# OMax

*The Software Improviser*


*© Martin Iartigues*

# Part II : OMaxVideo

Documentation by G. Bloch, May 2008
Omax is designed and developed by *The OMax Brothers:* G. Assayag, G. Bloch, M. Chemillier
OMaxVideo is designed and developed by G. Bloch, with contributions by E. Rossez, A.-S. Joubert, V. Robischung
http://www.ircam.fr/ircam/equipes/repmus/OMax

# OMaxvideo


*A typical OMaxvideo setup on one single computer. On the right, the rendering window on the top. Below, the OMaxvideo window itself, with two rectangles: acquisition (bottom, salmon pink) and cutting and mixing (top, green). The yellow green window on the bottom left of the OMaxvideo window is the Video_send extension of OMax, which send all instructions from OMax to the video program. The camera used for the picture is the own iSight camera of the computer; therefore the author is seen on the capture screen, wondering what all this mess is about.*

Readers of this part must be already familiar with OMax.
OMaxvideo requires jitter, the video extension of Max/msp.
OMaxvideo works on G4, G5 and Intel Apple Macintoshes. The most recent machines are recommended, but it also works quite easily on two older machines.

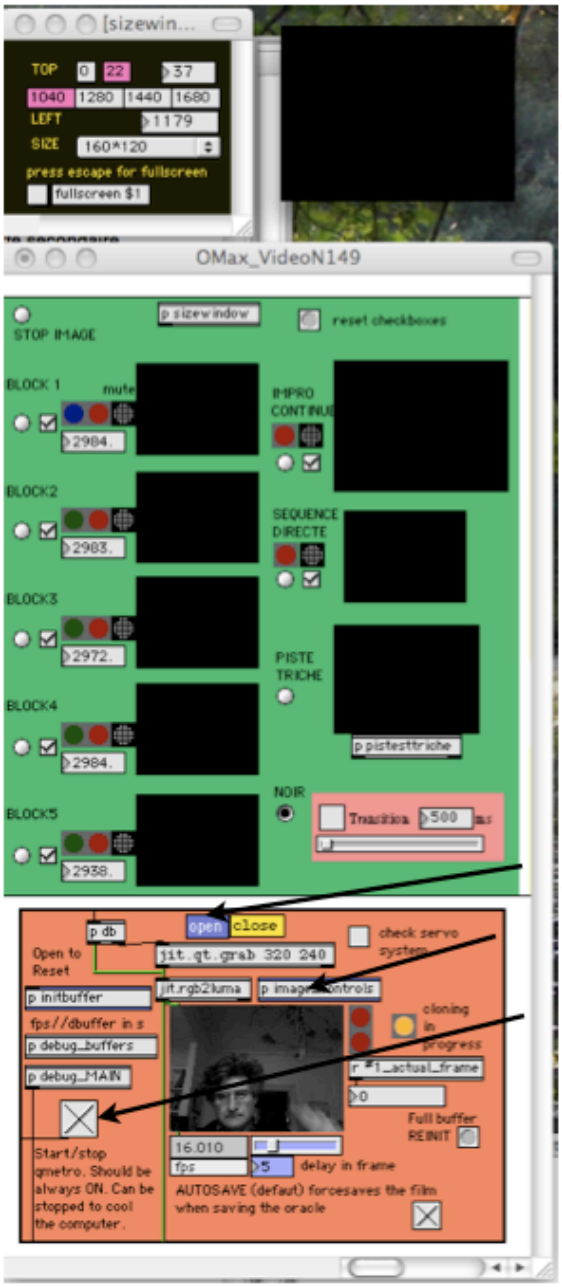
## *Introduction: Sound-driven Video*

OMax_Video follows basically the same principle as OMax audio. In OMax audio – whether pitch detection or spectral descriptors are used – to each oracle event is assigned a date in an audio buffer. In OMaxvideo, a video is realized as the musician is playing (it normally shows the performer, but, of course, anything can be filmed as the music goes along). When the musical improvisation goes on, the film is “cut” according to the re-injection of the played passages. So the filmed musician is always in sync with the improvisation and is actually seen playing something never played before.

Although it is not – by far – its most interesting feature, OMaxvideo gives a pedagogical view of OMax: actually, in its most successful improvisation patterns, OMax makes it very difficult to actually hear when a recombination takes place. But, of course, it can often be easily seen, since a cut in film generally is not as hard to catch.
OMaxvideo is just fascinating in the way it allows sound-driven video improvisation. You can even load films with no relationship with the music and have them be recombined according to the logic of the music played.

## *Setup: the Two-Computers Principle*

During the conception of OMaxvideo, the video was running on a (second) different computer from the one running OMax. There were historical reasons: both programs are quite CPU intensive and use a lot of memory. Also, it was (and still is) more convenient.
Now you can take a big breath: if you own a brand new Intel mac, YOU DON'T NEED TWO COMPUTERS. It even works on older G4 if you stick to a (very low) frame rate. However, it is easier to understand OMax_Video with two computers; it is also easier to manipulate, because you get two relatively large user interfaces, hard to master with one single keyboard and mouse. Finally, it is also easier to understand: you have C1_OMax, running OMax, and C2_Video, running the video, even if it is actually one single machine.
In any case, C1_OMax and C2_Video communicate with OSC (and a single computer uses OSC as well). C1_OMax, the OMax computer, is the boss, and the video computer (whether or not the same), is slaved to it.
By a way of consequence, OMaxvideo consists of two different programs:




*Rendering window [5]*

*[8] open acquisition and setup*

*video [7] ON/OFF*

- To_Video: for the OMax computer [C1_OMax], sending the OMax commands. It is basically an appendix to OMax allowing to redirect OMax instructions to the C2_Video
- OMax_Video: for the video computer [C2_Video], recording and playing video and following the instruction from OMax.

## Setup

**“Ideal” setup.**

- OMax computer (C1_OMax), with a sound setup. On it OMaxMax, OMaxLisp and To_Video
- OMax_Video computer (C2_Video), with jitter installed, and the OMaxVideo Package
- a Firewire camera to C2_Video
- a network connection between both computers (a Wifi computer to computer network is perfectly acceptable)

**“Minimal” setup.**

- One computer with OMaxMa, OMaxLisp, To_Video and OMaxVideo packages
- a built-in iSight camera or webcam.

