

# Temporal Control In the EyeHarp Gaze-Controlled Musical Interface

Zacharias Vamvakousis  
Universitat Pompeu Fabra  
Roc Boronat 138  
08018 Barcelona, Spain  
zacharias.vamvakousis@upf.edu

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

## ABSTRACT

In this paper we describe the EyeHarp, a new gaze-controlled musical instrument, and the new features we recently added to its design. In particular, we report on the EyeHarp new controls, the arpeggiator, the new remote eye-tracking device, and the EyeHarp capacity to act as a MIDI controller for any VST plugin virtual instrument. We conducted an evaluation of the EyeHarp Temporal accuracy by monitoring 10 users while performing a melody task, and comparing their gaze control accuracy with their accuracy using a computer keyboard. We report on the results of the evaluation.

## Keywords

Eye-tracking systems, music interfaces, gaze interaction

## 1. INTRODUCTION

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 [6]). These systems have been investigated in different domains such as cognitive psychology where eye movement data can help to understand how humans process information, as well as for understanding user-device interaction and for allowing physically disabled people to communicate with a computer using eye movements.

The first person who proposed gaze interaction in a musical context was Leo Theremin. As described by Albert Ginsky in his book “Theremin: ether music and espionage” [1], “...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...”. Andrea Polli in 1997 [11] developed a system which allowed performers to access a grid of nine words spoken by a single human voice by making saccadic movements in nine different directions. Hornof et al. [2], after abandoning the idea of the “EyePiano”, developed more interactive tools using Storyboarding. Kim et al. implemented Oculog [8]. The gaze’s position was mapped to PureData for generating and interacting with four sequences. A more detailed survey of existing gaze controlled musical interfaces can be found in [12]. However, none of these approaches allow the

user to perform even simple melodies in real time since there is no control over the basic parameters of common musical instruments such as pitch, timbre, articulation, or harmony.

In this paper, we describe the EyeHarp, a new real-time gaze-controlled musical instrument, focusing on the new features we recently added to its design. The EyeHarp gaze-controlled musical interface aims to provide similar expressive potentials to common musical instruments. In particular, we report on the EyeHarp new controls, the arpeggiator, the new remote eye-tracking device, and the EyeHarp capacity to act as a MIDI controller for any VST plugin virtual instrument.

The use of eye-tracking systems for controlling musical instruments raises a number of challenges and opportunities. Since in such scenarios the 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. The system should avoid the “Midas touch” problem [6], wherein all objects viewed are unintentionally selected. Furthermore, temporal control in eye-tracking-based musical instruments is of paramount importance. The user should be able to play a melody in tempo, with minimum latency. The EyeHarp intends to tackle these issues and advance the state-of-the-art in gaze-based musical instruments.

The rest of the paper is organized as follows: Section 2 describes existent gaze selection techniques. Section 3 describes the EyeHarp, in particular the new eye-tracking device and graphical interface. In Section 4 we evaluate the temporal control provided by the EyeHarp, and finally Section 5 presents some conclusions and future work.

## 2. GAZE SELECTION TECHNIQUES

In this section we overview the main selection techniques used in gaze controlled applications. We focus on techniques using solely the user’s gaze, i.e. we don’t consider other methods for clicking, for instance using the lips or hands.

*Blinking.* With blink-based selection, the user looks at a target object and blinks to select it. Humans blink involuntary every few seconds which would result in unintentional selections. Proposed solutions to this problem include prolonged blinking [9] and three deliberate blinks to indicate a selection [7]. Both methods are considered to be unnatural and introduce a big response time.

*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,

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

NIME’12, May 21 – 23, 2012, University of Michigan, Ann Arbor.  
Copyright remains with the author(s).

apart from anything related to playing real time melodies where the *screen button* method (see below) is used.

**Gaze gestures.** In this 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).

**Special Selection Area or Screen Button.** Another solution to the Midas touch problem is to use a special selection area [10]. For example, in the “quick glance” method developed by Ohno (1998), each object that could be selected was divided into two areas: command name area 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 can glance directly at the selection area associated with that command. With this method, it is possible to obtain better temporal control allowing to control the exact timing of selection events. This is of special interest in music applications where it is important that the user is able to play in tempo. Placing each command name and selection area in slices around a circle or “pie” results in the Pie menus selection technique.

