HANDMONIZER: A VOCAL IMPROVIZATION TOOL FOR LIVE PERFORMANCES

Pappas Antonios, Lionetti Davide

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano Piazza Leonardo Da Vinci 32, 20122 Milano, Italy [[antonio1.papa,davide1.lionetti]@mail.polimi.it](mailto:@mail.polimi.it)

ABSTRACT

The rapid evolution in technology has found its way tointroducing novelty in today’s live music performances. Music Interaction Design is a field dedicated to the enhancement of musical creativity by introducing new interaction methods between the performer and the system. This paper presents the development of an interactive instrument for a musical performance adapted to the needs of a specific singer. A key component of our work is the combination of hand motion recognition and audio signal processing which is regulated by this type of interaction. We describe the development methodology, but we also focus on our collaboration with a well-known singer to conceptualize and then refine this tool until the development of the final product. At the end of this report, we defined an evaluation strategy, collecting feedback with a questionnaire addressed to the singer. The aim of the latter is to help other engineers that would like to develop cutting-edge technologies working alongside artists.

Index Terms— Music interaction design, artist interaction, hand gesture recognition, vocal harmonizer, live performance

1. INTRODUCTION

The whole way of interacting with machines is changing, since gestures, movements, and direct graphic manipulation are co-existing with physical interfaces like keys, buttons, and knobs. In the musical context, these new interface technologies have given us countless possibilities in the creation of Digital Musical Instruments (DMI) or New Interfaces for Musical Expression (NIME), artifacts that connect inputs (interface controllers) and outputs (sound synthesis modules) according to a mapping strategy [1]. A DMI is composed of a control interface and a sound synthesizer, which work separately [2]. d[3].

In the case of a DMI, the design of the Human-Computer Interaction (HCI) is important to give a sense of musical meaning not only to the artist but also to the audience. The main goal indesigning such a system is to make HCI and software engineering sides work together seamlessly [4].Previous examples of similar interactive systems have been developed in the past few years (e.g.: [5], [6]).

In our specific case, we are supposed to design a DMI performance tool tailored to a specific artist, the well-known Italian jazz

singer, Maria Pia De Vito1. This scenario allows us to be creative, taking into consideration the artist’s needs based on the nature of her live performance. The main goal is to build a custom harmonizer to enhance her solo performance, a DMI with which she can interact creatively and intuitively. Collaborating and discussing her necessities with her, we develop the Handmonizer, an unusual and special har- monizer monitored in real-time by hand gestures. Just by moving her hand she can explore the different screen areas each associated to a specific setting, creating different harmonic combinations by exploiting the smooth transition between the different voices. At the same time, she can control different effects using hand gestures in an intuitive way for her and the audience. To achieve this effectively, the artist should only focus on her musical intention while performing, instead of having to think about switching between physical devices that slow the flow of her performance.

In this paper we first describe our interaction with the artist in order to set the goals as well as our approach towards this de- sign. Then we explain the technical aspects of our system, going through the tools and methodologies we used to develop the fi- nal product. Finally, we present the evaluation given by the artist through a questionnaire after experimenting with the Handmonizer during two demo sessions where she provided her detailed feedback on every aspect of her experience with this tool. The full source code is available on Git Hub in our repository.2

2. ARTIST ORIENTED SYSTEM DESIGN

The Handmonizer has been designed to cover the desires of the jazz singer Maria Pia De Vito and is aimed to be used during her vo- cal solo improvisation. In this section, we introduce the develop- ment process based on the artist’s constraints, the discussions and the decisions taken based on her feedback. This system consists of a vocal harmonizer and a hand gesture recognition interface able to map some parameters in real-time based on the movements of the singer’s hand.

2.1. The Artist needs

During our first meeting with the artist, we were presented with the challenges she was facing with her previous setup and her vision of her ideal performance scenario.

Her previous setup is made up of an Eventide guitar pedal harmonizer and an Echoplex looping machine to create the backing tracks along with which she used to improvise3. The main issues she faced with this setup were

1[Maria Pia De Vito Wikipedia page](https://it.wikipedia.org/wiki/Maria_Pia_De_Vito)

2[GitHub repository containing all the code](https://github.com/EllDy96/Handmonizer)

3[Maria pia de vito solo performance](https://www.youtube.com/watch?v=MFBWcX-iHVo)

related to the unnatural and mechanic interaction, changing the var- ious patches using switches is not a quick and comfortable task that reduce her creativity. At the same time, dealing with physical de-vices like the Eventide leads to clicking noises caused by switches and knobs that disrupt the musical flow.

