# BODILY BENCHMARK: GESTURAL/PHYSIOLOGICAL ANALYSIS BY REMOTE/WEARABLE SENSING

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

[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

## ABSTRACT

This paper presents a `numediart` project in collaboration with two artistic projects: *Musichorégraphie d'appartement* by André Serre-Milan [63] and *BioDiva* by Laurence Moletta (Nios Karma) [34]. The scope of this project was to offer technological forecasting and development consultancy to these two artists that both share a common goal of using gestures to control on audiovisual rendering on stage (see section 2).

We developed a first prototype for synchronized recording, visualization and editing of multimodal signals (audio, video, sensors) (see section 3.1). We updated our technologies for gesture recognition using Dynamic Time Warping (DTW) (see section 3.2) and mapping interpolation (see section 3.3). We ported our long-term attention computer vision algorithm from EyesWeb to Max / MSP (see section 3.4). We achieved initial promising results for the prototyping of the *BioDiva* gestural interface (see section 4.1) and for the analysis using the long-term attention model of the *Musichorégraphie d'appartement* recording sessions (see section 4.2).

We plan to improve our multimodal signal recording and analysis tool towards a more efficient annotation tool (see section 5.1). We plan to build a dedicated wireless sensor interface to overcome all the related issues we met so far (see section 5.2). We plan to use new textiles for the improvement of sensors, sensor interfaces and integration in performers' costumes (see section 5.3).

## KEYWORDS

Augmented performances, wearable sensors, synchronized multimodal recording, gesture recognition, gestural interface design, computer vision, intelligent textiles, textile sensors

## 1. SUMMARY OF PREVIOUS NUMEDIART WORKS ON SENSORS USED IN AUGMENTED PERFORMANCES

Within the `CoMedia` research axis, this project aims at analyzing gestural and bodily behaviors so as to augment audiovisual performances [9], by comparing and combining a certain selection of remote and wearable sensing techniques [45, 69]:

- biomechanical sensors, especially accelerometers;
- computer vision using cameras;
- biosensors, particularly for muscle tension (EMG), respiration and heart beats (ECG/EKG);
- intelligent wearable textiles.

A first comparison of sensors has been made through Sensor-Based Mini-CoMedia (#01.2) [39]. The idea is to take further the study of sensors (wearable, like flexometer, Wii Remote... or external, like video camera...) in order to have a strong expertise on the different possibilities of the sensors.

For an in-depth review of artistic works featuring sensors and/or cameras, we recommend [79].

### 1.1. Biomechanical

One of the goals of Dancing Viola (#04.2) [73] was to recognize gestures of a dancing viola player. Three elements of this project can be reused and taken further:

1. a module for realtime gesture recognition based on the Dynamic Time Warping (DTW);
2. a module for the extraction of features directly from the preprocessing of the sensors using wavelet analysis;
3. a Max/MSP/Jitter tool for mapping using a solar system metaphor-based interpolation.

All of these three elements were a very good basis to start our research on this present project.

### 1.2. Cameras for Computer Vision

In Tracking-Dependent and Interactive Video Projection (#03.1) [42] and Behavioral Installations (#05.2) [19], real-time video image analysis methods had been developed so as to extract features from a video stream in order to characterize the scene or to follow different elements on it.

### 1.3. Biosignals

In Breathing for Opera (#02.2) [20], we took further the early research done on project Sensor-Based Mini-CoMedia (#01.2) [39] aiming at studying breathing sensors. We tested several breathing sensors and extracted features from the received signals. We made an early recording / playing patch for signal analysis. Using this patch, data could be recorded from sensors, video and sound separately and every part of the recorded file could be accessed in order to focus on specific time frames.

## 2. TWO ARTISTIC PROJECTS IN COLLABORATION

### 2.1. *Musichorégraphie d'appartement* by André Serre-Milan

In *Musichorégraphie d'appartement* [63], we are following a woman in her appartement, in her daily life. Here, the body and his "traditional", daily attitudes and gestures, are used like musical and choreographic potentials in order to write a double partition.

To use daily gestures as musical potential, two directions will be followed :

1. an INNER one, with an embedded sensor's system on the interpreter's body: it will essentially be made around accelerometers and inclinometers in order to: on the one hand,

recognize gestures and and on the other hand, to extract the maximum features (it is possible from) speed of a gesture, direction of the gesture.

2. an OUTER one, made with two video cameras: one facing the stage from the foreground and an other one, facing the stage from above. This second system is used to have another scale, to use the body as a part of a larger space and to be able to track it in 3D.

Musichorégraphie d'appartement is a project of André Serre-Milan [63] (conception and musical composition) and Vanessa Le Mat [43] (choreographer and interpreter). This project was commissioned by Art Zoyd [84].

### 2.2. *BioDiva* by Nios Karma

Nios Karma [34] blends improvised borborygmous vocals and electroacoustic soundscapes on stage. As illustrated on Fig. 1, her performance setup is heavily constrained by the microphone standing beneath that records her voice, the laptop and soundcard laying below that process and synthesize the sound, and the MIDI fader box that controls the sound. As she naturally surrounds her vocal performance with bodily movements and yearns for moving in larger stage space, her goal from this collaboration is to set her gestures free from these hardware devices by using wireless sensors, the computed gesture analysis augmenting her performance.

Figure 1: Evolution of Nios Karma's performance setup constrained by microphone, computer stand, midi fader box, etc...

*Bio Diva* has been in residency from April to May at BRASS, Avenue Van Volxem 364, B-1190 Forest, Belgium; and in June at RTBF's Auditorium Abel Dubois, Esplanade Anne-Charlotte de Lorraine, B-7000 Mons, Belgium.

## 3. NEW AND IMPROVED TOOLS

### 3.1. Synchronized recording, visualization and editing tool

In the context of an artistic performance, the creation of this kind of tool had two main beginning points :

- record synchronously multimedia performances (data from sensors and / or video analysis, sound and video) so that not to be performer-dependent regarding to the analysis part
- being able to extract from the played back data, patterns, in our case gestures, that could directly feed the Dynamic Time Warping (DTW) algorithm for later possible gesture recognition. In the context of *Musichorégraphie d'appartement*, for example, as it is mostly based on daily repetitive gestures one may want to map a specific gesture, when recognized, to a specific triggered action (launching a soundfile, changing presets, etc...).

We did a recording session during which we had to record:

- two video sources : one from a zenithal camera and another one from a front camera
- data from sensors which were embodied on the performer

In order to keep a direct compatibility between previously made tools (like interpolation and DTW, started with the project Dancing Viola (#04.2) [73] and performance context used tools, we developed this recording and visualizing / editing tool for the Max/MSP environment [14] with help from the FTM framework [60, 30].

Synchronization of all sources was one of the key point as we got several different input signals and recording spaces : videos were recorded directly on video tapes, sensors data were received directly in Max/MSP and the different features of the video analysis were made by the HUM software developed by François Zajéga during project Behavioral Installations (#05.2) [19] and then sent to Max/MSP.

Multimodal software for data acquisition, processing and analysis in order to recognize gestures have already been done on various environments and for various purposes. We can name as a non-exhaustive list : the Gesture Follower [31] tool made by IRCAM and distributed in the FTM framework [30], Smart Sensor Integration (SSI) [64, 65], WiiGLE [55, 78] and Jensenius' toolbox [32].

#### 3.1.1. *Choice of file format and description*

In order to playback the data synchronously, we needed a time-tagged format. The Max/MSP environment proposes several tools to record data (e.g `coll` and `text` objects) and only one with time tagging (`mtr` object). Tests with this object during previous projects Breathing for Opera (#02.2) [20] and Dancing Viola (#04.2) [73] showed that this one is not that reliable when dealing with different sources at different frequency rates and that some resampling and reordering post-processing could be necessary, without ensuring that all these issues could be solved.

The file format we chose was SDIF (Sound Description Interchange Format) [62]. This format was made by CNMAT, IRCAM and IUA. In [80] we can find that the first goal of this format was to have a common file type so that data can easily be exchanged between labs working on sound descriptors. One of the main feature they emphasized on was to strike a balance between a strong file standard and a weak file standard in order to give everybody the possibility to customize it while staying in the same canvas. Firstly made to store sound descriptors values over time, it can easily be used to store any type of numerical data evolving through time, as one can define one's own frame and matrix type.

The SDIF format works with time-tagged and type-tagged frames. In the header of each SDIF file, Matrix Type Definition (MTD) and Frame Type Definition (FTD) can be defined and customized. A frame contains matrices and is time-tagged and matrix contains the data itself. After having first defined data types and the structure of these data, each frame contains the value of each defined-type with corresponding time tag. An example is illustrated in Fig. 2.

```
1TYP
{
  1MTD  XL01    {lig_data1}
  1MTD  XL02    {lig_data2}
  1MTD  XM01    {mag_data1}
  1MTD  XM02    {mag_data2}
  1MTD  XM03    {mag_data3}
  1MTD  XM04    {mag_data4}
  1MTD  XBIP    {lig_bip}
  1MTD  XA01    {axl_data1}
  1MTD  XA02    {axl_data2}

  1FTD  XL01    {XL01   lig_data1;}
  1FTD  XL02    {XL02   lig_data2;}
  1FTD  XM01    {XM01   mag_data1;}
  1FTD  XM02    {XM02   mag_data2;}
  1FTD  XM03    {XM03   mag_data3;}
  1FTD  XM04    {XM04   mag_data4;}
  1FTD  XBIP    {XBIP   lig_bip;}
  1FTD  XA01    {XA01   axl_data1;}
  1FTD  XA02    {XA02   axl_data2;}
}
```

```
XM04    1   3   0
  XM04  0x0004  1   1
    0.467773

XM03    1   2   0
  XM03  0x0004  1   1
    0.469727

XM02    1   1   0
  XM02  0x0004  1   1
    0.483398

XM01    1   0   0
  XM01  0x0004  1   1
    0.420898

XL02    1   7   0.005
  XL02  0x0004  1   1
    0.539062

XL01    1   6   0.005
  XL01  0x0004  1   1
    0.436523
```

Figure 2: On the right : definition of the frame and matrix types. On the left : an example of recorded frames

Several tools has been developed to work with the SDIF format [61]. The frameworks that we focused on are listed here:

- Command-line tools
- Max/MSP objects from the FTM framework
- Max/MSP objects from the CNMAT framework

As we were working with dancers and as one of the goal of our tool was to select a gesture and to make it available for DTW recognition, we also investigated gesture description file formats. A list of the different formats can be found in [33]. Our focus was evaluate the GDIF (Gesture Description Interchange Format) [26] format, because the beginning goal was to define a common file format with strong canvas that everyone can use and share, as we did when we chose the SDIF format. However, GDIF is still in research phase and for the moment, it only exists as an OSC namespace and no file type or tools are ready to work.

