If someone wants to design an eye tracking application, he should first know how it feels to use an eye tracking as a control device. And the best way to find out about the difficulties of controlling a computer with your eyes is to experience it.

Lately there is a lot of research on low-cost eye tracking devices and software. One of these is the eyeWriter initiative. I decided that from all the alternatives I had, the EyeWriter fulfilled most of my expectations: a low cost (50 €) do-it-yourself eye tracking device, with quite good accuracy and an open source software for getting the screen coordinates of the user' s gaze. So, after a week of trying to get all the parts I need, I built my own eye tracker. Then I could start testing my implemented interfaces.

The methodology I had decided to apply was repeating the following steps every day:

1. Brainstorming, new ideas about the interface. Draw them on paper and try to imagine if they would be handy.
2. Implement these ideas in Openframeworks.
3. Use the constructed eye tracking device to play some music and evaluate the usability of these ideas.
4. Decide if I should go on this idea, or not. If yes, then how could they be improved?

The main drawback of the built eye-tracking device is that it did not allow head movements. It takes some time and effort for someone to get accustomed to the device and learn how to calibrate it properly. People tend to move their heads and the calibration is lost. The ideal case would be to regularly perform usability evaluation tests on already experienced users on gaze control, using a very accurate commercial eye tracking device. Neither of these was feasible. For that reason I decided that I should be the only subject of the every-day usability evaluation of the interface. The advantage of that choice was that I was finally able to come up with a intuitive interface in a short time. Of course in the future the interface has to be evaluated on more users. Moreover, according to Saul Greenberg and Bill Buxton (2008):

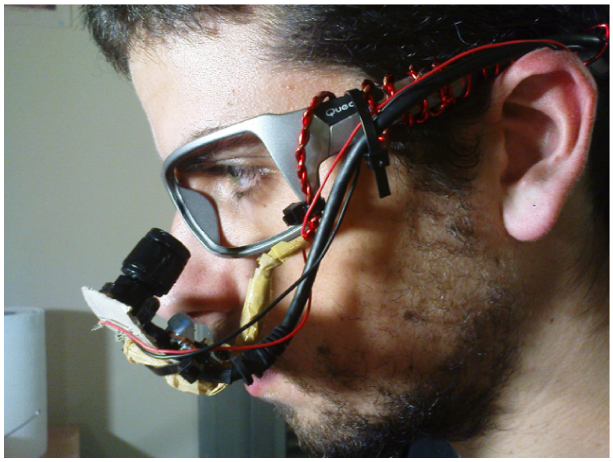

Figure 7.1: The PlayStation Eye digital camera is modified so as to be sensitive to infra-red light and mounted along with two infra-red leds on a pair of sun-glasses.

> "If done during early stage design, usability evaluation can mute creative ideas that do not conform to current interface norms."[11]

In order to read the input from the eye tracking device, we have used the libraries developed in the EyeWriter project. The eye-tracking software detects and tracks the position of a pupil from an incoming camera or video image, and uses a calibration sequence to map the tracked eye/pupil coordinates to positions on a computer screen or projection. The pupil tracking relies upon a clear and dark image of the pupil. The eye tracking device includes near-infrared leds to illuminate the eye and create a dark pupil effect. This makes the pupil much more distinguishable and, thus, easier to track. The software dealing with the camera settings allows the image to be adjusted with brightness and contrast to get an optimal image of the eye. When initializing the system, calibration takes place displaying a sequence of points on the screen and recording the position of the pupil at each point. The user focuses on a sequence of points displayed in the screen presented one by one. When the sequence is finished, the collected data are used to interpolate to intermediate eye positions.

Figure 7.1 shows the eye tracking device used in this work.

Chapter 8

Implementation

8.1 First Approach: The EyePiano

This was not the first attempt to try the EyePiano interface. As already mentioned, Hornof et al. [13] first tried the Eye-Piano interface. Firstly applying a very short dwell time in order to play the notes, and then working with the raw data of the eye tracker. In figure 8.1 we can see Hornof' s interface.

In figure 8.2 we can our version of the EyePiano interface. From the very beginning of trying an eye tracking device, it was obvious that it is easier for the visual system to focus on points or corners (where to lines cross). These points called *focus points*, and are common in gaze-controlled applications. This is the reason why there are red dots in the EyePiano interface. (figure 8.2) Instead of abandoning the idea of the eyePiano from the beginning, I tried to solve the problems mentioned by Hornof et al. The basic improvement was to add a region in the middle where the user can rest his eye (screen button technique). A region that nothing happens if the user' s gaze is there. Then I added one more octave. Here is how:

As we can see in figure 8.2 there are two piano keyboards: one in the top of the screen, and a second one in the bottom. The one in the top is mapped to the lower octave, while the one in the bottom is mapped to the higher octave. The two floor piano keyboard is actually divided into three regions:

- The top region for triggering notes in the lower octave. This region is starting at the top of the screen and is ending where the black keys end.
- The bottom region for triggering notes in the second octave. This region is starting where its black keys start, and is ending at the bottom of the screen.
- The neutral region in the middle. In this region nothing happens. It is just for moving the eye in the same horizontal position with the desired note, and then just move up or down in order to trigger it.

This neutral region improves the rhythmic control, as we can move our gaze close to the desired note, and then we can play in tempo and the right note as well. The reason for that has is that

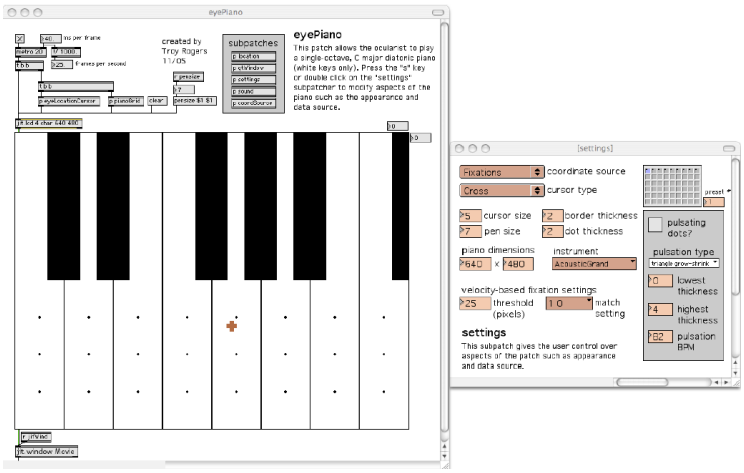

Figure 8.1: Hornof' s et al. EyePiano interface

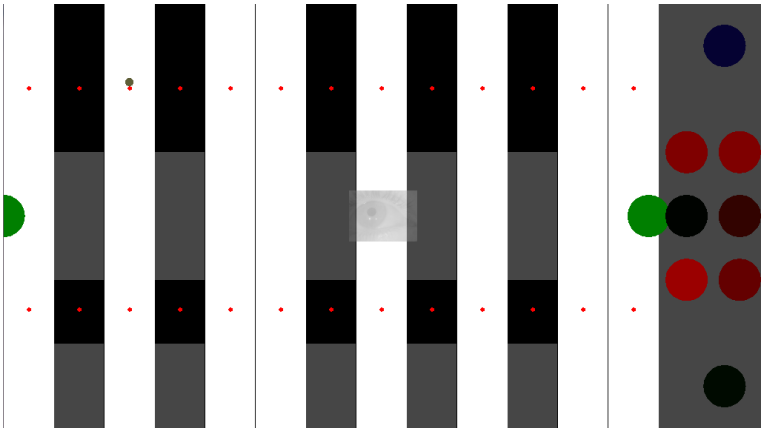

Figure 8.2: First Attempt: The EyePiano interface

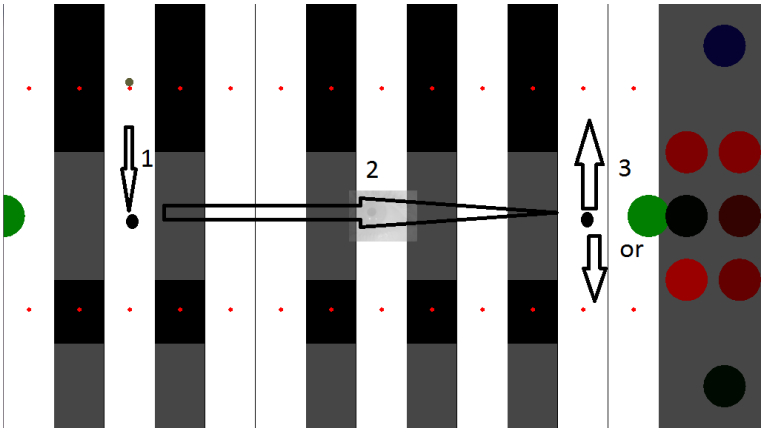

Figure 8.3: The EyePiano: We have to follow the arrows in order to take advantage of the neutral region added.

it takes less time to have a fixation in close region, than those that are far away. Long saccade eye movements are not so fast and accurate as close ones[8]. That way the spacial accuracy of the performer is improved, compared to Hornof' s approach. It is easier to play in tempo as well, since the screen button method was used instead of dwell time. By looking at the two green circles at the left and right sides of the keyboards we move one octave up or down. On the right of the two floor piano various controls are placed, like volume and other characteristics of the sound.

After playing music with that interface for a couple of days, I figured out that its the main drawback was that in order to make use of the neutral region, two additional fixations are required. This reduces maximum number of notes that we can play per second. Figure 8.3 explains why. This limitation is addressed in the next and final interface, the EyeHarp interface.

8.2 The EyeHarp

The EyeHarp[1] is a new musical instrument explicitly designed for eye gaze control, but it can be controlled with any technology that takes control of the mouse pointer.

The sound is generated with additive synthesis inside openframeworks. The EyeHarp consists of three basic parts:

- The EyeHarp layer for playing melodies and changing the chords in real time.
- The Step Sequencer layer for creating a rhythmic and harmonic background.
- A "harmonizer" engine that according to some input arguments generate up to four arpeggios. This is a fast way of having an interesting harmonic and rhythmic background. While in the step sequencer we choose the notes one by one, the harmonizer provides a higher level mapping in order to control the desired musical outcome.

[1]To better understand the functionality of the EyeHarp musical instrument visit the blog: theeyeharp.blogspot.com/ , to watch a performance using the EyeHarp along with explanatory annotations.

All these parts have different timbre and sound properties. Combined together they provide a real time composition tool.

In the following sections we will first describe the controls designed explicitly for gaze input in the EyeHarp interface, and then the basic parts of the EyeHarp mentioned above.

8.2.1 The GUI: Controls Designed Explicitly for Gaze Input

In figures 8.5, on the left and right side of the screen, there are the controls of the EyeHarp. Special controls had to be explicitly designed for gaze input. They were all designed having one principle in mind:

> For each control, two focus points are provided: one for deciding which function to perform, and another one for performing that action.

This is the screen button gaze selection technique described in 4.3. In most of the cases of the eyeHarp interface, when temporal control is not crucial this technique is combined with dwell time. So even if by mistake the user looks at a control button for a moment nothing will change unless dwell time is reached. That way we give the user the freedom to look around the screen while thinking of what he/she should do next. This is a natural behaviour in everything we do in our lives. For example, if someone wants to catch an pen, he will first look at it and then catch it. In the eyeHarp interface we are trying to provide this focus point, where actually no control is triggered. Here are the types of controls designed:

Slider

"A slider, is an object in a GUI with which a user may set a value by moving an indicator, usually in a horizontal fashion." (wikipedia).
In figure 8.5 in the left and right side there are two orange sliders. The difference between these sliders and common slider is that the user can set the value without clicking. He/she first looks at the desired value (number). Nothing is triggered in that region. Then by looking at the line with the circle (the line with the focus points), the value is set. A very short dwell time is applied on these focus points in order to avoid setting values accidentally.

Repeat Buttons

Alternatively the value can be set using the *repeat buttons* placed next to the slider. The repeat buttons are commonly used in many applications along with sliders to increase / decrease by one step. A Dwell time is applied in that case when looking on the increase / decrease button.

The increase / decrease buttons are connected with each other with a black line. When we increase the corresponding value, the increase button gets brighter, while the decrease button gets darker. In the middle of their distance, the set value is displayed. That way the user can check the set value, and if he / she decides to increase it, he / she looks at the increase button (up) or the decrease button (down). The slider control interface is more appropriate for

setting values that are far away with each other, while the repeat buttons are appropriate for fine tuning.