The main issue we mutually decided to consider was developing an interface that would be easy for her to use so that she can focus mainly on her improvisation without having to struggle with the in- teraction. We decided to implement a vocal harmonizer which gives her the opportunity to explore and experiment with the different har- monic configurations in a natural way, just moving her hand in front of a camera.

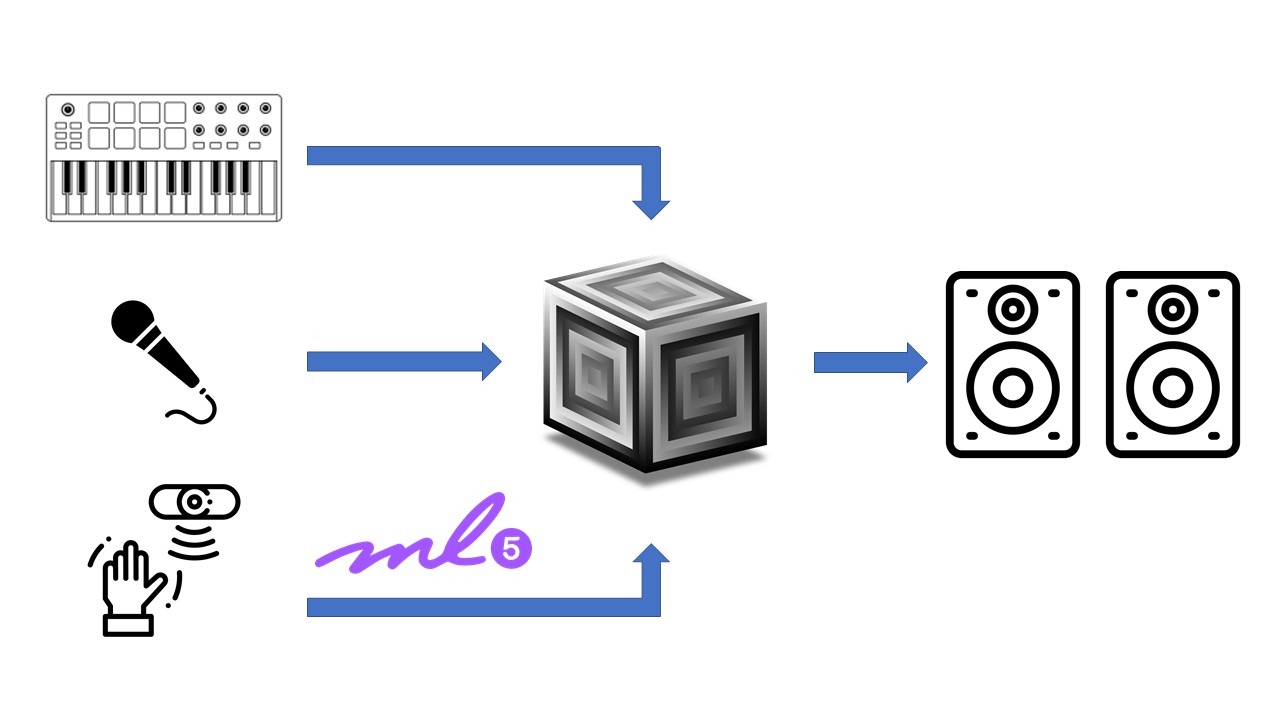


Figure 1: The Handmonizer signal flow

2.2. Interaction Design approach

Due to the separation between gesture input and sound output, the mapping strategies are the essence of digital musical instruments. Different mapping strategies for the same set of inputs and outputs affect how the performer reacts musically and psychologically to the instrument [1]. In general, while designing a DMI we need to follow three main principles [7]:

• Control: The way the interaction is mapped to the output, should give full control to the artist who is using the system. In other words, the output should exactly matchs the intention of the user.

• Legibility: also referred to as ”transparency”, this is related to the importance of having a mapping that is easily understood by both the artist and the audience. Whatever visual clue is pre- sented to the audience by the artist through gestures, should be translated to an audible output that they can easily distinguish.

• Sound: determining what kind of sounds the mapping is going to facilitate making. This could mean creating new sounds, processing an input sound or even both.