### *3.1.2. Synchronized Multimodal Recording*

The recording session which was realized for the *Musichorégraphie d'appartement* project, happened at the RTBF's Auditorium Abel Dubois, Esplanade Anne-Charlotte de Lorraine, B-7000 Mons, Belgium. As previously mentioned, we needed to record two video camera signals and sensors from a wireless system embodied on the performer.

Our first attempt of recording chain was to record sensors data and videos on the same computer via Max/MSP. This setup highlighted two problems: first two FireWire video cameras couldn't be accessed on the same computer from Max/MSP and secondly, the only video we managed to get recorded by the software was totally unsynchronized. As we had a tight schedule during this session we didn't manage to find whether it was a non fixed frame rate problem or whether it was a larger Max/MSP and Jitter (video part of Max/MSP) thread problem.

Our second recording chain was to use Max/MSP to record sensors data and record videos directly on the video camera tapes. The synchronization between sources was made with one sound clap and one visual clap. The sound clap was generated by the computer, recorded on the two tapes and on the SDIF file by defining a track with a decreasing number (from 3 to 0). An additional visual clap was made to add a security between the two video signals.

The recording tool we made is based on a Max/MSP toolbox made by Alexander R. Jensenius [32] which uses the FTM SDIF objects distribution. The overall use is quite simple : you first have to define the folder and the file in which you want to put your data, second you define the type of frames you're going to use in your SDIF file and then you just have to hit start to launch the recording. Clicking the start button launches the sound clap. The visual clap was done just on the last bip of the audio one. We, then, just have to click the stop button when the recording was over which launches an ending audio bip.

We have to mentioned that, concerning the structure of the SDIF file, you can define a Name Value Table header (NVT) where you can put details about what's in the file : type of sensors, what for, who, when, etc. We needed to define one frametype per sensor (instead of a 4×1 matrix as we had 4 1-axis accelerometers for example) as, at that time, import SDIF file option contained bugs. We needed to do one input per type as well. As mentioned above, we added the XBIP frame type to be able to store numerical values defining the beginning (from 3 to 0) and ending (-1) clap for each records, in order to synchronize everything.

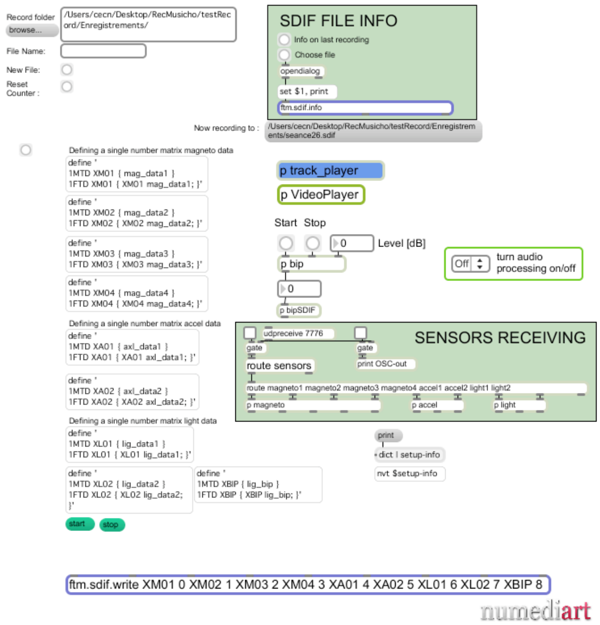


Figure 3: Recording tool

#### 3.1.3. Visualization and Edition

Before going into the details, we have to list here the elements and properties we needed this tool to have :

- Visualization of two videos synchronously
- Visualization of data (sensors, computer vision features, etc.) synchronized with the videos
- Editing features :
  - zoom in/out
  - show/hide specific tracks
  - select a certain part of the file and make the video loop this selection
  - mark the selection so that to recall it later
- Save selected gesture / pattern to fit the DTW fileformat

The FTM Framework provides a very promising tool (still in development and improvement) known as the `ftm.editor`. This is the tool that we used to visualize sensors signals. Our work with that tool helped the IRCAM team to correct some of the bugs present at that time in the `ftm.editor`. This visualization tool already has several properties we listed below like the zoom in/out and show/hide elements.

A `rslider` Max/MSP object (slider that gives you the possibility to define a region with mark in and mark out points) has been used to make the selection on all displayed tracks at the same time. Once the selection is made, the `ftm.editor` is updated to show only this portion and videos make a loop of this portion. As we zoom in the specified signals selection with `ftm.editor`, the `rslider` object becomes important as it shows the selection position and duration compared to the entire recording duration. If the first raw selection made doesn't fit perfectly the gesture meant to be extracted, by holding the [Shift ⇧] key, in and out points can be redefined separately, giving a more accurate control on the selection.

Once the selection is satisfactory, pressing the [R] key allows to type a region name in the window that just pops up. The Mark In and Mark Out menus are then updated. When all selections have been defined, hitting the "save marker" bang button launches the file writing.

A defined folder and files structure has been set up so that only a SDIF file has to be choosen in the menu so as to load all the corresponding video and markers files. An example of this structure is showed below :

```
|– VisualizationPatch.maxpat
|   |– MOV
|       |—- FACE
|           |—- seance1_01_FACE.mov
|       |—- ZENITH
|           |—- seance1_01_ZENITH.mov
|   |– SDIF
|           |—- seance1_01.sdif
|   |– MAT
|           |—- seance1_01.txt
```

Morover, choosing the SDIF file enables to dynamically create checkboxes with given frame type names linked to the show/hide parameter, so that you can choose to work more precisely and certain sensors signals. You can come back to the initial state whenever you feel like it just by unclicking the checkbox.

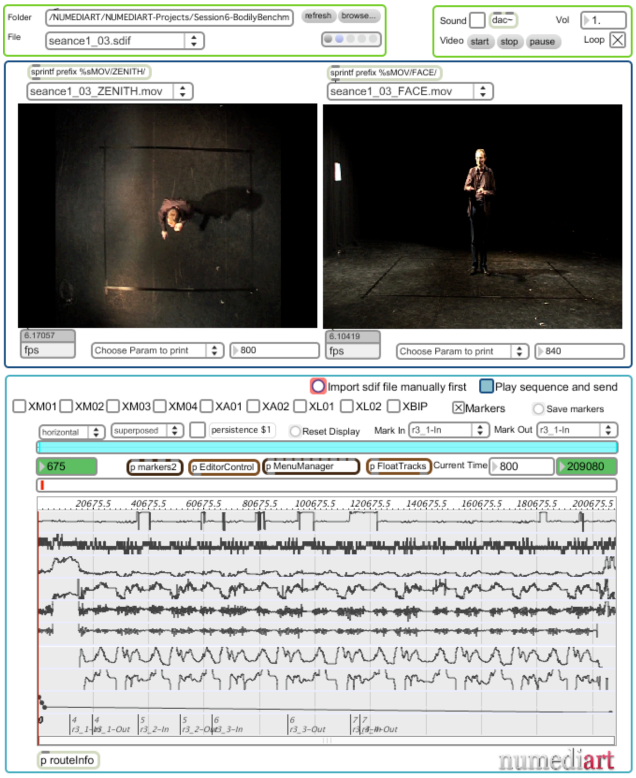


Figure 4: Visualisating and editing tool

For the moment, we only use markers to define the patterns we want to be recognized. Once all markers are defined, they can be saved in a text file. Export to the DTW file format hasn't been done here and further tests need to be done for the integration of a defined pattern in the algorithm. Two ways could be followed:

- add the possibility to the DTW object to read SDIF files
- choose the SDIF file format as standard format for the DTW (for the moment, it's an home-made format)

### 3.2. Updates of the gesture recognition Max external based on Dynamic Time Warping (DTW)

The `num.DTW` gesture recognition external was originally developed during the Dancing Viola (#04.2) project [73] [74] [5]. We implemented the following changes in order to improve the management of the database of reference gestures:

- New messages to allow to activate/deactivate reference gestures on the fly.
- Provision to comment reference gestures using Max messages.
- New messages to and from the `num.DTW` object allow to display a list of reference gestures, complete with ID, name, comment, number of samples and activity status using the `jit.cellblock` object. Names and comments as well as activity status may also be directly edited in `jit.cellblock`, see Fig. 5.

| id | name (may be edited in place) | comment (may be edited in place) | lenght | active toggle |
|---|---|---|---|---|
| 6 | debout_jmbG_GrdDev03 | Idem - version 3 | 313 | yes |
| 7 | debout_jmbG_GrdDevX04 | Idem étendu - Version 1 | 322 | no |
| 8 | debout_jmbG_GrdDevX05 | Idem étendu - Version 2 | 299 | no |
| 9 | debout_jmbG_GrdDevX06 | Idem étendu - Version 3 | 315 | no |
| 10 | debout_jmbG_GrdDevX07 | Idem étendu - Version 4 | 316 | no |
| 11 | sol_2jmbFluide02 | Mouvement au sol - 2 jambes fluides - Version 2 | 416 | yes |
| 12 | sol_2jmbFluide03 | Mouvement au sol - 2 jambes fluides - Version 3 | 406 | yes |
| 13 | sol_2jmbFluide04 | Mouvement au sol - 2 jambes fluides - Version 4 | 444 | yes |
| 14 | sol_2jmbFluide05 | Mouvement au sol - 2 jambes fluides - Version 5 | 407 | yes |
| 15 | sol_arcArr01 | Mouvement au sol - Arc Arrière - Version 1 | 504 | yes |
| 16 | sol_arcArr02 | Mouvement au sol - Arc Arrière - Version 2 | 481 | yes |
| 17 | sol_arcArr03 | Mouvement au sol - Arc Arrière - Version 3 | 425 | yes |
| 18 | sol_jmbD_cascade01 | Mouvement au sol - Cascade jambe droite - Version 1 | 93 | no |
| 19 | sol_jmbD_cascade02 | Mouvement au sol - Cascade jambe droite - Version 2 | 83 | no |
| 20 | sol_jmbD_cascade03 | Mouvement au sol - Cascade jambe droite - Version 3 | 75 | yes |
| 21 | sol_jmbD_cascade04 | Mouvement au sol - Cascade jambe droite - Version 4 | 137 | yes |

**note: in jit.cellblock, multiple spaces are removed and " and ", are replaced by a space**

Figure 5: New messages allow to view and edit names and comments as well as (de)activate all reference gestures contained in the database using `jit.cellblock`.

