Improvisio : towards a visual music improvisation tool for musicians in a cyber-human co-creation context

Sabina Covarrubias Stms

IMPROVISIO : TOWARDS A VISUAL MUSIC IMPROVISATION TOOL FOR MUSICIANS IN A CYBER-HUMAN CO-CREATION CONTEXT

Sabina COVARRUBIAS
STMS Lab : IRCAM
sabina.covarrubias@ircam.fr

ABSTRACT

Improvisio is a software for musicians who want to improvise visual music. Its development is part of the REACH project. It is useful to create visual improvisations in the frame of: audiovisual concerts, multidisciplinary projects, collaborations with visual artists or interactive audiovisual art installations. *Improvisio* interfaces with *Somax2* and *Djazz*. In the context of musician-machine performances, the main functions of the program are: 1. to provide a visual representation of the musical elements in play in real time; 2. to enable improvised musico-graphic interactions between AI agents and musicians; 3. to make the operating processes of the artificial intelligence algorithm intelligible through the real-time visualization of musical and sound data. This article describes the first phases of development of the program. We want to show how the objectives of the REACH project, the results of the previous research, and the technical and aesthetic needs arising from the use of the first prototype in the framework of an art research project, lead to the proposal of an architecture and the implementation of the functions that are part of the program.

1. INTRODUCTION

In this section we provide a general description of *Improvisio* and its development objectives.

1.1 Description

Improviso is a tool under development, programmed in Jitter, and GLSL. This program is a tool for musicians wishing to perform visual music improvisations for audiovisual concerts, multidisciplinary projects, collaborations with visual artists or interactive audiovisual art installations. In this project we adopt Sedes' [9] definition of visual music, in which the work is "a musical composition that uses the means of the visual to produce the musical". We agree. We also define visual music as a work that uses the means of the musical to produce the visual.

In the context of man-machine performances, where the program interfaces with the programs *Somax2*[1][12] and *Djazz*[2][13], the program has four main functions:

1. Create a visual representation of the musical elements in play in real time.
2. Transcode visual data into pertinent musical elements.
3. Enable improvised musico-graphic interactions between AI agents, musicians and visual artists.
4. Make the operating processes of the artificial intelligence algorithm intelligible by means of real-time visual representations of the musical elements and sound data.

Today, a first prototype has been implemented and tested during one experiment: The use of the software during the creation of an art project and its use within the framework of a music festival concert. This experiment will be described in detail in the second section of this article. As the development of *Improviso* is part of the REACH[3] project, its development aims to contribute to and facilitate the realization of the objectives of this project[4].

1.2 Objectives

In the following, we will describe the reasons and objectives that give place to the development of this tool and how it can contribute to their realization. In this way, we will show the relevance and necessity of developing such a tool in order to answer the question: In the context of human-machine performances, how can the inclusion of visual music improvisation contribute to an increase in co-creativity and audience engagement?

[1] For more information about Somax2, please visit the page : https://forum.ircam.fr/projects/detail/somax-2/
[2] For more information about DJazz, please visit the page :https://digitaljazz.fr/research/
[3] REACH stands for: Raising Co-creativity in Cyber-Human Musicianship. For further information about the project, please visit the official project page: https://reach.ircam.fr/
[4] The objectives of the Reach project are described on page 6 of the following document: http://repmus.ircam.fr/_media/reach/reach_erc_adg_2019_b2.pdf

1.2.1. Developing methods for augmenting human capabilities and enhancing computational creativity

Improvisio allows musicians to foster computational creativity through the use of a new tool for visual music improvisation. Why the need to provide a tool to create visual content for musicians? Several studies confirm that the trend of artists performing audiovisual concerts is indeed on the rise. According to Besada [2], it is now common to experience contemporary music concerts that combine or interact with live music performances and video projections. Besada explains that, the recording and presentation of this material to the audience is being influenced by video-sharing platforms on the Internet and by social media. The growing interest and investment in audiovisual technologies for music performance is demonstrated by the collaboration between the Key Laboratory of Intelligent Processing Technology for Digital Music and Professor Zhang in the audiovisual 3D immersive recording of concerts by Wang & Hu [11]. Further supporting the growing prevalence of audiovisual concerts, Budagyan & Zaytseva [3] highlight the importance of the transformation of modern music culture through the transition to new digital technologies for delivering audiovisual information to modern listeners.
Despite the increasing presence of visuals in music concerts, we have observed on several occasions that visuals accompanying live music tend to show:

