BEHAVIORAL INSTALLATIONS: EMERGENT AUDIOVISUAL INSTALLATIONS INFLUENCED BY VISITORS' BEHAVIOURS

Christian Frisson [1], *Loïc Reboursière* [2], *Jehan-Julien Filatriau* [1], *Todor Todoroff* [2], *Xavier Siebert* [3]

[1] Laboratoire de Télécommunications et Télédétection (TELE), Université Catholique de Louvain (UCL), Belgique
[2] Laboratoire de Théorie des Circuits et Traitement du Signal (TCTS), Faculté Polytechnique de Mons (FPMs), Belgique
[3] Laboratoire de Mathématique et Recherche Opérationnelle (MathRo), Faculté Polytechnique de Mons (FPMs), Belgique

ABSTRACT

This paper presents a numediart project in collaboration with two artistic projects: *Méta-crâne* by Thomas Israel [15] and *HUM* by François Zajéga [30]. The scope of this project was to offer technological forecasting and development consultancy to these two highly-interactive installations that both share a common goal of blending 'behavioral" recognition of crowd motion with audiovisual rendering. We achieved initial promising results for the *Méta-crâne* navigation by similarity in a video database and for the *HUM* analysis of crowd behaviors by computer vision techniques. We also provided a state of the art in domains such as sound spatialization, video projection on a 3D surface.

KEYWORDS

Interactive installations, gestural control, audiovisual rendering, video similarity, sound spatialization, 3D projection

1. BACKGROUND ON INTERACTIVE INSTALLATIONS

The following state of the art short study shows that there is still a high research potential both for public interaction methods (especially gestural) in interactive installations, and for more complex timelining methods in live/non-linear movies.

1.1. Interactive Installations

During the last century, following to the progress in technologies such as domotics, the domain of architecture has been showcasing many inventive constructions, as highly documented and ilustrated in Kronenburg's books [17]; not only homes and living spaces, but more specifically museums, concert rooms; and entertainment venues such as *The Sphere* [1] by the Belgian company Alterface using its Kiosk™technology. Lots of immersive multimedia installations have been created by artists, as documented in [13, 28]. Most of them use straightforward but simple computer vision techniques to analyse the user(s) behaviour.

Regarding interactive installations, besides Thomas Israel's experience in the field with for instance *Peeping Tom* (2006) and the *Le Lit TröM* (2005) [15], other notable artists in Belgium are Olivier Meunier with his interactive dome, *Real Unreal* [21], made in collaboration with the artists collective Foton and the hardware developer Periactes, and first presented at the Altitudes 1000 festival at Recyclart in 2006; and Pascale Barret with her interactive corridor *Synapse 2.0* created in residency at iMAL in 2008 [3].

Of interest for the numediart HyForge research theme on hypermedia navigation, George Legrady conceived *Pockets Full Of Memories* [18] in 2002, an installation where visitors bring objects that they encode in situ in a database (by scanning their image and giving semantical information) that organizes them by similarity by the Kohonen Self-Organizing Map algorithm, and displays them on a projection screen near each other in a two-dimensional map. Another example is the *Khronos Projector* by Alvaro Cassinelli in 2005 [7, 6] with which visitors can alter the playback of videos by touching and deforming the projection screen.

1.2. Interactive/Live Cinema

Still in the scope the HyForge numediart research theme, other works offer the possibility to break the fixed narrative timeline of cinematographic works, either at the audience will as previously illustrated for installations, or during performances by artists. A comprehensive retrospective is available in [19], its author having later on produced a series of three "soft cinema" movies [20] where the viewer can have a fairly limited impact on the timeline.

Experienced "locally", *Late Fragment* [8], an interactive video installation directed by Daryl Cloran, Anita Doron & Mateo Guez; has been shown during Brussels' Offscreen Festival in 2009 at Cinema Nova. At the end of each chapter, a sequenced is being looped until the viewer understands one can "choose" between two possible sequels using the remote control, thus offered a limited number of scenarios and possible plot explanations.

SLIDERS, [10] presented during a workshop and performed at iMAL in Brussels in 2008, offers a real-time audiovisual interface aimed at recomposing movies "live", from fragments hosted in a database. However, not much information is available on the methods with which fragments can be queried and put in relation to one-another.

2. TWO INSTALLATIONS, TWO ARTISTIC SUBPROJECTS

Two artistic projects have been in development in parallel with this numediart project. Both have been offered a full-time residency at:

BRASS
Avenue Van Volxem 364
B-1190 Forest
Belgium

2.1. *Méta-crâne* by Thomas Israel

2.1.1. Artistic intention

"*Méta-crâne* is a technical reconstruction of a symbolical process that we all know well but often repress: free association. A network of neural activities is being simulated and processes a certain amount of new events, past episodes and ancient memories that are presented to the "*spect-actor*" within an immersive environment. Its purpose: transposing self-disclosure and creating new meanings inside an interactive object."

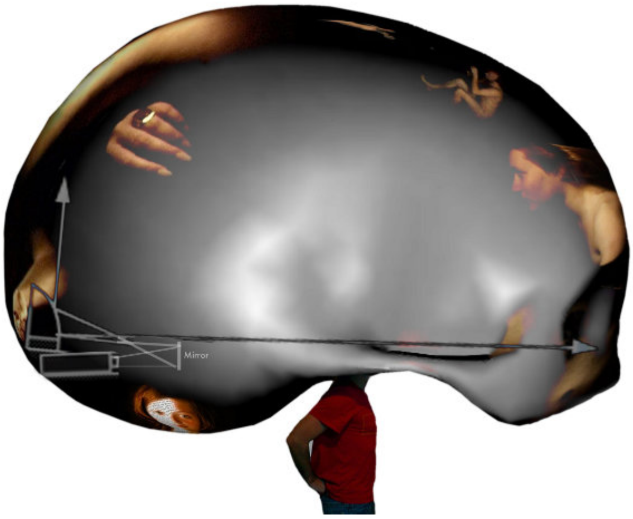


Figure 1: Side-view mockup of the installation being visited by a *spect-actor*.