Tabs

Analogous to the tabs in windows control panel, or in almost any web browser, there is a similar control in the EyeHarp interface. The purpose of this control is to map the same control buttons (like sliders) to different control variables, or to exclusively choose between some distinct choices (similar to the radio button). For example in figure 8.4 the user can assign the slider on the right just to one of the variables on the tab control. Once more dwell time is applied for making a selection.

Switches

In order to activate / deactivate a function switch buttons are used. Their color is dark when they are not activated, and they light up when they are activated. Dwell time is applied to activate / deactivate the switches. All the notes in the Step Sequencer described in section 8.2.3 are *switches*. When looking at a switch its focus point placed in the middle of it, turns green, indicating that if a the user keeps on looking on it, it will turn on. If the user has just turned on / off a switch, then in order to change again its value, he/she should move outside the switch' s region. The reason for that is that if someone wanted for example to turn on a switch, and kept on looking at it for double the dwell time, then the value would remain unchanged in the end. That was proven to be quite confusing and so it was fixed.

8.2.2 The global controls

The right part of the EyeHarp interface is dedicated to the global controls of the instrument. In figure 8.4 we can see these global controls. There are two "tab" controls:

- On the bottom right corner of the screen, there is a horizontal tab control, where the user can switch between the Step Sequencer and the EyeHarp layer.
- A vertical control for assigning the slider control in the right to the following global variables:
  - Transpose: choosing the tonality of the instrument. The value 0 corresponds to A ('la'), 1 to A#, -1 to G# and so on. Consequently it varies from -5 to +6.
  - Tempo: Setting the tempo of the step Sequencer (bpm to move to the next step). Minimum value: 60. Maximum value: 1200.
  - Master Volume: The master volume of the instrument.

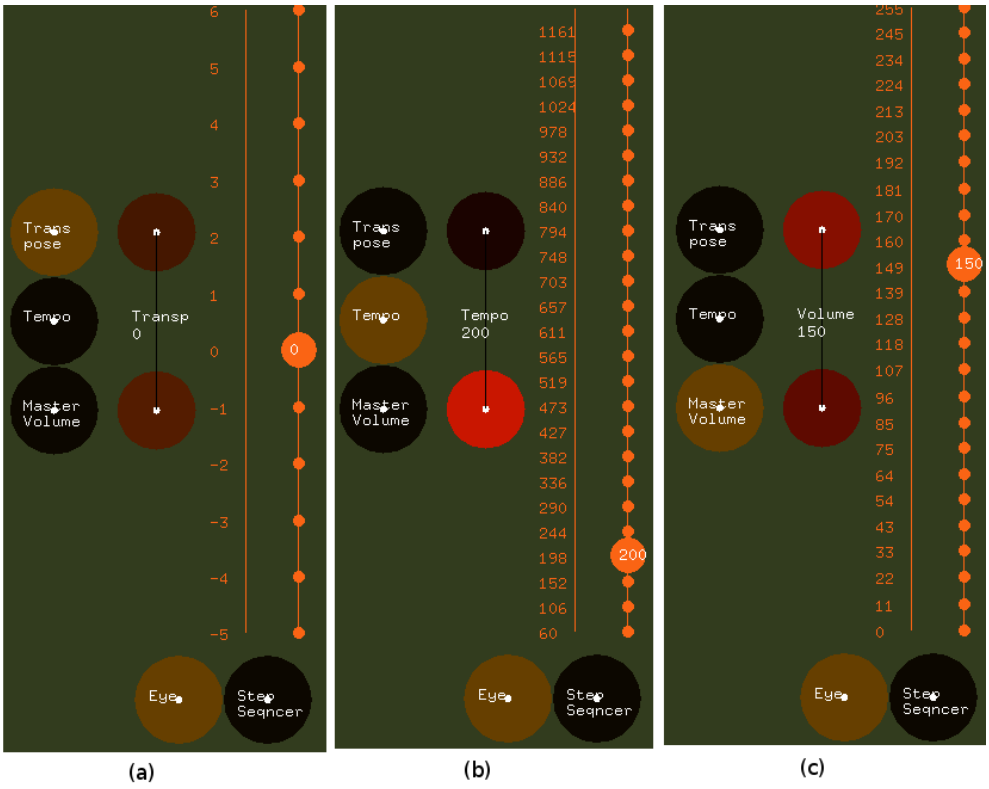

Figure 8.4: The global Controls: (a)Transpose (b)Tempo (c)Master Volume

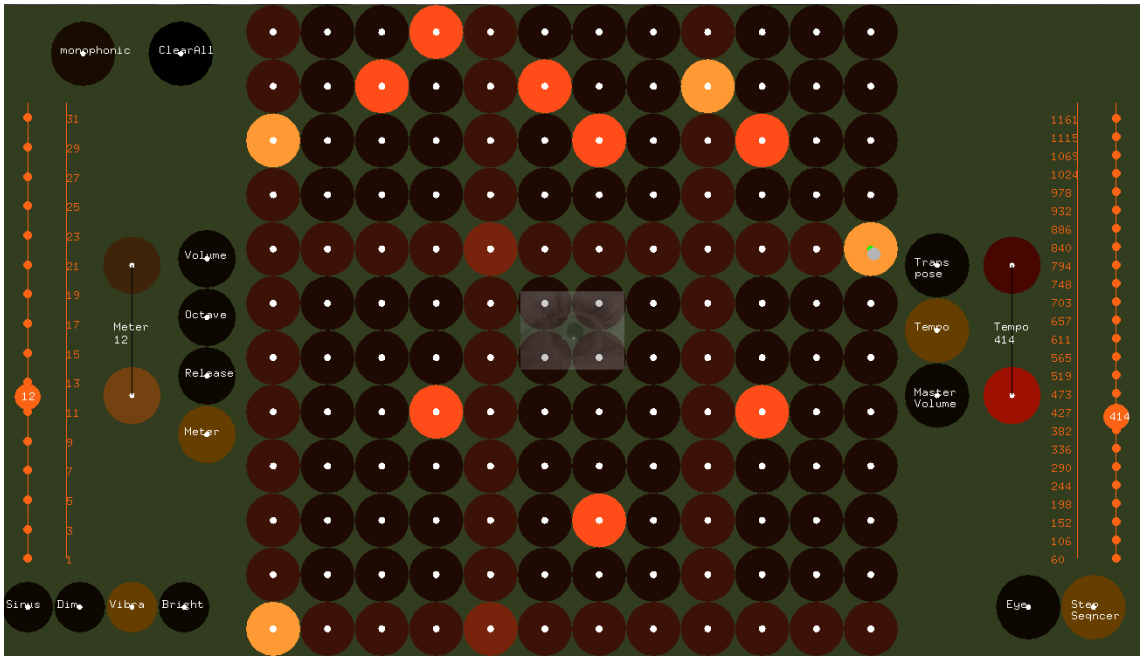

Figure 8.5: The EyeHarp Step Sequencer. Time Signature 12/16

8.2.3 The EyeHarp Step Sequencer: Building the harmonic and rhythmic background

The Step Sequencer Grid

Various interfaces based on step sequencers are available in different environments. Two commercial examples are the Tenori-on [27] and Max For Live Melodic Step Sequencer [2]. The EyeHarp Step Sequencer is implemented using similar ideas (see 8.5).

In the center of the screen there is a small transparent section which shows an image of the eye as captured by the camera in real-time. This is crucial for live performances, as it helps the audience to correlate the eye movements to the produced music. A small gray circle indicates the user's detected gaze point. Each circle / switch corresponds to a note. A note is selected when the user remains looking at it for more than one second. When a note is active, the color of the corresponding circle is brighter. To deactivate a note the user has to look at it again for a second. At the center of every circle, each of which corresponds to a note, there is a dot which helps the user to look at the middle of each circle. In every column we have the notes of the selected key with their pitch rising with direction from down to up.
A bright line is moving from left to right with a speed related to the selected tempo. When the line hits one of the green circles, the corresponding note sounds. So in this grid, in the horizontal dimension we have time and in the vertical dimension pitch (down→ low pitch, high→ high pitch). The horizontal brighter lines are for helping the user to understand where a new octave starts. Since the instrument is diatonic, a new octave starts every seven circles/notes. The vertical brighter lines appears every eight time steps and helps the user to better understand

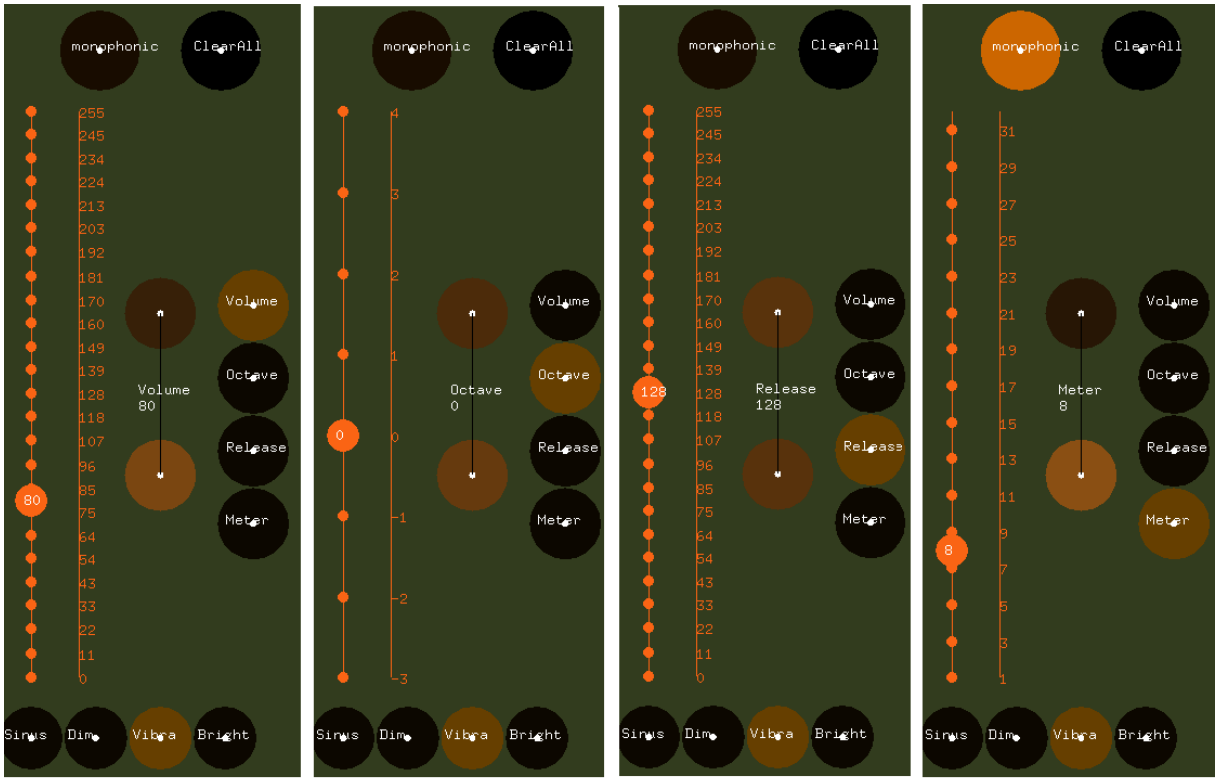

Figure 8.6: Controls of the Step Sequencer. (a)Volume (b)Octave (c)Release (d)Meter.

the temporal position of each column in the meter.

The controls of the Step Sequencer Layer

On the left side of the screen are placed the controls related to the step Sequencer layer (figure 8.6):

- Volume: determines the volume of the Step Sequencer.
- Octave: the octave of the Step Sequencer.
- Release: affects the release time of each note. The reverb of the the sound.
- Meter: The number of notes per loop in the step sequencer. In other words, the number of notes in the horizontal and vertical axis of the grid.

The slider along with its Repeat buttons can be assigned to any of the variables included in the tab in the right. In figure 8.6 we can also see the the range of these values.

The horizontal tab selector on the bottom of the screen determines the timbre of the Sequencer. Four Preset instruments are available.

On the top we can see two switches:

- 'Clear all' will erase all the notes of the step sequencer
- 'Monophonic' will limit the number of notes in each column of the step sequencer to one.