- When adding new reference gestures using the `record_gesture $1` and `store_in_ref_gesture $1` Max messages, the number of reference gestures is incremented automatically when entering a desired store ID number larger than those already attributed.
- Revision of the file format so that a single file now contains all the reference gestures and all the information concerning them: ID, name, comment, length and activity status. It also contains global parameters, like the number of reference gestures, the number of sensor axes and the decimation factor.

We discussed the possibility of using SDIF file format to store reference gestures. But SDIF files contain typically a fraction of a recording session, from which some bits only constitute desired reference gestures. Storing a large database of reference gestures would therefore imply using many SDIF files with start and end indexes for each gesture, which would be impracticable and difficult to maintain. The only usable solution would be to create a single SDIF file containing only the desired reference gestures with a list of indexes. We may implement in the future an SDIF export function if there is sufficient interest for it. Similarly, we may implement in the future a tool to import a desired reference gesture from a selected part of a SDIF file. The selection could be defined using the visualization and edit tool described above and shown in Fig. 4. But, as it is quite easy to simply play that selection while recording it simultaneously in `num.DTW`, we haven't deemed it a proirity.

### 3.3. Updates of the Interpolation Max external module

The interpolation tools developed during the Dancing Viola (#04.2) project [73] [74] [72] went through major revisions. Fig. 6 shows some of the new fonctionnalities, including multiple LCD and Jitter displays, and several cursors.

#### *3.3.1. Cursors and Points*

We implemented, as we suggested in the conclusion of *Dancing Viola*, the possibility to have several cursors moving in the same interpolation space. Coordinates of the cursors and associated interpolation weights are sent to the corresponding outputs of `num.interpol`.

And the number of cursors and points can now be changed dynamically, creating or erasing sprites for LCD and sending `enable` or `disable` messages to specific draw commands stored in `jit.gl.sketch` for the 1-D and 2-D Jitter representations or to the `jit.gl.gridshape` objects for the 3-D Jitter representation.

The coordinates of points and cursors can be sent to `num.interpol` either in 2-D, as $(x, y)$ coordinates, or in 3-D, as $(x, y, z)$ coordinates, but are always stored internally as 3-D coordinates.

#### *3.3.2. Integration of LCD and Jitter display modes in a single external*

The most important change is the fusion of the `num.interpol.lcd` and `num.interpol.jit` externals in a new `num.interpol` module, able to handle both a LCD and several Jitter displays in various ways, either `jit.pwindow`, included in a patcher window, or `jit.window`, in a separate window. The first creation argument now defines the type of display(s). This can be later modified at any time, using the message `display_mode` followed by one of the following arguments:

- `no_display` : when an interpolation space has been defined, one may want to stop the display in order to save cpu resources.
- `LCD` : sends only LCD commands on the left to rightmost output.
- `Jitter` : sends only Jitter commands on the rightmost output.
- `LCD_Jitter` : messages to both displays are sent out simultaneously, as seen on Fig. 6.

Great care has been taken to homogenize the look between the LCD and Jitter displays.

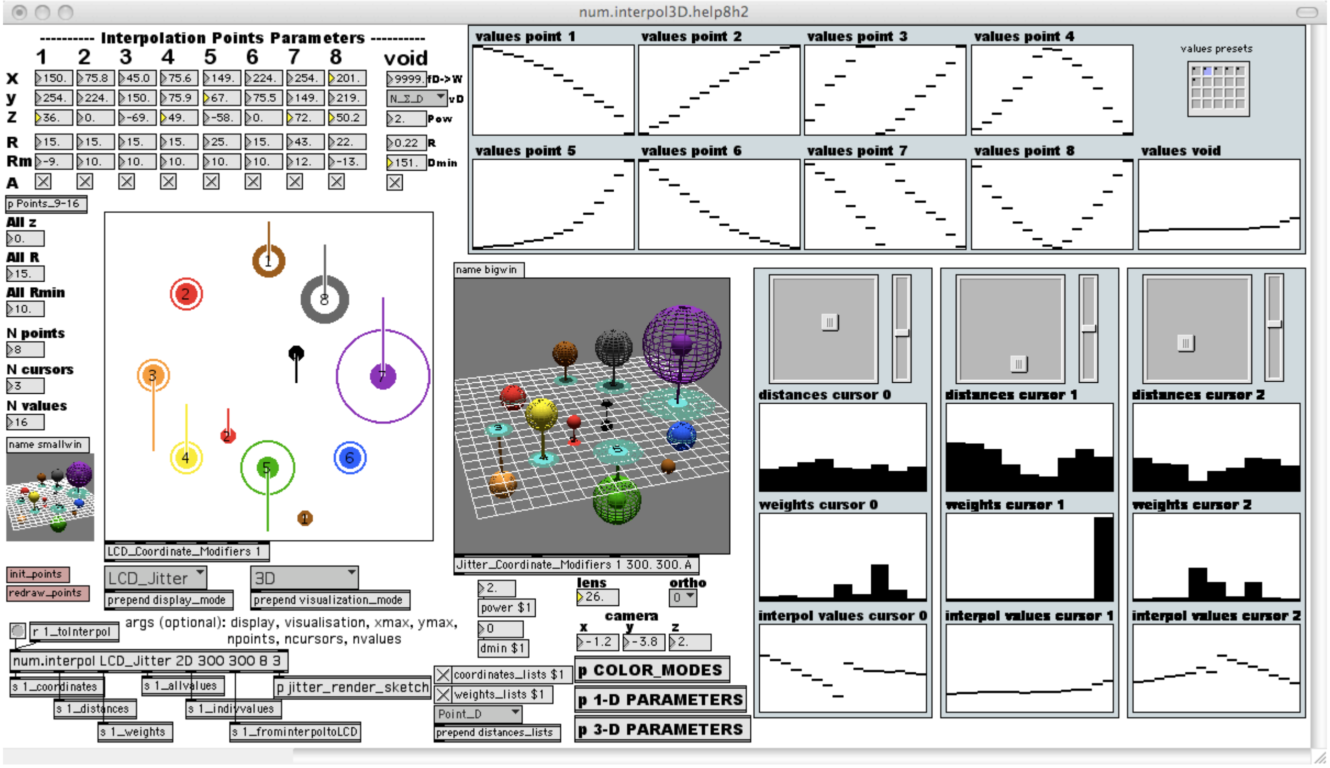


Figure 6: *Shown on one* LCD *and two* jit.pwindow *displays: eight interpolation points and three cursors with resulting distances, weights and interpolated values from the nine sets of values, including the one for the void.*

#### *3.3.3. Visualization modes*

We didn't go so far as to recreate a 3-D view in LCD, as the absence of alpha would have made transparency impossible and as it didn't make sense to implement a wide range of useful OpenGL commands equivalents. Instead, we created a pseudo_3D visualization mode that allows to display the $z$ coordinate in a 2-D plane. The second creation argument of num.interpol defines one of the visualization modes that can then be changed on the fly, using the message visualization_mode followed by one of the following arguments:

- 1D_hor : an horizontal one-dimensional visualization mode, with horizontal rectangles.
- 1D_vert : a vertical one-dimensional visualization mode, with vertical rectangles.
- 2D : a two-dimensional visualization mode, with circles.
- pseudo_3D : a 2-D visualization with circles with added $Z$ information, displayed as a vertical bar in both the LCD (as in Fig. 6) and Jitter displays.
- 3D : a 3D representation in the Jitter display and a peudo_3D representation in the LCD display. Fig. 6 shows both representations side by side. In the Jitter display, horizontal projections of points show clear user definable color circles if $Rmin \geq 0$ and darker ones if $Rmin < 0$, while the cursors horizontal projections are displayed in the cursors colors. The light on the $Rmin$ sphere also informs about its sign: it shines from above when $Rmin \geq 0$ and from underneath when $Rmin < 0$.

The laws used for distances and weights computation are automatically adapted when switching between 1-D, 2-D or 3-D mode. We need an extra Max subpatch with one jit.gl.text3D object per point and per cursor in order to display points and cursors names in a Jitter window. As before, several jit.gl.handle and jit.gl.gridshape objects are needed to represent the points and the cursors in a 3-D space. But num.interpol sends now automatically all the messages to the Jitter window in order to change the OpenGL context when the visualization mode is changed. And new input messages now allow the output of the jit.gl.handle modules to move the points and cursors in the 3-D space by sending the handle position to num.interpol which, in turn, updates the graphics. The position of the handles associated to each point and cursor can be modified by dragging their position while pressing the ⌥ or ⌘ key, in the usual Jitter way on Mac OS.

#### *3.3.4. Void*

The parameters associated to the *Void* have been transformed in order to make them more intuitive. The constant weight of the *Void* is therefore now expressed as a distance rather than the more abstract numerical value of the weight. It also eases the comparison of the effect of the constant part of the *Void* in regard with the distance related part of the *Void*.

Additional *Void* distances modes have been implemented. On top of the weighted sum and product of the distances from the cursor to all other active points, the distance to the closest point, Min_D, can now also be chosen. And those three distances can be either the real distances to the center of the active points or the mathematical distance that includes the `Rmin` offset. This gives us the choice between the following six distances that may be given as argument to the `void_distance` message: `Min_D`, `N_Σ_D`, `N_Π_D`, `mMin_D`, `mN_Σ_D` and `mN_Π_D`, where the $m$ prefix stands for the mathematical distance.

#### 3.3.5. Colors

The handling of colors has been improved:

- As we can now have several cursors, the same modes were implemented for the colors of the cursors and the points. We therefore have the messages `point_color_mode $1` and `cursor_color_mode $1`.
- The color modes arguments have been renamed in a clearer way: besides the `resistor_color` that can't be changed, we have two user modifiable modes: `user_single`, where all points or cursors share the same color, and `user_multiple`, where cursor, and $R$ and $Rmin$ for points, colors can be individually set up for each cursor or point using messages `cursor_color $1 $2 $3 $4 $5`, `R_color $1 $2 $3 $4 $5` or `Rmin_color $1 $2 $3 $4 $5`. Where `$1` defines the chosen cursor or point number, including `All` and `single`. `$2 $3 $4` are the RGB values and `$5` is the optional alpha value.
- The alpha can now be defined separately from the color with the messages `cursor_color $1 alpha $2` for cursor or `R_color $1 alpha $2` and `Rmin_color $1 alpha $2` for points.
- The `select_color $1 $2 $3 $4` message, complete with alpha, can now be modified by the user. This is the color of the surrounding rectangle or circle that shows which point is about to be graphically modified.
- The background color, that doubles as an erase color, associated with an erase time for Jitter, can now also be modified by the user with the `erase_color $1 $2 $3 $4` message, allowing to show a fading trace of the movements of the cursors,
- It is now possible to define the colors of the $R$ and $Rmin$ circles projected on the horizontal plane in a 3-D view.