"When the *spect-actor* enters the *Méta-crâne*, as depicted in figure 1, he gradually perceives the inherent and emergent behavior of the installation, while the flow of audiovisual fragments which are sequenced around him tends to take into consideration the color of his clothes, his body rhythm revealing his serenity or discomfort, and so on..., but without being granted "1:1" control. The initial content of the multimedia database presented to the *spect-actor* is composed of material extracted from Thomas Israël's previous productions [15], sequenced in real-time and processed with spatio-temporal effects, as in figure 2; and gets fed back from records of the *spect-actor*'s behavior."

2.1.2. Prototype setup

The first *Méta-crâne* prototype has been designed and realized by Thomas Israel and Thierry Sablon [23]. The dome of 3 m diameter features a webcam for gestural analysis of crowds, a Projection Design F2 sx+ wide SXGA+ video projector and a spherical mirror for the projection as illustrated in figure 3; 5 FAR Audio OBS active loudspeakers for sound spatialization. The software architecture of the installation has been developed by Thomas Israel and Laura Colmenares Guerra [14] using the Isadora modular visual programming environment [27].

Figure 2: A mashup of visuals extracted from Thomas Israël's installation *Caresse moi!* (2006).

2.2. *HUM* by François Zajéga

2.2.1. Artistic intention

"*HUM* is an audiovisual digital artwork where interactivity is used to enhance the dialog between the artwork and its audience. Through his motion, the visitor handles a visual and sonic shape and, in the same time, feeds and educates *HUM*. Once the visitor stops moving inside the installation, ready to listen, *HUM* gives him a response through the same media by mixing the long-term trends learned since the beginning of its life and the specific behaviour of the visitor, highlighting the potential richness of the installation. The aim, and unique criteria of quality, is to increase the energy emitted by the visitor by encouraging him to go out of standard behavioural scheme. The visitor is this both creator and spectator, guiding and guided by *HUM*."

2.2.2. Prototype setup

The first *HUM* prototype features a large projection screen, a projector, 4 speakers for sound spatialization, a webcam for crowd behaviour analysis, 3 computers to run the analysis and rendering.

3. *MÉTA-CRÂNE*: TECHNOLOGICAL CHALLENGES

This part of the project was done in the context of a close collaboration between `numediart` and the artist Thomas Israel.

3.1. Main contribution: navigation in an audiovisual database based on video similarity

One of the goals of the *Méta-crâne* installation was how to make the projected videos follow each other so that it feels natural for the spectator, according to her/his behavior. For example, if the spectator moves rapidly, a group of videos that convey a sense of speed should be displayed. This implies that the videos ought to be organized so that similar videos can be retrieved easily. Similarity between videos was computed using the MediaCycle software developed in the `numediart` project Media Cycle (# 5.3). This

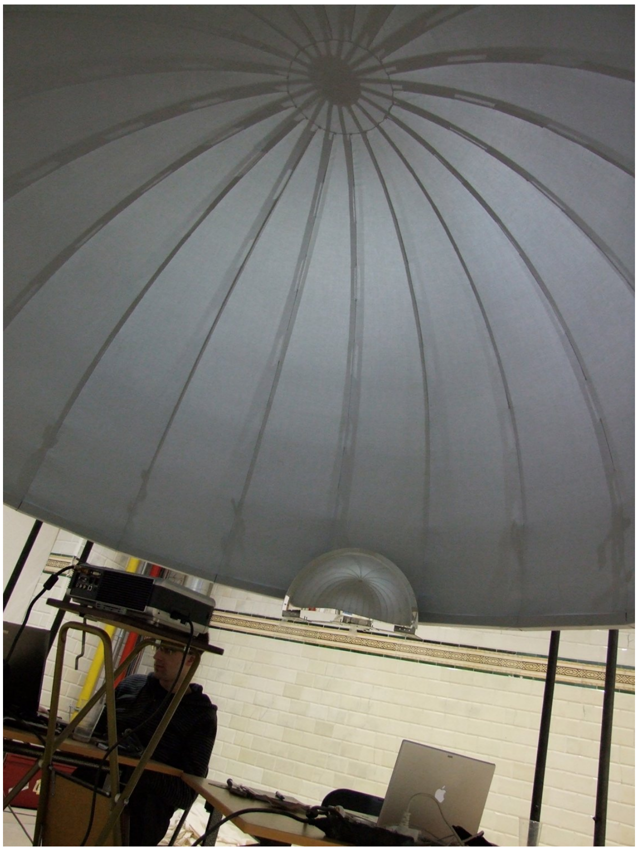


Figure 3: First *Méta-crâne* hardware prototype, designed and realized by Thomas Israel and Thierry Sablon, featuring a video projector and a spherical mirror

software was originally developed for sounds in project Audio Cycle (# 4.1), then adapted to images and now videos at the time of writing.

Features extraction is based on the OpenCV library and has been written in C++. In order to make the classification results easily available for the artist, a text file has been generated which respects the text-read format in the Isadora software [27].

Several criteria were investigated to compute similarity:

- color, luminosity and saturation (corresponding to hue, saturation and value of the HSV color space). These quantities were computed for each frame of the video, then the average and standard deviation were used for comparison between videos, as illustrated in figure 4.
- speed, defined as the pixel-per-pixel rate of change from a frame to another, either in RGB of HSV color spaces. The average and standard deviation of these quantities were also used for comparing between videos.

Preliminary tests with a few users indicated that speed-based descriptors correspond more closely to human perception of videos similarity, and was therefore used in the artistic installation.