1. An imbalance between the quality of the musical work and the quality of the visual work: which can lead to the presentation of repetitive animations for long periods of time.
2. The juxtaposition of two systems or languages whose operating principles are different and independent.
3. Presentation of visuals automatically generated by transcoding audio data into the visual domain for long periods of time.
4. A dominance of one system over the other, where the visuals are sometimes dispensable or a distraction from the music.

In light of these observations, and in an effort to explore new forms of visual creativity for audiovisual concerts, it is proposed to provide the musician with an instrument to generate improvised visual animations in musical interaction with other musicians and with AI agents improvising music. Indeed, *Improviso* is an instrument designed for musicians who want to explore visual music improvisation, so the program is controllable by musical tools such as MIDI controllers and musical instruments[5]. This tool expands the creative improvisational possibilities of the improvising musician and also helps to create a musical development of the visuals presented during a concert.

1.2.2 Explore and foster cyber-human co-creation strategies based on the visual embodiment of algorithms

Using real-time visualizations to show how algorithms work can be considered as a kind of embodiment of the algorithm. Real-time visualizations provide a dynamic and interactive way of representing the inner workings of algorithms. They allow users to observe the behavior of the algorithm as it unfolds. This approach is consistent with the concept of embodying the algorithm, as it allows users to visually engage with and understand the algorithm's processes in real time. By integrating visualization techniques into algorithms, users are able to interact with and guide the algorithm in real time, increasing their understanding and control over its operations. Farghally et al. [5] Furthermore, the use of real-time visualizations can facilitate the training and learning of embodied agents in visual environments, contributing to the development of embodied AI algorithms. Hernández et al. [6]
Considering these facts mentioned above, *Improviso* will allow the exploration of new improvisation processes by presenting visual representations of the AI agents' operations to the musicians. In fact, the visualization of the processes of the algorithm will have an impact on the outcome of the human-machine improvisations. These visualizations could allow musicians to anticipate to a certain extent the actions of the AI agents, which could have consequences such as: favoring the synchronization of musical events between man and machine.

1.2.3 Providing a touch-drawing control interface to influence the Somax2 players

It has been shown that the provision of new software control interfaces can contribute to innovation in musical language. Miranda, E. R., & Wanderley, M. M. [8] In this sense, with the aim of exploring new aesthetic proposals, *Improvisio* includes the modules: *drawing with tablet*, *interactive particle system* and *interactive physics-collisions*. These modules provide a new control interface that converts visual data into musical elements as MIDI messages. In this way, the software offers the possibility of interacting with a *Somax2* "player" by means of three types of graphic input: 1. drawing or hand writing on a tablet, 2. touch-interactive computer graphics where the visuals are generated by an engine that simulates the interaction of physical objects [6], and 3. touch-interactive particle system animations, where the position of each particle can be translated into OSC or MIDI messages for

[5] When connected to an audio analysis module, to obtain RMS amplitude, amplitude peaks, fundamental frequency, among others, the musician can use his instrument to control the software to generate or process graphs in real time.

[6] This module has been programmed using the jit.phys library in Jitter Max MSP, so that within an array of objects (spheres), each element is mapped to a MIDI note, and the collision between them is mapped to MIDI note-on messages. So when the objects collide, they generate MIDI note-on messages, and the strength of the collision determines the value of the velocity message. The resulting MIDI notes are then sent to a "Player" module to influence the *Somax2*'s improvisation. The pitch of each note assigned by the programmer will, in a future version, be assignable by the user.

sound spatialization[7]. The data generated by these visuals is transcoded into MIDI messages. Once transcoded into MIDI, the data is used to feed the *Somax2* MIDI type "influencer" module, which influences the improvisations generated by the "player" modules. In fact, the transcoding of graphic data into MIDI notes makes it possible to use a graphic interface to interact with the *Somax2* player modules. This interface allows the interaction of *Somax2* and visual artists drawing in real time during a music performance.