**“Reasonable” setup.**

- A reasonably powerful computer with OMaxMa, OMaxLisp, To_Video and OMaxVideo packages
- A good soundcard
- A firewire camera

## *Tutorial II-1: OMaxVideo Quick Start*

(If you don't understand steps 1 to 3, refer to the OMax Tutorials.)

1. Launch OMaxLisp. [on C1_OMax]
2. Launch OMaxMax [on C1_OMax]. Do the proper initialization setup: MIDI, event_audio or spectral_audio. In the last case, raise the continuity factor to a higher value (like 80).
3. Do **OMaxInitSession**
4. Launch To_Video [on C1_OMax]. A dialog box asks for the the IP address of the Video computer. If there is only one computer, declare 127.0.0.1 as the IP address, otherwise, give the IP address of the video computer [C2_Video] on the network
5. [On C2_Video] launch OMax_Video. A small rendering window appears at the top right of the screen. You can make it larger if you want through the black window called sizewindow; for the moment, keep it as it is.
6. If you are not in MIDI mode, TURN ON Audio [on C1_OMax]
7. Turn on the video on the **qmetro Start/stop** large toggle, bottom left of the OMax_Video window in the salmon colored rectangle.
8. Turn on the acquisition: plug in a firewire camera or use a webcam (or the included iSight webcam). Then press **open** on top of the pink rectangle. The image of the camera should appear on the screen with a delay of 5 frames. If you don't see the right image (not the firewire camera image but the iSight), either don't bother for the moment, or open the patcher **image_control** in the pink rectangle and play with the input menu. If it does not work, take a look at Tutorial II-3.
9. Turn on **RECORD ENABLE** [on C1_OMax]

Play. Normally, the video should be improvising in sync with the sound.

## *Tutorial II-2: Initialisation and performances*

Here are the default values for the video:

- fps: 15 frames per seconds
- black and white (1 plane) 320 * 240 video
- length of the buffer: 600 seconds

These values are comfortable for OMax_video running on a single recent Intel mac (may 2008).
WARNING: the video buffer is shorter than the audio buffer: by default, no video will be recorded after 10 minutes.
For G4 computer, make sure you use "number" display in OMax. OMax_Video runs on one single G4 computer with 10fps and a short buffer (less than 8 minutes, i.e. 480 sec) in MIDI or Audio-event mode (but not in spectral mode, more CPU expensive). A G4 can actually run up to 15 or 16 fps and a much larger buffer when the computer runs only OMax_video and has a sufficient RAM.

### Memory and performance considerations

With an Intel mac and 2GB of RAM, it is possible to go up to 20 frames a second with a 600 seconds buffer. Too high values create a lost of sync between audio and video.
Large buffer sizes will induce a lot of disk swapping. Generally, the performance drops heavily when the system starts to write on disk; with a fast enough disk, speed often goes back to normal, but with such a delay that it becomes impossible to make up for it. In that case, the video recording stops as a message **STOP: fps TOO HIGH** appears in the max window.
The solution is, of course, adding more RAM and, for performances, a better system and/or the use of two computers, with one dedicated to video.

Actually, the size of the buffer bears at least as much importance to the performances than the fps rate.
Actually, a frame represents 320*240 characters (77Kbytes). At 16 fps and for a buffer of 600 seconds, the buffer size is 737 Megabytes big, which takes a lot of memory (to which are added the 160 Megabytes of the sound buffer).
The optimal size (20 fps, 20 minutes buffer) would end in a 1844 Megabyte video buffer, and more than 2 Gigas with the audio!!!

### OMax_Video servodrive

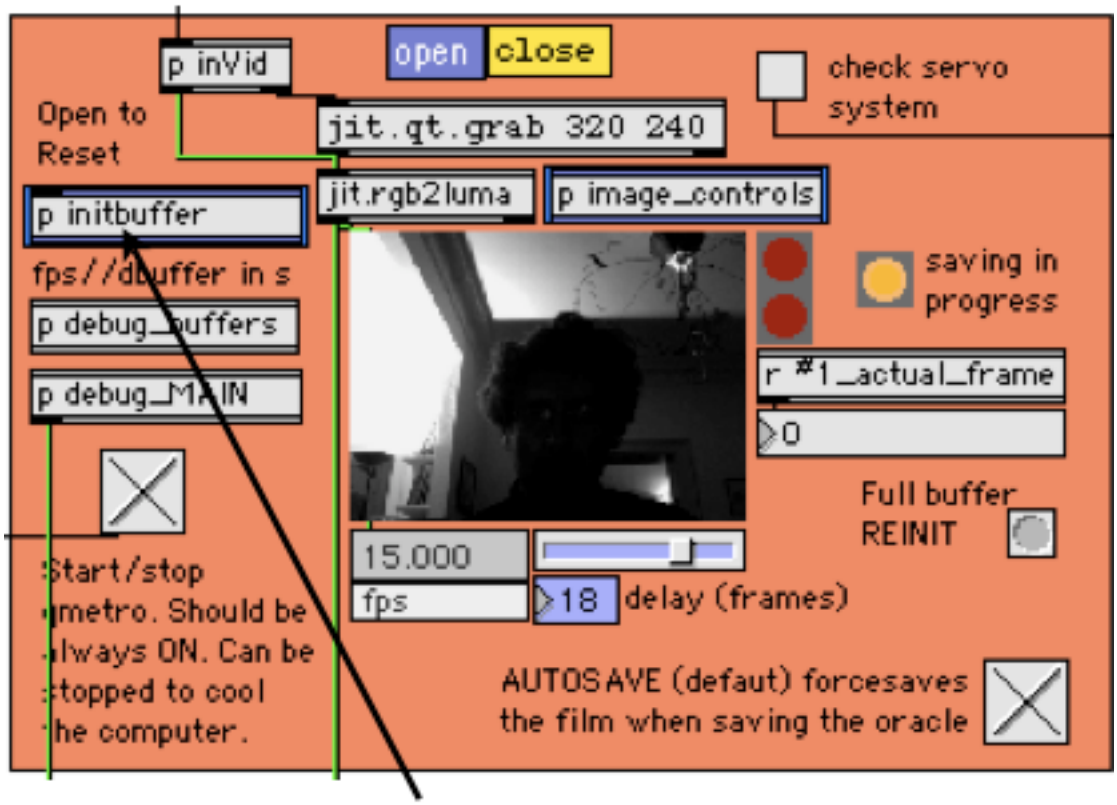
OMax video is sound driven, that is, the date of the sound event decides of the date of the video event. By a way of consequence:

1. One should "drift" in time during the performance. In other terms, better some local imprecisions than a sync losing frames, then seconds, then minutes as the performance goes
2. Of course, dropped frames are strictly forbidden (black frames in the middle of a film are always problematic).