Moreover, when having to design an interface tailored to a spe- cific artist, we as developer should try and put ourselves in the posi- tion of the artist in order to develop something meaningful for her. A good starting point to decide on the gesture-sound mapping is to use a bottom-up approach. We start with simple interactions that trigger a certain response and at the same time think of how we can

and sound is not only an issue for the musicians, but also for the audience. Since the gestures and the produced sound have not necessarily a direct connection, the audience may not understand what is happening, and consequently, does not feel engaged during the performance. [1] [6].

3. HANDMONIZER IMPLEMENTATION

The Handmonizer structure is based on a Node server that allows the communication between the two main parts: the hand ges- ture recognition and the real time sound harmonization algorithm which communicate through OSC messages. Moreover, a MIDI controller is connected to the harmonizer to switch between dif- ferent patches during the live performance. In this section we first describe each component separately and finally their communica- tion and parameter mapping.

3.1. Hand gesture recognition

The harmonizer can be controlled through hand gestures captured from a webcam. For the hand pose recognition, we use one pre- trained model from ml5.js 4, a JavaScript framework for creative coding built on top of TensorFlow.js that allows the use of GPU- accelerated machine learning algorithms in a web browser. The TensorFlow ecosystem provides an easy-to-use tool to convert pre- trained ML models trained in Python or C++ into web targets. Some of these models were specifically designed for creative applica- tionsto facilitate the development of real-time music re- lated Web-application [8].

We use the model called Handpose5 that performs hand- skeleton finger tracking. It takes the video stream frame by frame and returns the coordinates of 21 hand keypoints over the palm of the hand as shown in Figure 2. This process is GPU intensive. To achieve the best performance a system with a dedicated GPU is ad- vised. Based on the coordinates of these 21 points we compute three main parameters

• The hand centroid: displayed as the light-green central dot is the arithmetic mean position of the 21 dots.

• The palm length: defined as the distance between the tip of the middle finger and the base of the palm (displayed as the length of the white line)

• The hand orientation: computed as the slope of the white lines between [0, π].

With a custom mapping of that three parameters we define 5 hand gestures that the user can use to change the harmonizer settings in real time. We retrieve the two coordinates of the centroid in the screen xc, yc and we map each to a specific parameter. The palm length is used to implement two gestures. The user could close the hand to fade out the harmonics or we could move the hand farther or closer to the screen to modify the volumes of the harmonized voices. In that way we are able to also map the z-position of the hand in the space. The slope of the palm controls the amount of effect (Reverb or Delay) that the user wants to add to the voice, as a dry-wet knob; when the white line is perpendicular to the bottom border the voice has no effect, while it reaches its maximum value when the line is parallel to the lower border.

make this interaction meaningful for the audience. Audience en-

engagement is strongly related to the mapping problem, the creation of meaningful and perceivable connections between human action

4ml5.js: Friendly machine learning for the Web. <https://ml5js.org>

5[https://learn.ml5js.org//reference/handpose](https://learn.ml5js.org/)



Figure 2: The hand is mapped by Handpose model into 21 points

3.2. Harmonizer implementation

The audio signal processing part of the Handmonizer is developed entirely on SuperCollider. The algorithm of the harmonizer is com- posed of separate pieces of code, including pitch tracking and shift- ing, effects and communication protocol definitions (OSC, MIDI) for parameter mapping. Here we explain each component sepa- rately.

3.2.1. Pitch tracking and shifting

The very first part of the harmonizer aims at recognizing the signal input. Since we need to perform pitch shifting in order to create harmonic voices, the first thing we need to do is track the funda- mental frequency of the input audio signal. For this purpose we use an external class called Tartini instead of the standard Pitch class from SuperCollider since it performs the pitch tracking more pre- cisely and in a shorter amount of time [9].

To avoid harmonies that sound very unnatural, the pitch shifting needs to be performed using the Pitch-Synchronous Overlap and Add (PSOLA). One of the main advantages of this method is the preservation of the formant positions (spectral envelope) which al- lows us to keep the original timber [10]. Here we use a SuperCol- lider class called PitchShiftPA which is based on PSOLA. We pass the fundamental frequency value tracked by Tartini as well as the pitch shift ration which we fix depending on the harmony we want to achieve. The same procedure is used to generate three harmonic voices, both higher and lower than the input signal (six voices in total). The harmonic voices are mixed together in a sep- arate channel so they can be processed without affecting the lead voice.