1.2.4 Augmenting innovative collaborative strategies

Improvisio facilitates the realization of interdisciplinary improvisation projects, allowing collaboration between musicians and visual artists such as painters, filmmakers, set designers, photographers and illustrators. In this collaboration, visual artists and musicians improvise by interacting with each other and with AI agents. This experience can be realized by interfacing *Improvisio* with *Djazz* or *Somax2* software. *Improvisio* can be an interface between the visual music improviser, the musicians, the visual artists and the AI agents of *Somax2* and *Djazz.*
The performance of the visual music performer is the key to establishing the link between the visual and musical elements. The visual music performer gives musical development to the visual discourse by playing with other musicians using musical interfaces to produce the visual animations.

1.2.5 Contribute to the engagement, diversification and enlargement of the experimental music audience

One of the aims of the REACH project is to disseminate the results of this research, and for this reason we estimate it is important to highlight the benefits of including improvised visual music animations in concert performances of experimental music produced using *Djazz* and *Somax2*.
The inclusion of live visuals in these concerts will help to engage, broaden and diversify the audiences for this music. This claim is supported by research. In the field of musical performance, the integration of audio-visual cues has been found to have a significant impact on the audience's understanding of the emotions and themes embedded in the music. Influenced by Kim and Pellegrino's [7] research, this study investigates the impact of visual aesthetics and multimedia elements on audience perception and emotional engagement during live concerts. Based on the findings presented in the Journal of Student Research, the research emphasizes that the incorporation of visual elements, such as body movements, facial expressions and synchronized multimedia, plays a crucial role in shaping the audience's interpretation of the musical performance. The study also highlights the positive influence of visual synchrony on the audience's evaluation of the music, suggesting that a well-integrated audiovisual experience can lead to a deeper emotional connection and understanding of the music, supporting the notion that the use of live video during concerts helps to broaden and diversify the audience. However, the study also highlights the potential for multimedia to act as a distraction, emphasizing the importance of thoughtful integration to ensure an immersive and engaging audience experience[7]. Belfi et al. [1] explore the effects of congruence between musical artist and song, suggesting that audiences may particularly enjoy seeing their favorite artists perform their best-known song in a live setting. This suggests an increased interest in live music consumption. These findings justify the presence of the "live camera" module in the *Improvisio* software. This module allows a camera to be connected to the software in order to show the performer's gestures, performance modes, etc. These images could be modified in real time by the visual music performer using video effects.
In terms of audience engagement, research from Shoda[10] and Beli [1] show that the use of live video enhances the overall concert experience, allowing audience members to have a more immersive and engaging interaction with the performance. Live concerts allow audience members to experience a personal relationship with the performer, and the inclusion of live video further enhances this connection Furthermore, using live video in concerts provides a multi-sensory experience, which contributes to the collective behaviors observed in concert settings. This variety and magnitude of stimuli contributes to the communal experience and overall enjoyment of the event.

II. DEVELOPMENT AND USE OF THE FIRST PROTOTYPE

In this section we will describe the stages of development of the first *Improvisio* prototype and supplementary modules in the context of one artistic project. The emphasis will be on showing how the development and use of *Improvisio* was useful in solving the aesthetic and technical needs that arose in the context of interdisciplinary collaboration, collective improvisation and human-machine improvisation. The development of the first *Improvisio* prototype and the additional modules to be included in the program were carried out during and for the production of the art-research project described below.