#### 3.3.6. Coordinates, distances and weights outputs

In order to widen the potential use of the visual part of interpolation external for other applications, like spatialization, `num.interpol` can output the coordinates of the points when moved, depending on the value of the argument, 0 or 1, of the message `coordinates_lists $1`. The same holds for the weights with the message `weights_lists $1`.

It either doesn't output distances or outputs the distances from the cursor to the center of the points or the ones used in the computation of the weights, i.e. the mathematical distances that depend on the values of $Rmin$, depending on the argument, `Off`, `Point_D` or `Math_D`, sent with the `distances_lists` message. Lists of distances and weights are sent out for each cursor, each time the position of that cursor is modified. Distances and weights are shown for each cursor on Fig. 6.

### 3.4. Long-term attention model: port from EyesWeb to Max/MSP

#### 3.4.1. Global Overview of the model

The long-term motion attention model has been developed by Matei Mancas at TCTS Lab in projects Tracking-Dependent and Interactive Video Projection (#03.1) [42] and MATRIX: Natural Interaction Between Real and Virtual Worlds (#05.1) [41]. The model was made to detect region of interest in a video. By region of interest, we mean region where something new, something unusual, is detected. As described in [40], this model is based on the computation of the rarity index on position, direction, and velocity so as to obtain highlighted regions where rare behaviors were observed. If current motion has the same features as the model at the same locations, the motion detection will be inhibited: it is an already seen one, it is not rare, and thus it is not worthy of attention. If motion occurs with different features as those from the corresponding model, the motion detection will not be inhibited: it is a novel movement which is rare and which should attract attention.

Here is the procedure of the model in the following pseudo-code (IR stands for inhibition rate) :

```
a= 0.6, Eps = 10^-5
Nb_frames_salient_motion=0, Nb_frames_detected_motion=0

if sum(moving_pixels_after_inhibition)/sum(initial_moving_pixels)>a
Nb_frames_salient_motion = Nb_frames_salient_motion+1
if sum(initial_moving_pixels)>0
Nb_frames_detected_motion = Nb_frames_detected_motion+1

IR = Nb_frames_salient_motion /(Nb_frames_detected_motion+Eps)
```

Figure 7: Procedure of the model in pseudo-code

Such a model was quite interesting for *Musichorégraphie d'appartement* because as it is based on daily routine the dancer will make repetitive things, take repetitive paths, etc. The recording session that has been done went in that sense. By implementing such a model, we'll be able to detect when the performer is starting to do repetitive schemes and when she's starting to change her "activity". A way to track her daily gesture to turn them into artistic potential.

#### 3.4.2. Model implementation

The long-term attention model follows several steps which are showed on Fig. 8 :

- Background Extraction : this block, as its name tells us, performs a background extraction followed by a threshold
- Memory : the moving pixels coming out of the background extraction are recursively added to themselves to draw the track. Two parameters are tunable, then : the track opacity and the remembering factor. The last one defines the duration of the memory of the model (e.g, if this parameter is set to 1 the system will remember everything he added). The track opacity is not only a display parameter as it defines the time the system will need to draw a "correct" track (i.e, that we can then recognize)

  This track is then subtracted to the entire moving pixels image comming out from the background extraction, so that we isolate only the pixels which are part of the region of interest

- Quantity of motion : compute the quantity of rare pixels found and compare them to the defined threshold

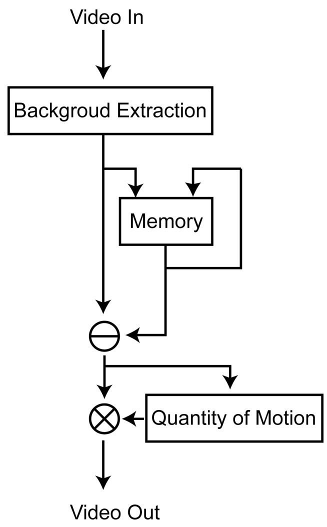


Figure 8: Schema of the long-term attention model

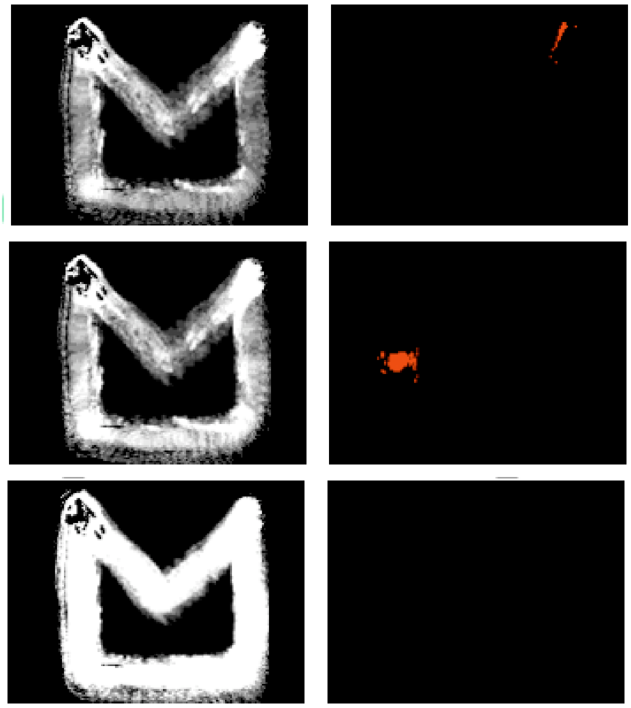

Figure 9: Display of the memory of the model (left) and of the region of interest (right) at three different times

This model was first implemented in EyesWeb [18] software and as all the previous described tools were developed for the Max/MSP Environnement and as Eyes Web is only for Windows, we decided to redevelop it for Max/MSP rather than building a bridge between Max and EyesWeb, in order to limit the number of needed computers.

The Fig. 9 shows the displays given by the implementation of the long-term attention model at three different times. As said on the caption of the figure, you can find the memory of the model on the left and the detected region of interest on the right. The whiter the graphics on the left are the longer the performer stayed and the smallest the region of interest is when she comes back at that position (first set of images). On the other hand, the whiteless the graphics on the left is, the shorter the performer stayed and the bigger the region of interest is when she comes back at that position (second set of images). After several passages the memory is “white” enough, no region of interest is detected and the memory image can be saved for further recognition if desired (third set of images).

## 4. APPLICATIONS

### 4.1. Prototyping of the *BioDiva* Gestural Interface

So as to design the best user-centered wearable gestural interface [46] for the *BioDiva* project, we first tested together with the artist several wearable sensors and produced a first prototype version with a mapping to her audio performance software application.

#### *4.1.1. Preliminary Sensors Tests*

- Biomechanics (Accelerometers):

  On January 7th 2009, we have been welcomed at Serge de Laubier’s personal studio so as to receive a personal demo of Puce Muse’s Méta-Instrument 3 [37]. First released in 1986, this “exo-instrument” allows the simultaneous control of up to 54 parameters, transmitted through an Ethernet connection with a resolution of 16 data bits for about 2 ms of latency (wired) and 4 ms (wireless). *Méta-Instruments* (or mappings of the Méta-Instrument) can be configured and created using the associated 2PIM multimodal interactive development platform developed in Max/MSP. While this solution affords an large number of degrees of freedom and high quality specifications, the mechanical structure of this instrument was too constraining for our purpose: providing a non-invasive free-form gestural-based interface dedicated to a given set of gestures, instead of using a gestural instrument to which the user has to adapt his gestures to.

  We have more intensively tested Julien Stamatakis’ (UCL-TELE) sensor interface design [67] featuring two CrossBow TelosB Motes [51] (one base/receiver connected to the computer thru the serial USB port, one node/emitter communicating wirelessly to the base using the IEEE 802.15.4 ZigBee protocol); and 4 3-axes digital accelerometers with a 10-bit resolution, digitally wired to the node using the I2C protocol. While the data reception software was originally developed in Matlab, we developed custom made adaptations in Max/MSP and Pd based respectively on the `serial` and `comport` objects. We met several issues when using this interface in Max/MSP and Pd: when the connexion between node and base is lost, the trackpad of the laptop (running OSX) would be set inactive and entailed the need of rebooting the computer. We believe this issue might be solved by replacing the USB serial connexion of the base by an Ethernet connexion, as it is not uncommon that several USB peripherals (using FTDI chips and drivers) conflict when plugged and used simultaneously as Human Interface Devices (HID). This CrossBow TelosB Mote sensor interface is cheap, relying on a power-saving and reliable communcation protocol (ZigBee), available in a small form factor; but not as widely available as Arduinos, and a bit more difficult to program (embedded TinyOS [70]).

- Biomechanics (Capacitive):

  We have also tested a Moog Etherwave Theremin during a one-day session. The Theremin is an instrument based on capacitive sensing and whose 2 antennas allow to control pitch and amplitude of a "sine-like wave". Yet offering a free-form interaction, the player is still constrained by the need of surrounding the instrument even if the instrument is remotely controlled.

- Biomechanics (Gloves):

  Sensor Gloves, first designed for virtual reality applications, have also been used in musical and dance performances, notably Laetitia Sonami's Lady's Glove [66] and Michel Waisvisz' The Hands [75] (that we could consider as a glove-sized Méta-Instrument).

  We tested an Essential Reality P5 Glove (released in 2002, now discontinued). This low-cost glove is equipped with 5 bend sensors (one per finger), 4 buttons and 8 IR leds wired to a IR receiver stand for 3D position tracking. We used Carl Kenner's GlovePie so as to easily map sensor values to MIDI Control Change values. The flexion sensors can be set to output continuous values or 3 positions (finger closed or half-closed, or open). We rejected this device for several reasons not compliant with artistic performance considerations: the need to point towards the IR receiver stand for 3D position tracking, the wired connexion from the glove to the stand, the non ergonomic form factor (exoskeleton-like). This discontinued device can yet be reverse-engineered: it is often cheaper and easier to rip the FSR flexion sensors off a second-hand version of this device than buying small amounts of sensors from electronics equipment suppliers. We plan to produce later prototypes of "data gloves" using textile sensors, more ergonomic, less invasive and possibly washable.