In consequence of item 2, no frame is going to be written in the video buffer before the preceding one has been completely recorded into the buffer. This can slow down the system more than the qmetro can manage. The principle of the qmetro is that any loss of time is caught up on the very next bang of the metro. If the rate is 50. milliseconds (corresponding to 20 fps), and if an unexpected CPU expensive event delays the scheduler by 112 milliseconds, at least a frame will be lost and never made up. To insure a long-term sync (Item 1), OMax video regularly checks the actual position on the buffer compared to the ideal position – that is, the frame position corresponding to the actual position on the **sound** buffer. The fps rate is recomputed each 2 seconds in order to make up for the eventual delay (or advance) in the video buffer in less than 10 seconds.
If the values asked are beyond the system capacities, either the calculated fps will become unrealistic or the out-of-sync will grow to impossible values: in these cases, the video.recording stops.

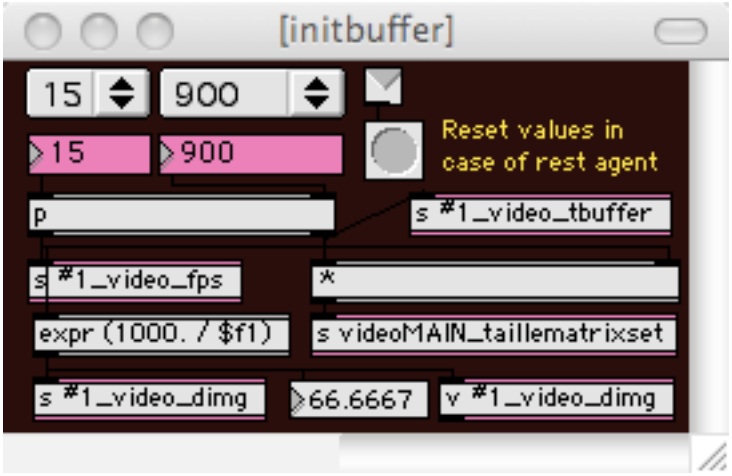
## Changing the init values

The patcher init_buffer allows to change the default values. You simply can double-click on it:




*A double click on initbuffer allows to change the default values for fps and buffer size*

It opens a subpatcher proposing different choices through menus or number boxes:




We don't have to stick to one of the menu choices (even if we suggest it); in any case, your choice depends on the system and RAM you have, and whether you use one or two computers. The cheapest setup are on top of menus. The most expensive on bottom. 15 to 18 fps and 600 seconds is reasonable for an Intel mac with 2GB of RAM. 15 fps and 900 sec also works. On a single G4 the only possible setup is 10 fps and 300 or less than 600 sec.

Using two computers is the easiest way to improve the performances.

## Compatibility considerations

The frame rate is important when you save the MAIN oracle and reload it. A reloaded MAIN will be played at the frame rate of the actual session, not the recorded one. Therefore, it is important to keep record the of the frame rate in mind if you plan to continue the session. However, if you load a file oracle, OMax_Video adapts the frame rate [See Tut.II-5]

## *Tutorial II-3: Acquisition*

### Setup

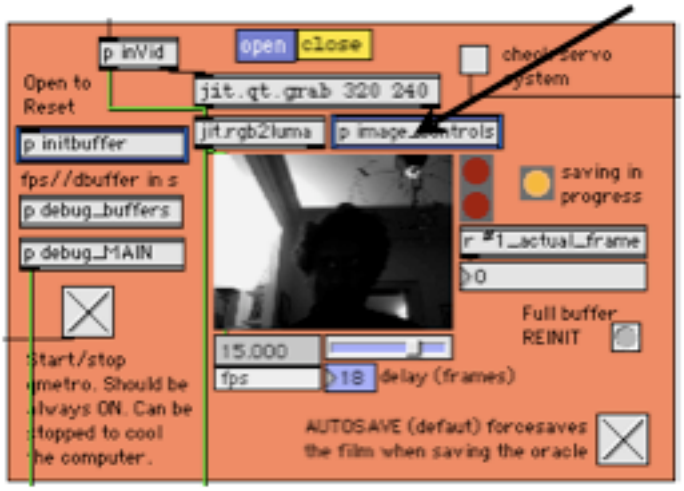
- You need a camera going into the computer with a USB or Firewire cable. A webcam or the iSight can be used.
- The acquisition is made in black and white: the color image is transformed into a one-layer matrix using rgb2luma. The picture size is 320 by 240 pixels.
- However, as always, the optical quality of the camera matters, and its capacity to adapt to different kinds of light.
- You can even plug different cameras and choose from them. Since it takes some time to switch, you must do it when the musician is not playing.

WARNING: Don't forget to TURN ON the **Start/stop** button and to OPEN the grabber! The delayed image from the camera must appear on the screen-window of the acquisition rectangle.

### Controls

The **Start/Stop** button for the video [Tut. II-1] and the **initialization** sub-patcher [Tut. II-2] are situated on the acquisition rectangle. The window also includes:

- The **image_controls** setup. By double-clicking into the patcher box, you open the actual control window (arrow on the picture)
- **Debug sub-patches** to allow you to examine the content of your audio buffer
- an advanced tune up window to verify the **servodrive** system
- a subwindow **inVid** in case you want to use a film as input a film [See Tut. II-6]
- two displays to show current **fps** and the current recorded frame (**actual frame**)




- and a value for the **delay** of the Video recording. The audio is recorded with 400 ms delay. In theory the video delay should be the same, but the camera and the jitter acquisition already induce a certain delay. I suggest more the equivalent of 250ms. With webcams, an even shorter value becomes necessary (like 1 or 2 frames delay).

**Full Buffer REINIT (IMPORTANT!)**

In case you record to the end of your video buffer, you need to bang on that button to be able to record again (generally, you do it if you reset your main oracle or your whole session after filling up the buffer).