2.1 Art-Research Project: Justin Vali meets Jacek Woźniak. Drawn improvisations and virtual choreographies around the baobab tree

The development of *Improvisio* began in February 2023 to give birth to the spectacle: *Justin Vali meets Jacek Woźniak. Drawn improvisations and virtual*

[7] This is an experimental feature which will work with the Somax2 version which is currently being written in *Super Collider.*

choreographies around the baobab tree [8]. The work was created over a period of 4 months, from February 2023 to May 2023. It was presented during the *Festival de l'Imaginaire* on 8 June 2023. The participants of the work were: Justin Vali[9], Jacek Woźniak[10], Marc Chemillier[11], and Sabina Covarrubias[12]. The contribution of each participant was as follows:

- Justin Vali: music composition and lyrics, singing, performance of sitars *valiha* and *marovany*, and *kabosy* guitar.
- Jacek Woźniak. The artist provided three types of images in .png files with transparency: 1. images with cut-out outlines showing "characters" to be animated in real time. 2. Images to be used as the "background" of a visual composition. 3. Drawing in real time on a tablet connected to the *Improvisio* software.
- Marc Chemillier: artistic direction, keyboard performance and *Djazz* improvisation software.
- Sabina Covarrubias: Development of the *Improvisio* software and real-time visual improvisations.

The collective contributions were: The visual choreographies: how the characters move on the screen, at what moment they appear and which ones appear were decided collectively by the four members of the group.

2.2. Description of Improvisio's development stages

2.2.1 Context

Musician Justin Vali wanted to collaborate with painter Jacek Woźniak to create an audiovisual concert. The idea is to present the painter's paintings and live drawings on a screen while Vali and Chemillier play music using *DJazz,* traditional Madagascan instruments and vocals. Woźniak, who has experience of audiovisual concerts with the musician Ariche Shepp, had already worked with the artist Claire Roygan, who animated the painter's drawings using software she wrote in Jitter, allowing the artist to draw over the animations in real time during the audiovisual concert. In February 2023, Chemillier invited Sabina Covarrubias to develop a program for the live animation of the visuals, as part of a collaboration with Justin Vali and Woźniak. A recording of this work is available in this webpage : https://digitaljazz.fr/2024/02/16/film-festival-de-limaginaire/

2.2.2 Main functions of the first prototype

Chemillier, the project director, asked Covarrubias to develop a prototype program, specifying that it should be written in Jitter and have the same functions as the program developed by Claire Roygan, plus additional functions to visualize the parameters used in the *Djazz* program to carry out the improvisations. The main functions of the first version of the prototype include:

- Animation of .png image files based on changes in scale and vertical or horizontal movement across the screen. These animations are the result of a mapping between the amplitude of the signal from a live musician and the scale or position of the image file displayed on the screen.
- Up to 15 layers of images can be juxtaposed to create a visual composition.
- Drawing module for use with a drawing tablet to display drawings as a layer on top of the overall visual composition.
- Loading of pre-defined image files between each song.
- Preset system for creating different visual compositions, with the possibility of recalling presets without interrupting the visual animations.
- Application of video effects such as contrast, brightness and saturation in real time.
- Visualization of data coming from *Djazz:* the pulsation, the chapter number (or section within the musical structure), the acceleration rate of the musical motifs used for improvisation, the pitch transposition rate of the musical motifs used for improvisation. *Djazz* must run on a different computer than the visuals and must send the data to be displayed via OSC to the program that generates the visuals.

2.2.3 Implementing and optimizing the first prototype

The first prototype was implemented in June 2023. In contrast to the version programmed by Claire Roygan, the first prototype has the following advantages

- Exclusive use of the GPU to process textures and shape animations.
- Reception of *Djazz* data from another computer via OSC protocol.

[8] The original name in French of this project is : *Justin Vali rencontre Jacek Woźniak. Improvisations dessinées et chorégraphies virtuelles autour du baobab.*

[9] Justin Vali is a famous Magash composer and performer of traditional Madagascan music.

[10] Wozniak is a well-known Polish painter, artist and illustrator who works in the press. For more information, please visit the site https://www.wozwoz.net/

[11] Marc Chemillier is a researcher and musician, conceptor of the DJazz program. For more information visit : https://www.ehess.fr/fr/personne/marc-chemillier

[12] Sabina Covarrubias is a Composer, multimedia artist and researcher, for more information please visit : www.sabinacovarrubias.com

2.2.4 Integrating the visual improvisation and visual choreography modules

In order to meet the aesthetic requirements of the painter Woźniak, we found it necessary to include modules that would allow musical improvisation of the visuals.
The artist has precise requirements: he asks for certain animated movements of the drawings, which must correspond to certain moments of the music. Here are some examples:
"A sun that rises from the right side of the screen; once it has risen, it slowly moves to the left and this trajectory must last exactly the time of the song; when a certain sound is heard in the music, the sun sets and then you must see the image of a landscape where it is night".