- Biosignals (Respiration Belt):

  We tested one of the twin respiration belts based on the M.E.C. hardware used in project Breathing for Opera (#02.2) [20]. As its was initially meant for analyzing vocal features, we found it difficult to use as a controller for applying digital effects to the voice, as singers need to control their breathing accordingly to the vocal characteristics they want to produce vocally, and can not dissociate both easily.

#### *4.1.2. Chosen Setup for BioDiva*

Nios Karma [34]'s chosen software application for sample playback and digital effects processing of samples and voice is Ableton Live [1]. In this software application, an audio sequencer optimized for live performances, sound samples are organized visually by rows and columns on the "Session View", as illustrated in Fig. 10. Samples can be played at the same time when located on the same row. Digital audio effects can be assigned by columns.


Figure 10: Ableton Live's Session View of one of Nios Karma's sets: the parameters of "Grain Delay" effect applied to a voice track are being visualized

We decided to use a 5-column setup, as illustrated in Fig. 10, so as to allow up to 4 samples to be played simultaneously; and so that these 4 samples and 1 voice channel could be processed each by 3 possible types of digital effects: granular-like processing, reverb and delay. If we follow Claude Cadoz' instrumental gesture typology [8], the required gestural interface should allow two types of gestures:

1. one for the *modification* of the continous values of the digital audio effects parameters,
2. another one for the *selection* of soundscapes/voice channels to be associated with a selection of digital audio effects at a given time.

We decided to use the Interpolation Tool developed in Max/MSP (portable to Pd) in project Dancing Viola (#04.2) [73] as follows:

- MIDI Control Changes (named `Fader`, `FX` and `Send`), sent to Ableton Live, are associated to *interpolation parameter values* (see the central-right list of drop-list menus in Fig. 11);
- different *values points* allow to define MIDI CC presets corresponding to MIDI CC values assigning soundscape/voice channels to associated effects and values (see the top-right corner histograms in Fig. 11);
- *cursors* (here 1) allow to navigate in a 2D space of interpolation values points, thus interpolating between MIDI CC presets (see the central-left multi-color 2D scatter plot in Fig. 11).

Most of our gestural interface prototypes created for the preliminary sensor tests (see previous section 4.1.1) were developed inside Max/MSP or Pd. Our last test in date consisted in mapping 2 accelerometers (1 dimension of each) to the 2D position of the

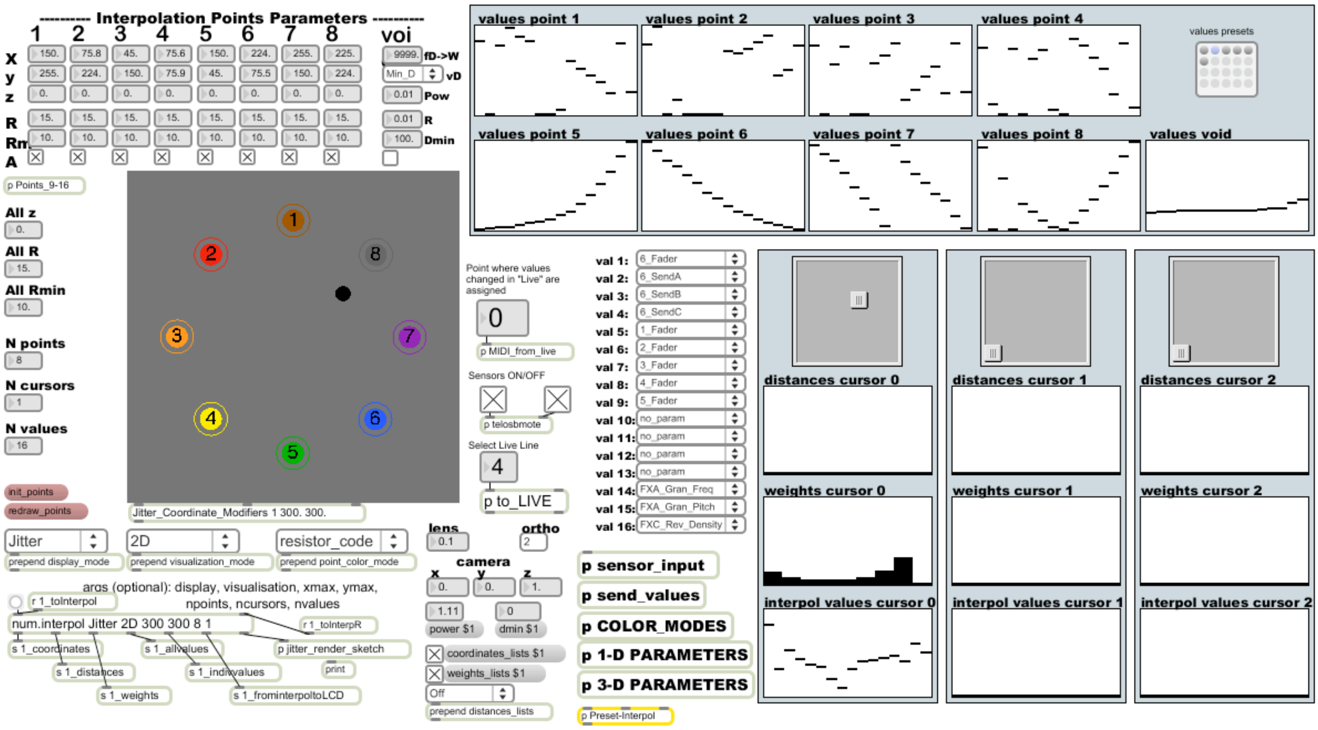


Figure 11: Control by accelerometers in Max/MSP of an interpolation of MIDI Control Change parameters sent to Ableton Live

interpolation cursor, thus controlling all soundscape/voice channel selection and digital effects modifications with both forearms movements.

For this test, we used two computers: one running Ableton Live, one running Max/MSP, both connected thru MIDI interfaces. Solutions to run everything on a single computer depend on the choice of OS and communication protocol (MIDI or OSC):

- J. Sarlo's PdVst [57] enables to run a Pd patch in a VST host (such as Ableton Live) as VST plugin (Steinberg's Virtual Studio Technology virtual effects plugin standard) only under Microsoft Windows, and our Interpolation Tool is not yet available in Pd;
- the LiveAPI, an unsupported Third Party tool for getting access to Ableton Live's Python API, allows to have a limited control of the Ableton Live user interface thru the OSC protocol, has only been reported to work on specific Ableton Live versions under Microsoft Windows;
- OscVstBridge [28], a Java VST plugin, allows the MIDI to OSC and OSC to MIDI conversion under Microsoft Windows;
- Max for Live (M4L) is a conjunct effort of Ableton and Cycling'74 to control the Ableton user interface and processing chain using an embedded version Max/MSP, to be released in late 2009. Features and flexibility are not yet clear for our case;
- Pete Yandell's MIDI Patchbay [81], under Apple OSX, the solution we successfully used.

#### 4.1.3. Possible Future Sensors Tests

- Biomechanics (Accelerometers):

  In addition to our Dynamic Time Warping (DTW) gesture recognition Max/MSP tool developed in project Dancing Viola (#04.2) [73], other solutions for mapping and gesture recognition could be tested:

  - Arshia Cont's Pd Realtime Gesture Analysis using Neural Networks in Pd [13, 12] ,
  - Frédéric Bevilacqua's Gesture Follower using Hidden Markov Models [7, 31] using the FTM and MnM toolboxes [6].
- Biomechanics (Gloves):

  We would like to try another setup featuring accelerometers to change digital effects parameters, and switches on fingertips so as to select soundscapes/voice channels and digital audio effects to apply to the played soundscapes and the voice.

  A textile glove, using conductive fabric so as to sew and wire electrical switches on finger tips, is currently being prototyped.
- Biosignals (Electrocardiograms):

  We might try the Biogene ECG belt bought for project Sensor-Based Mini-COMEDIA (#01.2) [22, 21] so as to first monitor and analyze how the heart rate variates along an artistic performance and then, if it appears to be relevant, map some features to the audiovisual rendering of the performance instrument.

### 4.2. Analysis of the *Musichorégraphie d'appartement* Recording Session

#### *4.2.1. Setup*

As previously mentioned in section 3.1.2, the setup was made with two video cameras and an embodied and wireless sensors system. The two video cameras were Sony HVR-V1E each one of them mounted with a wide-angle lens on. The wireless system was one of the early prototype of La kitchen's Kroonde [11]. The transmission was an HF one and the group of sensors was made of four magnetometers, two 1-axis accelerometers and two light sensors.

#### *4.2.2. Gestural script*

A gestural script has been made for this recording sessions. The script was divided into two part : the first one was build around 4 basis patterns and the four variations around these patterns. The second part was improvisations around daily gestures, like watching television, reading, going to bed, etc. The first one suited best our tests for the long-term attention model as they were repetitive patterns and variations. An example taken from the script is showed in Fig. 12.

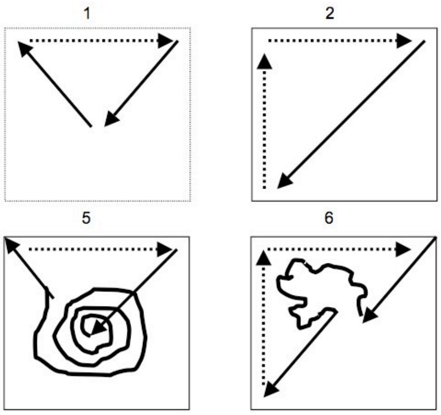


Figure 12: On the top : basis pattern. On the bottom : these basis patterns with variation.

#### *4.2.3. Analysis*

In [40], results on the use of the long-term attention model using video material from *Musichorégraphie d'appartement* recording session has been published. Fig. 13 and 14 shows that long-term attention module has been able to recognize the basis pattern part and novel motion during the improvised parts. By thresholding the inhibition rate, we can easily make the model recognize predefined patterns.

|  | Path 1 | Path 2 | Path 3 |
|---|---|---|---|
| Model 1 | **0.03** | 0.35 | 0.35 |
| Model 2 | 0.15 | **0.01** | 0.20 |
| Model 3 | 0.11 | 0.13 | **0.07** |

Figure 13: The comparison of the referent three models (1, 2, 3) with the three corresponding paths (1, 2, 3) shows a low inhibition rate (IR) value when matching between model and path is high and a high IR values in the opposite case

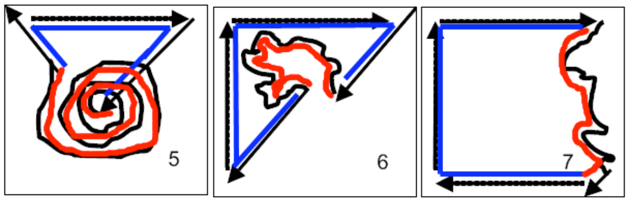


Figure 14: In black: trajectories 5, 6 and 7. In blue: already detected motion of the models 1, 2 and 3 from figure 6. In red: novel/salient motion

## 5. PERSPECTIVES

### 5.1. Towards a more efficient Multitrack Annotation Tool

During the development of the synchronized recording, visualization and editing tool (see section 3.1), we faced several shortcomings:

- when multiple signal tracks are displayed, the GUI tends to be slowed down;
- the monitoring of multiple sensor signal tracks while recording is not yet possible;
- the recording of more than one live video feed at the same time is not yet possible (using the FireWire DV cameras described in section 3.1.2)
- the time series interaction techniques offered by the `ftm.editor` object are currently less intuitive than timeline-based software applications (ex: audio sequencers, video editing): scroll and zoom for the navigation, region and markers for the annotation;

New releases starting from July 2009 of the `ftm.editor` might address some of these issues. Though we haven't tested EyesWeb [18] and Smart Sensor Integration (SSI) [64, 65] for audio/video/sensors synchronized recording, we propose here below some tracks for improving our tool.