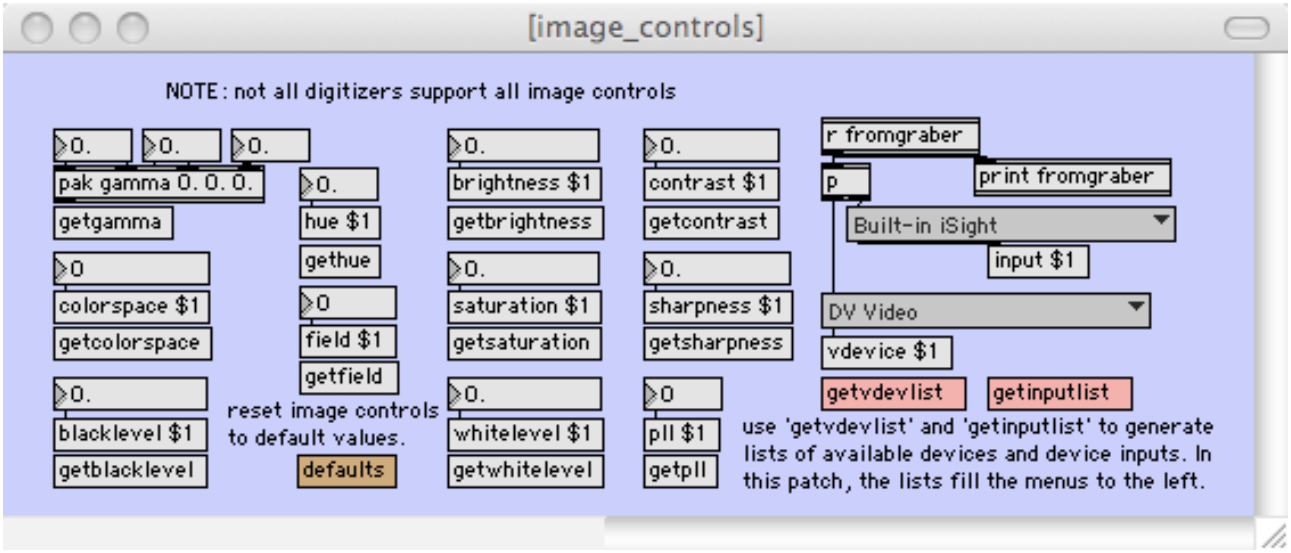
**AUTOSAVE**

By default, the film is saved on your Video computer if you save your Oracle on the OMax computer (once more, whether or not it is the same computer). You MUST save your oracle with a ".or" ending. If you save it under the name *MyNamedOracle.or*, it also saves the audio buffer as *MyNamedOracle.aif* AND the video buffer as *MyNamedOracle.mov*.

If you don't want this facility, you uncheck the **AUTOSAVE** button. When you save your oracle, it will prompt you with a dialog box to save (or not) your film. More about saving is said in Tut. II-5.

## image controls

When you open the image_controls subpatch, here is what you see:

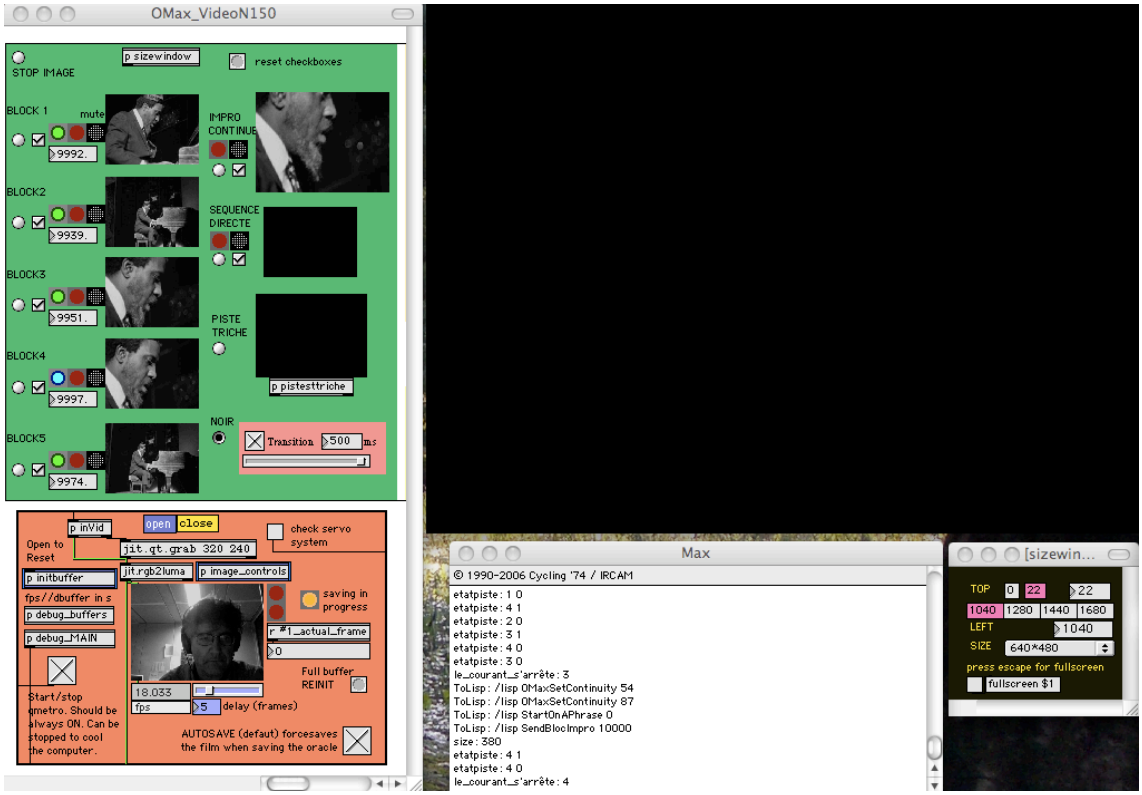



This is actually a generic jitter help patch with a little modification.
In order to get the right camera, you have to click in **getdevlist** and **getinputlist**. The list of available Input will be printed in the max video. By choosing the right input (iSight on the above figure) and device (if you choose USB video, your camera riks not working even if it is selected), you get the right picture in the acquisition screen.
The other values are for image control. As written, not every digitizer supports these parameters and, sometimes, they support them only locally, which means you have to do the setup on the camera itself.