Other features like textures, shapes or faces detection could be added in the future and will be investigated in upcoming numediart projects, e.g., 'Video navigation tool: application to browsing a database of dancers' performances (#07.3) and can potentially refine the current measures of similarity between videos.

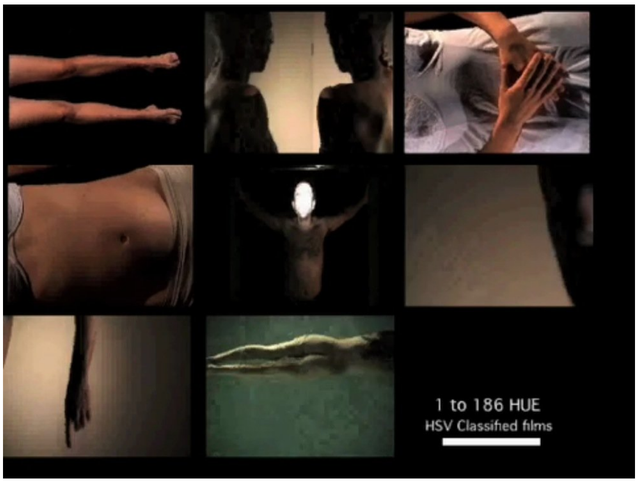


Figure 4: *Thumbnails of images classified by similarity along the Hue component of the HSV color space*

3.2. Technological forecasting: 3D projection

Many 3D projection screen geometries have been developed so far, namely the Panoscope 360 [9], and its upside-down equivalent the Panodome, both from Luc Courchesne and the Société des Arts Technologiques of Montreal, Canada. Thomas Israel chose a top hemisphere, ressembling the most to a skull, and allowing visitors to move easily under.

So as to reduce costs as much as possible, we decided to use Paul Bourke's design [5, 4] requiring only one projector, as opposed to multiple projectors, and a spherical lens instead of fisheye lenses.

To enhance the viewer's immersion, 2D images or videos displayed on a non-plane surface should be pre-warped accordingly to the projection screen geometry so as to have them look accurate. Many solutions exist, but wouldn't suit our needs (ie integrated as most as possible inside the Isadora framework [27] running under OSX): Eluminati's OmniMap API [11] is running on Microsoft Windows only, Territoires Ouverts' lightTWIST [26] didn't support hemispherical surfaces at the time of writing. Olivier Meunier used the 3D modeller Blender and the Ashvid video texture plugin for his installation *Real Unreal* [21], but it wasn't made available to the public.

Closer to a promising result, we tested Paul Bourke's algorithm released as a Quartz Composer object (a node-based visual programming language available on MacOS X system) and that could be used as a FreeFrame plugin (a cross-platform open framework for developing video effects plugins) inside Isadora. This object is then quite easy to use : an incoming image is distorded by the chosen matrix file and the resultant image is then output. If the distortion matrix is well defined accordingly to the dome geometry, the projection works quite perfectly. Paul Bourke provides another software application for the generation of the distortion matrix file that is characteristic of the chosen screen. Two useful matrix files are provided by default : the one of a fish eye lens and the one of a spherical mirror. Due to screen/matrix calibration issues, the result using this pre-warping feature wasn't satisfying enough to justify its use, especially because of to the added processing cost.

3.3. Technological forecasting: sound spatialization

Many spatialization techniques have been developed so far and can be somehow classified in two main categories: “individualized spatialization”, working for one listener at a time preferably with headphones and customized accordingly to one’s skull sound characteristics (namely Head-Related Transfer Functions), and “room spatialization”, including techniques such as Vector Based Amplitude Panning (VBAP), Wave Field Synthesis, Ambisonics... [24]. Improving such techniques was out of the scope of this project, we focused on the integration of sound spatialization inside Isadora [27] according to the movement of the visitors. The minimal number of loudspeakers can be theoretically determined specifically to some spatialization techniques, especially Ambisonics. We opted a 5-speaker system (a quadriphony - 4 speakers in a square evenly spaced around the listeners’ ears in the same plane - and a zenith speaker) that we found cost-effective in terms of hardware and software.

Spatialization is difficult in Isadora as there is no way to connect the sound output of a movie to physical sound outputs. Indeed, neither “Movie Player” nor the “Sound Movie Player” objects have a sound output! Rather, the sound is sent directly to only one output, the default one for “Movie Player” and one that can be chosen for the “Sound Movie Player”. The only way to send that same movie sound simulateously to other sound output channels is to add, for each additional channel, a “Sound Movie Player” that plays the sound of the movie.

We therefore planned to use the Zirkonium [31], a software developed at ZKM that allows to place the loudspeakers in a virtual space and then set the coordinates of the desired sound position using OSC messages and let the algorithm compute the volumes to each speaker. The Zirkonium is a stand-alone application that creates a virtual interface driver that programs can connect to in order to send it their sound channels. We have used it successfully before with Max/MSP. Zirkonium then outputs the spatialized sounds to the sound interface defined by the user in the preferences settings. Channels can be freely assigned to individual speakers. We managed to generate the needed OSC messages from inside Isadora to move the sound position in Zirkonium.

Unfortunately, in Isadora, in order to choose a sound output, you have to go to the "sound Output Setup" to define the mapping between the internal Isadora sound channel outputs to virtual external channels. The conversion to the “real channels” of an interface can only be done in Mac OS “Audio MIDI Setup”: Isadora can only access the interface defined as the “Default Output” in Mac OS “Audio MIDI Setup”.

As the Zirkonium driver is a virtual interface, it would mean choosing as Mac OS default sound output a virtual interface rather than to a physical one. It therefore proved impossible to connect the output of Isadora to the input of the Zirkonium. Every attempt was granted with a crash.

