Ribbons: a live cinema instrument.

Tomás Laurenzo
Universidad de la República.
laurenzo@fing.edu.uy

Abstract.

Ribbons *is a relatively small real-time visual instrument created for maximizing the expressiveness of the performer within its aesthetic paradigm. Its design questions the basic assumption of a flat rectangle traditional as projection space by presenting a virtual three-dimensional space where the footage (or live video) can be projected and deconstructed, adding a new dimension of expressiveness* orthogonal *to traditional narrative.*

We worked with the following design axes: playability vs. autonomy, expressiveness vs. narrative, and originality, and applied some of the basic techniques of Human-Computer Interaction and digital lutherie.

Introduction.

Live Cinema

Live cinema is a term recently coined for a long-standing practice[1]: real-time audiovisual performances, which –in its current incarnation– are real-time collaboration between sonic and visual artists [5].

Although the aesthetics of live cinema has been shaped mainly by VJing[2] –i.e. club-based visual performances– live cineastes have been performing at different spaces, with their oeuvres being shown in places ranging from traditional art galleries to multitudinous rock performances, and expanding traditional narrative cinema with a much broader conception of cinematographic space [5].

This expansion, together with the images of club VJing has led the production to very abstract and synaesthesia-focused works that somewhat deny traditional cinematographic narrative techniques and methods.

This biases the production, by focusing only in "the transitions, the movements, the pure visual beauty" [5]. By claiming freedom from the narrative strings, the

---

[1] It indeed is a long-standing practice, since the Wayan Kulit –Indonesian shadow theatre projected by fire– in the XX century; there have been different *projected* real-time performances. Live cinema has its direct predecessors in the Magic Lantern performances of late 1700s and early 1800s, also –of course– in cinema and –perhaps more notably– in the synaesthetic efforts of Color Music and Lumia. For a history of Live Cinema and its artistic language, see [5] and [2].

[2] We refer to VJing as real-time video *mixing* of footage, while live cinema also includes footage creation and its aesthetic, including but transcending the appropriation of footage of VJs.

performer is not allowed to convey a potentially denser stream of images that benefits from less abstract images.

Live cinema's performances, beyond their particular characteristics, are constructed by real-time editing live or stored visual media (often both), using many gestures of traditional cinema (such as slow motion) and effects (such as *scratching*) that belong to VJing.

In order to permit these on-the-fly manipulations, different tools –both software and hardware– have appeared.

The software tools range from the most general and low-level –for example Cycling74's Max/MSP/Jitter or Apple's Quartz Composer– which are full programming languages, albeit visual ones, to more application-like environments such as Resolume, Oscil8, etc. Hardware tools include video mixers, effects, and –of course– playback and output hardware.

Visual Lutherie and Human-Computer Interaction.

These performance-oriented tools that produce moving images are called visual instruments, and therefore, their crafting should be called *visual lutherie*[3].

As Miller Puckette said about computer music software: "The design of the software cannot help but affect what computer music will sound like" [6], visual lutherie (as any tool used in art production does) influences visual production.

Reciprocally –yet, still talking about music–, Bahn and Trueman present the concept of "composed instruments" [1]. If we believe that "new music tends to be the result of new techniques, which can be both compositional or instrumental" [3], we conclude that a possible approach for art production consists in the creation –the composition– of new tools of artistic performance, new instruments, and that its creation may no longer be a stage previous to the art, but became part of it.

But this artistic approach to instrument creation should not forget that many guidelines and techniques of HCI[4] are applicable[5] and aid in the instrument's design [4].

Two of the most important methodologies of HCI are user-centered design and iterative design, where the user becomes part of the development team –because he or she is an expert on his or her area of knowledge– and the team assumes that their work is perfectible and iterates creating many versions that get closer and closer to what the user needs.

Also, a very important interaction style is Direct Manipulation, which stands for interactive systems with continuous representation of the domain of interest, with rapid, reversible, incremental actions and continuous feedback. This allows

[3] Although a rather obvious name, we have never seen it before writing this paper, nor were we able to find any paper or work that uses it.
[4] Although these are to be taken with a grain of salt, of course.
[5] If consonant with the artist's desires and objectives.

the user to *feel* that he or she is operating directly with the objects presented to them with a direct representation of the domain of interest.