8.2.4 The EyeHarp Layer

Playing Melodies in Real Time

The EyeHarp interface was designed having in mind that it can be controlled without any dwell time applied or even without any gaze fixation detection algorithm. A velocity based fixation detection algorithm can be optionally activated. The velocity is computed by two successive frames and is given by the equation:

$$Velocity = \frac{\sqrt{(x_{t+1} - x_t)^2 - (y_{t+1} - y_t)^2}}{dt} \tag{8.1}$$

where $x_t, y_t, x_{t+1}, y_{t+1}$ are the screen coordinates of the gaze detected in each frame, and dt is the time between two successive frames. If the fixation-detection algorithm is not activated, the response time is expected to be equal to dt. As the frame-rate of the EyeHarp is set to 30 frames/second, dt is around 33 milliseconds. If the fixation-detection algorithm is active the response time of the system will be at least $2 \cdot dt$.

In any eye tracking device, noise will be registered due to the inherent instability of the eye, and especially due to blinking [8]. The EyeWriter software used for tracking the gaze coordinates, provides some configurations that can help to reduce that noise. By setting the minimum and maximum of the pupil' s size to the proper values the system might ignore the blinks in most of the cases. Another possible adjustment is the smoothing amount. The smoothed coordinates are given by the equation:

$x_n = S * x_{n-1} + (1 - S) * Gx_n$

$y_n = S * y_{n-1} + (1 - S) * Gy_n$

where x, y are the smoothed gaze values, Gx_n, Gy_n are the raw data of the gaze detection and S is the smoothing amount. $0 \leq S \leq 1$. For maximum temporal control, the smoothing amount should be set to zero.

Spatial Distribution of the Notes

The EyeHarp layer is displayed in 8.7. This interface is the evolution of the EyePiano interface. Two basic improvements were made:

- The instrument is diatonic. That way less notes have to be displayed on the screen. This is good, because we need big buttons, so as to balance the noise of a possible inaccurate eye tracker.
- The notes are placed around a circle. That way, the three fixations limitation in the EyePiano interface does not stand any more. Moreover, it is possible to play the instrument even without a fixation-detection algorithm, in case we want better temporal control.

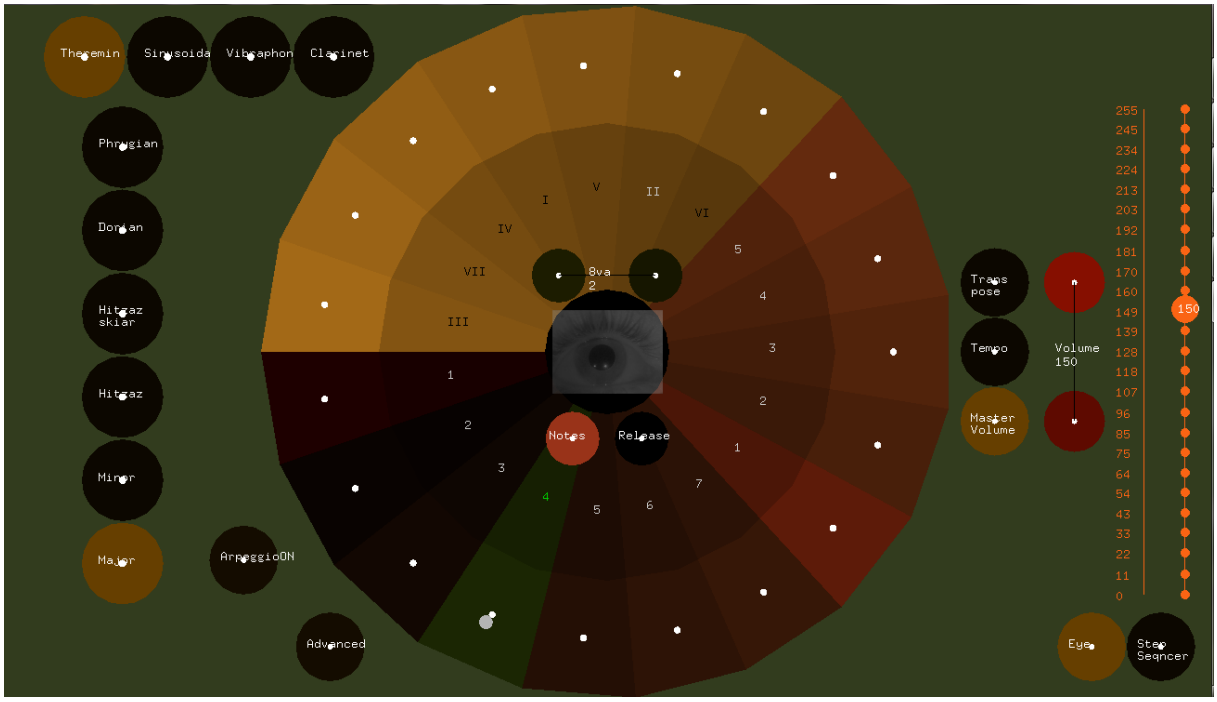

Figure 8.7: The EyeHarp layer.

Only two control buttons are placed inside the circle:

- *Octave up and down.*
- *NotesONOFF*: we press that button (with dwell time about one second) in order to move outside the circle without accidentally triggering any note.
- *Release*: If the user is looking at this button, the played note is released. The fixation detection algorithm is always active for this specific region. The reason for this is that the user should be able to play any melodic interval without accidentally releasing the played note.

In the middle of this circle there is a small black circle, where the performer' s eye is displayed. Using that spatial distribution, the user can have control over the articulation of the sound (staccato, legato). When not in percussive mode, in order play staccato, after triggering a note the user' s gaze should quickly return to the release button in order to release it soon. One more advantage of the circular spatial distribution is that all the notes are relatively close to each other, so it is easier to play every possible melodic interval. At the center of every note there is a focus point. In case the *Distance Volume/vibrato width* option is enabled -where the volume and vibrato width depend on the distance from the center- four focus points appear. A note is triggered immediately when the gaze of the user is detected inside its region. The *active region* -where the notes are triggered- placed at the periphery of the circle is a little bit brighter than the neutral region.

Almost no controls are placed in the neutral region inside the circle. The user' s gaze can move freely inside this region without triggering anything. Before playing a note the user can first look on the corresponding number inside the circle and then play it by looking at the white spot placed at the periphery. This is a common technique in the pEYEs and select button gaze selection techniques. This way of "clicking" provides an optimum temporal control, since the note is triggered exactly when the user looks at it without any dwell time. If the fixation-detection algorithm is inactive, the response time increased by one frame.

The dark color indicates a low pitch, while a bright color indicates a higher pitch. So the pitch increases in a counter clockwise order, starting from the most left note of the circle. If the gaze of the performer is between the small black circle in the center and the notes at the periphery of the main circle, nothing happens. The last triggered note will keep on being generated until a new note is played or it is released. As already mentioned the instrument is diatonic, so every seven notes we have a new octave.

The last seven notes of the disc are optionally dedicated to the chord selection. The order that the chords are placed is similar to the way the chords are placed in the left hand of an accordion. We are moving a clear fifth up to find the next chords to the right. So the order of the chords is: III, VII, IV, I, V, II, VI. The most commonly used IV, I, V chords are placed in the middle so as to be more accessible when playing a melody.

The controls of the EyeHarp layer

Basic Mode In figure 8.7, on the right part of the screen there are the controls of the basic mode of the EyeHarp layer. On the top of the screen we can see a tab control, where the user can choose between some preset timbres for the EyeHarp layer.

The vertical tab control is for choosing a preset musical mode. This is a global control, as it affects the tuning of both the step sequencer and the harmonizer.

Finally there are two switch controls:

- "ArpeggioON" for activating the harmonizer
- "Advanced" for switching to advanced mode, where more complicated controls appear.

Advanced Mode In figure 8.8 the advanced mode is active. When the "advanced" switch is active a tab selector appears next to the "advanced" button with the following options:

Controls When the 'controls' tab is selected, another tab control appears in the vertical axis. This tab control determines which variable will be assigned to the slider and repeat button on the left. All these variables will only affect the sound and graphical properties of the the EyeHarp layer for playing melodies in real time. The following options are available:

- Notes Number: In figure 8.8 the 'NotesNum' option is selected and its value is set to 22. This means that the disc for playing the melodies and changing the chords has a resolution of 22 notes. The minimum resolution is 7 notes, while the maximum is 36 notes.

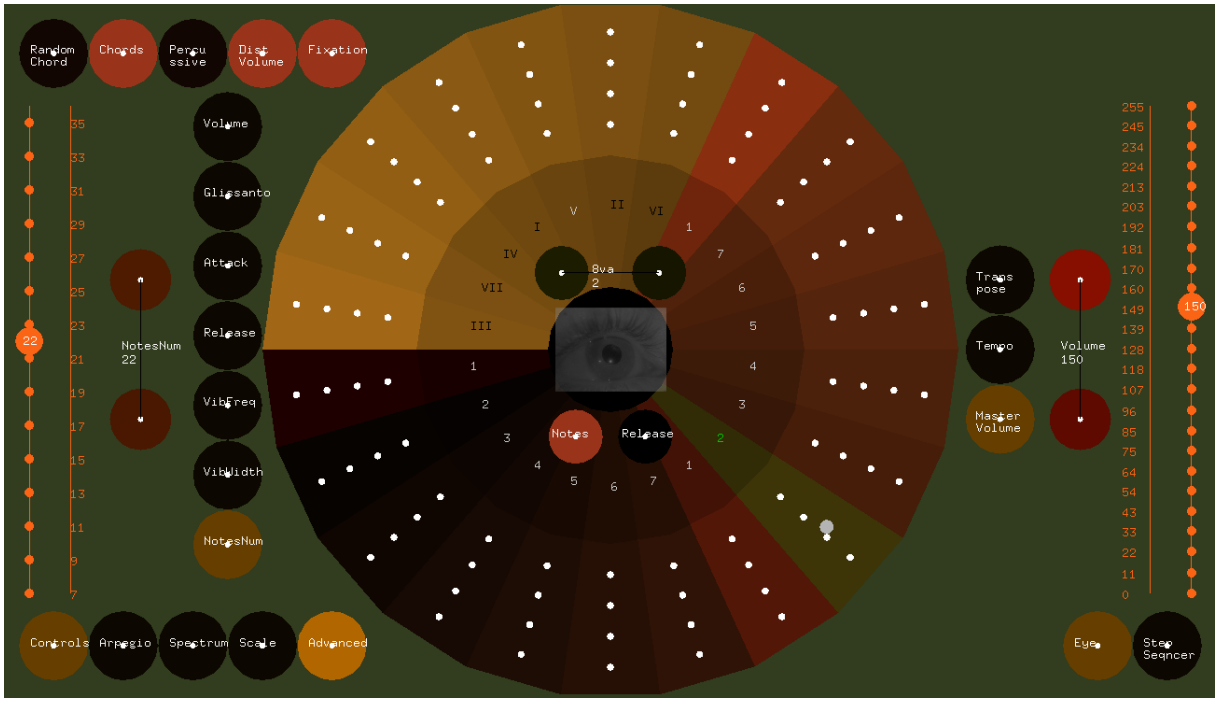

Figure 8.8: The EyeHarp layer with advanced controls, 22 notes and DistVol active.

- Vibrato Width: With this variable we set the maximum vibrato width of the EyeHarp additive synthesis part. When 'Distance Volume' is active, the width of the vibrato is also determined by the distance from the center of the circle (as well as the volume).
- Vibrato Frequency: Determining the frequency of the vibrato in centihertz.
- Release: Setting the release factor of the EyeHarp' s sound.
- Attack: The attack factor of of the EyeHarp' s sound.
- Glissanto: The glissanto factor when playing a melody.
- Volume: The volume of the EyeHarp' s sound.

Four more switches appear at the top of the screen:

- Fixation: This switch determines whether the velocity based fixation detection algorithm will be applied when playing melodies in real time. The advantage of having the fixation detection is that the noise introduced by blinking can be ignored. If the fixation detection is not activated, then during a blink, a random note might be triggered. Moreover the fixation detection prevents any note to be triggered during a saccade movement.
- Distance Volume: In figure 8.7 the Distance Volume switch is off, while in figure 8.8 it is on. When it is on, the notes width increases and four focus points appear instead of one. The reason is that the volume and vibrato width of the melody is depending on the distance from the center of the circle (the further, the louder).