A possible solution would be to spatialize sounds in Max/MSP, with or without the Zirkonium. But that would mean that all sound files should be extracted from the video files and that the engine that selects the videos, depending on the criteria defined elsewhere in this document, would have to synchronously start the video sequences in Isadora and the sound sequences in Max/MSP. This hasn’t been done yet.

We also considered using audioTWIST [25] and its successor Audioscape [29, 2], but the current version weren’t satisfying enough due to extensive compilation issues.

4. *HUM*: TECHNOLOGICAL CHALLENGES

This part of the project was done in the context of a close collaboration between numediart and the artist Francois Zajéga, who was in the process of creation of his new art installation named *HUM*. *HUM* is a visual and sonic interactive installation, where the behavior of the visitors is captured by means of a video camera and then analyzed in order to control both visual and sound rendering modules. *HUM* was presented in BRASS cultural center (Forest, Belgium) in May 2009 and will be part of Digital Arts festival in Brussels during fall 2009.

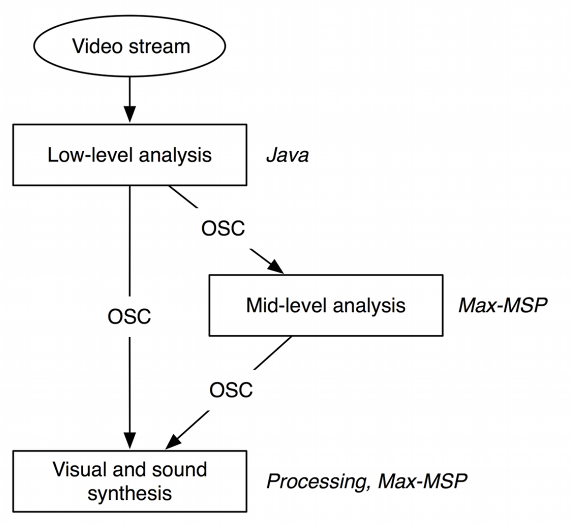


Figure 5: *Generic architecture of HUM*

During this three months project we mainly focused on two blocks of the pipeline introduced in Fig.5 :

- the low-level video analysis, which takes as input the video flux grabbed by a camera and provides a low-level description of the scene, mainly corresponding to the position of each visitor moving in the scene (i.e. ’blob’)
- a high-level long-term analysis of the scene, which takes as input the result of the low-level analysis i.e. the position of each blob detected in the scene. The aim of this second stage of analysis is to provide a more precise description of the scene, by considering the temporal evolution of one blob behavior in several time spans.

In the following sections, we will first describe the low-level analysis module, developed in Java, and then the mid-level analysis modules written in Max-MSP, a programming environment dedicated to audio and video interactive applications and widely used in digital arts community. The communication between each of these modules is based on the Open Sound Control protocol (OSC).

4.1. Main contribution: low-level analysis by video motion tracking

By low-level analysis we mean the analysis of the video stream grabbed by a camera placed on top of the scene and capturing the visitors. A video motion tracking analysis is achieved and provides a number of basic features characterizing the scene, such as the position of each detected blob, its size etc... By video motion tracking, we mean the process of locating one or several moving objects in time using a camera. An algorithm analyses the video frames and outputs the location of moving targets within the video frame.

For the creation of *HUM*, F. Zajéga has implemented, in collaboration with numediart researchers, a video motion-tracking module. This module can be used as a standalone application and is able to communicate with others programs with the Open Sound Control protocol. It is written in Java, and both the application and the source code have been made publicly available at the end of the project on the numediart website.

We give here a brief technical description of this module; as shown in Fig.6, the process is divided in five steps:

1. Initialization of the process, which mainly consists in creation a reference image that will be used for subtracting the background of the incoming stream.
2. Background subtraction: the incoming image is compared with the reference image in order to find elements having appeared.
3. Pixels grouping: pixels are grouped in cells according to a grid over the image
4. Cells grouping: cells of pixels are grouped in 'cells groups'
5. Cells groups tracking: cells groups are then compared to the ones of the previous frame

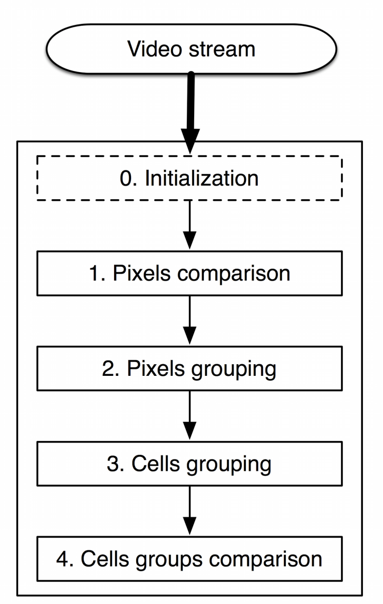


Figure 6: *Overview of the video tracking analysis*

This chain, except the initialization, is executed for each incoming frame. Here are some details about each of these steps:

4.1.1. Step 0: initialization

The initialization phase mainly consists in creating a reference image that will be used in the sequel for subtracting the background of the incoming stream. This can also be done again in case of changes in lightning conditions in the scene.

4.1.2. Step 1: pixels comparison

The reference image and the current image are represented as an array of pixels, each pixel being characterized by three value corresponding to its red, green and blue levels. Each pixel of the current frame and the reference image are compared and a threshold, which can be set in the graphical user interface of the application (Fig.9), is used to determine if a pixel is considered as different or identical than its parent in the reference image. A pixel is considered different if one of these conditions is true:

- if the absolute value of the difference between the current pixel red or green or blue value and the reference pixel red or green or blue is bigger than the threshold;
- or if the absolute value of the addition of two differences (red+green for instance) is bigger than the threshold

If at least one of these conditions is true, the pixel is considered as different than the reference and also set as "active". This implies the activation of the second step.