## *Tutorial II-4: Rendering*

### Rendering Window

The upper part of the OMax_Video window is occupied by a green rectangle: the *mixing* part. Actually, two other parts are also important for rendering: the window **sizewindow** and the actual jitter window – we will call it the *screen window* – in which the image is actually seen.

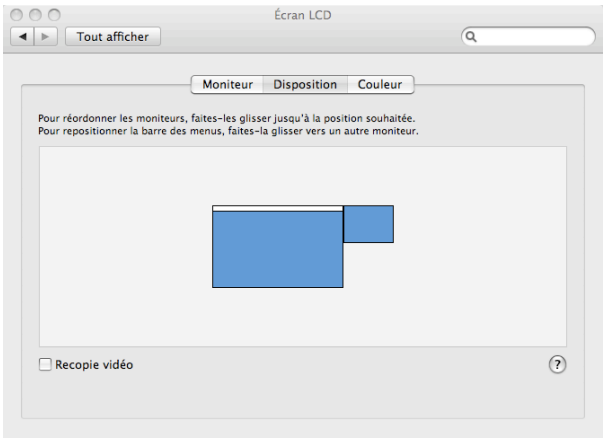



*The* mixing *window is the green part of the OMax_Video window. But the actual rendering is made on the so-called* screen window *(here on top right). The size and position of the screen window can be set with the small black window* ***sizewindow*** *on the bottom right of the picture.*

### Size and position of the screen window

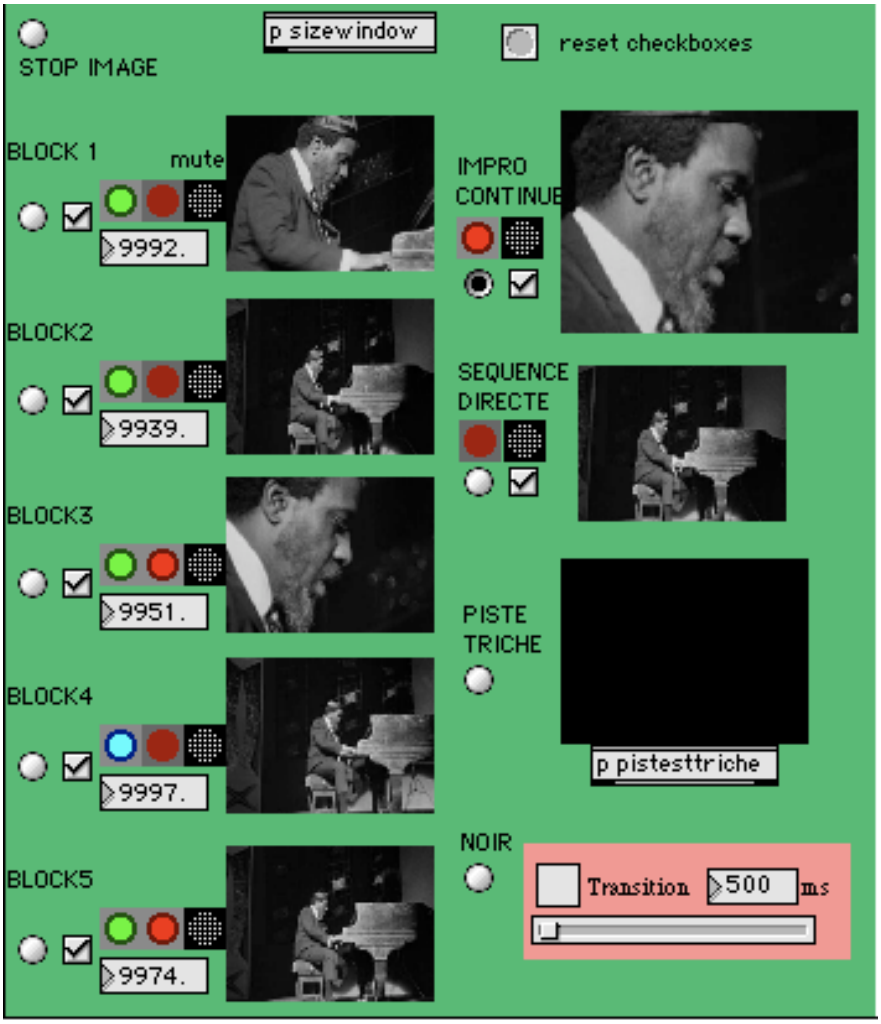
IMPORTANT: for the moment, OMax_Video renders a 320 * 240 image. It is useless (and costly) to use a much bigger definition for the screen window, since it does not actually improve the image. The choices given are half-size, 100% size and 640 * 480, a useful definition for a second monitor or for a video projector.

The small black window called **sizewindow** proposes these choices of size, as well as the position in pixel of the top left of the screen. With a video projector (or a second monitor), the choice 0 –for the top – and your monitor width (for the left) allows to easily place the window at the top right of your monitor (as shown is the side picture), and with 640*480 as your monitor definition you have the image on the whole of the video monitor. Another (easy) way is to move the video image into the second monitor and to push the button **fullscreen** or press the **escape** key.



## Video Mixer

The video mixer is pretty straightforward.




You actually get nine video inputs:

- The "continuous" OMax improvisation (top right)
- The five "blocks" corresponding to the equivalent OMax blocks [See OMax Tutorial 9]
- The "direct sequence" corresponding to the equivalent reading of the original ***SeqDir*** in OMax [ See OMax Ref 1]
- A "Cheating track" allowing to play a different movie called ***PISTETRICHE*** [See Tut. II-6]
- And, last but not least, a black screen called ***NOIR***
- The leds mirror the lights in OMax. For the blocks, the left light is yellow when loading, green when loaded and blue for the last one loaded. The red light is on when the track is playing (on the picture, continuous impro and blocks and 5 are playing). The black/white light shows if the track is muted. Muting a track is a way of having an image playing not in sync with what has been heard.

Otherwise, a crossfade value can be set (default 500 ms) for the track change crossfade.

WARNING: the crossfade is only applied when changing track. The OMax editing is cut, for obvious reasons: the video passages can be very short.