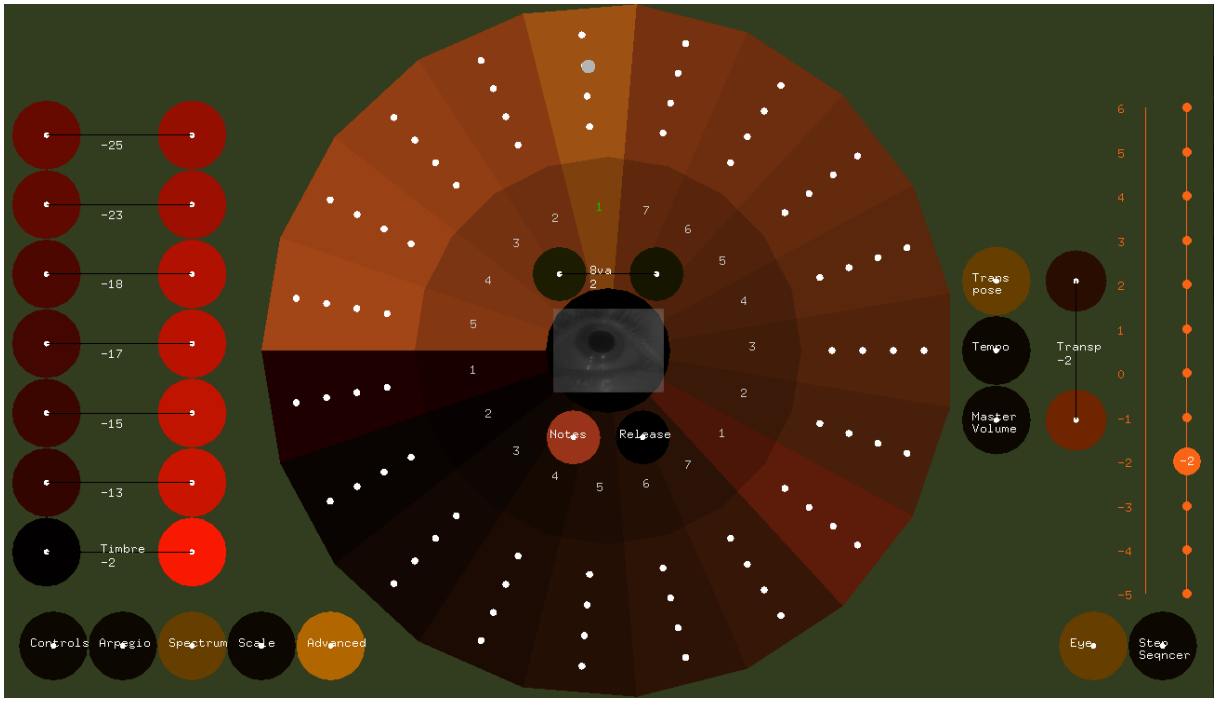

Figure 8.9: The EyeHarp layer without chords. Spectrum tab selected.

- Percussive: If the percussive switch is on, then when a note is triggered, the release phase starts immediately after the attack phase. Otherwise the release phase starts only if the user has a fixation on the 'release' button next to the center of the circle.
- Chords: This switch determines whether the last seven notes of the EyeHarp layer will be assigned for choosing the chords or not. In figure 8.9 for example chords do not appear in the interface.
- Random Chord: When this switch is on, then every meter, the chord is changing randomly. This creates an interesting harmonic background in order to focus on playing just a solo on top of it.

Spectrum In figure 8.9, the Spectrum tab is selected. In that case seven Repeat buttons appear for assigning the dB value of the amplitude of each harmonic of the additive synthesis of the melody sound.

Scale In figure 8.10 the Scale tab is selected. In this case we set the musical mode manually by assigning one of the twelve possible semitones to each of the seven notes of the desired scale. '0' corresponds to the first semitone of the scale, and '11' to the last one. In figure 8.10 the scale set is the Hitzazskiar mode.

Arpeggio When the arpeggio tab is selected, we can set the variables for generating up to four arpeggios. The "arpegiator" implemented is better described in the following section.

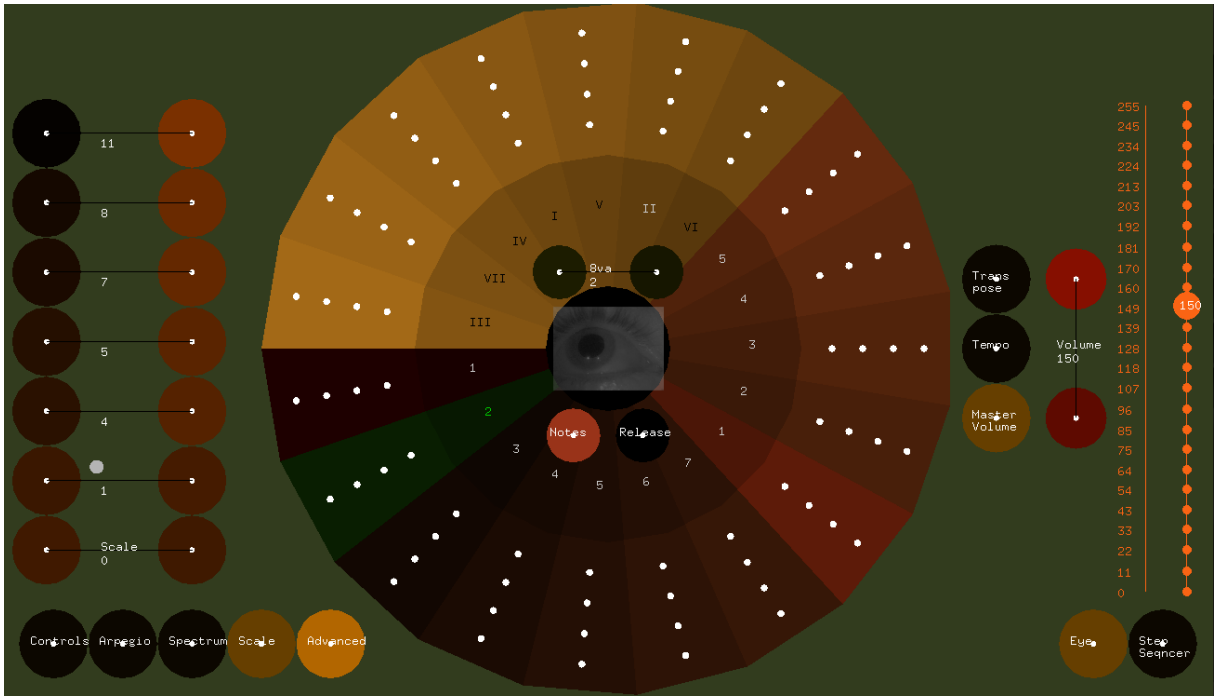

Figure 8.10: Setting the musical mode manually

8.2.5 Arpeggios generator

When the Arpeggio tab is selected, the controls for determining the produced arpeggios appear (figure 8.11). We can generate up to four arpeggios that will sound together in the end. At the top of the screen there is a tab selector, where we choose which arpeggio we need to modify.

The vertical tab selector determines which of the arpeggio variables will be set by the slider and repeat button on the left of the screen. In Figure 8.12 we can see the range of these controls:

- Starting Note: This variable determines the starting note of the arpeggio. The value '0' corresponds to A1 (55Hz).
- Meter: The total notes of the arpeggio. The tempo is always related to the step Sequencer. This means that if the meter of the step sequencer is set to '8', and an arpeggio has meter 16, then the notes of this arpeggio will sound two times faster (like playing 8ths and 16ths). The first note of all generated arpeggios and of the sequencer will sound together at the beginning of the meter.
- Notes Included: if value is set to '1', then only the tonal note of each chord will be included in the arpeggio. If the value is set to '2' then the tonal and 5th of the chord will be included. '3' $\rightarrow$ + 3rd, '4' $\rightarrow$ + 7th, '5' $\rightarrow$ + 2th, '6' $\rightarrow$ + 4th, '7' $\rightarrow$ + 6th.
- pattern Size: Each arpeggio is formed by repeated patterns. This variable determines how many notes are forming this pattern.

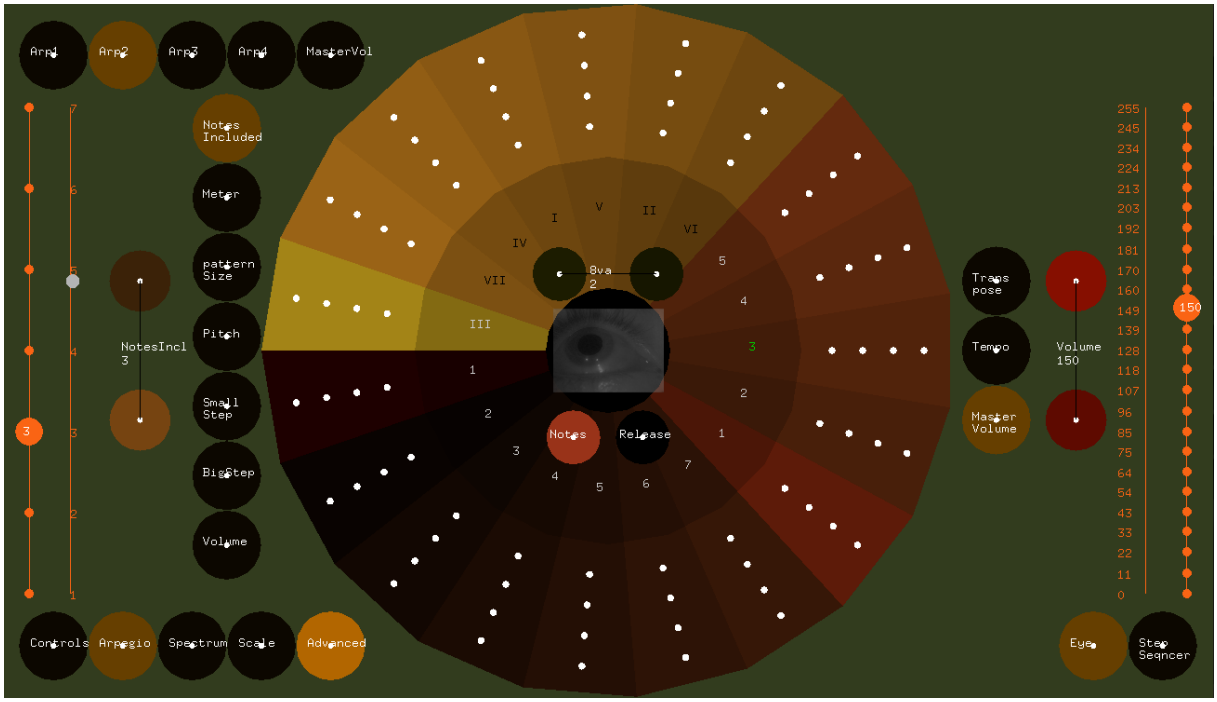

Figure 8.11: Arpeggio controls appear on the left of the screen.

- pattern Step: Along with the 'notes Included' determines the intervals between the notes in the pattern.
- Global Step: Along with the 'notes Included' determines the intervals between the starting notes of the patterns.
- Volume: The volume of each arpeggio.

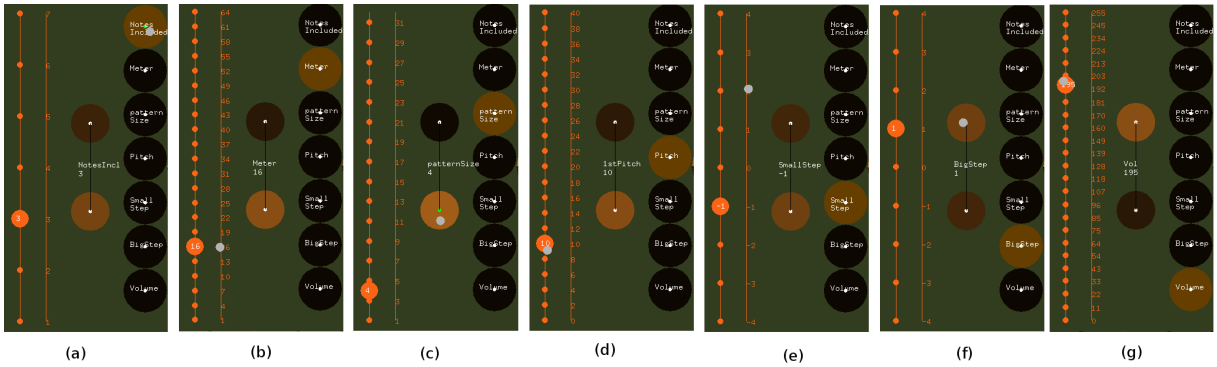

Figure 8.12: The Harmonizer controls.