4.1.3. Step 2: pixels grouping

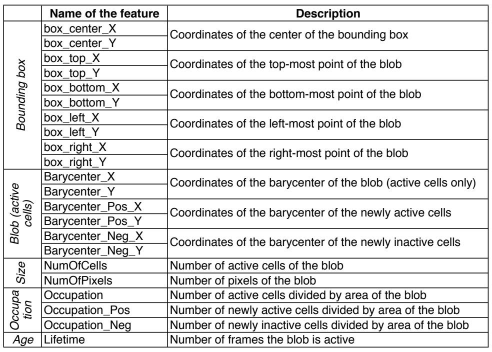
The image is virtually covered by a grid. When an active pixel is detected, it is placed in one of the cells of this grid. A cell is not only an array of pixels and includes additional information:

- It has a short memory (several frames) used to smooth its 'occupation'. The occupation of one cell is a division of the number of pixels it contains on the size of the cell. The short-term memory is a way to know if the cell is getting more or less active, depending on the rise or the diminution of pixels number.
- It contains also the position x and y of the extreme pixels (top, right, bottom and left).
- It obviously contains the information of the current occupation of the cell, and the average of the red, green and blue value of the pixels.

Once last pixel is compared, the application jumps to step 3.

4.1.4. Step 3: cells grouping

This step consists in grouping cells of pixels in entities called "cells groups" or "blobs". To start a cells group, there must be at least one pixel in the cell. Once the group is setup, any cell containing pixels or still active (due to the smoothing, a cell can be active even if it is empty) can respond positively to the test explained here below. Starting with the most active cell, the grouping function searches around the current cell if there are other not empty cells next to it. There are eight possible positions: above, sharing right-top corner, right, sharing right-bottom corner, bottom, sharing left-bottom corner, left and sharing left-top corner. The analysis is made in this order, each position represented by a letter from A to H. If it finds some, it stores the cells in the cells group. Once no more cells are found in the neighborhood of all the cells stored in the cells group, the group is closed and, if it remains active cells, the function declares a new cells group. A cells group is then encapsulated in a rectangular bounding box (Fig.8), as it is done in most of the motion tracking systems. The cells group also contain average of the features of each cell and position of the extreme points of the bounding box (Fig.7).



	Name of the feature	Description
Bounding box	box_center_X	Coordinates of the center of the bounding box
	box_center_Y	
	box_top_X	Coordinates of the top-most point of the blob
	box_top_Y	
	box_bottom_X	Coordinates of the bottom-most point of the blob
	box_bottom_Y	
	box_left_X	Coordinates of the left-most point of the blob
	box_left_Y	
	box_right_X	Coordinates of the right-most point of the blob
	box_right_Y	
Blob (active cells)	Barycenter_X	Coordinates of the barycenter of the blob (active cells only)
	Barycenter_Y	
	Barycenter_Pos_X	Coordinates of the barycenter of the newly active cells
	Barycenter_Pos_Y	
	Barycenter_Neg_X	Coordinates of the barycenter of the newly inactive cells
	Barycenter_Neg_Y	
Size	NumOfCells	Number of active cells of the blob
	NumOfPixels	Number of pixels of the blob
Occupa tion	Occupation	Number of active cells divided by area of the blob
	Occupation_Pos	Number of newly active cells divided by area of the blob
	Occupation_Neg	Number of newly inactive cells divided by area of the blob
Age	Lifetime	Number of frames the blob is active

Figure 7: *Features provided by the video motion-tracking analyser*

4.1.5. Step 4: blobs tracking

At the beginning of the step 3, the previous cells groups are saved. Once the cells grouping process is finished, the new cells groups are compared to the former ones. The mapping is done on basis of the distance of each border and of the center. The cells group id is adapted and the lifetime of the cells group incremented.

This application has been implemented in Java, and is based on libraries from the programming language and integrated development environment (IDE) Processing [22]. Processing is widely used by the interactive arts community and builds on the graphical capabilities of the Java programming language, simplifying features and creating a few new ones. A Graphical User Interface allows to easily set some parameters of the application, such as the sensitivity of the video segmentation, the minimum size of a blob and the maximum number of blobs detected in the scene (Fig. 9).

4.2. Main contribution: high-level analysis

The second part of this project aimed to develop tools for a high-level long-term analysis of the scene grabbed by the video camera. It takes as inputs the positions of the blobs detected by the motion tracking analyser (low-level analysis) described above and provides features characterizing the temporal evolution of the blobs behavior on different time spans.

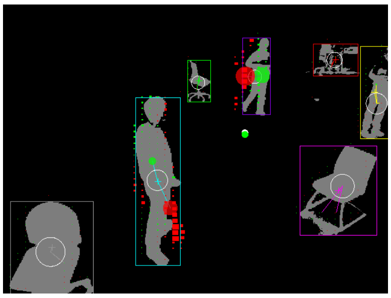
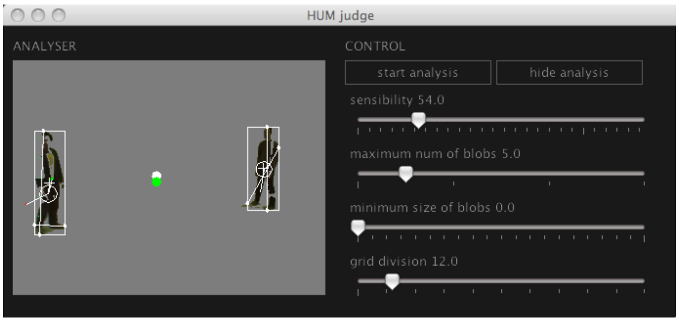


Figure 8: *Result of a motion tracking analysis*



sensibility 54.0

maximum num of blobs 5.0

minimum size of blobs 0.0

grid division 12.0

Figure 9: *Snapshot of the video motion tracking module*

4.2.1. Preprocessing of the data

We develop some tools in the Max/MSP programming environment, mainly patches and abstractions, for the long-term analysis of some motion features extracted by the tracking module. By long-term analysis we mean using basic statistical tools to describe the evolution of a variable during a certain time span. It could also be interesting to analyze the same feature on different time spans.