"Every time Marc plays the sound of thunder and Justin moves his head to indicate that moment, you should see the effect of lightning on the screen".

"In a kind of slide show with 1000 pictures of different faces: The images should change very slowly at first and then gradually faster and faster, but there should be moments of rhythmic contrast to avoid a linear accelerando from beginning to end, each change of image should be synchronized with the pulsation, but the changes should not be regular to avoid monotony".

These requirements make it necessary to include the "visual choreographies" module, which has the function of storing and recalling, without latency, a sequence of movements, sizes and positions of shapes on the screen in synchronization with improvised musical events during a song. This module contains the writing of the movements of the shapes, the musician performing the visuals is in charge of triggering the different parts of the choreography at precise moments, such as the change of section within a song. In this show, a choreography was written for each song. Each choreography is made up of several parts. The performer of the visuals starts each part of the choreography in synchronization with each section of the music, for example:

Music section	Associated choreography
introduction	choreography01_part01
verse 1	choreograghy01_part02
refrain	choreograpy01_part03
verse 2	choreograpy01_part02
interlude	choreograghy01_part04
refrain	choreograpy01_part03
finale	choreograpy01_part04

Table 1. Each song is associated to one choreography. A is composed by several parts. Each part corresponds to one preset, or memorized set of parameter values.

In addition, in response to the needs of the painter, it was necessary to implement modules for visual music improvisation with the functions:

- The possibility of synchronizing visual events, such as the application of special effects or the transformation of shapes, with musical events typical of improvised music, such as: the variable duration of an improvised solo section, or the moment of the end of a song determined by a gesture from a musician addressed to all the musicians.
- The possibility of performing different rhythms within the same pulse and tempo received from *DJazz,* thus avoiding the monotony that results from visualizing the same pulse throughout a song.
- MIDI controller input to modify visual parameter values such as: shape rotation, size, speed of movement, position, colors, among others. In addition to triggering the different sections of the visual choreography, the performer of the visuals can improvise within each section of the choreography, adding contrast and musicality to the visual animations by interacting with the musicians' playing.

2.2.5 File upload system

In this project it is necessary to emphasize the optimization of loading and managing images.
The artist Woźniak asked to show details of images with a weight of 2M by zooming in on them, this implies optimizing the system to be able to process graphics by means of the "visual choreography" and "visual improvisation" modules, so that the GPU renders the projected image at no less than 24 frames per second. To achieve this, the image and video file loading module has been implemented, which, after reading the files from the hard disk, transfers the data to the GPU, thus freeing the CPU's RAM memory. The images are loaded by the CPU into <jit.matrix> type objects. Then the data is transferred to the GPU and the images are read as textures. Once the images are treated as textures, the program is ready to activate the choreography and improvisation modules.

2.2.6 Ending the development of the first prototype

At the end of this project, we realized that although the program was effective for the purposes of this show, it was of little use for other artists and for other projects. This is because the programming of the "visual-choreography" module was implemented to meet the aesthetic needs of the painter Woźniak. In fact, the choreographies are fixed, they determine the positions of the images on the screen, their size and their movements, therefore these choreographies only make sense when applied to the painter's images in the context of Justin Vali and Chemillier's music and are hardly useful or adaptable for other projects or artists. For this reason, and as a

conclusion to this first phase of development, we asked ourselves: how can this tool be adapted to be useful for other projects and other artists?

The first clue is that it is necessary to provide the program with a variety of modules for the synthesis and processing of texture-type images in real time, and that the choreographies are not fixed but dynamic: the user will be able to write his own choreographies or will have the possibility of using pre-recorded choreographies with and choosing between several options. For this reason, the following artistic research project has been carried out.