Both the methodologies and the interaction style are applicable to visual lutherie. In the following section we will present our live cinema instrument, which was created with these HCI concepts in mind.

The Instrument.

Design.

Traditional cinema *projects* its narrative onto the flat canvas of the projection screen: everything that happens in the film is under the "tyranny-of-the-rectangle". The live cineaste is also constricted by the same limitations, although many times it can be altered by using multiple projection screens that break the traditional rectangle or by using *projection mapping*[6] techniques.

But even in the most extreme cases, once the projection surface or surfaces have been chosen all the narrative occurs on those pre-defined canvases.

While Ribbons –like many visual instruments– is, on its core, a video player[7] and its able to reproduce the videos in a standard way (a full-screen flat representation), and to apply some basic effects, such as transparency, scratching and direct access, its design challenges the flat representation by projecting the cinematic material (pre-recorded or live) onto a three-dimensional, virtual, radically deformable canvas (see Image 1).

To be able to do so, Ribbons creates a grid of three-dimensional particles with each one tinted with the corresponding color of the video. The particles can be manipulated by the performer in novel ways, thus adding a new dimension of expression, orthogonal to the footages' original one and distinct of common VJing techniques.

This new dimension may or may not compete with the traditional one, and it is the performer's call to keep the images intelligible or completely deconstructed.

These particles can then be used as input for different *visualizations* (such as triangle trips, cubes or lines), which we shall discuss later.

---

[6] Id est, projecting over with non-flat surfaces, see –for example– White Void's Polygon Playground at http://www.whitevoid.com] or AntiVJ's works at http://antivj.com.

[7] The user, previously to the performance, must add different banks of video loops that can be trigged (played) by the instrument. It is to notice that Ribbons also has some generative abilities, and could be used without any video source.

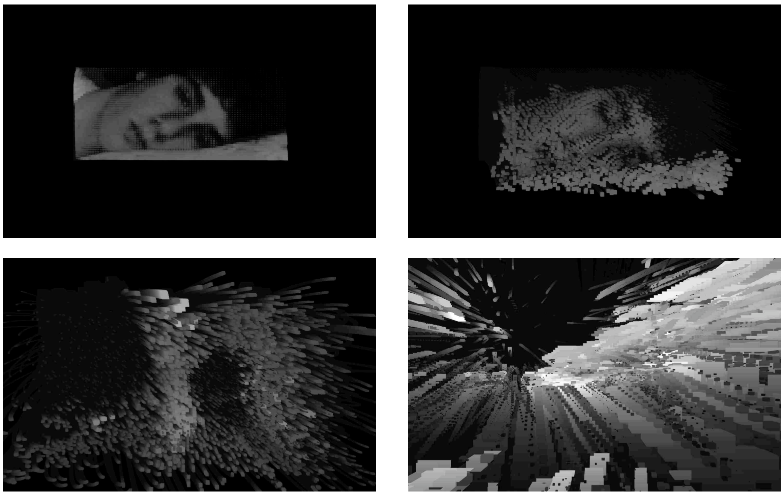

Image 1 - Different stages of deconstruction of the filmed image by applying a Perlin wind.

In the design and construction of the instrument, three axes guided our work: playability vs. autonomy, expressiveness vs. narrative, and originality.

Playability vs. autonomy.

The defining characteristic of an instrument its that it is playable. Ideally, the user should feel that both the manipulation is as direct as possible, even to the point that its manipulation disappears from the cognitive universe of the user as he or she focus on the results: the interaction becomes a metaphor of a world instead of the metaphor of a conversation, that is, the manipulation is direct.

In order to reinforce the directness, all the commands built trigger an immediate response[8], and the user can directly control parameters (such as camera orientation), select representations, or set off some visual response (e.g. drawing text).

However, we wanted the instrument to be able to "play by itself", that is, it should be able to keep on producing visual output even if the performer is not interacting with it. This was mainly because in real-time performances some times one needs to focus on something else (e.g. a hardware video mixer) and *the show must go on.*

Two things were implemented to achieve this: sound reaction (the instrument processes the audio captured by the computer's microphone and modifies the visual output) and inertial representation.