The first step of the process is a smoothing of the data coming from the video analysis module. Typically these may be noised by some jitter caused by changes of lightning conditions on the room and need to be cleaned before further analysis. The smoothing step consists of a median filtering followed by a low-pass filtering. The median filter is a non-linear digital filtering technique, often used in image processing to remove noise from signals. The idea is to examine a sample of the input and decide if it is representative of the signal. This is performed using a sliding window consisting of an odd number of samples. The values in the window are sorted into numerical order; the median value, the sample in the center of the window, is selected as the output (cf. Fig.10).

The output of the median filter is then smoothed using a first-order lowpass filter, where the current output sample *y(n)* depends on both the current input *x(n)* and the previous output sample *y(n-1)*:

$$y(n) = y(n-1) + ((x(n) - y(n-1))/\alpha), \qquad (1)$$

alpha being a parameter to tune the smoothing effect of the filter.

This smoothing stage is implemented in Max-MSP using the *ej.mmmm* object developed by Emmanuel Jourdan for the median filtering and the *slide* object for the low-pass filtering. It allows to remove noise and artifacts introduced in the video analysis stage (Fig.11)

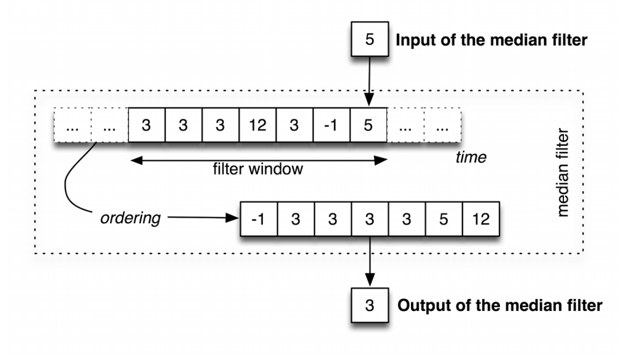


Figure 10: *A median filter is used to remove noise in a signal*

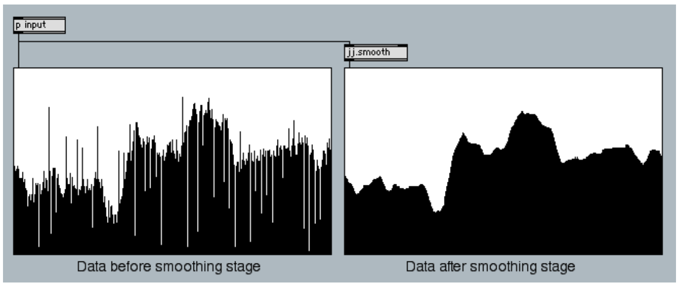


Figure 11: *A preprocessing stage is used to smooth data coming from the motion tracking analyzer before mid-level analysis*

4.2.2. Long-term analysis

Once the data streamed from the motion-tracking module are smoothed, the next step of the analysis chain consists in buffering the data in a fixed length analysis window. The length of the window can be parameterized and corresponds to the analysis time-span. The data buffering works as a shifting register (Fig.12): at the beginning of the analysis the window is filled by the incoming data, and once the window is full, a first in-first out strategy is used to manage the input/output of the window: the incoming data takes the first place of the window each data is shifted and the oldest value is dropped. This process enables to maintain a short-term memory of the system and gives it a reminiscence of the behavior of a variable. In the following, this is referred as the "sliding analysis window".

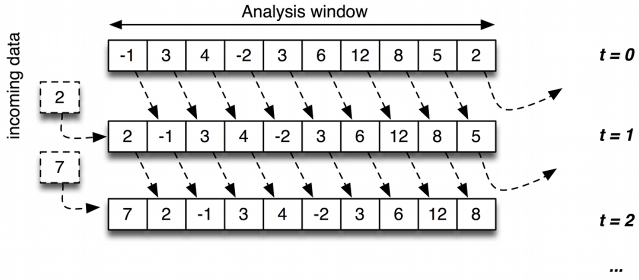


Figure 12: *A preprocessing stage is used to smooth data coming from the motion tracking analyzer before mid-level analysis*

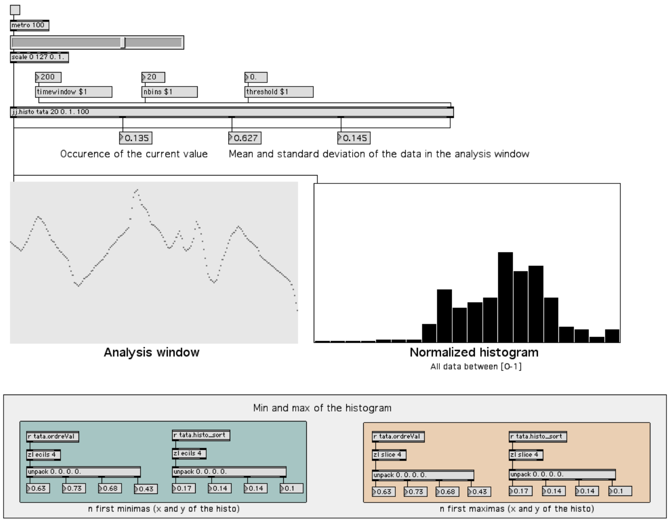


Figure 13: *A preprocessing stage is used to smooth data coming from the motion tracking analyzer before mid-level analysis*

The simplest way to describe the data in the analysis window is to compute the statistical moments of the sequence. The most relevant moments are the mean and the standard deviation. The median of the sequence could be also useful. The mean and standard deviation of a sequence are given by the following formulas respectively:

$$\bar{x} = \frac{1}{n} \cdot \sum_{i=1}^{n} x_i \tag{2}$$

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2} \tag{3}$$

The mean describes the central location of the data, whereas the standard deviation describes their dispersion/variability in the sequence. A low standard deviation indicates that the data points tend to be very close to the same value (the mean), while high standard deviation indicates that the data are 'spread out' over a large range of values.

