

M_PLAYER V4.3β

Documentation

Video Player, Maker and Disassembler For the Atari 68030+ computers

Author : guillaume.tello@orange.fr 1996-2023

WEB Pages:

- (in english) <https://gtello.pagesperso-orange.fr/mplayere.htm>
- (in french) <https://gtello.pagesperso-orange.fr/mplayerf.htm>

Thanks to **Dieter Fiebelkorn** for sharing its original C work on AVI and MOV.

Thanks to **Didier Mequignon** for sharing IV32 and IMA4 codecs.

Thanks to **Douglas Little** for providing the fast Byte to Planes conversion for TT-Low resolution

Thanks to **Rémi Vanel** for FLH and partnership on his Mountain software using M_PLAYER.

Thanks to **Olivier Landemarre** for helping me on multitasking and GEM management.

Thanks to **Martin Griffiths** for his fast DCT routine on MPEG.

Thanks to **T. Ferguson** for his web page with the idea of the CRAM encoder.

Thanks to **Nima Montaser** for his partnership on his great Infopedia driver using M_PLAYER.

Thanks to **Eric Da Cunha** for his module JPEG_68K.RIM (from parx M&E system).

Contents

I Introduction.....	5
I.1 RSC discussion.....	5
I.2 Graphic requirements.....	6
I.3 Sound requirements	6
I.4 OS requirements.....	6
I.5 M_PLAYER as an accessory.....	7
II General Menu.....	8
III Load and play a video.....	9
III.1 AVI, MOV and Extended FLM.....	9
VDI button.....	9
W button.....	9
R button.....	9
>>Ram button.....	9
Play Sound button.....	9
Enhanced / Record button.....	10
Synchronize button.....	10
Save As button.....	10
Options button.....	10
Play ! Button.....	10
III.2 MPEG files.....	10
Grey Levels button.....	11
Dirt mode button.....	11
III.3 MJPEG files/Codec in AVI-MOV.....	11
III.4 Speed up some slow animations.....	12
III.5 GIF files.....	13
III.6 Old Atari/PC animations files.....	13
FLI, FLC, FLH.....	13
SEQ.....	13
PI1+DLT.....	13
FLM.....	13
DL.....	13
FLM.....	14
FLM.....	14
CDH, CDL, CDV.....	14
IV Disassemble or Convert.....	15
IV.1 Disassembling a video.....	15
Save Sound button	15
TGA 24b and TGA 16b buttons.....	15
Calc Ratio button.....	15
Interactive saving.....	16
IV.2 Converting a video.....	17
V Build a video from images/Slideshows.....	20
V.1 The integrated panel.....	20
V.2 Limitations and links between image type and animation.....	21
V.3 Structure of a batch file, create it by hand.....	21
The identifier.....	22
The header.....	22
The data.....	24
V.4 QuickTime VR movies.....	25

Fixed cells.....	27
Animated cells	30
Panoramas.....	30
V.5 GIF movies.....	31
V.6 FLM Movies.....	31
V.7 Example of a BATCH FILE.....	32
V.8 Key-Frames use.....	33
VI Albums.....	35
VI.1 Create an album by hand.....	35
Load, Save.....	37
Reorder.....	37
VI.2 Create an album automatically.....	37
VI.3 Linking an album to a CD-Rom.....	38
Associate CD / Dissociate CD.....	38
CD's Album.....	38
VI.4 Using an album.....	38
▶ (play).....	38
Skip Dialog.....	38
VII Capture the screen to an animation.....	39
VII.1 The manual mode.....	39
VII.2 The auto capture mode.....	41
VII.3 Capturing using a batch file.....	41
VIII Enhanced files.....	42
VIII.1 The header.....	42
VIII.2 The different versions.....	42
VIII.3 The events during replay.....	43
VIII.4 An example.....	43
VIII.5 Pausing.....	44
VIII.6 Recording an Enhanced file.....	45
IX Set Options and Paths.....	48
IX.1 Replay Options.....	48
Default path.....	48
Always count MPG frames.....	48
Sound system.....	48
MJPEG frames per second.....	49
Max button (for MJpeg).....	49
Enh button.....	49
Win button.....	49
IX.2 Save and convert options.....	49
Create Folder with Anim name.....	49
Auto Name.....	50
Disable.....	50
Stats after conversion.....	50
Warnings.....	50
Set Sound.....	50
X Search a disk for animations.....	52
X.1 Slideshow of animations.....	53
XI EASY_BAT.PRГ (discontinued).....	54
XI.1 Image type selection.....	54
XI.2 Size of the animation.....	55
XI.3 Output file selection.....	56
XI.4 Time and sound.....	56