#### *5.1.1. GPU Acceleration*

Methods for accelerating the computation of algorithms by using the Graphical Processing Unit (GPU) so as to decrease the load of the Central Processing Unit (CPU) have been available for several years, chronologically: in the past by reverse-engineering shaders using the OpenGL Shading Language (GLSL); currently by using Nvidia's Compute Unified Device Architecture (CUDA) language on compliant graphics cards equipped with selected Nvidia chipsets; towards the OpenCL (Open Computing Language) standard from the Khronos Group. For further details and application of GPU acceleration for visualization and data mining, we recommend [27, 77]. We are interested in accelerating image processing and computer vision [3, 4], and waveform visualization [38, 56] by algorithms processed directly by the GPU.

These GPU programming languages are not yet all integrated in the patcher environments we use: shaders can be programmed by integrating portions of GLSL code into the visual programming environment, by using the `jit.gl.shader` object in Max/MSP /Jitter [14] and the `glsl_vertex`, `glsl_fragment`, and `glsl_program` objects in PureData/GEM [15].

We have undertaken initial tests of memory allocation of recorded signal samples into the GPU memory so as to speed up the display. 1D signal samples could be defined as OpenGL Vextex Buffer Objects (VBOs) or Pixel Buffer Objects (PBOs) and then would already be available inside the GPU memory for transformations (scrool, zoom) without passing back by the GPU <-> CPU bus. We need to produce more tests possible memory allocation bottlenecks, depending on the type(s) of memory allocation chosen, as the CUDA manual [49] identifies and supports 6 types of memory: global, local, constant, texture, shared and registers.

#### 5.1.2. Adapting existing Multimedia Annotation Tools

We plan to test other multimodal annotation tools [17, 54] so as to compare features and check if future development could be switched on one of these platforms:

- Advene [53, 2]
- ANVIL [36, 35]
- Lignes de Temps [52]
- Smart Sensor Integration (SSI) [64, 65]

### 5.2. Towards a dedicated `numediart` Wireless Sensor Interface

#### 5.2.1. Analysis of models we've been using for far

When one wants to buy a sensor interface, usually one can only rely on datasheets or manufacturers' websites, and sensors interfaces are rarely demoable on local electronics supply shops. User-fed websites can also cross mutual information, such as the SensorWiki.org [76]. In table 1, we gather most sensor interfaces that are available locally in Belgium, or not far around. Throughout the past projects of the CoMedia research axis, we have been using and testing many low-cost wireless sensor interfaces, thus all the interfaces listed on table 1 but the Eowave and Interface-Z products:

- Arduinos Diecimila BlueTooth and USB versions and the ARTeM WiFi interface were benchmarked with regularity and latency tests in project Sensor-Based Mini-CoMedia (#01.2) [39], the latter provided the best results in ensuring the data would be transmitted and received in time;
- Arduinos Diecimila USB version connected to MaxStream XBee modules in project Breathing for Opera (#02.2) [20]
- The ARTeM WiFi interface used for the dance performance *De deux points de vue* [47] choreographed by Michèle Noiret and in project Dancing Viola (#04.2) [73] with Dominica Eyckmans. It features four 11-bit analog inputs (10-bit ADC with 4×oversampling), four 16-bit frequency counters in order to constitute a four-antennas Theremin with additional hardware, and a 400kHz I2C bus that allows reading up to 20 axes of I2C sensors at 125Hz sampling frequency, with a resolution defined by the sensor ICs. In our case, 12-bit accelerometers and 14-bit gyroscopes.
- La kitchen Kroonde (section 4.2.1) and CrossBow TelosB Motes (section 4.1.1) in this project

Here follows a quick analysis of each feature of comparison used in table 1:

- **Wireless Protocol** - We found that BlueTooth is the most battery consuming protocol and tends to disconnect from the host computer. ZigBee is power-saving (good for battery, less good for data transmission that is designed to be interrupted). WiFi is power-consuming regarding the emitter, but discards all possible USB serial communication issues for the receiver (we met some in this project, see (section 4.1.1). In an ideal case, a ZigBee protocol could be used between the node and the receiver and an Ethernet connexion between the receiver and the host computer.
- **Configuration Software** - Some solutions for changing the sensors settings are more user-friendly than others: Arduinos and TelosB mote need be reprogrammed in that case (what is easier when using the Arduino Processing IDE); in other cases data reception can be reformatted in Pd or Max or dedicated editors more easily. Attempts such as $\mu$OSC [59, 10] and Firmata [68, 23] try to simplify the software configuration of the sensor interface by defining a communication protocol between nodes and the base. $\mu$OSC allows to send OSC messages to the emitter so as to reconfigure the sensor settings.
- **Inputs** - Most interfaces use analog wiring between sensors and the interface, thus a limited number of analog and digital inputs. CrossBow TelosB motes and the ARTeM interface rely on the I2C protocol, thus analog converters are located closer to sensors and the wiring is digital, which is better for a noise-proof data transmission; and the number of sensors is then limited by the designers program and the interface transmission limitations.
- **Maximum Sampling Rate** - The information regarding this is often confusing because manufacturers mention either the wireless protocol data transmission speed (in baud or b/s) or the microcontroller operating speed (in Hz) or the number of samples acquired per second (S/s) often without stating the number of inputs and related resolutions. We thus can not compare sensor interface here along this feature.
- **Resolution** - We've been using sensor interfaces mostly for movement analysis so far, requiring a good resolution for slow movements and a high speed for rapid movements. We can not yet estimate the lowest resolution necessary for this purpose.
- **Power Supply** - The sensor power supply is dependent on the power supply provided by the node (or emitter) and the power supply required by the sensor chips (for accelerometers, gyroscopes...). The node power supply has implications on the choice of the battery, increasing the node size and limiting the autonomy. Factory-designed CrossBow TelosB Motes can unfortunately not run on rechargeable AA batteries (often providing 1.2 V), as 3V is the minimal supply voltage.
- **Size** - The battery is excluded from all the size dimensions provided in the table.
- **Price and country** - Price and country of origin/sales are coupled in our comparison because shipping fees for receiving orders and sending back interfaces for repair and maintenance share a great percentage of the total price, especially for low-cost interfaces. Long distance has also implications on environmental issues and allows a less human troubleshooting support. Open-hardware Arduinos tend to

| Brand / Model | Wireless Protocol | Configuration Software | Inputs A | Inputs D | Inputs I2C | $F_s$ max (Hz) | Resol. (bit) | Power Supply Sensor | Power Supply Node | Size (mm) | Price (€) (Country) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Arduino Bluetooth | BlueTooth | Dev (Processing) | 6 | 14 | | N/A | 10 | 5 V | ~5 V | 75×50×5 | 105 (be) |
| Arduino LilyPad XBee | ZigBee | Dev (Processing) | 6 | 14 | | N/A | 10 | 5 V | ~5 V | 50(⌀)×5 | 58 (nl) |
| ARTeM WiFi System | WiFi | Patch (Max) | 4 | 4 | 20 | 125 | 11-16 | 3.3 V | 1.8-6 V | 110×80×25 | 1000 (be) |
| CrossBow TelosB [51] [67] | ZigBee | Dev (TinyOS) | | | 30 | 100 | 10 | 3 V | 3 V | 65×31×6 | 300 (de) |
| Eowave Eobody2 HF [24] | ZigBee | Editor (XP/OSX) | 16 | | | N/A | 12 | 3.3 V | 9 V | 65×60×10 | 400 (fr) |
| Interface-Z MiniHF | HF | Patch (Pd/Max) | 8 | 8 | | N/A | 11 | 3.3 V | 9 V | 57×40×19 | 400 (fr) |
| Interface-Z Wiwi | WiFi | Patch (Pd/Max) | 10 | 8 | | 200 | 12 | 3.3 V | 6 V | 80×60×15 | 600 (fr) |
| La kitchen Kroonde [11] | HF | Patch (Pd) | 16 | | | 200 | 10 | ~4.5 V | 4.5 V | 65×55×10 | ~~1200 (fr)~~ |

Table 1: Comparison of low-cost wireless sensor interfaces available around Belgium with features evaluated to be of low, mid or high quality

be manufactured on more and more countries, yet the use of the ZigBee protocol (for the associated module) requires license fees for commercial purposes. The ARTeM WiFi system is a prototype built on demand. La kitchen's Kroonde is not available anymore.

#### 5.2.2. *Specifications for an upcoming dedicated `numediart` Wireless Sensor Interface*

We have been gathering data on various interfaces, comparing their pros and cons. The jitter and latency measurements started within the Sensor-Based Mini-CoMedia (#01.2) project [39]. And the present project allowed us to test other wireless sensor interfaces.

We have also accumulated practical experience in musical and dance performances, like the *Quartet Project* [44] lead by Margie Medlin, Todor Todoroff's *Around, above and weightless...* and Stevie Wishart's *The Sound of Gesture* at the *Bipolar Festival* [83], *De deux points de vue* [47] choreographed by Michèle Noiret, Dancing Viola (#04.2) [73] and the artistic projects concerned by this report.

We decided to use this knowledge and experience in order to built, for the project *Fire Experiences and Projections* [71], a `numediart` wireless sensor interface that could also be used in the future for various other projects.