Another useful tool for characterizing the evolution of a sequence of data is its histogram. The histogram of a sequence is a summary graph showing a count of the data points falling various ranges. It provides a rough approximation of the frequency distribution of the data. No tool was available in the Max-MSP programming environment for a real-time histogram-based analysis of an incoming stream of data; we also developed an abstraction *num.histo* relying on objects natively available in the Max-MSP distribution and third part objects publicly available. In the future, we plan to write a Max external in order to optimize the computation of the histogram. The values of the histogram are normalized between [0-1] so that individual bins represent the fraction of the total number of events assigned to the entire histogram. It is also possible to threshold the histogram so that all bins whose value is below the threshold factor are set to 0. Finally the abstraction provides the n first minima and maxima of the histogram, and the bins associated to those values (Fig.13).

This tool has been used in *HUM* for analyzing the way a visitor was occupying the space during a certain amount of time (i.e. the sliding analyzing window). By computing the histogram of the position taken by the visitor within the analysis window, we were able to provide a cartography of 'cumulated space occupation' of a visitor or a group of visitors. The scheme of this kind of analysis is described in Fig.14: first the space, considered as a 2-D plane, is divided in a grid of n x n cells, and position (x,y) of a blob is converted in a z position in the grid defined as z = x + n*y (step 1). Z positions are then stored in a sliding analysis window in order to provide a memory of the last positions of the studied blob (step 2), and a histogram of this sequence of positions is computed for each incoming video frame (step 3). This histogram is then used to construct a map of the areas which have been occupied by the blob within the analysis window. This map provides for each cell of the grid - i.e. corresponding to each bin of the histogram - the number of time the visitor has occupied this position within the analysis window (step 4); we used a display inspired by meteorological cartography to visualize the result of this analysis: the space is visualized as a 2-D plane, and a color is associated to each cell of the grid following the amount of time it has been occupied; a black cell corresponds to a cell which has never been occupied, a red one to a cell occupied during a long-time. This tool allows to characterize the displacement of a visitor in the installation by providing information such as:

- has the visitor stayed in a close area or has he/she visited a large part of the space
- which places have been more occupied by the visitor
- is the current position a 'hot' or 'cold' position, i.e. a position which has been a lot occupied or not

Those information characterize the displacement of one visitor in the installation but one could also apply the same kind of analysis to a group of visitors. These information can then be used by the artist in the design of the interactions. In *HUM*, the result of this analysis was used to control sound spatialization parameters and triggering of sound samples. We plan to keep on investigating more complex strategies of mapping relying on this analysis tool in future versions of the installation.

5. PERSPECTIVES

A 3-month project is quite short to cover such a long list of tasks. We hope that our technological forecasting will prove to be helpful to the artists for their ongoing efforts to finalize all the modules of their installations.

We need to implement an OSC communication in the next HyForge applications so as to allow such artistic projects to benefit from our future progress in the field, here enabling to query similar videos directly from the Isadora framework.

We will continue the development of the MediaCycle software that was used to compute similarities between videos. In particular we will explore other descriptors of similarity between videos, as well as means of organizing more efficiently the videos when the database is very large.

HUM has already been presented publicly at BRASS in May 2009, and an improved version of the installation will be presented by the end of this year in Brussels. The major improvements will aim to enrich the interaction between the visitor and the resulting sound and image. The possibility of using algorithms inspired by artificial life [12, 16] to educate the system will also be explored by François Zajéga and the numediart team, who plan for the future to keep working together in a close collaboration.

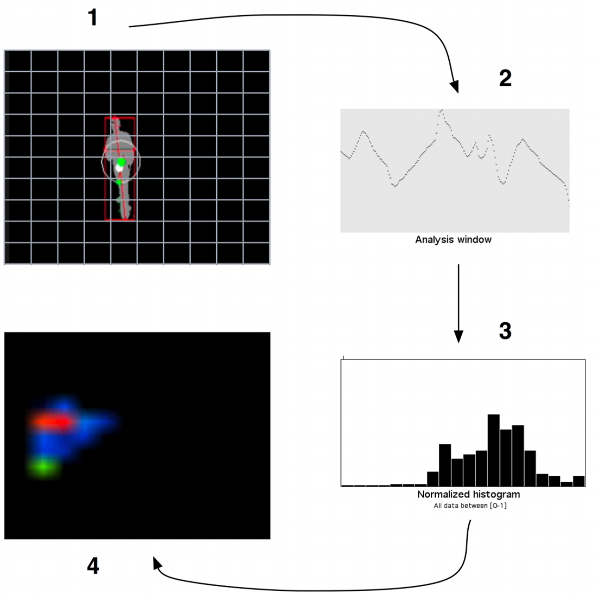


Figure 14: *A preprocessing stage is used to smooth data coming from the motion tracking analyzer before mid-level analysis*

6. ACKNOWLEDGMENTS

This work has been supported by the numediart research project, funded by Région Wallonne, Belgium (grant N°716631).

We would like to thank all the artists that made this collaboration possible, in order of appearance: Thomas Israel [15], François Zajéga [30], Laura Colmenares Guerra [14] and Thierry Sablon [23].