## Automatic Mixing Principles: OMax is the boss

The mixer is very simple in principle: Any "checked" track can be played. And by pushing the radio button of the desired track (on the above figure, ***IMPRO CONTINUE***) this very image is sent to the rendering window.

But what is simple in principle is not so simple in practice: if you chose a non playing track, the image will stop. This is the reason why OMax is the boss. The actual interface has been made to be remotely directed from OMax, with an automatic mixing system.

**Priorities**

This system functions with a list of priorities:

1. The last starting track is always rendered, except for the PISTETRICHE (prerecorded video) which is never interrupted
2. When the current rendered track stops:

- if current = CONTINU, next = BLACK
- if current = TRICHE (video), next = CONTINU, otherwise BLACK
- if current = BUFFER, next = BUFFER, otherwise CONTINU, otherwise BLACK

This is a sort of logical setup, since the continuous improvisation is generally more present than the others and that a prerecorded video has always a large priority.

## Work in Progress or toolkit?

We should emphasize that there are a lot of things that OMax_Video does NOT do. Although we have seven tracks available, we present them one at a time. Outside of the technical improvements (more powerful machines will permit larger images, colors, etc. with very little change to the actual version), many things could be done according to the individual project.
Therefore OMax_Video could be seen as a work in progress or, rather, as a toolkit using the OMax structure and allowing basic video capture and rendering.

## *Tutorial II-5: Saving and Importing Video files*

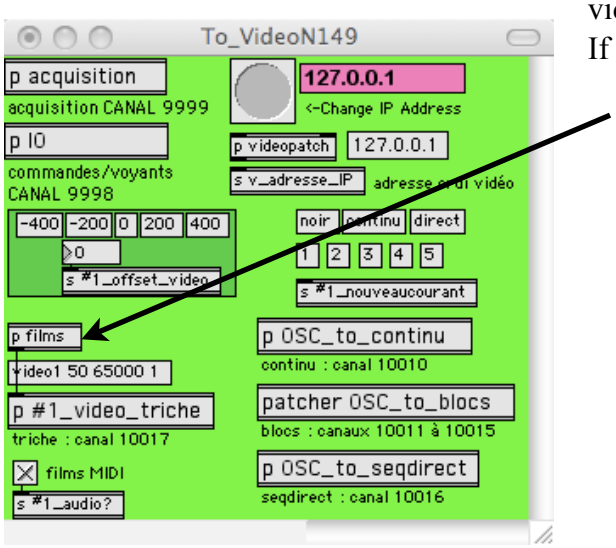
In theory, everything is made from OMax, especially if you use the AUTOSAVE feature.
WARNING: saving films takes time.

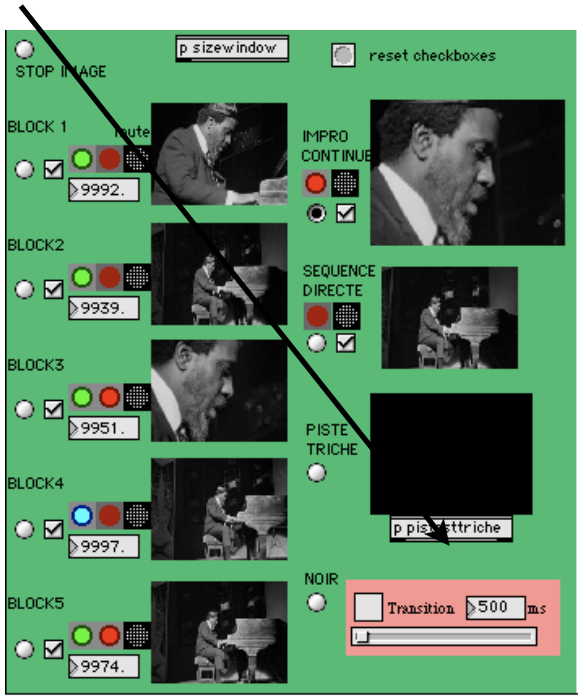
### Saving

**AUTOSAVE or not?**

By default, the film is saved on your Video computer when you save your Oracle on the OMax computer (once more, whether or not it is the same computer). You MUST save your oracle with a ".or" extension [See OMax tutorials 4 and 7]. If you save it under the name *MyOracle.or*, it also saves the audio buffer as *MyOracle.aif* AND the video buffer as *MyOracle.mov*.

If you don't want this facility, uncheck the **AUTOSAVE** button. In any case, when you save your oracle, it will prompt you with a dialog box asking to save (or not) your film.



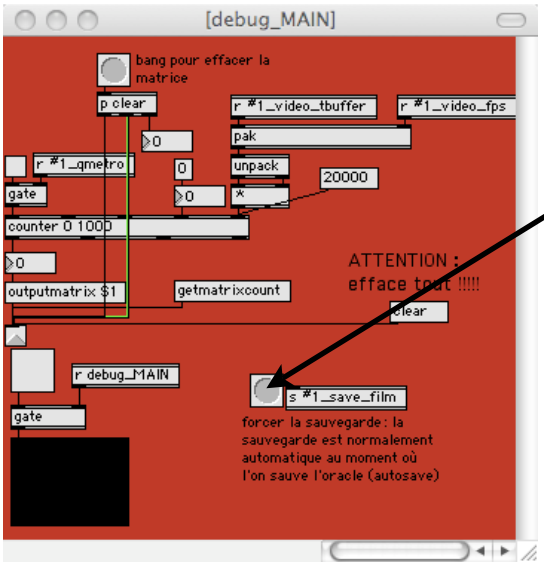

Any auto-saved film is saved in the subfolder **ovideo_films** of the OMaxMaxVideo folder.

WARNING: the name (ovideo_films) and position (inside the OMaxMaxVideo folder) of this folder should remain the same. Not only does this folder contains the autosaved films: it is also reserved for the films you want to include in your OMax_Video session [see Tut. II-6].

In general, we recommend you place your OMax_Video films there, whether or not you use the Autosave feature. Autosave or not, if you save the Oracle, there is little danger to forget saving your film because you will be prompted to save the film. However, it makes a lot of dialog boxes in a row. You save the Oracle under *MyOracle.or* and a second dialog box comes to save the film. With Autosave, you just have one Dialog.

WARNING: Saving a film is a long (and blocking) process.