III. ARCHITECTURE AND FUNCTIONS FOR A SECOND PROTOTYPE

In this section we will describe our architectural proposal and the functions of a program intended for musicians who need to perform visual music improvisations in situations of human-machine co-creativity. This proposal was conceived taking into account the objectives and research results described in the first section of this article. Likewise, the proposal responds to the aesthetic needs and technical problems that arose during the creation and presentation of the artistic projects described in the second section of this paper.

3.1 Modularity and Portability with Max for Live

We propose a modular architecture inspired by the *Somax2* program architecture, where modules can be used as part of an application or as building blocks within the Max MSP programming environment. The first advantage of this type of architecture is the possibility to extend the program by adding synthesis and image processing modules. The second advantage is portability: it is possible to adapt the modules to the *Max for Live* programming environment. In fact, the modules programmed in Jitter, as well as the shaders written in GLSL, are compatible and usable in *Max for Live*. The advantage of using these modules in *Max for Live* is to broaden the audience by making Ableton Live available to musicians using *Ableton Live*. The use of the modules in Max for Live would also offer the possibility to easily map different MIDI instruments to control the parameters of the program.

3.2 Strategic CPU/GPU Process Division

A good programming practice when using jitter in Max MSP is to run the CPU processes first and then the GPU processes, avoiding data transfers from the GPU to the CPU. In addition, minimize data copies (loads) from CPU to GPU and make sure they occur as early in the processing chain as possible[13]. *Improvisio* consists of modules that are grouped into module categories. There are categories of modules whose tasks are processed exclusively on the CPU, and categories of modules that run exclusively on the GPU. The figure below shows the program's module categories and the distribution of their tasks between CPU and GPU. To avoid frequent data exchange between processors, we can see that each module category works exclusively on the CPU or on the GPU. The data transfer function allows to convert data from a matrix processed on the CPU to a texture processed on the GPU.

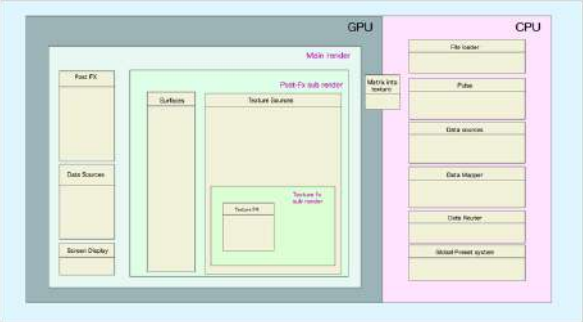

Figure 1. The categories of modules of the program Improvisio are divided into two categories: those whose modules work with the CPU and those whose modules work with the GPU. A function allows the transfer of data from the CPU to the GPU and is represented in the middle of both processors.

3.2.1 Functions Processed by the CPU

The following are the categories of modules that the CPU processes and a summary of their main functions.

- *File Loader*: loads and reads image and video files to be used during improvisations. It contains a database to remember the files used in each project.
- *Pulse:* Receives the beat from the DJazz program or generates a beat. The pulse can be divided or multiplied with rhythmic values or used to generate rhythmic motifs with a sequencer. The module outputs a bang message.
- *Data sources Receives OSC data:* float data in the range 0 to 1 and bang messages, receives MIDI control and note messages.
- *Data Router:* Allows routing data to the parameters of the visual composition that can be controlled in real time.
- *Global Preset System:* Manages the storage and recall of parameter values throughout the program.

3.2.2 GPU Processed Functions

The module categories processed on the GPU are distributed among different *rendering engines.* It is convenient to divide the rendering process into rendering

[13] The description of this practice is described in detail by Jitter developer Robert Ramirez in the article Jitter Best Practices, part 1. Accessible at this link : https://cycling74.com/tutorials/best-practices-in-jitter-part-1

threads so that the processes can be enabled or disabled when not in use to free up GPU resources.