**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. Anke Huckauf et al., in 2008 introduced pEYES [4], where pie menus were suggested for gaze control. A slice opens without dwell times by looking in the outer selection area of a slice. A slice can lead to another pie menu. The advantage of this selection technique for playing melodies in real time is that it allows better temporal control: the command is sent the exact moment that the user looks at the outer region of a slice. Between playing two notes, the user can have a fixation inside the internal neutral command area of the pie. This helps in avoiding accidental notes (Mida’s touch) and thus no Dwell time is necessary. Fixations detection algorithms can be optionally activated. A one-level “pEye” menu is used in the EyeHarp for playing melodies.

### 3. THE EYEHARP

The EyeHarp is an open source gaze controlled musical interface [13] implemented in OpenFrameworks open source C++ toolkit. To access the source and binaries visit <http://code.google.com/p/eyeharp/>. At the moment there are implementations for windows and linux systems. Probably the best way to demonstrate its functionality is to watch a video demonstration with explanatory annotations at <http://theeyeharp.blogspot.com/>. In this section we describe the main components of the EyeHarp with emphasis in its new features. A more extensive description of the functionality of the EyeHarp can be found in [12].

#### 3.1 Eye-Tracking Device

The eye-tracking device (figure 1) used for evaluating the EyeHarp consists of a modified PS3 eye camera and two infra-red illuminators. The infra-red cut-off filter of the PS3 eye camera was replaced with infra-red pass filter and its lens was replaced by a 16mm m12 lens. This setup was used along with the ITU Gaze Tracker open source software [5]. This software is computing the user’s gaze by detecting the center of the pupil and the corneal reflec-

tions of the infra-red illuminators. Both monocular and binocular remote eye tracking are supported. The term *remote* implies that both camera and infra-red illuminators are placed on the desk (usually under the computer screen). An alternative solution is a head-mounted eye-tracker. The advantage of the corneal reflections remote eye tracking technique is that slight head movements are possible without affecting the accuracy of the system. Using a head mounted eye-tracker it was impossible to conduct a proper evaluation, as with slight head movements the calibration was lost [13]. The cost of building the remote eye-tracking device varies from 50 to 100 euros, depending on the delivery cost of the components. More details about how to build such an eye-tracker can be found online at: <http://www.gazegroup.org/forum/>