Part IV

Evaluation and Validation

Chapter 9

Evaluation procedure

The eyeHarp is a musical instrument designed for people with disabilities. The proper evaluation procedure would be to evaluate it on different cases of people with disabilities.

The EyeHarp has already been tried once on some people with cerebral palsy of the "Catalan foundation of Celebral Palsy" (figure 9.1). Eight different users tried the interface using with head tracking, using the free software 'camera mouse'. No proper evaluation process occurred, but the users were all excited playing with the EyeHarp, even though they had no musical background. The fact that the EyeHarp is a diatonic instrument and consequently does not produce dissonant intervals, makes it more fun for people with no musical knowledge to experiment with it.

There are indications that the eyeHarp could not only improve the quality of living some cases of people with disabilities, but also help in rehabilitation procedures. In order to become an expert in the EyeHarp the user has to learn how to control the mouse pointer quite accurately. The instrument can be adapted to novice and expert users. In the case of people with cerebral palsy, this can me a motivation for these people improve the control over their movements and thus help in rehabilitation procedures.

In order to confirm this hypothesis and measure the level of motivation that the eyeHarp provides an extensive evaluation process should happen on different groups of people with disabilities. Unfortunately there was not enough time to both develop and evaluate the EyeHarp properly for this master thesis. This is part of the future work.

Nevertheless, we performed a discount usability research on people without any disability.

As a preliminary evaluation we have asked four people, three persons completely novice to the instrument (playing the EyeHarp for the first time), and another more experienced person who had spent many hours using the EyeHarp, to each perform two tasks: perform a two octave scale using the EyeHarp interface as accurate and speedy as possible, and generate a note pattern on the EyeHarp melodic step sequencer as speedy as possible. in addition, for comparison purposes we have asked the same two people to perform the same tasks using a video-based head tracking software [4]. Figure 9.2 shows the results of the experiment.

All participants agreed that proficiency in the EyeHarp improves with practice. The quality of the calibration process seemed to affect a lot the results. In the case of the first novice user,

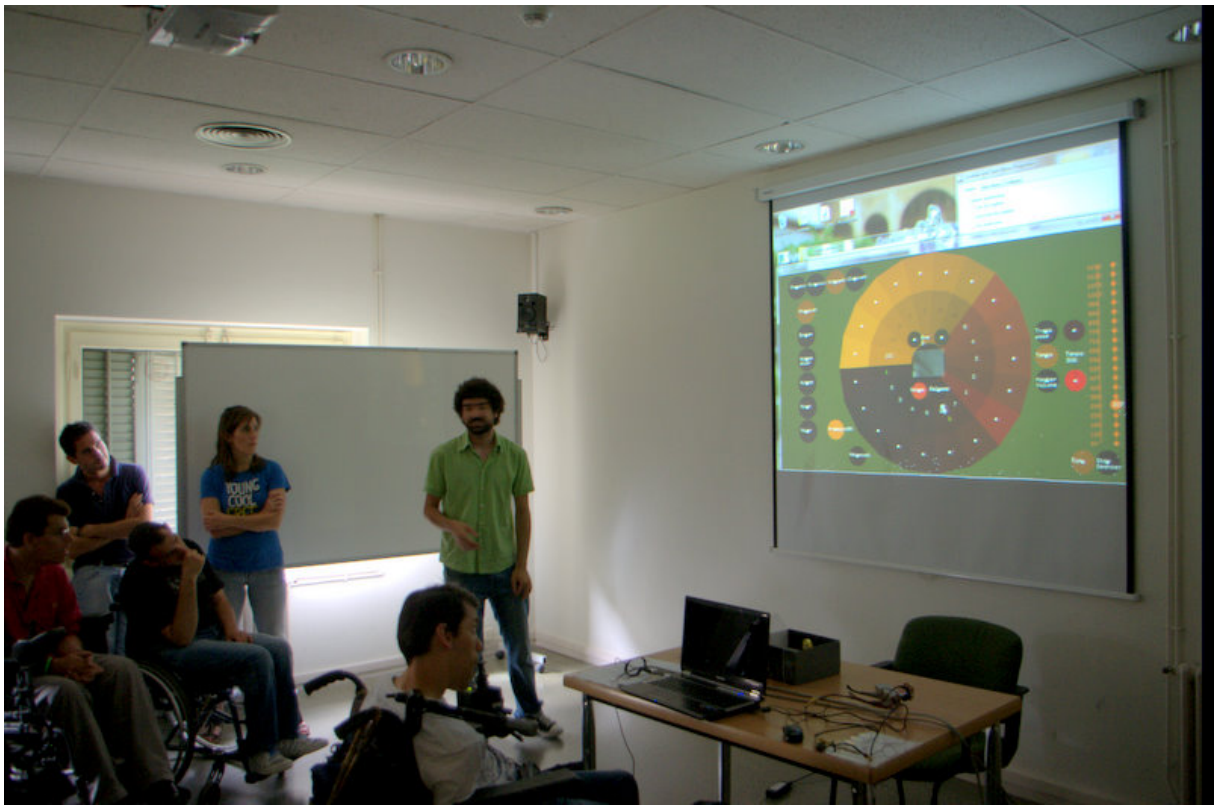

Figure 9.1: Visit at the "Catalan foundation of Celebral Palsy". The free software camera mouse[4] was used to control the applications with head movements.

User	Tracking	Two octave scale in the EyeHarp: Seconds	Two octave scale in the EyeHarp: Accuracy	Arpeggio in the Step Sequencer: Seconds
Expert	Eye	**12**	100%	**15**
	Head	36	100%	18
Novice	Eye	**18**	94%	30
	Head	40	75%	**29**
Novice	Eye	**30**	75%	28
	Head	46	75%	**24**
Novice	Eye	45	67%	40
	Head	**38**	93%	**38**

Figure 9.2: Eye-Tracking and Head-Tracking for an experienced and a novice user.

the calibration was good, and resulted to good results with the eye tracker. On the other hand, in the case of the last novice user, the calibration was not that good, so the head tracking outperformed the eye tracking input. In the case of the expert user, it seems that the eye tracking is a much faster way to control the EyeHarp interface for playing real time melodies (12 seconds with the eye tracker and 36 with the head tracker). Observing the participants interact with the EyeHarp after the experiment, it seems that the fixation-detection algorithm is indeed very helpful for a novice user and can be activated for increasing the spatial accuracy of the system. The smoothing amount can be adjusted as well. The user can choose between better spatial (not pressing notes accidentally) or temporal control by adjusting these two parameters.

It has to be noted that the accuracy of the implemented eye-tracking device was not explicitly evaluated (this is out of the scope of this paper). However, the EyeHarp interface can be used along with more accurate commercial eye tracking systems. It is very likely that the temporal and spatial control would be even better in that case.

Probably the best way to evaluate the potential of the EyeHarp as a musical instrument is to listen to performances produced using the instrument that can be found online in the blog of the EyeHarp.

Part V

Conclusions

Chapter 10

Assesment of the results and future work

In this thesis we presented the EyeHarp: a gaze controlled musical instrument. The project is free and open-source. At the moment it runs only under Microsoft Windows operating system (XP, vista, 7). The source code and the executable can be found at:
http://code.google.com/p/eyeharp/
A version for controlling the EyeHarp with the mouse is also available.

The reader may listen (and watch) one performance with gaze input at:
http://youtu.be/XyU8FyB0nZ8

In this video it is evident that using the EyeHarp makes it possible to create complex music with just eye movements in real time. The EyeHarp could be a musical interface appropriate for people with different level and kind of disability and has to be tried and evaluated on different groups of disabled people. Depending on the physical ability of the user, any device that can take control of the mouse pointer can be used for playing music. This might improve the quality of life of the user and provide a motivation for fine controlled physical activity, depending on the input device used. That way it could also be used for rehabilitation purposes.

All these have to be measured and evaluated. Evaluating and adapting the EyeHarp to people with different kind of disabilities is the main priority for future work. For that purpose we are already in contact with foundations and institutes for people with disabilities. In the near future the results of that evaluation process will be available.

In order to better evaluate the EyeHarp, a more accurate eye-tracking device should be used. Since the commercial eye-trackers cost more than €4000, another low-cost solution would be to use the ITU open source gaze-tracker [1], along with the suggested "Raven" setup [2]. This system is expected to have better accuracy than the EyeWriter system used during the implementation of the EyeHarp, since it is not head-mounted and allows head movements -due to the commonly used corneal reflections technique-. A high definition infra-red sensitive camera is used along with two infra-red illuminators. It costs around €400 to buy all the required parts.

[1] http://www.gazegroup.org/downloads/23-gazetracker
[2] http://www.gazegroup.org/forum/

Moreover the EyeHarp might also be evaluated as a musical instrument of a normal band. Its functionality could be extended as well. For example the user might have the option to record his compositions, listen to them and modify them.

Part VI

Appendixes

Appendix A

Digital resources

To keep updated with the future work regarding the EyeHarp project, the reader can follow the blog:
www.theeyeharp.blogspot.com
The EyeHarp software is open-source. The code is hosted online at:
http://code.google.com/p/eyeharp/

Appendix B

The EyeHarp at the Sound and Music Computing Conference, 2011

THE EYEHARP: AN EYE-TRACKING-BASED MUSICAL INSTRUMENT

Zacharias Vamvakousis
Universitat Pompeu Fabra
Roc Boronat 138
08018 Barcelona, Spain
zackbam@gmail.com

Rafael Ramirez
Universitat Pompeu Fabra
Roc Boronat 138
08018 Barcelona, Spain
rafael.ramirez@upf.edu

ABSTRACT

In this paper we present the EyeHarp, a new musical instrument based on eye tracking. The EyeHarp consists of a self-built low-cost eye-tracking device which communicates with an intuitive musical interface. The system allows performers and composers to produce music by controlling sound settings and musical events using eye movement. We describe the development of the EyeHarp, in particular the construction of the eye-tracking device and the design and implementation of the musical interface. We conduct a preliminary experiment for evaluating the system and report on the results.

1. INTRODUCTION

Traditionally, music performance has been associated with singing and hand-held instruments. However, nowadays computers are transforming the way we perform and compose music. Recently, music performance has been extended by including electronic sensors for detecting movement and producing sound using movement information. One early example of this new form of music performance is the theremin and terpsitone [1]. More recent examples of new music performance paradigms are systems such as The Hands [2] and SensorLab [3]. The creation of these kinds of musical electronic instruments opens a whole new door of opportunities for the production and performance of music.

Eye tracking systems provide a very promising approach to real-time human-computer interaction (a good overview of eye tracking research in human-computer interaction can be found in [4]). These systems have been investigated in different domains such as cognitive psychology where eye movement data can help to understand how humans process information. Eye tracking systems are also important for understanding user-device interaction and to allow physically disabled people to communicate with a computer using eye movements.

In this paper, we present the EyeHarp, a new music instrument based on eye tracking. We have built a low-cost tracking device based on the EyeWriter project [5] and implemented various musical interfaces for producing sound. The resulting system allows users to perform and compose music by controlling sound settings and musical events using eye movement.

The rest of the paper is organized as follows: Section 2 describes the background to this research. Section 3 presents the EyeHarp, in particular it describes the construction of the eye-tracking device, the design and implementation of the musical interface, and the evaluation of the system. Finally Section 4 presents some conclusions and future work.

2. BACKGROUND

2.1 Eye tracking systems

Several approaches for detecting eye movement have been proposed in the past. These have included electrophysiological methods [6, 7], magnetic search coil techniques [8], infrared corneal reflectance and pupil detection methods.

Electrophysiological methods involve recording the difference potentials generated between electrodes placed in the region around the eyes. However, this method has been found to vary over time, and is affected by background activation of eye muscles [7]. The disadvantages of search coil systems are that its use involves quite invasive procedures, and it relies on expensive hardware (i.e. around US$40,000).

In recent years video-based eye movement detection has gained popularity due to the fact that it offers a solution to some of the limitations of other methods. For instance, it allows reliable tracking of the pupil as well as tracking of the iris as it rotates torsionally around the optic axis [9, 10] at rates of up to 250 frames per second. However, one limitation of this type of system is the need for greater intensity of infrared illumination to allow adequate passing of light from the eye to the camera sensor.
Combined pupil detection and corneal reflection techniques are becoming more and more popular lately for interactive systems. The reason is that with this combined method the head of the user does not have to be fixed.