The main rendering engine, implemented from a <jit.world mainrender> object, processes the *PostFx, Data Sources*, and *Screen Display* module categories.
The following are the categories of modules processed by the GPU and an overview of their main functions:

- *Screen Display:* Display of the main image on the screen of the
- *GPUPostFx:* Application of special video effects to the final composition - Data Sources
- *Data Sources:* Generation and output of OSC data generated from the visuals, to be used to control a MIDI instrument or for sound spatialization.

The first secondary rendering engine, implemented from an object: <jit.gl.node mainrender @name postfx>, handles the following modules :

- *Texture sources* : Different types of image sources that are applied as textures to a 3D shape to be rendered.
- *Surfaces :* Generation of 3D shapes to which a texture (image) can be applied.

The third secondary rendering engine, implemented from an object: <jit.gl.node postfx @name texturefx> , the module:

- *Texture Sources:* processes the special effects applied to the images generated or processed by the Texture Sources module.

3.2.3 Data Transfer from CPU to GPUs

Video or image files are loaded into the system by the CPU, and these data are then converted to matrix format in Jitter. The conversion of Jitter matrix type data into Jitter texture type data allows data to be transferred from the CPU to the GPU. In the program, this operation is represented as an intermediate module between the CPU and the GPU: Matrix to Texture.
The modules of the category "Data sources", which are processed by the GPU, output data generated from images to the outside of the system via the OSC protocol, so that they can be used as control messages in other systems.

3.2 Functional Classification of the Modules

As described above, *Improvisio* consists of several modules that belong to different categories. The figure below shows a list of the modules in each category.

The categories of modules working on the CPU are not modifiable: they are essential for the functioning of the system. On the other hand, the categories of modules working on the GPU can be modified: Their modules are interchangeable, they can be extended by adding more modules. A detailed description of the functions of each module is beyond the scope of this article, so we will limit ourselves to providing a list of the names of the modules that have been implemented so far. The following list shows the names of the main GPU modules implemented grouped by category. The attributes available from each of these modules is mappable to OSC or MIDI messages[14].

Texture Sources modules:
- Movie player with scratcher
- Poly~ texture loader
- Loader external
- Web-cam input
- Tablet drawing
- Shader image synth
- Shader loader/player

Surface modules:
- Poly~videoplane
- Poly~ shapes
- Multiple shapes
- Particle system
- Deformed mesh
- Iso_surface

Texture FX modules:
- Image correction
- Noise distortion
- Kaleidoscope
- Zoom
- Offset

Post FX modules:
- Noise distortion
- Tint colorizer
- Pixel sorting
- Kaleidoscope
- Zoom/offset

Data Sources modules:
- Estrange attractors particle system
- Coords "xyz"
  from a points cloud physics system
- Midi notes

The following list shows the names of the main CPU modules grouped by category:

File Loader modules:
- Loader
- Player

Pulse modules:
- Receiver
- Internal
- Tap tempo
- Sequencer
- Clock divider/multiplier

[14] The description the attributes of each module is out of the scope of this paper.

- Pulse router/sender

Data Sources modules:

- Audio Analysis
- OSC receiver : bang/float 0. to 1.
- MIDI receiver

Data Mapper modules:

- Automatic range mapping
- Mapping Calibration (target range adjustment)
- Reactivity factor (scalar multiplication)

Data Router modules:

- To 3D surface attribute
- To Post FX attribute

Global Preset System modules:

- Presets UI
- Preset UI mapper

3.3 Data flow between the module categories

Data flow in the *Improvisio* program. The system receives data from the outside through the OSC protocol or the MIDI protocol, and outputs data in OSC format. The modules of the "Pulse" and "Data Router" categories can send data to the categories of the modules operating on the GPU in order to modify the parameter values available in these modules.
The following figure describes the data flow when the program is running.