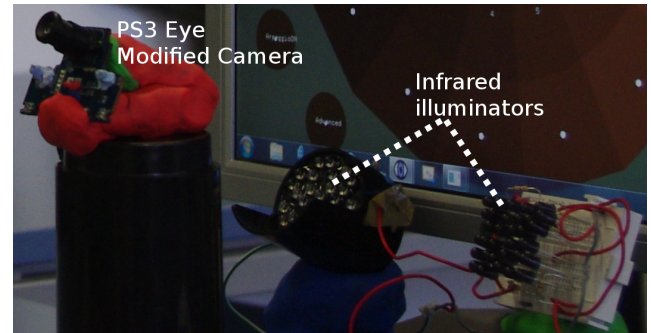


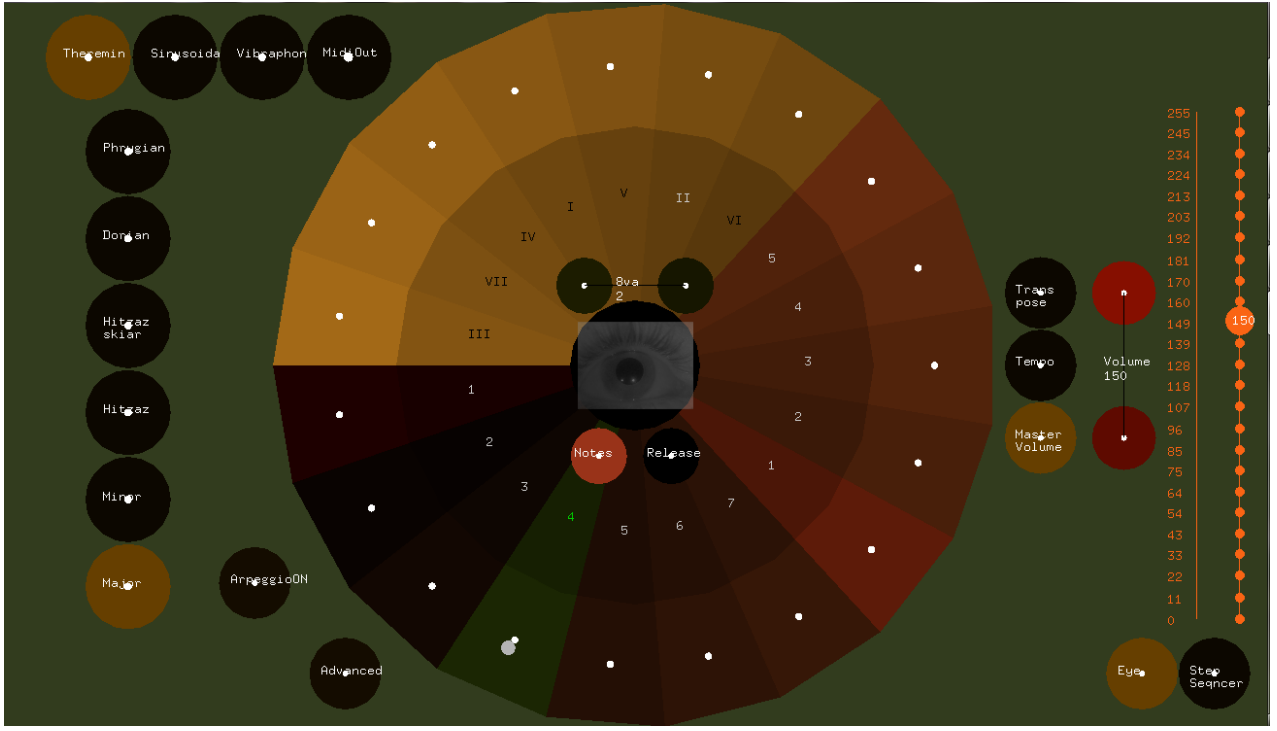
Figure 1: The Eye Tracker used for the experiment

#### 3.2 The EyeHarp Interface

The EyeHarp interface consists of three basic layers: (i) The pEYE menu, (ii) the Step Sequencer, (iii) The Arpeggiator. The last two are used for building a rhythmic and harmonic background in the musical composition, while the pEYE menu for playing melodies and changing the chords in real time. The EyeHarp also consists of various controls specifically designed for gaze input. These controls can be assigned to local parameters of each layer separately (like the timbre, volume, pitch range of each layer) or global features of the composition (like scale, transposition, tempo). Focus points are placed at the center of each control. This technique is necessary whenever we use the gaze as a pointing input.

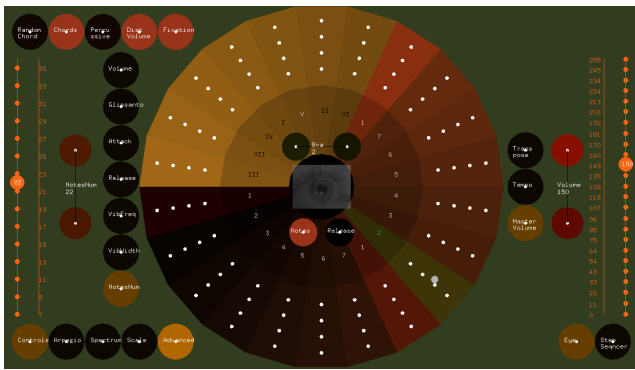
##### 3.2.1 The EyeHarp Peye

The Peye is displayed in figure 2. Initially, the EyeHarp design included an EyePiano interface for playing melodies similar to the one introduced by Hornof et al. [2]. The interface implemented by Hornof et al. consisted of a fixation-based eye-controlled piano in which users played notes by looking at particular keys in the screen for a short period of time. However, they reported that users were only slightly able to improve their ability to move to the intended piano keys and were not at all able to improve their rhythmic accuracy. In order to address these problems we introduced the screen button gaze selection technique into our EyePiano interface (figure 4). The resulting interface consisted of two piano keyboards (each representing a different octave), one at the top of the screen and another at the bottom. The area in the middle is the “command area”, where no notes are triggered. When the user wants to trigger a note, he first looks at the desired note in the command area and then he looks up or down, depending on the desired note. This improved interface allows better spatial and temporal accuracy when playing a melody.