We warmly greet Roger Burton for our welcoming at BRASS, granting us residencies for both artistic projects, a venue and proper catering for the end project presentation.

We gratify Rémy Labbé, M.Sc. student at UCL-TELE, for his help and participation throughout the project workshop, when trying to find the best-matching sound spatialization software.

We also want to thank Paul Bourke and Sebastien Roy for giving us time to discuss about dome projection.

7. REFERENCES

7.1. Scientific references

[4] Paul Bourke. "Digital Fulldome, Techniques and Technologies". In: *Course prepared for Graphite (ACM Siggraph)*. 2007. URL: http://local.wasp.uwa.edu.au/~pbourke/papers/graphite2007/graphite2007.pdf. P.: 11.

[5] Paul Bourke. "Using A Spherical Mirror For Projection Into Immersive Environments". In: *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia, Graphite (ACM Siggraph)*. 2005. URL: http://local.wasp.uwa.edu.au/~pbourke/papers/graphite2005/graphite.pdf. P.: 11.

[7] Alvaro Cassinelli and Masatoshi Ishikawa. "Khronos Projector". In: *International Conference on Computer Graphics and Interactive Techniques, ACM SIGGRAPH Emerging technologies*. 2005. P.: 9.

[9] Luc Courchesne, Guillaume Langlois, and Luc Martinez. "Where are you?: an immersive experience in the panoscope 360". In: *Proceedings of the 14th annual ACM International Conference on Multimedia (MM)*. 2006. P.: 11.

[10] Jean-Marie Dallet, Christian Laroche, and Frédéric Curien. "SLIDERS: a collective experience of interactive cinema". In: *Proceedings of the 14th annual ACM International Conference on Multimedia (MM)*. 2006. P.: 9.

[12] Dario Floreano and Claudio Mattiussi. *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. Intelligent Robotics and Autonomous Agents. The MIT Press, 2008. ISBN: 9780262062718. P.: 16.

[13] Oliver Grau. *Virtual Art: From Illusion to Immersion*. Leonardo. The MIT Press, 2003. ISBN: 0-262-07241-6. P.: 9.

[16] Eduardo Kac, ed. *Signs of Life: Bio Art and Beyond*. Leonardo Books. The MIT Press, 2006. ISBN: 9780262112932. P.: 16.

[17] Robert Kronenburg. *Flexible: Architecture that Responds to Change*. Laurence King Publishers, 2007. ISBN: 9781856694612. P.: 9.

[18] George Legrady and Timo Honkela. "Pockets Full of Memories: an interactive museum installation". In: *Visual Communication* 1.2 (2002). Pp. 163–169. P.: 9.

[19] Lev Manovich. *The Language of New Media*. Leonardo Books. The MIT Press, 2002. ISBN: 9780262632553. P.: 9.

[20] Lev Manovich and Andreas Kratky. *Soft Cinema: Navigating the Database*. DVD & Booklet. MIT Press, 2005. ISBN: 9780262134569. P.: 9.

[24] Alois Sontacchi and Robert Höldrich. "Getting Mixed up with WFS, VBAP, HOA, TRM... From Acronymic Cacophony to a Generalized Rendering Toolbox". In: *DEGA Wave Field Synthesis Work*. 2007. P.: 12.

[28] Stephen Wilson. *Information Arts: Intersections of Art, Science, and Technology*. Leonardo. The MIT Press, 2002. ISBN: 0-262-23209-X. P.: 9.

[29] M. Wozniewski, Z. Settel, and J.R. Cooperstock. "AudioScape: A Pure Data library for management of virtual environments and spatial audio". In: *Pure Data Convention*. 2007. P.: 12.

7.2. Artistic references

[1] Alterface. *The Sphere*. 2009. URL: http://www.alterface.com/en/science_centers/the_sphere/. P.: 9.

[3] Pascale Barret. *Synapse 2.0*. Interactive video installation. URL: http://www.imal.org/synapse/. P.: 9.

[6] Alvaro Cassinelli. *The Khronos Projector: a video time-warping machine with a tangible deformable screen*. Online description containing videos, slides and a Processing demo applet. URL: http://www.k2.t.u-tokyo.ac.jp/members/alvaro/Khronos/. P.: 9.

[8] Daryl Cloran, Anita Doron, and Mateo Guez. *Late Fragment*. 2009. URL: http://www.latefragment.com. P.: 9.

[14] Laura Colmenares Guerra. URL: http://www.ulara.org. Pp.: 10, 16.

[15] Thomas Israel. URL: http://www.thomasisrael.be. Pp.: 9, 10, 16.

[21] Olivier Meunier. *Real Unreal*. 2006. URL: http://www.ogeem.be. Pp.: 9, 11.

[23] Thierry Sablon. URL: http://www.tysablon.eu. Pp.: 10, 16.

[30] Franccois Zajéga. URL: http://www.frankiezafe.net. Pp.: 9, 16.

7.3. Software and technologies

[2] "Audioscape". URL: http://www.audioscape.org. P.: 12.

[11] Eluminati. "OmniMap API". URL: http://www.elumenati.com/products/omnimap.html. P.: 11.

[22] "Processing IDE". URL: http://www.processing.org. P.: 14.

[25] TOT [Territoires Ouverts - Open Territories]. "audioTWIST". immersive 3D audio architecture. URL: http://tot.sat.qc.ca/logiciels_audiotwist.html. P.: 12.

[26] TOT [Territoires Ouverts - Open Territories]. "lightTWIST". image deformation for 3D projection. URL: http://tot.sat.qc.ca/logiciels_lighttwist.html. P.: 11.

[27] TroikaTronix. "Isadora". URL: http://www.troikatronix.com/isadora.html. Pp.: 10–12.

[31] ZKM. "Zirkonium". URL: http://www.zkm.de/zirkonium. P.: 12.