By inertial representation we mean that Ribbons allow the performer to deform the grid of particles by applying forces to them, and the particles act as if attached to strings (and then will oscillate and eventually converge to its original position) creating an effect of deconstruction and reconstruction of the original frame that can be controlled by the performer.

---

[8] This depends on the hardware being used to run Ribbons. In the tested setup (a Intel MacBook Pro laptop) the achieved performance allowed for immediate response.

This allows the performer to deform the grid in such a way that it will keep on moving coherently even if there is no user input with the synaesthesia reinforced by the before-mentioned sound reaction.

The deformations can be completely random or coherently random (by using Perlin noise) and the performer can have medium to little control of each particle actual movement but can always modify global parameters like the strength of the strings, the direction of the particles, etc.

The final product is a visual instrument where the user can completely engage into the performance, yet is able to let the instrument perform by itself without the change being noticeable by the audience.

Expressiveness vs. narrative

As we mentioned, the performer can, for example[9], apply a Perlin "wind"[10] to the particles and deform the projection surface, even to the point of de-constructing the video frame, re-signifying its components (the pixels) as elements capable of independently convey meaning.

This dichotomy between the narrative encapsulated on the cinematic material and the expressiveness of its manipulation conformed our second design axis.

Both the controllable deformations and the usage of the videos as raw data for the representations allow the performer to maintain the expressive language of traditional cinema while adding an orthogonal channel of information, expanding it for real-time performance.

Originality

Our third and last axis of work simply consisted in the attempt of generating a distinct, recognizable visual output.

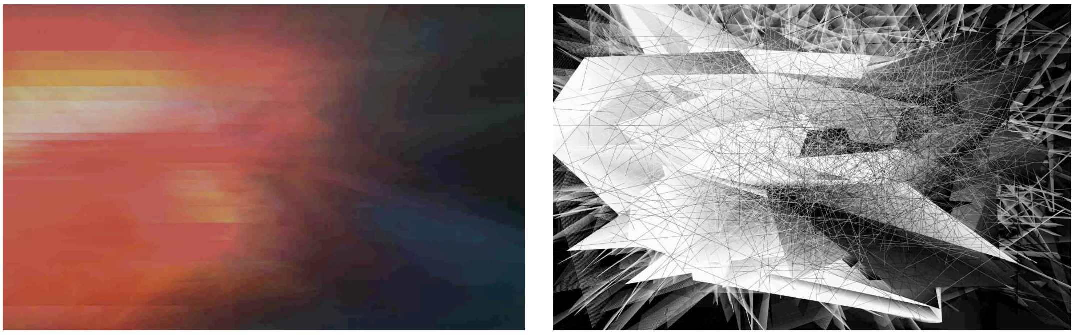

Image 2 – More Ribbons screenshots: triangles (left) and triangles + lines (right) visualizations.

Although we believe that we were moderately successful at it, we also coded some visualizations that are well known by the live cinema aficionado. For example, one of the completely sound-reactive outputs of Ribbons is directly

---

[9] To see all the performative gestures available, please refer to the next subsection, Operation.

[10] A commonly used technique in computer graphics consists on calculating a field of Perlin noise and using it as a vector field of speed for –for example– particles. In these cases, the particles appear to be *blown* by the Perlin noise.

inspired by and reminiscent of the visual output used by Alva Noto in his latest tour[11].

But it is obvious that the choices on whether or not use these visualizations or how to combine them is on the performer.

Operation.

Ribbons is to be controlled with one hand in the computer's keyboard and the other one in a drawing tablet[12]. There are four different types of commands:

- **Selectors** select a video source or visualization with a keystroke.
- **Triggers** trigger an immediate visual response (such as drawing some text on screen or reversing the particles' rotation direction). Also with a keystroke (usually augmentable or modifiable using the shift key)
- **Faders** change a continuous value, such as rotation or return speeds. These are controlled by holding a key pressed and moving the pencil.
- **Control commands** are meta-commands (i.e. not belonging to a Ribbons' performance but commands for settings, quitting, saving, etc.).

In the current version of the instrument the following commands have been implemented:

Selectors.

The topmost and bottommost lines of the computer keyboard are destined to selectors. The user can select up to ten videos from ten different banks: by pressing shift plus a number key the user selects a bank and the videos are mapped to the 10 numbers of the keyboard.