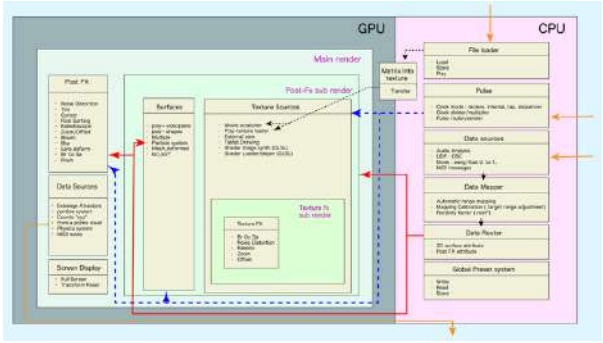

Figure 2. Data flow when the *Improvisio* program is in operation

The first process performed by the system is reading audio or video files; then the rendering processes must be activated. The following processes can be performed in any order:

- Image data is transferred to the GPU.
- If the modules of the *Pulse* category are activated, they can send the received pulse data to the modules of the *Texture Sources, Surfaces* or *Post Fx* categories to change the values of the attributes of the modules of these categories by means of "bang" messages.
- When activated, the modules of the *Data Source* category can send data to the *Data Mapper* and *Data Router* categories and then distribute the data to the modules of the *Texture Source*, *Surface* and *Post Fx* categories in order to change the parameter values of the modules of these categories using OSC or MIDI messages.
- The Data Source module can translate data from the GPU into OSC messages so that they can be used as control messages in *Somax2* systems.

The system allows simultaneous modification of parameter values of modules processed in the GPU by both impulse and external data. The Data Mapper module makes it possible to adapt the data areas of the *Somax2* and *DJazz* systems to the modification of the visual animations.

5. CONCLUSION

The design of the architecture of the *Improvisio* program, as well as the functions that are part of it, are based on the aesthetic and technical needs that arise from artistic practice and research objectives. The improvisation of the visual elements by a musician in the context of an audiovisual concert gives integrity and unity between the visuals and the music. The next stage of research, creation and development consists to assemble the modules into a single application according to the proposed architecture, and to use this version in the context of several multidisciplinary art-research projects by musicians using Somax2 and DJazz.

6. REFERENCES

1. Belfi, A., Samson, D., Crane, J., & Schmidt, N. Aesthetic judgments of live and recorded music: effects of congruence between musical artist and piece. Frontiers in Psychology, 12. 2021.

2. Besada, J. L. Cover, Custom, and DIY? Memetic Features in Multimedia Creative Practices. Contemporary Music Review, 41(4), 382-400. 2022.

3. Budagyan, R. and Zaytseva, M. Digital technologies in the modern music space. Observatory of Culture, 17(4), 368-378. 2020.

4. Brown, S. C., & Knox, D. Why go to pop concerts? The motivations behind live music attendance. *Musicae Scientiae*, *21*(3), 233-249. 2017.

5. Farghally, M., Koh, K., Shahin, H., & Shaffer, C. (2017). Evaluating the effectiveness of algorithm analysis visualizations. 2017.

6. Hernández, A., Risquet, C., Rodríguez, R., & García, R. Integration of visualization techniques to algorithms of optimization of the metaheuristics ant colony. Computación Y Sistemas, 22. 2018.

7. Kim, J., & Pellegrino, N. The Effect of Audio-Visual Cues in Understanding Musical Performance. Journal of Student Research, 12(1). 2023.

8. Miranda, E. R., & Wanderley, M. M.New digital musical instruments: Control and interaction beyond the keyboard. A-R Editions.2006

9. Sedes A., Musique visuelle, extension de la mixité. Lecture given as part of the study days on intermediate extension and perenniality in mixed music: image, improvisation, preservation and "recasting" ed. Portuguese Catholic University, Porto "Music mixte" MSH Paris Nord programme. 2012.

10. Shoda, H., Adachi, M., & Umeda, T. How live performance moves the human heart. Plos One, 11(4), 2016.

11. Wang, X. and Hu, T. A symphony of sound images: spatial composition strategies in zhang xiaofu'syarlung zangbo. Organised Sound, 27(3), 346-357. 2022.

12. Reach [REACH]. (n.d.). Retrieved 16 February 2024, from http://repmus.ircam.fr/reach

13. Djazz—Digital Jazz | Research. (n.d.). Retrieved 16 February 2024, from https://digitaljazz.fr/research/

This research is supported by the European Research Council under Horizon 2020 program (ERC REACH project, PI Gérard Assayag, Grant 883313).