**Figure 2: The Peye for playing real time melodies.**

After experimenting with the EyePiano interface, we decided to further improve it by replacing the piano interface by a Peye menu: we positioned the notes at the periphery of a circle. In addition, we converted the instrument to a diatonic one which allowed us to have more space for other interface controls. In a gaze controlled interface big buttons are required, as the accuracy of the eye tracking system might not be sufficient. In the new diatonic interface, users determine the musical mode before starting to play. Once this is done, only the notes included in the selected scale appear in the Peye menu. If the distance volume switch (figure 6-a) is active, then 4 focus points appear in the selection area of each note (figure 3). The closer to the Peye perimeter the the gaze is, the louder and more vibrato in the the note.



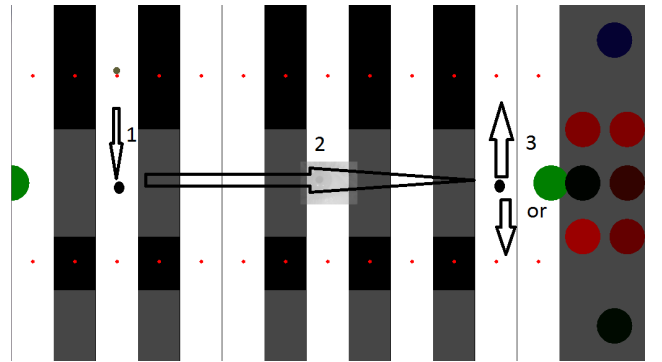
**Figure 3: When distance volume is active, 4 focus points appear at the selection area of each note. The further out the gaze of the user is, the louder and with more vibrato the note will sound.**

The user can choose between some pre-set sounds, build his own timbre by changing the parameters of the additive synthesis or just use the EyeHarp as a midi controller in

combination with any sample bank by selecting the midiOut tab (figure 2 top left corner). An additional virtual midi driver software (such as LoopBe), a vst virtual instrument host application (such as reaper) are required to connect the EyeHarp with a vst virtual instrument.

The notes mapped to each slice can be assigned automatically by choosing one of the pre-set scales in the basic mode (figure 2) or can be assigned manually and thus create any possible 12-semitone musical mode (figure 6-d).

If the “chord” button is activated, then the last 7 slices of the pEYE menu are dedicated for changing the chords of the composition (figure 2). The chords are placed in a order of fifths. That way, the most common chords (IV, I, V) are placed in the middle and thus are more easily accessible while playing melodies.



**Figure 4: Improved version of the EyePiano**

### 3.2.2 Step Sequencer

Figure 5 displays the implemented Step Sequencer. The notes assigned to the sequencer belong to the pre-determined scale. Moving from the bottom to the top of the screen we go up in the pitch. Consecutively the columns of the grid are flashing, indicating the current position in the sequencer

loop. The user can activate a note in the grid by looking at it for about a second. In the case of the step sequencer the temporal accuracy is not critical. For deselecting a note the user looks again at the same note for one second. The number of notes on the grid, the octave, the scale, and the tempo can be changed using the controls described in the following sections.

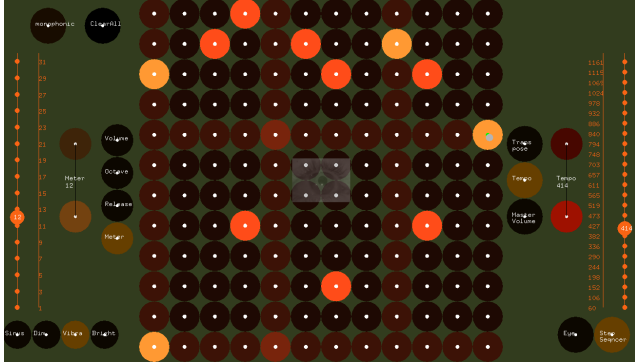


Figure 5: The dwell-based step sequencer

### 3.2.3 EyeHarp Controls

In order to set the various parameters of the EyeHarp interface, explicit controls for gaze input were designed.