We will take into account the following key factors: small form factor for the sensors nodes, ZigBee [82] or similar wireless technology for its low power consumption and its small size, connection between nodes and motes using I2C protocol [29] in order to limit the number of signal cables, and using all possible means to lower latency and jitter, like implementing a wireless to Ethernet bridge in order to get the data as fast as possible to the destination applications, on one computer or distributed over a network. As the resolution of each type of sensor (accelerometer, gyroscope, magnetometer) is crucial for their suitability for a given application, an extensive survey of the existing sensor chips on the market will be performed.

With this new interface, we could still use our CrossBow TelosB Motes as emitter/nodes. We could alternatively try to recycle them as well as receivers/bases if we can replace the USB serial FTDI chip by adding an Ethernet bridge (see section 4.1.1).

We might as well consider attempts such as $\mu$OSC [59, 10], Firmata [68, 23], or [50] on the TinyOS platform [70], trying to simplify the software configuration of the sensor interface by defining a communication protocol between nodes and the base. $\mu$OSC allows to send OSC messages to the emitter so as to reconfigure the sensor settings.

We might at last try to implement time stamps between the emitter, transceiver and host computer, as in [25] on the TinyOS platform [70] or as in [58] using the OSC protocol.

### 5.3. New Opportunities with intelligent textiles, wearable sensors

Though the aim of the *Fire Experiences and Projections* project [71] is not the integration of sensors in garment, we would like to be able to integrate the cables in clothes. We are therefore going test the suitability of conductive threads and fabric to transmit I2C signals with spread signal lines. And we will measure the influence of skin resistance under sweaty circumstances to find out which adaptations in terms of I2C buffers, bus frequency and/or routing of the threads within the clothes are necessary to make it work.

Fabric sensors have already been used in a few artistic applications, notably for sound synthesis and control [48] and dance performances [16].

## 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: Laurence Moletta [34], André Serre-Milan [63] and Vanessa Le Mat [43].

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

We thank Clarisse Bardiot and Nicolas Guichard from CeCN for having arranged for us the availability of RTBF's Auditorium for the recording sessions and residency, including help for the heavy hardware setup.

We thank Serge de Laubier and the Puce Muse team for having welcome us at Serge Laubier's studio for a showcase of the Méta-Instrument.

We would like to thank Alexander Refsum Jensenius and Kristian Nymoen for having allowed us to test their Multilayered GDIF-based Setup [32].

We thank Julien Stamatakis from UCL-TELE for having let us use many times his tailored wireless accelerometers interface.

We thank Matei Mancas from FPMs/TCTS for his advice on the long-term attention model implementation and on the recording sessions.

## 7. REFERENCES

### 7.1. Scientific references

[3] Yannick Allusse et al. "GpuCV: An OpenSource GPU-Accelerated Framework for Image Processing and Computer Vision". In: *Proceedings of the Open Source Software Competition, ACM International Conference on Multimedia*. 2008. URL: http://www-public.it-sudparis.eu/~horain/Publications/MM2008/ACM_MM2008_OST-Allusse.pdf. P.: 51.

[5] F. Bettens and T. Todoroff. "Real-time DTW-based Gesture Recognition External Object for Max/MSP and PureData". In: *Proc. SMC '09*. Porto, Portugal 2009. Pp. 30–35. P.: 44.

[6] Frédéric Bevilacqua, Rémy Müller, and Norbert Schnell. "MnM: a Max/MSP mapping toolbox". In: *Proceedings of the 2005 International Conference on New Interfaces for Musical Expression (NIME05)*. 2005. URL: http://recherche.ircam.fr/equipes/temps-reel/articles/mnm.nime05.pdf. P.: 50.

[7] Frédéric Bevilacqua et al. "Wireless sensor interface and gesture-follower for music pedagogy". In: *Proc. of the International Conference of New Interfaces for Musical Expression (NIME 07)*. 2007. URL: http://mediatheque.ircam.fr/articles/textes/Bevilacqua07a. P.: 50.

[8] Claude Cadoz. "Le timbre, métaphore pour la composition". In: ed. by J. B. Barrière. Christian Bourgois, 1991. Chap. Timbre et causalité. P.: 49.

[9] Claude Cadoz and Marcelo M. Wanderley. "Trends in Gestural Control of Music". In: ed. by Marcelo M. Wanderley and Marc Battier. IRCAM - Centre Georges Pompidou, 2000. Chap. Gesture-Music, pp. 71–94. URL: http://www.idmil.org/_media/wiki/cadoz_wanderley_trends.pdf. P.: 41.

[11] Thierry Coduys, Cyrille Henry, and Arshia Cont. "TOASTER and KROONDE: High-Resolution and High-Speed Real-time Sensor Interfaces". In: *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME-04)*. 2004. Pp.: 51, 53.

[13] Arshia Cont, Cyrille Henry, and Thierry Coduys. "Real-time gesture mapping in Pd environment using Neural Networks". In: *New Interfaces for Musical Expressions 2004 (NIME04)*. 2004. URL: http://cosmal.ucsd.edu/arshia/papers/LaKitchen_NIME04.pdf. P.: 50.

[17] L. Dybkjaer and N. O. Bernsen. "Towards general-purpose annotation tools: how far are we today?" In: *Proceedings of the Fourth International Conference on Language Resources and Evaluation LREC'2004*. 2004. URL: http://www.nis.sdu.dk/~nob/publications/LREC2004-annotation-DybkjaerBernsen.pdf. P.: 52.

[19] Jean-Julien Filatriau et al. "Behavioral Installations: Emergent audiovisual installations influenced by visitors' behaviours". In: *QPSR of the numediart research program*. Ed. by Thierry Dutoit and Benoît Macq. Vol. 2. 1. numediart. 2009. Pp. 9–17. URL: http://www.numediart.org. Pp.: 41, 42.

[20] Jean-Julien Filatriau et al. "Breathing for Opera". In: *QPSR of the numediart research program*. Ed. by Thierry Dutoit and Benoît Macq. Vol. 1. 2. numediart. 2008. Pp. 53–65. URL: http://www.numediart.org. Pp.: 41, 42, 49, 52.

[22] Jehan-Julien Filatriau and Loïc Kessous. "Visual and sound generation driven by brain, heart and respiration signals". In: *Proceedings of the 2008 International Computer Music Conference (ICMC'08)*. 2008. URL: http://www.tele.ucl.ac.be/%7Ejjfil/publis/Filatriau_ICMC08.pdf. P.: 50.

[24] Emmanuelle Gallin and Marc Sirguy. "Sensor Technology and the Remaking of Instruments from the Past". In: *Proc of the 8th International Conference on New Interfaces for Musical Expression (NIME)*. 2009. P.: 53.

[25] Saurabh Ganeriwal, Ram Kumar, and Mani B. Srivastava. "Timing-sync protocol for sensor networks". In: *Proceedings of the 1st international conference on Embedded networked sensor systems*. 2003. P.: 53.

[27] Sudipto Guha, Shankar Krishnan, and Suresh Venkatasubramanian. "Tutorial: Data Visualization and Mining using the GPU". In: *The Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-05)*. 2005. URL: http://www.research.att.com/~krishnas/MY_PAPERS/kdd05_tut.pdf. P.: 51.

[32] A. R. Jensenius, K. Nymoen, and R. I. Godøy. "A Multilayered GDIF-Based Setup for Studying Coarticulation in the Movements of Musicians". In: *Proceedings of the International Computer Music Conference (ICMC)*. 2008. URL: http://www.hf.uio.no/imv/forskning/forskningsprosjekter/musicalactions/PDF/Jensenius_2008a.pdf. Pp.: 42, 43, 54.

[33] Alexander Refsun Jensenius et al. "Panel: The need of format for streaming and storing music-related movement and gesture data". In: *International Computer Music Conference*. 2007. P.: 43.

[36] Michael Kipp. *Multimedia Annotation, Querying and Analysis in ANVIL*. Ed. by M. Maybury. MIT Press, to appear. P.: 52.

[37] Serge de Laubier and Vincent Goudard. "Meta-Instrument 3: a look over 17 years of practice". In: *Proc of the International Conference on New Interfaces for Musical Expression (NIME)*. 2006. P.: 48.

[38] Charles Loop and Jim Blinn. "GPU Gems 3". In: ed. by Hubert Nguyen. Addison-Wesley Professional, 2007. Chap. Rendering Vector Art on the GPU, pp. 543–562. ISBN: 978-0321515261. URL: http://http.developer.nvidia.com/GPUGems3/gpugems3_ch25.html. P.: 51.

[39] Céline Mancas-Thillou et al. "Sensor-based Mini-COMEDIA". In: *QPSR of the numediart research program*. Ed. by Thierry Dutoit and Benoît Macq. Vol. 1. 1. numediart. 2008. Pp. 17–34. URL: http://www.numediart.org. Pp.: 41, 52, 53.

[40] M. Mancas et al. "Gestures Saliency: a Context-Aware Analysis". In: *Lecture Notes in Computer Science* (to appear). Pp.: 47, 51.

[41] Matei Mancas et al. "MATRIX: Natural Interaction Between Real and Virtual Worlds". In: *QPSR of the numediart research program*. Ed. by Thierry Dutoit and Benoît Macq. Vol. 2. 1. numediart. 2009. Pp. 1–7. URL: http://www.numediart.org. P.: 47.

[42] Matei Mancas et al. "Tracking-Dependent and Interactive Video Projection". In: *QPSR of the numediart research program*. Ed. by Thierry Dutoit and Benoît Macq. Vol. 1. 3. numediart. 2008. Pp. 87–96. URL: http://www.numediart.org. Pp.: 41, 47.

[45] Eduardo Reck Miranda and Marcelo Wanderley. *New Digital Musical Instruments: Control And Interaction Beyond the Keyboard*. Computer Music and Digital Audio Series. A-R Editions, 2006. ISBN: 9780895795854. P.: 41.

[46] Jin Moen. "From Hand-Held to Body-Worn: Embodied Experiences of the Design and Use of a Wearable Movement-Based Interaction Concept". In: *First International Conference on Tangible and Embedded Interaction (TEI'07)*. 2007. URL: http://www.movintofun.com/Documents/Research/Moen_TEI07.pdf. P.: 48.

[48] Doug Van Nort et al. "Extraction of Gestural Meaning from a Fabric-Based Instrument". In: *Proceedings of the 2007 International Computer Music Conference (ICMC2007)*. 2007. URL: http://www.idmil.org/_media/wiki/dvnt_icmc07_wysiwyg_final.pdf. P.: 53.

[50] Enkhbold Ochirsuren et al. "Distributed Embedded Systems: Design, Middleware and Resources". In: Springer Boston, 2008. Chap. TinyOS Extensions for a Wireless Sensor Network Node Based on a Dynamically Reconfigurable Processor, pp. 161–170. P.: 53.