2.2 Eye-tracking-based music systems

The first system using eye tracking devices to produce music in real time was proposed by Andrea Polli in 1997 [11]. Polli developed a system which allowed performers to access a grid of nine words spoken by a single human voice

by making saccadic to nine different directions. After trying different artistic implementations Polli concluded that improvising with the eye-tracking instrument could produce the same feeling for the performer as improvisation with a traditional instrument [11].

In 2001 she performed "Intuitive Ocusonics", a system for sound performance using eye tracking instruments to be performed live. Instruments were played using distinct eye movements. Polli's compositions responded to video images of the eye, not specifically the pupil center which are parsed and processed twelve times per second using the STEIM's BigEye software (www.steim.org). With this technology it is impossible to calibrate the pupils position to the computer screen coordinates, thus the user does not have precise control of the system.

Hornof et al. [12] propose a system based on a commercial eye tracking system, the LC Technologies Eyegaze System, which provides accurate gazepoint data using the standard pupil-center corneal-reflection technique. In the system, the coordinates of the user' s gaze are sent to MAX/MSP for generating sound. They study both the case of using fixation detection algorithms for choosing an object and the raw data form the eye tracker. When trying to implement an eye-piano they report that the musicians that tried the system preferred to work with the raw data instead of a dispersion-based fixation-detection for playing the notes. The problem with fixation detection is that it reduces the temporal control, which is very critical in music. A velocity-based fixation-detection algorithm is suggested instead. They do not consider other techniques, such as blink detection, as a method for choosing objects. The authors consider designing more interactive tools using Storyboarding. The performer moves an eye-controlled cursor around on the screen, and makes the cursor come into direct visual contact with other visual objects on the screen, producing a visual and sonic reaction. The user interacts with objects that appear on the screen, through a series of interaction sequences (like a scenario).

Hornof and Vessey in a recent technical report evaluate four different methods for converting real-time eye movement data into control signals (two fixation based and two saccade-based methods). They conduct an experiment comparing the musicians' ability to use each method to trigger sounds at precise times, and examined how quickly musicians are able to move their eyes to produce correctly-timed, evenly-paced rhythms. The results indicate that fixation based eye-control algorithms provide better timing control than saccade based algorithms, and that people have a fundamental performance limitation for tapping out eye-controlled rhythms that lies somewhere between two and four beats per second [13]. Hornof claims in [12] that velocity-based (as opposed to dispersion-based) fixation-detection algorithms work better for rhythmic control with the eyes. Fixation-detection algorithms typically employ a minimum fixation duration of 100 ms which would impose an upper bound of ten eye-taps per second.

Kim et al. [14] present a low cost eye-tracking system with innovative characteristics, called Oculog. For selecting objects, blink detection is implemented. The data from the eye tracking device are mapped to PureData for generating and interacting with four sequences. In their user interface the performer's field of vision is divided into four discrete quadrants. The direction of eye movement detected by the Oculog camera software is encoded as a combination of horizontal position (pitch) and vertical position (velocity): pitch 0 is produced by looking to the extreme left, note number 127 to the extreme right; velocity 0 is produced by looking down, velocity 127 by looking up. Assigned to each quadrant is a real-time tone generator. Each tone generator is driven by a cyclic sequence. Oculog also detects torsional movement of the eye, but this is not mapped to any control feature. The authors claim that eye tracking systems are appropriate for micro-tonal tuning (they used a 15 note scale).

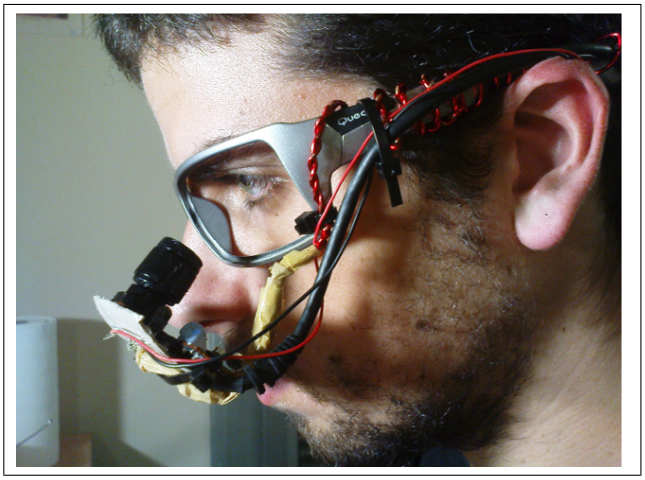

Figure 1. The PlayStation Eye digital camera is modified so as to be sensitive to infra-red light and mounted along with two infra-red leds on a pair of sun-glasses.

3. THE EYEHARP

3.1 Eye tracking device

There are a number of commercial systems available specifically designed to enable people to communicate using their eyes. However, these systems are expensive, costing in the range of US$20,000. In order to create a reproducible system we decided to make the most simple and inexpensive eye-tracking head-set possible. We built our own eye tracking system based on the EyeWriter project [5] . Thus, the resulting system emphasizes low-cost and ease of construction and as a consequence has several limitations such as robustness and appearance. Figure 1 shows the eye tracking device used in this work.

In order to read the input from the eye tracking device, we have used the libraries developed in the EyeWriter project. The eye-tracking software detects and tracks the position of a pupil from an incoming camera or video image, and uses a calibration sequence to map the tracked eye/pupil coordinates to positions on a computer screen or projection. The pupil tracking relies upon a clear and dark image of the pupil. The eye tracking device includes near-infrared leds to illuminate the eye and create a dark pupil effect.

This makes the pupil much more distinguishable and, thus, easier to track. The software dealing with the camera settings allows the image to be adjusted with brightness and contrast to get an optimal image of the eye. When initializing the system, calibration takes place displaying a sequence of points on the screen and recording the position of the pupil at each point. The user focuses on a sequence of points displayed in the screen presented one by one. When the sequence is finished, the collected data are used to interpolate to intermediate eye positions.

3.2 Music interface

The ultimate goal of this project is to create a real musical instrument with the same expressive power as traditional musical instruments. The implemented instrument should be suitable for being used as a musical instrument for performing in a band, as well as a standalone composition tool. The following decisions have been taken in the EyeHarp design:

- More than one different layer should be available. One of them could be used for building the rhythmic and harmonic musical background, and another for playing accompanying melodies on top of the musical background.
- The performer should be able to control in real time the rhythmic, harmonic and melodic aspect of his/her composition, as well as to control the timbre of the instrument. The instrument's timbre is determined by having control over (i) the spectral envelope, (ii) the attack-decay time of the produced sound. In addition, the performer should have control over the articulation and other temporal aspects of sound such as glissando and vibrato.
- The buttons on the screen for playing a note should be big enough in order to reduce the possibility of playing neighbor notes, due to errors of the eye tracking system. To save space and avoid dissonant notes the produced instrument should be diatonic (like e.g. the harmonica). The user should be able to determine the musical mode while performing.
- Temporal control in music is crucial. This is why we should avoid using blink detection or fixation detection algorithms for playing real-time melodies. Music should be controlled by making use of just the user' s gaze. Thus, the process of designing an eye tracking musical instrument is similar to designing an instrument in which the input is a pencil (eye gaze) drawing on a paper (screen), where the pencil should always be in touch the surface of the paper. Consequently the performer should be able to play every pair of notes with a straight saccade eye movement without activating any other note. This would allow working with the raw data of the eye tracker and skip the use of any fixation detection algorithm that would increase the response time of the instrument [12].

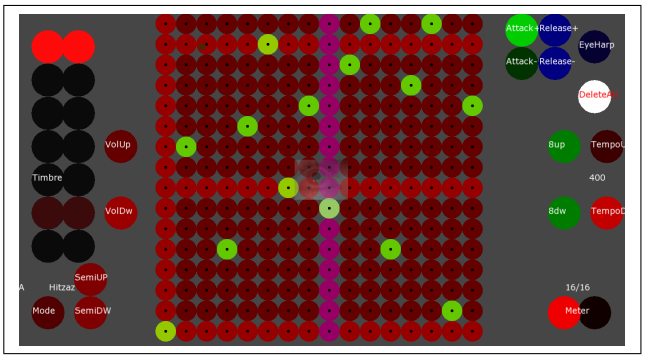

Figure 3. The EyeHarp Melodic Step Sequencer. Time Signature 16/16

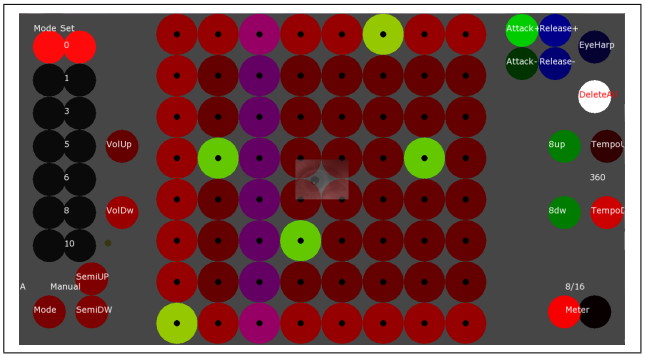

Figure 4. Setting the musical mode manually. In this case [0,1,3,5,6,8,10] corresponds to the mixolydian mode.

3.2.1 The EyeHarp Melodic Step Sequencer: Building the harmonic and rhythmic background

Various interfaces based on step sequencers are available in different environments. Two commercial examples are the Tenori-on [15] and Max For Live Melodic Step Sequencer [16]. The EyeHarp Melodic Step Sequencer is implemented using similar ideas (see Figure 2).

In the center of the screen there is a small transparent section which shows an image of the eye as captured by the camera in real-time. This is crucial for live performances, as it helps the audience to correlate the eye movements to the produced music. A small green circle indicates the user's detected gaze point. Each circle corresponds to a note. A note is selected when the user remains looking at it for more than one second. When a note is active, the color of the corresponding circle is green. Only one note can be selected for each step of the sequence. To deactivate a note the user has to look at it again for more than a second. At the center of every circle, each of which corresponds to a note, there is a black dot which helps the user to look at the middle of each circle. In every column we have the notes of the selected key with their pitch rising with direction from down to up.

The purple line in Figure 2 is moving from left to right with a speed related to the selected tempo. When the line hits one of the green circles, the corresponding note is played. So in this grid, in the horizontal dimension we have time and in the vertical dimension pitch (down→ low pitch, high→ high pitch). The horizontal brighter lines are for

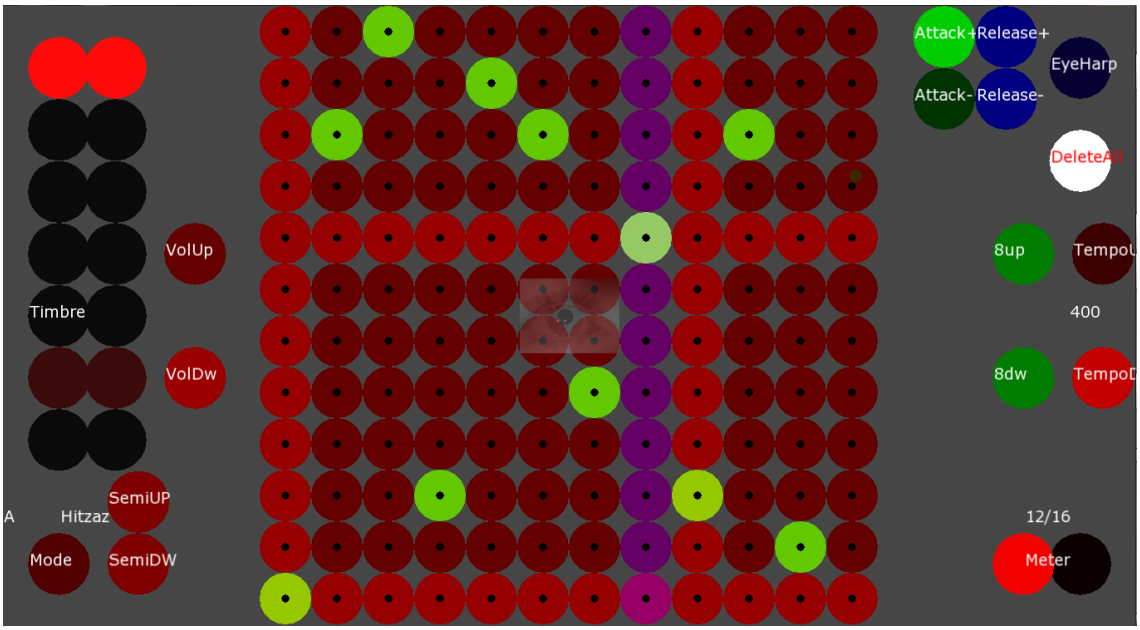

Figure 2. The EyeHarp Melodic Step Sequencer. Time Signature 12/16

helping the user to understand where a new octave starts (every 5 or 7 notes depending on the mode) The vertical brighter lines repeat every eight time steps and help to visualize beats.