#### Sliders.

On both, the left and right hand side of figure 5 two orange sliders are shown. Users can set the desired value without clicking by looking at the desired value in the command area (nothing is triggered in that region), and then gazing at the actual slider. A short dwell time is applied on these focus points in order to avoid setting values accidentally. The screen button technique is applied in this case.

#### 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 this case. 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. The set value is displayed in a area between the buttons. The slider control interface is more appropriate for setting distant values, 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 6 the user can assign the slider on the left to any of the available controls determined by the 2 or 3 levels of multiplexing determined by the tabs.

#### Switches.

In order to activate / deactivate a particular function, switch buttons are used. Their color is dark when they are not activated, and light when they are activated. Dwell

time is applied to activate / deactivate the switches. All the notes in the Step Sequencer are *switches*. When looking at a switch its focus point placed in the middle of it, turns green, indicating that if the user continues to look at it, it will turn on.

### 3.2.4 Arpeggiator

Figure 6-b shows the available controls of the arpeggiator. Up to four arpeggios can be generated by setting parameters for each of them. When sounding all together, they provided a rich harmonic and rhythmic background to the composition. Chords can be changed in the pEye. The arpeggiator parameters are:

*Starting Note:* the first note of the arpeggio. The value '0' corresponds to A1 (55Hz).

*Meter:* The total notes of the arpeggio.

*Notes Included:* if the value of this parameter 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. Up to seven notes (all the notes of the scale) can be included in this order: I, V, III, VII, II, IV, VI.

*Pattern Size:* Each arpeggio is formed by repeated patterns. This variable determines how many notes are forming this pattern.

*Pattern Step:* Together with the 'notes Included' this parameter determines the intervals between the notes in the pattern.

*Global Step:* Together with the 'notes Included' this parameter determines the intervals between the starting notes of the patterns.

*Volume:* The volume of each arpeggio.

## 4. EVALUATION

The hypothesis made when using a Peye menu for playing melodies in real time was that the screen button gaze selection technique used in Peye menus would give optimal temporal control. In the current system, the user decides to play a note with just one saccade eye movement. Saccadic movements are very fast eye movements. In fact the results of the experiment we performed indicate that users tend to play the notes earlier than they intended to.

Ten users participated in the experiment. All users had a minimum of 5 years of musical training. First they were asked to perform 3 ascending scales in the EyeHarp interface. Between every performed scale they practised for 2 minutes in the EyeHarp interface trying to play in tempo while they were asked to look at the command area of every note (that is the number corresponding to this note) before triggering that note. As this area is closer to the selection area, according to the Fitts's law this should improve their performance. Although some users preferred to look at the centre of the interface before performing each note. Afterwards they were asked to play an octave interval for 40 times. These notes were placed diametrically in the Peye menu, so the distance between the notes is the farthest possible. The tempo in both cases was set to 60bps and each note had to be triggered every 2 beats (every 2 seconds). The data were recorded with the same frame rate as the EyeHarp running (60fps). A simple arpeggio along with a flash on the computer screen was providing the users with the sense of the rhythm. Afterwards they were asked to perform the same task using the computer's keyboard. In figure 7 we can see the results of the gaze input, while in figure 8 the results of the keyboard input in the case the ascending scale measurement.