XI.5	<i>Start image selection.....</i>	56
XI.6	<i>Global repetition.....</i>	59
XI.7	<i>Turn into a QuickTime VR/1.....</i>	59
XI.8	<i>Save the batch file.....</i>	59
XI.9	<i>Create the animation.....</i>	60
XI.10	<i>Replay the animation.....</i>	60
XII	Command line options.....	61
	+d, -d, -D Display Dialogs.....	61
	+p, -p Play Sound.....	61
	+s, -s Synchronize.....	61
	+a, -a Save As.....	61
	+e, -e Error Messages.....	61
	+i, -i Repeat.....	62
	+r nnnn Return infos.....	62
	+xnnn, +ynnn Coordinates.....	62
	+w, -w Set window use.....	63
XIII	Calling M_PLAYER as an accessory.....	64
	XIII.1 <i>The "Mountain" mode.....</i>	64
XIV	Supported file formats.....	67
	QuickTime *.MOV (VR1 Objects & Vr2 Panoramas supported).....	67
	Video for Windows *.AVI.....	67
	JPEG *.JPG.....	67
	MJPEG *.MJP.....	67
	MPEG *.MPG.....	67
	Extended Lexicor Film *.FLM.....	67
	Compuserve *.GIF.....	67
	Video Master *.FLM.....	68
	CD Stratos Magazine *.CDH, *.CDL, *.CDV.....	68
	Old ST Low Atari animations.....	68
	Old PC animations.....	68
	Batch Files *.BAT.....	68
XV	Translating the interface.....	69
	XV.1 <i>File selector titles.....</i>	69
	XV.2 <i>General info box.....</i>	69
	XV.3 <i>Step by step box.....</i>	69
	XV.4 <i>Free strings.....</i>	70
	XV.5 <i>Alert strings.....</i>	70
	XV.6 <i>General menu.....</i>	70

I Introduction

M_PLAYER is a **video player** for Atari computers running TOS or similar operating system on a Motorola **68030 CPU or better**.

It is also a **video disassembler** and **encoder** that handle various types of formats.

It was written mostly for the Atari TT with a graphic card in 16 bits colors.

The goal of the program was not to make a GEM-friendly application. Knowing that a 32MHz machine is limited in power, I wanted to take the best of the hardware to get a good replay experience : not jerky sound, not jerky display.

That's why M_PLAYER doesn't use the VDI for graphic output when direct access is available, reserves the screen, seldom shares the CPU under a multitasking environment.

M_PLAYER can be used as a **GEM program** but also as a **desk accessory**. Just rename it as M_PLAYER.ACC and copy it in the root of your boot drive. The file M_PLAYER.RSC must be located in the same folder.



I.1 RSC discussion

The program comes with **different resources files** according to **different languages**. They are named M_PLAYER.nnn where nnn is the country code:

- 000 English (US & UK)
- 001 German (and Swiss German)
- 002 French (ans Swiss French)
- 004 Spanish
- 005 Italian

The program **determines the country code** and tries to load the corresponding resource. If this one is not available, then it tries to load a default M_PLAYER.RSC. Again, if this one is not available, then you get an alert and the program stops. For now, two resources are available:

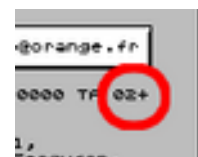
M_PLAYER.RSC, the default english one

M_PLAYER.002, the french one.