The bottommost row keys select the visualizations by turning them on and off with a key press (the used keys, so far, are z for particles, x for triangles, v for video, b for lines and n for UniTXT-like lines).

Tab switches from live video to pre-recorded footage and vice-versa.

Triggers.

Triggers and faders use the middle keyboard rows. There are triggers that start, stop or mirror the camera rotation, change the way new frames are drawn (old frames can be erased or faded out), add different levels and directions of Perlin or pure random flow to the particles, draw rectangles or text, that turns on or off some filters, etc.

These commands –together with the faders– are mapped onto the two middle rows of keys.

---

[11] UniTXT, see http://www.raster-noton.net/.

[12] Although it can be controlled with a standard computer mouse instead of the drawing tablet, the direct mapping from tablet-coordinates to screen-coordinates allow Ribbons to give an implicit feedback of the current level of the parameter being manipulated and allows the performer to manipulate the instrument with much more precision and speed than what can be achieved using a mouse.

Faders

As we mentioned, Ribbons offer some gestures for the direct manipulation of continuous (real, in [0,1]) parameters. The available gestures are: positioning, moving, dragging and clicking (touching the tablet with the pen's tip).

All the faders are mapped in an absolute way to one either the vertical or horizontal displacement. The selection of the parameter is done by holding a key pressed and choosing one of the gestures (direction plus touching or not the tablet with the pen).

The user can control, for example the transparency of a visualization, the speed of the video playing or the rotation of the camera, or can access a specific point on the video (and scratch by dragging the pen).

Control.

By pressing the F keys on the computer's keyboard the user is able to record the performance on the HDD, to edit the camera's parameters, or to turn on or off the unprocessed monitoring of the camera's input.

Implementation.

Ribbons was fully implemented in C++ and OpenGL using OpenFrameworks[13] as a programming framework.

We can't praise enough OpenFrameworks, an application framework for C++, which wraps GLUT, which in turn wraps OpenGL, an API for controlling graphic hardware.

OpenFrameworks combine the raw performance of C++ and OpenGL while providing a much more gentle environment.

Conclusions and future work.

We have shown our visual instrument Ribbons, which is not only theoretically consistent, but also has been successfully used in "real life" performances (see Image 3), where it provides the performer engagement that is expected from an instrument, while also being able to perform autonomously[14] (for brief periods).

The instrument allows us to investigate and question the basic need of expressive footage, and its relation with, on one hand more abstract, generative visuals and on the other its real-time manipulation by the performer.

It also questions, by virtually projecting the footage onto a three-dimensional space where the camera can be moved around and the projected image can be deformed, the classic assumption of a flat orthogonal projection without the costs and rigidity of more actual solutions.

---

[13] See http://openframeworks.cc

[14] We find particularly interesting this delegation to an automated process, which also means to delegate some very actual performative decisions to a previous ourselves.

Finally we would like to praise frameworks such as OpenFrameworks or Processing[15], that allow coders to create working early prototypes very fast, offering artists the invaluable gesture of sketching.

Future work.

We plan to keep on working on Ribbons by adding new visualizations, new automated sound-reactive behaviors, and the ability to use simultaneously multiple cameras.

We would also like to work on its performance (in terms of speed), as it can be improved by using computer graphics' techniques.

Image 3 - Ribbons' performing with a band (name blinded). Video, lines and particles visualizations.

References

[1] C. Bahn and D. Trueman. Interface: electronic chamber ensemble. *In ACM SIGCHI Workshop on New Interfaces for Musical Expression*, electronic proceedings, Apr 2001.

[2] M. Faulkner, editor. *VJ, audio-visual art culture*. Lawrence King Publishing, 2006.

[3] S. Jordá, *Digital Lutherie. Crafting musical computers for new musics' performance and improvisation.* Ph.D. thesis, Universitat Pompeu Fabra, 2005.

---

15 See http://processing.org

[4] T. Laurenzo. *HCI in new media art practices*, proceedings of Interacción 2008, June 2008.
[5] M. Makela. *Live Cinema, language and elements*. Master Thesis, Helsinki University of Art and Design, May 2006.
[6] M. Puckette, *Max at 17,* Computer Music Journal, 2002.