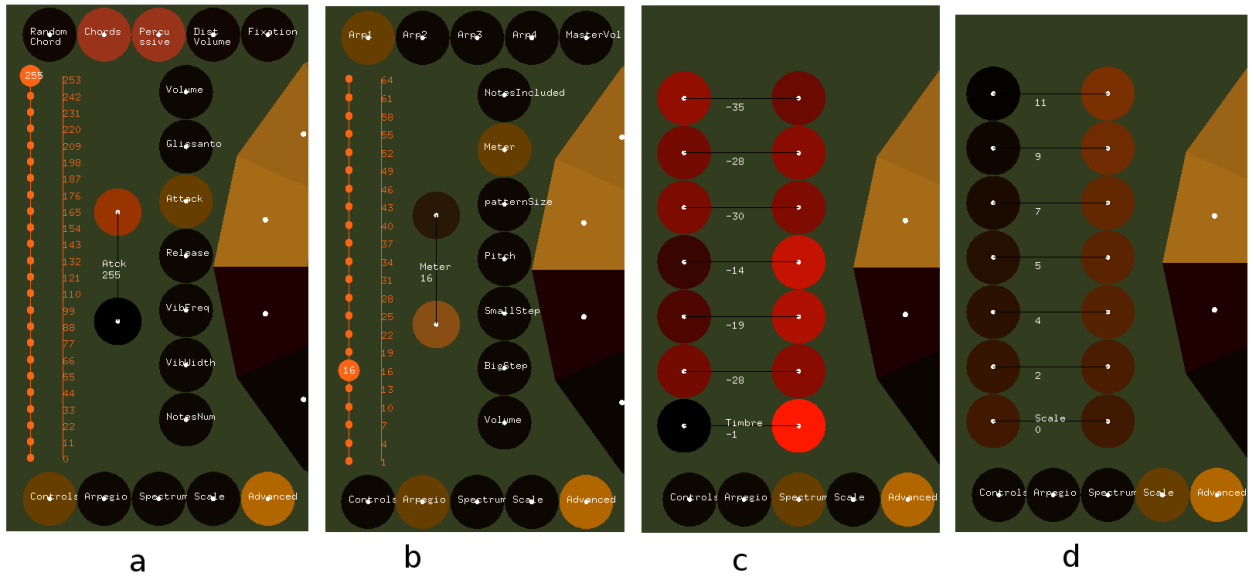


Figure 6: When the advanced switch is active, the advanced controls appear. An horizontal tab at the bottom determines the first level of multiplexing. Depending on the button selected there, another vertical tab might appear. This tab assigns the slider and repeat button to the desired control. In the case of the Arpeggiator (b) one more tab appears at the top for determining which of the 4 available arpeggios is to be modified. In the case of the general controls (a), a few switch controls appear at the top.

- In the case of Gaze input the commands are triggered earlier. This confirms the results reported by Hornof et al. [3], where in the case of midline saccade detection algorithm (which is similar to what was applied in our case as well - the note is triggered once the gaze enters the selection area without any dwell time) the users played the notes 57ms earlier (when asked to play 1 note every second).
- When performing a scale -where the notes are close to each other- with gaze input, the average asynchrony measured was -94ms. When performing at octave interval -where the distance between the notes is longer- the average asynchrony was -46ms. This indicates that the closer the target, the earlier the event is triggered. Applying an appropriate fixation detection algorithm may improve this asynchrony.
- The users improved their performance through practice. In the first attempt the variance is very high, while it falls in the second and third attempt.
- In figure 9 we can observe each users' overall asynchrony for gaze and keyboard input. The size of each circle corresponds to the average variance of each user. An ideal performance is one with zero variance and zero asynchrony. Most users' average asynchrony has a negative value for gaze input and a positive value for keyboard input.

## 5. CONCLUSIONS AND FUTURE WORK

We have described the EyeHarp, a new gaze controlled musical interface for controlling the melodic, harmonic and expressive aspects of a musical composition in real time. In particular, we have described the new features of the EyeHarp, such as its new controls, the arpeggiator, and the new remote eye-tracking device. We have conducted an evaluation to assess the EyeHarp Temporal accuracy using

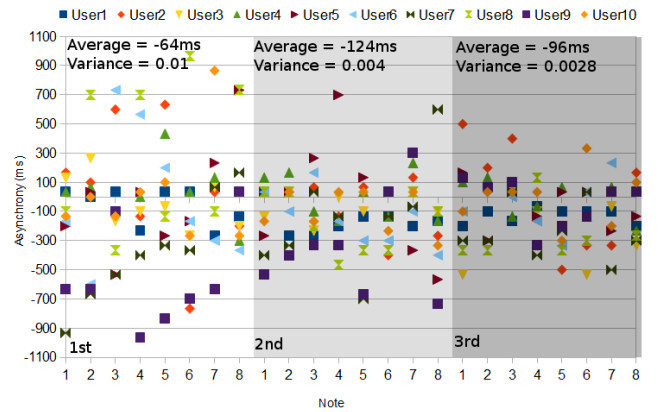


Figure 7: Performance with gaze input. Users practised for 2 minutes before performing each of the 3 scales.