If you don't know the **country code of your system**, it is displayed in the **credits dialog** box at the end of the line using the small font:

Here "02+" means french system and that M_PLAYER.002 could be loaded.

For example, "05-" would mean italian system but M_PLAYER.005 was not found and the default M_PLAYER.RSC was loaded.



I.2 Graphic requirements

- ✓ An **Apollo Vampire Standalone V4+** in 16 bits (the best) but other modes are supported.

- ✓ An **Atari TT with a graphic card in 16 bits** is the best.

- ✓ A **Falcon (CT060 supported, CTPCI+Radeon too) in High Color** mode.

In this mode, the video is replayed in a window if it fits the screen, else the whole screen is used.

- ✓ An **Atari TT with a graphic card in 256 colors**.

In this mode, the screen turns black, the video is centered and any animation with more than 256 colors is replayed in grey levels.

- ✓ An **Atari TT with no graphic card**.

The planes management of the Atari video memory makes this mode slower than the others. Anyway, the TT switches to TT Low to replay animations in 256 colors/greys. For earlier animations on Atari, the corresponding video mode is used (ST Low for example).

- ✓ An **Atari TT with the AlberTT** graphic card.

As the card is limited to 16 colors, the displayed is redirected to the standard TT output and works with the TT Low resolution. So you need a second monitor or a switch.

- ✓ A **Falcon in any other mode than High Color**.

In this mode, the Falcon switches to 320x240xTrue Color hoping that the animation fits the screen. This is included but not certified to work...

- ✓ Other **machines with 256/TC modes** (Hadès, MagicMac)

They work as the TT.

- ✓ Any **machine with 24 or 32 bits display**. (Aranyx for example)

In this mode, M_Player has no specific routines, so the 16 bits native routines are used, and, after a reencoding, the VDI is used to display every image.

I.3 Sound requirements

Sound is replayed via the **DMA system** on Atari TT, Falcon and Aranyx.

If no DMA sound is available, then the **Yamaha** chip is used (This is really slow with a low quality). The **Psound card** (parallel sound card) is no more detected with the PSND cookie, you have to set it manually through the Option panel if you own one.

See at Set Options and Paths.

On the Apollo Vampire, the **SAGA sound chip** is fully supported and works like the DMA on the TT.

I.4 OS requirements

M_Player was primarily intended to run with the **standard TOS** system.

It supports **MagiC, MagicMac, Mint, Geneva and EmuTOS**. But in any multitasking system, the program reserves the resources and don't allow the user to work on another application while it is replaying a video.

However, while the dialog boxes are displayed, or during a pause, or when an animation is encoded, then it can be considered as a multitasking application.

There is no more problem with multitasking systems and the Control key as from version 4.00 M_PLAYER doesn't use the supervisor mode. All routines have been rewritten to play in user mode.

I.5 M_PLAYER as an accessory

On some systems, the player can't find its resource file or option file when installed as an accessory.

So, you have to respect this organization:

- ➔ M_PLAYER.ACC in the root of the boot disk or in the ACC folder (under Mint for example)
- ➔ M_PLAYER.RSC, M_PLAYER.OPT, M_PLAYER.AVR, JPEG_68K.RIM in the root of the boot disk, even if the executable is located in the ACC folder.

II General Menu

When you run M_PLAYER, this menu appears giving access to every feature of the program.

You can:

- ◆ **Load and play a video**, see the list of supported formats. The program will do its best according to the capabilities of your system concerning video and sound.
- ◆ **Disassemble or Convert**, you can turn a video into a set of individual images, a sound file or just convert the video from one format to a limited target of formats.
- ◆ **Build a video from images**, you can reassemble a set of images and eventually a sound into various formats of videos.
- ◆ **Albums**, let you create or open an album with a set of videos that you have organized and can play without searching on the disk.
- ◆ **Capture the screen to an animation**, the program lets you define a rectangle and capture the screen (manually or automatically) that is assembled on the fly into a video.
- ◆ **Set Options and Paths**, this is a very important part that you should rapidly fill with your own preferences to greatly ease your work
- ◆ **Search a disk for animations**, a search tool that returns a text file with every video found (selectable extensions) from a particular folder on disk.
- ◆ **Credits**, opens the credits panel.
- ◆ **Quit**, returns to the desktop.