3.2.2. Patches and effects

As mentioned previously, one of the main features needed by the artist is the possibility to switch between the following chord types:

• 3rd - 5th

• 4th - 5th - 7th

• 4th - augmented 4th - 7th

• Octaver

These patches include a delay effect that can also be controlled by the user. The harmonizer also includes two patches that allow the artist to control only the effect amount for the reverb and delay with- out having any harmony. These two patches can be used for single voice improvisations over looped backing tracks.

3.2.3. Cross-fading

A special feature of the Handmonizer is the smooth transition be- tween the voices, creating different patterns during the performance without have any sudden jumps in volume level. Here we manip- ulate the different amplitudes of each voice group before mixing them. A similar procedure is used for the volume fader of the har- monies for which we use a dB scale mapping for a more natural level perception. Finally, we use XFade2 as a cross-fade knob to control the dry/wet ratio for the reverb and delay effects.

3.2.4. In-scale harmonizer

Finally, we developed a second version in a separate script file. In- stead of fixed intervals, the Handmonizer can be used as a classic harmonizer following a specific scale, where the user can set the key and scale type (major, minor, etc). By hard-coding the first MIDI note for each key, we use an external class called MiscFuncs to retrieve the array of MIDI notes for the selected scale. Then to retrieve in real time the precise MIDI note sang by the singer, we use another external class called MyKFiddle. Finally, the algorithm checks if the input note is part of the scale. If this is the case, it computes the pitch ratio and feeds it to the pitch shifter.

The two classes mentioned above, are developed by Matthew Yee King6 and slightly modified by us to be better adapted to our purpose.

3.3. Communication protocols and architecture

To switch between patches, we use a MIDI controller where we assign each pad to a patch by changing the necessary parameters. In addition to the patches mentioned above, we use one pad as an ON/OFF toggle button and another pad as a bypass for the harmonic voices.

The user interface is hosted as a web page/application in an Express server, the connection is set up through the framework Socket.io. All the control parameters mentioned above are computed in the client and then sent to the server. From the server, the parameters are written in OSC messages and forwarded to SuperCollider.

The hand motion recognition features are sent to SuperCollider as OSC messages in real time and are used to control different pa- rameters that define the harmonizer’s performance. We use the x- coordinate of the palm centroid to add more voices as we move from left to right. We can imagine the screen divided into three columns where on the first we only have one additional voice and every time we visit the next column we add one more voice. Similarly, we map the y-coordinate as a switch between low octave and high oc- tave harmonics. We can imagine the screen divided into two rows where the upper row represents the high octave harmonics and the bottom row represents the low octave harmonics. All these changes in number of voices and octaves are performed in a smooth way as explained previously, so the artist can explore different sounds.

6[Matthew Yee King Supercollider collection of classes](https://github.com/yeeking/myksupercollider)

Another feature that we use is the palm length represented by the white line in Figure 1. We have mapped this feature to the harmony fader using a dB scale to control the volume of the harmonic voices. There are two ways to exploit this feature. The first and most intu- itive way is to open and close our hand and the second is to move our hand back and forth. If the artist wants to emphasize the har- monic voices she can simply move her hand closer to the camera. Finally, we use the hand orientation as an imaginary knob that con- trols the dry/wet level of the reverb or delay effects. When we keep our hand straight we have a fully dry signal (e.g.: no effect). While we rotate our hand either left or right we add the amount of the wet signal and decrease the amount of the dry signal using a cross-fade effect.

4. TESTING AND EVALUATION

For the final evaluation, we take into consideration the final singer’s performance basing on the specific initial constraints and we create a evaluation questionnaire following the typicalDMI evaluation methodologies described on [8]. During the Demo we can notice that the artist learns in a fast way how to inter- act with the tool and in a few trials she is able to come up with very interesting vocal solos. She is not bothered by the interaction, instead she is stimulated to find new ideas and patterns by moving freely her hand. To collect the stakeholder perspective, we ask her to evaluate our tool with some questions giving a grade from 1 to

10 and providing a short comment.

• playability: How much control do you feel that you have over the tool while using it?

• learnability : How easy was it to learn how to use it?

• expressiveness : How much does it help you to enhance your creativity and express your musical intention?

• enjoyability : how enjoyable it is your experience while using the handmonizer?

• novelty: how much novelty does it introduce in your perfor- mance?

• effectiveness: how much does this tool manage to solve the issue with your previous setup?