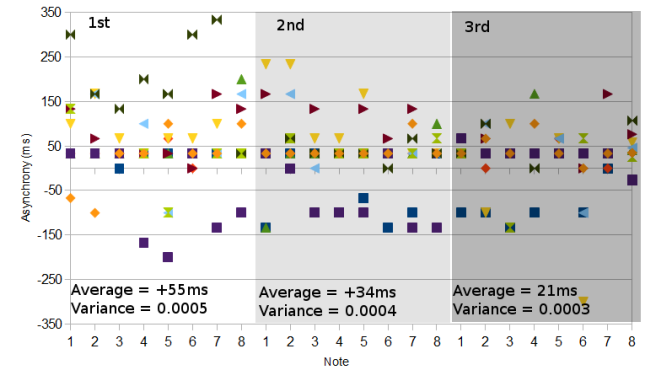
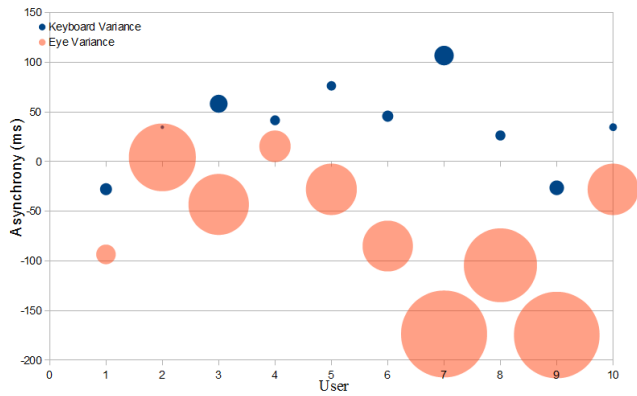


Figure 8: Performance with keyboard input.





**Figure 9: Comparing Gaze and Keyboard performance for each user.**

the pEYE menu. The results show that the user's temporal accuracy improves with practice, as it is the case with traditional musical instruments. Motivated by the results obtained, we plan to apply a fixation detection algorithm that would depend on the distance to the target note. We think this could improve the negative asynchrony observed in the evaluation. We plan to conduct a more extensive evaluation using various fixation detection algorithms in order to confirm this hypothesis.

## 6. REFERENCES

- [1] A. Glinsky. *Theremin: ether music and espionage*. University of Illinois Press, 2000.
- [2] A. Hornof. Bringing to life the musical properties of the eyes. Technical report, University of Oregon, 2008.
- [3] A. J. Hornof and K. 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.
- [4] A. Huckauf and M. H. Urbina. Gazing with peyes: Towards a universal input for various applications. In *2008 symposium on Eye tracking research & applications*.
- [5] S. A. J., H. Skovsgaar, H. J.P., and H. D.W. Low-cost gaze interaction: ready to deliver the promises. In *CHI 2009, ACM Press (2009)*, page 4453–4458.
- [6] R. J. K. Jacob and K. S. Karn. Eye tracking in humancomputer interaction and usability research: Ready to deliver the promises. In . H. D. J. Hyona, R. Radach, editor, *The Mind's Eyes: Cognitive and Applied Aspects of Eye Movements*. Elsevier Science, pages 573–605. 2003.
- [7] 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.
- [8] J. Kim. Oculog: Playing with eye movements. In *Nime 2007*, 2007.
- [9] 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.
- [10] 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.
- [11] A. Polli. Active vision: Controlling sound with eye movements. *Leonardo*, 32(5):405–411, 1999. Seventh New York Digital Salon.
- [12] Z. Vamvakousis. The eyeharp: A gaze-controlled musical instrument. Master's thesis, Universitat Pompeu Fabra, 2011.
- [13] Z. Vamvakousis and R. Ramirez. The eyeharp: An eye-tracking-based musical instrument. In *8th Sound and Music Computing Conference*, July 2011.