On the left and right of the "score" region described above, there are various buttons (i.e. circles) affecting different sound and musical aspects of the composition. On the left of the score region there are two circle buttons for controlling the volume. Again a time threshold is used for triggering a volume change: every 0.25 seconds that the user keeps looking at the "volumeUp" button, the volume is increased one step. The color of the corresponding button gets brighter or darker corresponding to its value. When the volume reaches its maximum value the "VolumeUp" button is bright red and "VolumeDown" button is black. That way the user has feedback about when a control parameter has reached its minimum or maximum value. The same applies to most of the circle buttons for controlling all different input variables: "SemitoneUp" - "SemitoneDown", "AttackUp" - "AttackDown", "ReleaseUp" - "ReleaseDown", "TempoUp" - "TempoDown", "OctaveUp" - "OctaveDown", "MeterUp" - "MeterDown". The "MeterUp" - "MeterDown" buttons change the dimensions of the sequencer's grid. That way the time signature of the composition changes as well (ranges from 1 to 64).

The two columns of circles at the left of the screen are for controlling the amplitude of each of the seven first harmonics of the synthesized sound. The left column is for decreasing the contribution of each harmonic and the right for increasing it. The "mode" button is for switching between different musical (scale) modes. The available modes are the major, Hitzaz [1] and pentatonic. The "mode" can also been set to "manual". In this case the buttons that normally were for setting the amplitude of the harmonics can be used for setting the musical mode manually: each note of the scale can be assigned to any semitone. For example in Figure 4 the mode is set manually to the mixolydian. Transposition to all the different semitones is available as well. Finally "DeleteAll" sets all the notes to inactive. The "EyeHarp" button is for switching to the EyeHarp layer for playing a melody on top of the composed loop.

The EyeHarp Step Sequencer is not designed for playing real time melodies, but for building the harmonic and temporal background of the composition. The decisions mentioned at the beginning of this section apply mostly to the next proposed layer for playing real time melodies.

3.2.2 The EyeHarp

Playing Melodies in Real Time The EyeHarp interface was designed having in mind that it can be controlled with or without a gaze fixation detection algorithm. A velocity based fixation detection algorithm can be optionally activated. The velocity is computed by two successive frames and is given by the equation:

$$Velocity = \frac{\sqrt{(x_{t+1} - x_t)^2 - (y_{t+1} - y_t)^2}}{dt} \quad (1)$$

where $x_t, y_t, x_{t+1}, y_{t+1}$ are the screen coordinates of the gaze detected in each frame, and dt is the time between two successive frames. If the fixation-detection algorithm is not activated, the response time is expected to be equal to dt. If the fixation-detection algorithm is active the response time of the system will range from $2 \cdot dt$ to $4 \cdot dt$. In any eye tracking device, noise will be registered due to the inherent instability of the eye, and specially due to blinks [17]. The EyeWriter software used for tracking the gaze coordinates, provides some configurations that can help to reduce that noise. By setting the minimum and maximum

[1] Hitzaz is a mode used in Eastern music (e.g. Greece, Turkey and some Arabic countries) and flamenco music

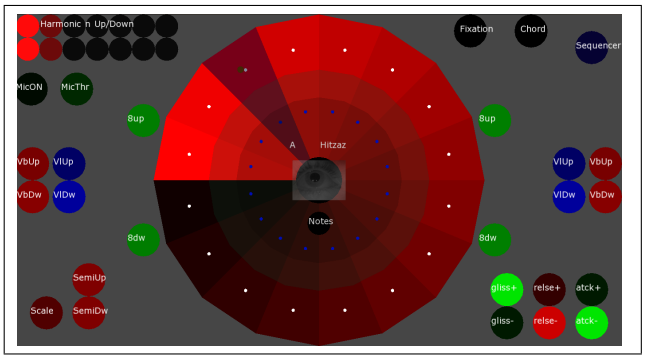

Figure 6. The EyeHarp without chords.

of the pupil' s size to the proper values the system might ignore the blinks in most of the cases. Another possible adjustment is the smoothing amount. The smoothed coordinates are given by the equation:

$$x_n = S * x_{n-1} + (1 - S) * Gx_n$$
$$y_n = S * y_{n-1} + (1 - S) * Gy_n$$

where x, y are the smoothed gaze values, Gx_n, Gy_n are the raw data of the gaze detection and S is the smoothing amount. $0 \leq S \leq 1$. For maximum temporal control, the smoothing amount should be set to zero.

The PlayStation Eye camera that is used in this project is capable of capturing standard video with frame rates of 75 hertz at a 640×480 pixel resolution. The program has been tested on a Intel Core i5 460M processor with 4GB of RAM and an nVIDIA GeForce GT 330M graphic card. For the sound to be generated smoothly, the refresh rate is set at 30 frames/second. Thus, without the fixation-detection algorithm the response time is 25ms, while with the fixation algorithm it is 50-100ms.

Spatial Distribution of the Notes The EyeHarp layer is displayed in Figure 5 and Figure 6. In order to be able to play the instrument without a fixation-detection algorithm all the notes are placed at the periphery of a circle. In the middle of this circle there is a small black circle, where the performer' s eye is displayed. If the performer looks at this circle, then the played note is released. The fixation detection algorithm is always active for this specific region. The reason for this is that the user should be able to play any melodic interval without accidentally releasing the played note. So the release region in the center is triggered only when a fixation is detected. Using that spatial distribution, the user can have control over the articulation of the sound (staccato, legato). To play staccato, after triggering a note the user' s gaze should quickly return to the center of the circle in order to release it soon. One more advantage of that spatial distribution is that all the notes are relatively close to each other, so it is easy to play every possible melodic interval. At the center of every note there is a white spot that helps the user to focus on it. A note is triggered immediately when the gaze of the user is detected inside its region. Almost no controls are placed in the region inside the circle. The user' s gaze can move freely inside this region without triggering anything. A second row of blue dots are placed inside this region. Before playing a note the user can first look on the corresponding blue spot inside the circle and then play it by looking at the white spot placed at the periphery. This way of "clicking" provides an optimum temporal control, since the note is triggered exactly when the user looks at it. If the fixation-detection algorithm is inactive, the response time is only limited by the frame rate. The dark color indicates a low pitch, while a bright color indicates a higher pitch. So the pitch increases in a counter clockwise order, starting from the most left note of the circle. If the gaze of the performer is between the small black circle in the center and the notes at the periphery of the main circle, nothing happens. The last triggered note will keep on being generated until a new note is played or it is released. As already mentioned the instrument is diatonic, so every seven or five (for pentatonic) notes we have a new octave.

Spatial distribution of the control buttons All the control buttons work in the same way as described in the step sequencer layer: there is a time threshold -different for every button- for moving the corresponding variable one step up or down.

The only control button that is inside the main circle is the one that deactivates all the notes. Obviously, there was a need for the gaze to move outside the circle in order to change several aspects of the synthesized sound. If the fixation-detection is inactive, in order to go outside the circle without triggering a note, the notes should be deactivated first. They can be activated again by looking at the black circle in the middle of the interface.

On the upper right corner of the screen, there is the "fixation" button for activating or deactivating fixation-detection. Next to it, there is the "chord" button. When it is active the notes at the upper part of the circle are assigned for changing the harmonies of the Step Sequencer. The user can build an arpeggio in the sequencer layer and then change the harmonies of his composition in the EyeHarp layer. The closest buttons to the main circle are the ones for changing octaves, and they are placed close to the lowest and highest pitches of the interface. If the fixation-detection is not active, when pressing any of the buttons for changing octave, the notes are automatically deactivated, so the user can enter inside the main circle again without triggering any note accidentally.

As it can been seen in Figure 5, there are buttons for adjusting the glissando, attack, release, volume, vibrato, amplitude of each harmonic, tonality (semitone up, down and "mode"), and switching to the sequencer layer. The two layers have their own sound properties, apart from the ones related to the tonality. That means that the timbre, articulation, temporal aspects of the sound, octave of each layer can be set to different values (e.g. choose a percussive timbre for the sequencer and a harmonic timbre for the melody). The user can also activate the microphone input for blowing in the microphone and having control over the amplitude of the melody that he is performing (a very dynamic microphone is recommended). The minimum sound level to be considered as an input can be set through the "MicThr" button.

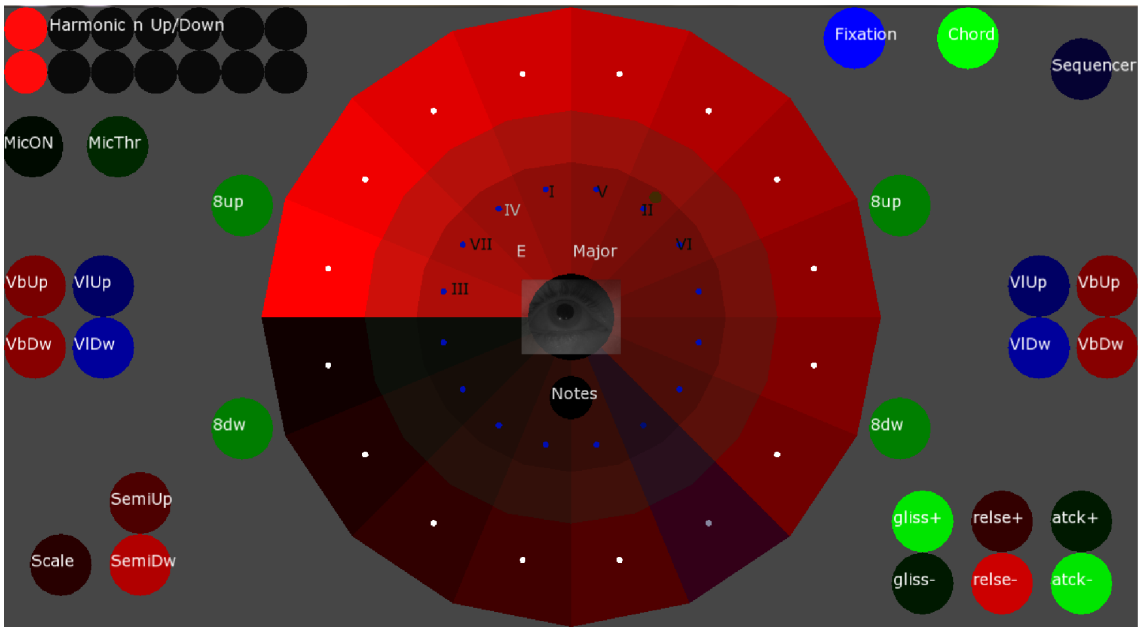

Figure 5. The EyeHarp layer. The selected scale is E major. The 4th chord of the key is played (A minor) and the melody note is do. The user is about to play the second chord of the key (F# minor).

3.3 Implementation

The interface and sound synthesis of the EyeHarp were implemented in Openframeworks [18], an open source C++ toolkit for creative coding. Openframeworks is used in all the stages of the system: (i) tracking the pupil of the eye and calibrating (based on the EyeWriter Project), (ii) designing the different modes of the EyeHarp (iii) synthesizing the sound.

3.4 Evaluation

Evaluating a new musical instrument is a difficult task. Ideally, the instrument should be evaluated at different stages. It should be evaluated on how accessible or "playable" it is for novice performers, how easy/difficult it is to improve with experience, and what is the potential of the instrument performed by experts. As a preliminary evaluation we have asked two people, one person completely novice to the instrument (playing the EyeHarp for the first time), and another more experienced person who had spent many hours using the EyeHarp, to each perform two tasks: perform a two octave scale using the EyeHarp interface as accurate and speedy as possible, and generate a note pattern on the EyeHarp melodic step sequencer as speedy as possible. in addition, for comparison purposes we have asked the same two people to perform the same tasks using a video-based head tracking software [19]. Figure 7 shows the results of the experiment.