• Sound quality: Rate the sound quality

• give like a thumbs up a subscribe to out YT channel if you like it!

• feedback from the first workshop

• feedback from the second workshop, concert simulation,

• questionnaire/evaluation from the artist

5. CONCLUSIONS AND FUTURE WORK

We have explained in detail how we decided to approach the whole design process keeping into account the feedback from the artist in each step. We have also explained the methodology we used includ- ing what available technology we exploited and how we created the architecture of our Handmonizer. We have shown that in the end we managed to create a fully working harmonizer that interacts effectively with the control interface. However, there are a number of improvements to make the experience even smoother. Instead of the ml5.js framework, leap motion could be used for a faster and



Figure 3: Workshop demo

smoother interaction. Another issue is the fact that it is almost im- possible for the artist to run the application on her own. To solve this issue we should consider the setup and running of the tool as a task that will be carried out by a technician or engineer before and during the performance. Finally, a slight latency especially while using the delay can prevent the artist from looping her voice in a rhythmical way, which is something that should be considered.

6. ACKNOWLEDGMENTS

You can put acknowledgment in the 5th page, the extra page, counting it as the references

The preferred spelling of the word acknowledgment in America is without an “e” after the “g.” Try to avoid the stilted expression, “One of us (R. B. G.) thanks ...” Instead, try “R.B.G. thanks ...” Put sponsor acknowledgments in the unnumbered footnote on the first page.

• demos descriptions in details

• check the comments in the superCollider code

• write the readMe

• fulfil the questionnaire

• do a schematic of the whole architecture

• acknowledgements

• abstract

7. REFERENCES

[1] R. Medeiros, F. Calegario, G. Cabral, and G. Ramalho, “Chal- lenges in designing new interfaces for musical expression,” in International Conference of Design, User Experience, and Usability. Springer, 2014, pp. 643–652.

[2] E. R. Miranda and M. M. Wanderley, “New digital musical in- struments: Control and interaction beyond the keyboard (com- puter music and digital audio series),” 2006.

[3] I. Bergstrom, A. Steed, and R. Lotto, “Mutable mapping: gradual re-routing of osc control data as a form of artistic per- formance,” 01 2009, pp. 290–293.

[4] J. Borchers and M. Muhlhauser, “Design patterns for inter- active musical systems,” IEEE MultiMedia, vol. 5, no. 3, pp.

36–46, 1998.

[5] Y. Ikawa and A. Matsuura, “Playful audio-visual interac- tion with spheroids,” in Proceedings of the International Conference on New Interfaces for Musical Expression, R. Michon and F. Schroeder, Eds. Birmingham, UK: Birmingham City University, July 2020, pp. 188–189. [Online]. Available: [https://www.nime.org/proceedings/2020/](https://www.nime.org/proceedings/2020/nime2020_paper36.pdf) [nime2020 paper36.pdf](https://www.nime.org/proceedings/2020/nime2020_paper36.pdf)

[6] J. Leonard and A. Giomi, “Towards an interactive model- based sonification of hand gesture for dance performance,” in Proceedings of the International Conference on New Inter- faces for Musical Expression, R. Michon and F. Schroeder, Eds. Birmingham, UK: Birmingham City University, July 2020, pp. 369–374. [Online]. Available: [https:](https://www.nime.org/proceedings/2020/nime2020_paper72.pdf)

[//www.nime.org/proceedings/2020/nime2020 paper72.pdf](https://www.nime.org/proceedings/2020/nime2020_paper72.pdf)

[7] T. West, B. Caramiaux, S. Huot, and M. M. Wan- derley, “Making Mappings: Design Criteria for Live Performance,” in NIME 2021, may 3 2021, https://nime.pubpub.org/pub/f1ueovwv.

[8] A. Correya, P. Alonso-Jime´nez, J. Marcos-Ferna´ndez, X. Serra, and D. Bogdanov, “Essentia tensorflow models for audio and music processing on the web,” in Web Audio Con- ference (WAC 2021), 2021.

[9] E. Kermit-Canfield, “A comparison of real-time pitch detec- tion algorithms in supercollider,” Journal of the Audio Engi- neering Society, october 2014.

[10] N. Schnell, G. Peeters, S. Lemouton, P. Manoury, and X. Rodet, “Synthesizing a choir in real-time using pitch syn- chronous overlap add (psola),” in ICMC, 2000.

the shifted voices preserving the formants and includes different presets that the user can change using a MIDI keyboard