[51] Joseph Polastre, Robert Szewczyk, and David Culler. "Telos: Enabling Ultra-Low Power Wireless Research". In: *The Fourth International Conference on Information Processing in Sensor Networks: Special track on Platform Tools and Design Methods for Network Embedded Sensors (IPSN)*. 2005. URL: http://www.cs.berkeley.edu/~culler/papers/spots05-telos.pdf. Pp.: 48, 53.

[53] Yannick Prié, Olivier Aubert, and Bertrand Richard. "Démonstration: Advene, un outil pour la lecture active audiovisuelle". In: *IHM'2008*. 2008. URL: http://liris.cnrs.fr/advene/doc/advene-demo-ihm08.pdf. P.: 52.

[54] Vincent Puig et al. "Interfaces pour l'annotation et la manipulation d'objets temporels : une comparaison des outils et des paradigmes dans le domaine musical et cinématographique". In: *Atelier IHM 2007*. 2007. URL: http://www.iri.centrepompidou.fr/seminaires/ihm.php. P.: 52.

[55] M. Rehm, N. Bee, and E. André. "Wave Like an Egyption - Accelerometer Based Gesture Recognition for Culture Specific Interactions". In: *Proceedings of HCI 2008, Culture, Creativity, Interaction*. 2008. URL: http://mm-werkstatt.informatik.uni-augsburg.de/files/publications/199/wave_like_an_egyptian_final.pdf. P.: 42.

[56] Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. "Accuracy of GPU-based B-Spline Evaluation". In: *Proc. Tenth IASTED International Conference on COMPUTER GRAPHICS AND IMAGING (CGIM)*. 2008. URL: http://www.dannyruijters.nl/docs/GPU_AccuracyBSpline.pdf. P.: 51.

[58] Andy Schmeder and Adrian Freed. "Implementation and Applications of Open Sound Control Timestamps". In: *Proc. ICMC*. 2008. URL: http://cnmat.berkeley.edu/system/files/attachments/OSC-time-ICMC2008-2.pdf. P.: 53.

[59] Andy Schmeder and Adrian Freed. "uOSC: The Open Sound Control Reference Platform for Embedded Devices". In: *Proc of the 8th International Conference on New Interfaces for Musical Expression (NIME)*. 2008. Pp.: 52, 53.

[60] Norbert Schnell et al. "FTM - Complex Data Structures for Max". In: *Proceedings of the International Conference on Computer Music (ICMC)*. 2005. URL: http://recherche.ircam.fr/equipes/temps-reel/articles/ftm.icmc2005.pdf. P.: 42.

[61] Diemo Scwartz and Matthew Wright. "Extensions and Applications of the SDIF Sound Description Interchange Format". In: *International Computer Music Conference*. 2000. P.: 43.

[64] "Smart sensor integration: A framework for multimodal emotion recognition in real-time". In: *Affective Computing and Intelligent Interaction (ACII 2009)*. 2009. Pp.: 42, 51, 52.

[67] Julien Stamatakis. "Etude et mise en oeuvre d'un réseau sans fil d'accéléromètres pour l'analyse du mouvement". MA thesis. Université catholique de Louvain (UCL), 2008. Pp.: 48, 53.

[68] Hans-Christopher Steiner. "Firmata: Towards making microcontrollers act like extensions of the computer". In: *Proc. NIME*. 2009. URL: http://www.nime2009.org/proceedings/NM090182. Pp.: 52, 53.

[69] Atau Tanaka. "Trends in Gestural Control of Music". In: ed. by Marcelo M. Wanderley and Marc Battier. IRCAM - Centre Georges Pompidou, 2000. Chap. Musical Performance Practice on Sensor-based Instruments, pp. 389–406. URL: http://www.csl.sony.fr/downloads/papers/2000/AtauIRCAM.pdf. P.: 41.

[71] Todor Todoroff. "Fire Experiences and Projections". In: *CFP of the numediart research program*. Ed. by Thierry Dutoit and Benoît Macq. Vol. 2. 4. numediart. 2009. Pp. 3–5. URL: http://www.numediart.org/files/numediart_2009_s08_cfp.pdf. P.: 53.

[72] Todor Todoroff and L. Reboursière. "1-D, 2-D And 3-D Interpolation Tools for Max/MSP/Jitter". In: *Proc. ICMC '09*. Montreal, Quebec, Canada 2009. Pp. 447–450. P.: 45.

[73] Todor Todoroff et al. "Dancing Viola". In: *QPSR of the numediart research program*. Ed. by Thierry Dutoit and Benoît Macq. Vol. 1. 4. numediart. 2008. Pp. 129–146. URL: http://www.numediart.org. Pp.: 41, 42, 44, 45, 49, 50, 52, 53.

[74] Todor Todoroff et al. "Extension du Corps Sonore - Dancing Viola". In: *Proc. NIME '09*. Pittsburgh, Pennsylvania, USA 2009. Pp. 141–146. Pp.: 44, 45.

[76] M. M. Wanderley et al. "SensorWiki.org: A Collaborative Resource for Researchers and Interface Designers". In: *Proceedings of the 2006 International Conference on New Interfaces for Musical Expression (NIME06), Paris, France*. 2006. URL: http://recherche.ircam.fr/equipes/temps-reel/nime06/proc/nime2006_180.pdf. P.: 52.

[77] Daniel Weiskopf. *GPU-Based Interactive Visualization Techniques*. Mathematics and Visualization. Springer, 2006. ISBN: 3-540-33262-6. URL: http://www.vis.uni-stuttgart.de/~weiskopf/gpubook/. P.: 51.

[79] Stephen Wilson. *Information Arts: Intersections of Art, Science, and Technology*. Leonardo. The MIT Press, 2002. ISBN: 0-262-23209-X. P.: 41.

[80] Matthew Wright et al. "New Applications of the Sound Description Interchange Format". In: *International Computer Music Conference*. 1998. P.: 42.


### 7.2. Artistic references


[34] Nios Karma. URL: http://www.nioskarma.com. Pp.: 41, 42, 49, 53.

[43] Vanessa Le Mat. URL: http://www.vlemat.com. Pp.: 42, 53.

[44] Margie Medlin. *Quartet Project*. 2004–2007. URL: http://www.quartetproject.net/. P.: 53.

[47] Michèle Noiret. *De Deux Points de Vue*. URL: http://www.michele-noiret.be. Pp.: 52, 53.

[63] André Serre-Milan. URL: http://www.andreserre-milan.com. Pp.: 41, 42, 53.

[66] Laetitia Sonami. *Lady's Glove*. 1991–present. URL: http://www.sonami.net. P.: 49.

[75] Michel Waisvisz. *The Hands (I & II)*. 1984–2000. URL: http://www.crackle.org. P.: 49.

[83] ZKM. *Bipolar Festival*. 2007. URL: http://container.zkm.de/musik/downloads/ZKMBipolarBrochure.pdf. P.: 53.

[84] Art Zoyd. URL: http://www.artzoyd.fr. P.: 42.


### 7.3. Software and technologies


[1] Ableton. "Live". URL: http://www.ableton.com. P.: 49.

[2] "Advene (Annotate Digital Video, Exchange on the NEt)". URL: http://www.advene.org. P.: 52.

[4] Yannick Allusse et al. "GpuCV: GPU-accelerated Computer Vision". CeCILL-B license. URL: https://picoforge.int-evry.fr/cgi-bin/twiki/view/Gpucv/Web/WebHome. P.: 51.

[10] CNMAT. "uOSC". URL: http://cnmat.berkeley.edu/research/uosc. Pp.: 52, 53.

[12] Arshia Cont. "Pd patches and externals for Realtime Gesture Analysis using Neural Networks and Pattern Recognition mapping techniques". 2004. URL: http://cosmal.ucsd.edu/arshia/index.php?n=Main.Lakitchen. P.: 50.

[14] Cycling'74. "Max/MSP". URL: http://www.cycling74.com. Pp.: 42, 52.

[15] Mark Danks et al. "GEM (Graphics Environment for Multimedia)". v.0.91. 2009. URL: http://gem.iem.at. P.: 52.

[16] Maurin Donnneaud and Vincent Roudaut. "XYinteraction". URL: http://xyinteraction.free.fr. P.: 53.

[18] "EyesWeb". URL: http://www.eyesweb.org. Pp.: 48, 51.

[21] Jehan-Julien Filatriau. "Example of bio-controlled sound generation". URL: http://www.tele.ucl.ac.be/~jjfil/biosignals_demo/biosignals_demo.html. P.: 50.

[23] "Firmata". URL: http://www.firmata.org. Pp.: 52, 53.

[26] "GDIF (Sound Description Interchange Format)". URL: http://www.gdif.org. P.: 43.

[28] Jason Hector. "OscVstBridge". v 1.0 (GPL). 2009. URL: http://oscvstbridge.sourceforge.net/. P.: 50.

[29] "I2C Bus". URL: http://www.i2c-bus.org/. P.: 53.

[30] IRCAM. "FTM". URL: http://ftm.ircam.fr. P.: 42.

[31] IRCAM. "Gesture Follower using FTM and MnM". v0.3. URL: http://ftm.ircam.fr/index.php/Gesture_Follower. Pp.: 42, 50.

[35] Michael Kipp. "ANVIL: The Video Annotation Research Tool". URL: http://www.anvil-software.de. P.: 52.

[49] *NVIDIA CUDA™ Programming Guide*. v. 2.2.1. 2009. URL: http://www.nvidia.com/object/cuda_get.html. P.: 52.

[52] IRI / Centre Pompidou. "Lignes de Temps". URL: http://www.iri.centrepompidou.fr. P.: 52.

[57] Joseph A. Sarlo. "PdVst: VST - Pd Bridging Plugin". v0.0.2. 2004. URL: http://crca.ucsd.edu/~jsarlo/pdvst/. P.: 50.

[62] "SDIF (Sound Description Interchange Format)". URL: http://sdif.sourceforge.net. P.: 42.

[65] "Smart Sensor Integration (SSI)". URL: http://mm-werkstatt.informatik.uni-augsburg.de/project_details.php?id=49. Pp.: 42, 51, 52.

[70] "TinyOS". URL: http://www.tinyos.net. Pp.: 49, 53.

[78] "WiiGLE". URL: http://mm-werkstatt.informatik.uni-augsburg.de/project_details.php?id=46. P.: 42.

[81] Pete Yandell. "MIDI Patchbay". v.1.0.3, Pete's Public License. 2008. URL: http://notahat.com/midi_patchbay. P.: 50.

[82] "ZigBee". URL: http://www.zigbee.org. P.: 53.