Both (the experienced and novice) participants agreed that proficiency in the EyeHarp improves with practice. Observing the participants interact with the EyeHarp after the experiment, it seems that the fixation-detection algorithm is indeed very helpful for a novice user and can be activated for increasing the spatial accuracy of the system. The smoothing amount can be adjusted as well. The user can choose between better spatial (not pressing notes accidentally) or temporal control by adjusting these two parameters.

		Two octave scale in the EyeHarp		Arpeggio in the Step Sequencer
User	Tracking	Seconds	Accuracy	Seconds
Expert	Eye	12	100%	15
	Head	36	100%	18
Novice	Eye	18	94%	30
	Head	40	75%	29

Figure 7. Eye-Tracking and Head-Tracking for an experienced and a novice user.

It has to be noted that the accuracy of the implemented eye-tracking device was not explicitly evaluated (this is out of the scope of this paper). However, the EyeHarp interface can be used along with more accurate commercial eye tracking systems. It is very likely that the temporal and spatial control would be even better in that case.

Probably the best way to evaluate the potential of the EyeHarp as a musical instrument is to listen to performances produced using the instrument. The reader may listen (and watch) one such performance at:
http://www.dtic.upf.edu/~rramirez/eyeharp/EyeHarpDEMO.wmv

4. CONCLUSIONS

We have presented the EyeHarp, a new musical instrument based on eye tracking. We have built a low-cost eye-tracking device which communicates with a melody and step sequencer interface. The interface allows performers and composers to produce music by controlling sound settings and musical events using eye movement. We have described the development of the EyeHarp, in particular design and implementation of the melody and step sequencer interface. Finally, we have conducted a preliminary experiment for evaluating the system and compare its usability with a similar video-based head tracking controller. The results are encouraging but are still preliminary because the evaluation included only one experienced performer and one novice performer. The EyeHarp interface is still under development and many aspects, such as the choice of colors and spatial distribution of the control buttons, are still under reconsideration.

The eyeWriter project team has provided "a low-cost eye-tracking apparatus and custom software that allows graffiti writers and artists with paralysis resulting from Amyotrophic lateral sclerosis to draw using only their eyes". The eyeHarp is a musical instrument that could give to these people the opportunity to express themselves through music, but can also be used by anyone as musical instrument in a traditional way.

5. REFERENCES

[1] B. M. Galayev, "Light and shadows of a great life: In commemoration of the one-hundredth anniversary of the birth of leon theremin," *Pioneer of Electronic Art. Leonardo Music Journal*, pp. 45–48, 1996.

[2] M. Waiswicz, "The hands: A set of remote midi controllers." in *Proceeding s of the 1985 International Computer Music Conference*. San Francisco: Computer Music Association, 2003, pp. 573–605.

[3] A. Tanaka, "Musical technical issues in using interactive instrument technology." in *Proceedings of the International Computer Music Conference*. San Francisco: International Computer Music Association, 1993, pp. 124–126.

[4] R. J. K. Jacob and K. S. Karn, "Eye tracking in human-computer interaction and usability research: Ready to deliver the promises," in *The Mind's Eyes: Cognitive and Applied Aspects of Eye Movements. Elsevier Science*, . H. D. J. Hyona, R. Radach, Ed., 2003, pp. 573–605.

[5] The eyewriter project. [Online]. Available: http://www.eyewriter.org/

[6] J. A. Werner, "A method for arousal-free and continuous measurement of the depth of sleep in man with the aid of electroencephalo-, electrooculo- and electrocardiography (eeg, eog, and ekg)," in *Z Gesamte Exp. Med.*, 1961, vol. 134, pp. 187–209.

[7] S. Iwasaki, L. A. McGarvie, G. M. Halmagyi, A. M. Burgess, J. Kim, J. G. Colebatch, and I. S. Curthoys, "Head taps evoke a crossed vestibulo-ocular reflex." in *Neurology (in press)*, December 2006.

[8] D. A. Robinson, "A method of measuring eye movement using a scleral search coil in a magnetic field," *IEEE Trans Biomed Engineering*, pp. 137–145, 1963.

[9] S. T. Moore, T. Haslwanter, I. S. Curthoys, and S. T. Smith, "A geometric basis for measurement of three-dimensional eye position using image processing. vision research," in *Vision Research*, 1996, vol. 36, no. 3, pp. 445–459.

[10] D. Zhu, S. T. Moore, and T. Raphan, "Robust and real-time torsional eye position calculation using a template-matching technique." in *Comput. Methods Programs Biomed.*, 2004, vol. 74, pp. 201–209.

[11] A. Polli, "Active vision: Controlling sound with eye movements," *Leonardo*, vol. 32, no. 5, pp. 405–411, 1999, seventh New York Digital Salon.

[12] A. Hornof, "Bringing to life the musical properties of the eyes," University of Oregon, Tech. Rep., 2008.

[13] A. J. Hornof and K. E. V. Vessey, "The sound of one eye clapping: Tapping an accurate rhythm with eye movements," Computer and Information Science University of Oregon, Tech. Rep., 2011.

[14] J. Kim, "Oculog: Playing with eye movements," in . Nime 07, 2007.

[15] Y. Nishibori and T. Iwai, "Tenori-on," in *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME06), Paris, France*, 2006.

[16] Max for live. [Online]. Available: http://www.ableton.com/maxforlive/

[17] A. Duchowski, *Eye Tracking Methodology: Theory and Practice*, 2nd ed., Springer, Ed., 2007.

[18] openframeworks. [Online]. Available: http://www.openframeworks.cc/

[19] M. Betke, J. Gips, and P. Fleming, "The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities," *IEEE Transactions on neural systems and rehabilitation engineering*, vol. 10, no. 1, p. 110, 2002.

Part VII

Bibliographic references

Bibliography

[1] Grid 2. http://www.sensorysoftware.com/thegrid2.html.

[2] Max for live. http://www.ableton.com/maxforlive/.

[3] R. Bates, H.O. Istance, and S. Vickers. Gaze interaction with virtual on-line communities. *Designing Inclusive Futures, Springer*, pages 149–162, 2008.

[4] Margrit Betke, James Gips, and Peter Fleming. The camera mouse: visual tracking of body features to provide computer access for people with severe disabilities. *IEEE Transactions on neural systems and rehabilitation engineering*, 10(1):1–10, 2002.

[5] Adam Boulanger. *Music, Mind and Health: How Community Change, Diagnosis, and Neuro-rehabilitation can be Targeted During Creative Tasks.* PhD thesis, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, September 2010.

[6] E. Castellina and F. Corno. Accessible web surfing through gaze interaction. In *Proceedings of the 3rd Conference on Communication by Gaze Interaction (COGAIN)*, pages 74–77, 2007. Available at http://www.cogain.org/cogain2007/COGAIN2007Proceedings.pdf (accessed 14 February 2009).

[7] M. Dorr, M. Böhme, T. Martinetz, and E. Barth. Gaze beats mouse: a case study. In *Proceedings of the 3rd Conference on Communication by Gaze Interaction (COGAIN 2007)*, pages 16–19.

[8] Andrew T. Duchowski. *Eye Tracking Methodology: Theory and Practice.* Springer, 2 edition, July 2007.

[9] M. Farbood, E. Pasztor, and K. Jennings. Hyperscore: A graphical sketchpad for novice composers. *IEEE Computer Graphics and Applications*, January-March 2004.

[10] Albert Glinsky. *Theremin: ether music and espionage.* University of Illinois Press, 2000.

[11] Saul Greenberg and Bill Buxton. Usability evaluation considered harmful (some of the time). In *CHI 2008*, Florence, Italy., April 2008,.

[12] A. J. Hornof and A. Cavender. Eyedraw: enabling children with severe motor impairments to draw with their eyes. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 161–170, 2005.

[13] Anthony Hornof. Bringing to life the musical properties of the eyes. Technical report, University of Oregon, 2008.

[14] Anthony J. Hornof and Kyle E. V. Vessey. The sound of one eye clapping: Tapping an accurate rhythm with eye movements. In *Proceedings of the 55nd Annual Meeting of the Human Factors and Ergonomics Society*, 2011.

[15] Anke Huckauf and Mario H. Urbina. Gazing with peyes: Towards a universal input for various applications. In *2008 symposium on Eye tracking research & applications*.

[16] P. Isokoski and B. Martin. Eye tracker input in first person shooter games. In *Proceedings of the 2nd Conference on Communication by Gaze Interaction (COGAIN 2006)*, pages 76–79.

[17] Poika Isokoski, Markus Joos, Oleg Spakov, and Benoît Martin. Gaze controlled games. *Universal Access in the Information Society*, 8(4):323–337, 2009.

[18] R.J.K. Jacob. What you look at is what you get: Eye movement-based interaction techniques. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '90), Seattle, United States*, 1990.

[19] Q. Ji and Z. Zhu. Eye and gaze tracking for interactive graphic display. In *Proceedings of the International Symposium on Smart Graphics*, page 79 – 85, Hawthorne, NY, United States, 2002.

[20] Juno Kim. Oculog: Playing with eye movements. In *Nime '07*, 2007.

[21] C. Lankford. Effective eye-gaze input into windows. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA'00)*, pages 23–27, New York, 2000. ACM Press.

[22] H. Lund and J.P. Hansen. Gaze interaction and access to library collection. *Research and Advanced Technology for Digital Libraries, LNCS, Springer Berlin/Heidelberg*, pages 423–424, 2008.

[23] David MacKay. Dasher - an efficient keyboard alternative. *ACNR*, 3(2), May/June 2003.

[24] Päivi Majaranta. *Text Entry by Eye Gaze*. PhD thesis, University of Tampere, August 2009.

[25] A. Meyer and M. Dittmar. Conception and development of an accessible application for producing images by gaze interaction - eyeart.

[26] Y. Nakano, A. Nakamura, and Y. Kuno. Web browser controlled by eye movements. In *Proceedings of the IASTED International Conference on Advances in Computer Science and Technology (ACST'04)*, pages 93–98.

[27] Yu Nishibori and Toshio Iwai. Tenori-on. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME06), Paris, France*, 2006.

[28] T. Ohno. Features of eye gaze interface for selection tasks. In *Proceedings of the 3rd Asia Pacific Computer-Human Interaction (APCHI'98)*, pages 176–182. IEEE Computer Society, 1998.

[29] Andrea Polli. Active vision: Controlling sound with eye movements. *Leonardo*, 32(5):405–411, 1999. Seventh New York Digital Salon.

[30] M. Porta and M. Turina. Eye-s: a full-screen input modality for pure eye-based communication. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '08)*, pages 27–34, New York, 2008. ACM Press.

[31] J. David Smith and T.C. Nicholas Graham. Use of eye movements for video game control. In *Proceedings of the SIGCHI International Conference on Advances in Computer Entertainment Technology (ACE'06)*, New York. ACM Press.

[32] Zacharias Vamvakousis and Rafael Ramirez. The eyeharp: An eye-tracking based musical instrument. In *Proceedings of 8th Sound and Music Computing Conference*, July 2011.

[33] Martin Stephen van Tonder. *The Development and Evaluation of Gaze Selection Techniques*. PhD thesis, Nelson Mandela Metropolitan University, April 2009.

[34] S. Vickers, R. Bates, and H. Istance. Gazing into a second life: Gazedriven adventures, control barriers, and the need for disability privacy in an online virtual world. In *Proceedings of the 7th International Conference on Disability, Virtual Reality and Associated Technologies (ICDVRAT'08)*, September 2008.

[35] David J. Ward. *Adaptive Computer Interfaces*. PhD thesis, University of Cambridge, November 2001.

[36] C. Ware and Mikaelian H.H. An evaluation of an eye tracker as a device for computer input. In *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface (CHI and GI '87)*, pages 183–188, New York, 1987. ACM Press.

[37] J.O. Wobbrock, J. Rubinstein, M.W. Sawyer, and A.T. Duchowski. Longitudinal evaluation of discrete consecutive gaze gestures for text entry. In *Proceedings of the Symposium on Eye Tracking Research & Applications (ETRA'08)*, pages 11–18, New York. ACM Press.

[38] M. Yamada and Fukuda T. Eye word processor (ewp) and peripheral controller for the als patient. In *IEEE Proceedings Physical Science, Measurement and Instrumentation, Management and Education*, pages 328–330, 1987.