**Saving the Film and not Necessarily the Oracle or the Audio**




You may want to save the film and not the oracle, or the film independently of the oracle. You can force save the film. By opening the subpatch **debug_MAIN** in the acquisition (salmon pink) rectangle, you get this window:

By clicking on the button, you force the film to be saved. Of course, it is mandatory to do this when OMax is not running. It is even advised to stop the main metronome in the acquisition window to speed up the saving process.

**The whole buffer is saved.**

It is just faster, because saving just the film with its good length would oblige to make a copy of it, that is, twice the buffer space (a very good way of crashing)!

The black frames can easily be cut in iMovie. It is strongly recommended to do so before importing the new oracle.

## Importing files

As seen in OMax tutorial 4 (Housekeeping), you can import and export oracles to play them alternatively with the MAIN (that is, learning) oracle. You can even import a MAIN oracle and learn on top of it (that is, in sequence, the new session starting at the end of the preceding one). The caution about audio buffer becomes particularly acute with video buffers. Five ten minutes video buffers will use a lot of memory! This is why you want to cut the black frames at the end of the movies.

Importing is straightforward: OMax asks for an Oracle file (*MyOracle.or* and, for an audio oracle, the associated *MyOracle.aif)*. Then it opens another dialog asking for a movie. The movie can be anywhere and any name as long as jitter understand its format. In fact, as long as the movie is long enough, it can bear no relationship with the sound; and even if it is to short, your image will "freeze" when it will be asked to edit on a non existing frame. This is a quite interesting way of "cheating" [See next tutorial], by applying a strict musical edit on non related images.

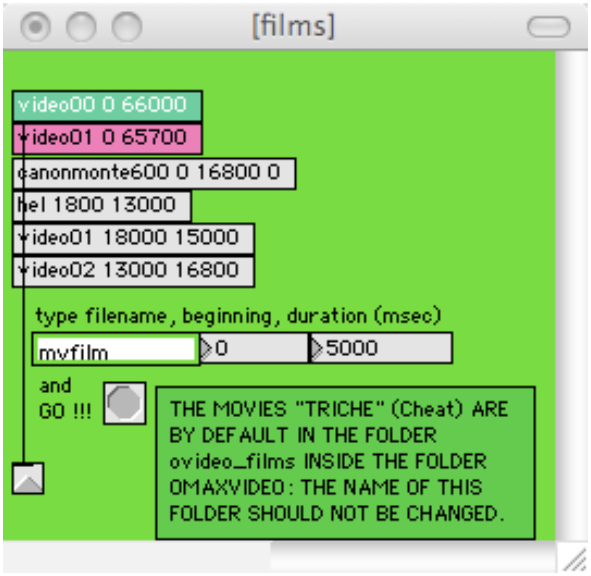
WARNING: the movie is transformed in black and white. If you have a color movie to start with, it can be quite time consuming.

## *Tutorial II-6: Cheating*

The basic principle of a video slaved to audio can become tedious. You could, sometimes, want to drift from the one to one relationship of what is heard and what is seen. OMax_Video has several solutions to that: the last tutorial just showed how a non-related film can be applied to a given oracle. You can also play prerecorded video (cheating track); replay parts of the sequence as originally played without applying the oracle, or use a film as an input as the performance is playing.

### Piste triche = Cheating Track in French

On the OMax computer, on ***To_Video***, you get a sub-patcher named ***films***. The same sub-patcher exists also on the mixing window under the name ***pistetriche***. In case you use two computers, you can get a command from the OMax computer (That is, the audio computer).

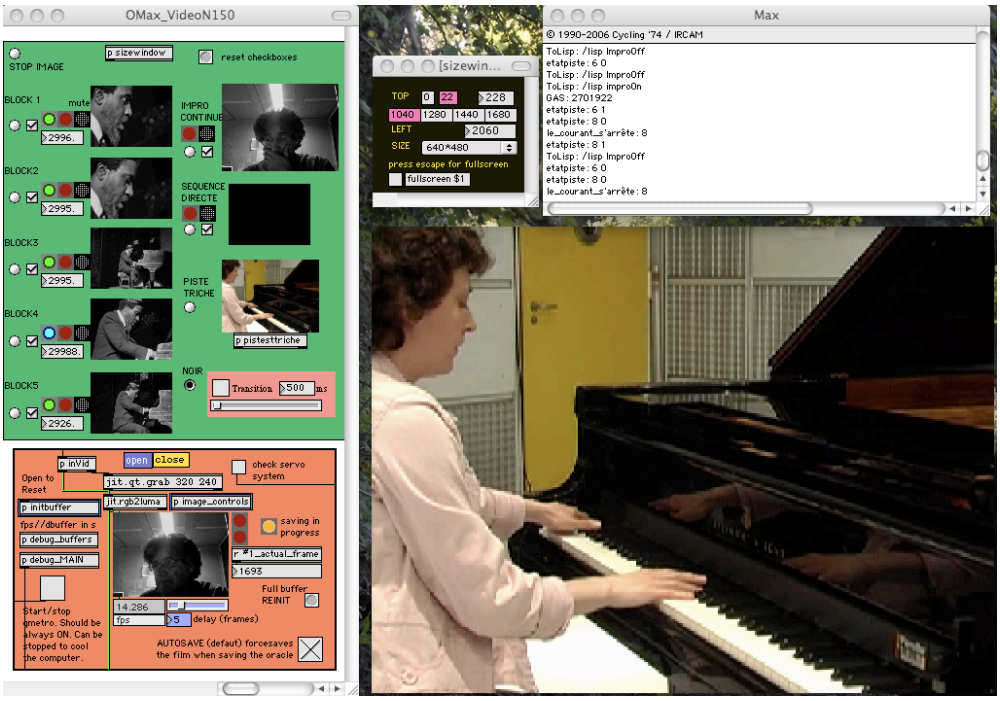



Here is what you get when you open the subpatch: you get a list of films that you can edit (if you have max/Msp), or you can enter the name of the film, its beginning time and duration in milliseconds. Then you bang and the film is immediately played on the rendering window.

NOTE: the PISTETRICHE has always the highest priority. There is no way of stopping it once it is gone, except by launching another one (it can be an extremely short one).

WARNING: as written on the window, the movies should be in the ovideo_films subfolder of the OMax_Video folder. Do not change the name nor displace the folder. If the film is non existent, a dialog box opens, but it is difficult in terms of time precision.

As you can see in the picture below, the “cheating track” displays a real film, with its eventual colors.

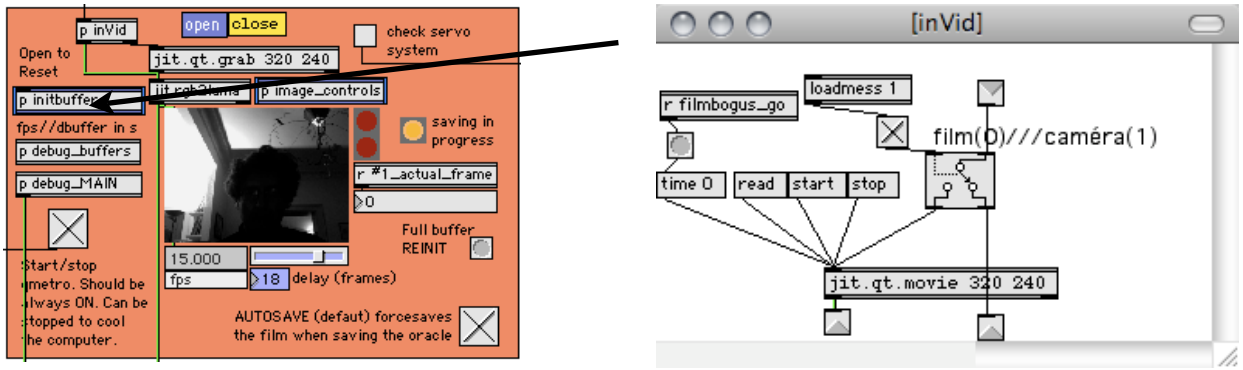


## Sequence

As in OMax, you can replay part of the original sequence, with the film that goes with it. It corresponds to the ***SeqDir*** process explained in the OMax tutorial.

## Using a film as input

You can also use a film as input in real time, that is, read a film as you are recording the audio. In that case the film in transformed in back and white and recorded into the buffer. It just means that, instead of taking the camera input, you are reading a quicktime movie. This is the purpose of the inVid subpatch, on top of the acquisition window:




The window is self-explanatory. By putting the toggle on 0, you take the the input from a film: you can read the file and start it when you feel like it.

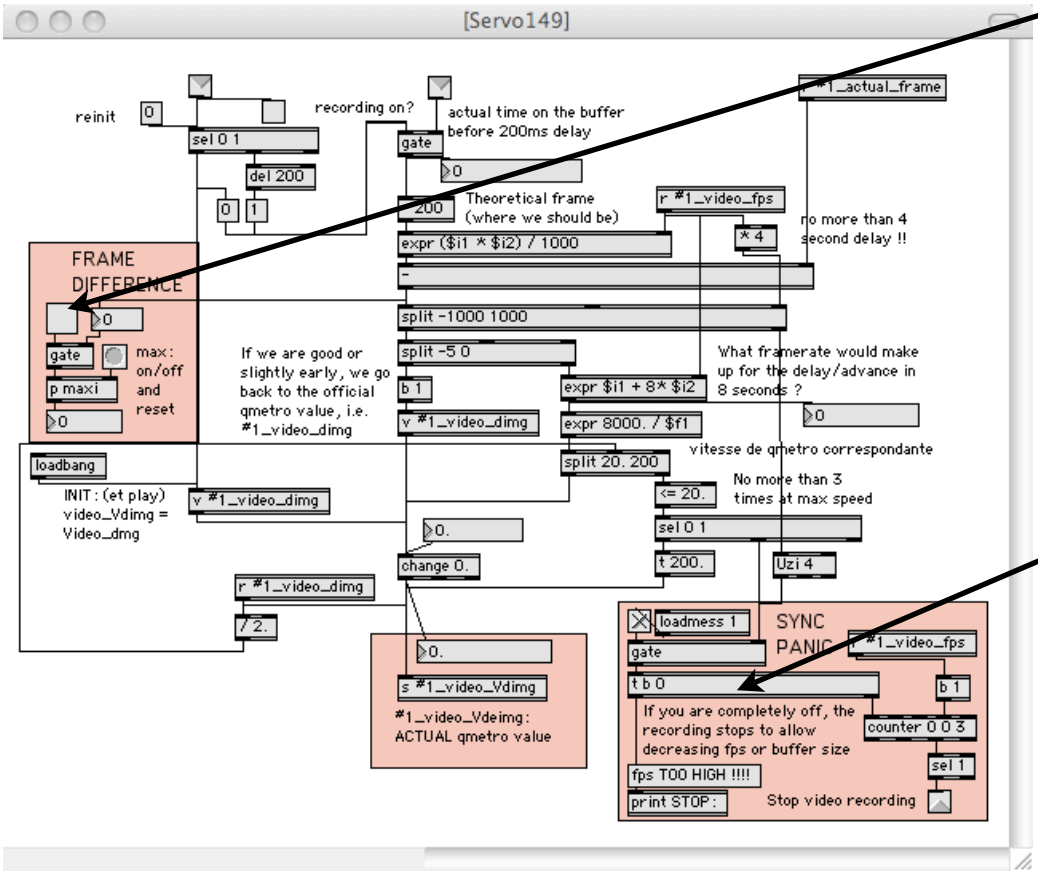
WARNING: Since reading the file can be expensive, it is recommended you close the grabber (by pressing the yellow button ***close*** if you use a film input.

## *Ref II-7: Servo*

Normally, you do not have to open this window. If you get real trouble of sync and want to check on it, this is a good place to start, after you checking that your values of fps and buffer size are not unreasonable given the machine and RAM yo have.

WARNING: disk access applications, like Spotlight (when it reindexes the disk) are strongly to avoid when using OMax or OMax_Video.

Here is the way the Servo window looks:




The left pink zone shows the frame difference. By checking the toggle, you get the maximum number of delay on the bottom. It should not be above 3 frames.

The "panic" zone explains the way recording can be stopped. Either the metro has reached its minimum value more than 4 times, or the delay is over 4 seconds, which is rightly considered as unacceptable. In any case, if you get very high delays, you have either a problem of setup or of other applications.