It's time to have a look at every entry of this menu...





III Load and play a video

This **opens the file selector** where you can select a video to be played.

When the program reaches the **end of a video**, it **returns to the file selector**, so you can chain rapidly animations without going back to the menu.

To go **back to the menu**, just click on **Cancel** in the file selector.

III.1 AVI, MOV and Extended FLM

Those three types have a similar behavior : they come with images and sound.

After the file has been selected, the main dialog opens and displays information about the video. It also tells you whether the display and sound codecs are supported or not.

Several buttons are available :

VDI button

If set, you're in 24 or 32 bits display and the VDI will be used instead of direct routines (or in 16 bits under emulators such as Aranyem). *This button is read only.*

W button

If set, the display can be done within a window. You can deselect it to get a black background. When running on a TT with no graphic card, if W is set, then the replay is done in monochrome in a window, else the computer switches to TT Low to use 256 colors/Grey levels.

R button

If set the animation will be repeated until the Control key is pressed.

>>Ram button

If set, the whole video is loaded first in memory, it speeds up the display. If not enough memory available, then this button is disabled.

Play Sound button

If set, the sound will be played, else mute.



Enhanced / Record button

See chapter Enhanced files.

Synchronize button

If set, the images will be synchronized with sound or with the file time settings. If not set, images are replayed at full speed.

Save As button

Opens a dialog where you can disassemble or convert a file to another format. This can be done directly from the General Menu entry: **Disassemble or Convert**.

Options button

Opens the option panel, see at Set Options and Paths.

Play ! Button

Replays the animation with your settings.

At the end of the replay, you get some information about the quality.

How many frames per second were displayed.

How many frames were actually displayed compared to the total. For sync needs, some frames may be skipped.

Hold CONTROL to stop animation prematurely. Hold Alt+Control to Pause. See Pausing.



If a sound is to be loaded, a little dialog "Loading sound..." appears. There you can read the hardware information for the sound. Typically, if you use the DMA, the frequency will be 12,5 or 25 or 50KHz. If you use the Yamaha or PSound card, it will be 9,6 KHz.

If you don't have the time to read the information, you can **hold the SHIFT key** while clicking on **Play !**, the dialog will remain until you release the key.

III.2 MPEG files

When loading an MPG file, the **number of frames is not available**. This is the default.

To get it, M_Player has to parse the whole file and that can **take a lot of time**.

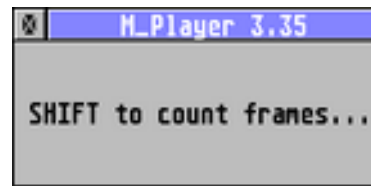
But if you really want this information, when the file selector is opened, click on you MPG file and then **hold the Left Shift key while validating** your choice.



At this moment, you'll see M_Player working for a while and finally display the complete information :



According to the power of your machine, M_PLAYER decides weather or not it will count the frames. If the animation is considered small enough to require only a few seconds, frames will be counted. Else, you'll get this box during one second :



If you hold Shift, then counting will be forced and during this operation, another box appears.

Take care ! For example, on a TT, one MB of animation will take approximately 1 second. So one minute for a 60MB video.

See at Set Options and Paths to force count.

Because **MPG replay is very slow** on a standard Atari machine, you can select :

Grey Levels button

If you select this, the MPG will only decode grey levels and ignore color data, this can speed up a bit the replay.

Dirt mode button

If set, then only a short part of the encoding is used to offer a reduced view of the animation in very low quality, but then again, more speed.

As an example, here are the times in seconds for the different modes for the above animation on a TT with graphic card:

FRANCE2.MPG	Color	Grey
Normal	173 sec	119,6 sec
Dirt mode	65,4 sec	46,6 sec

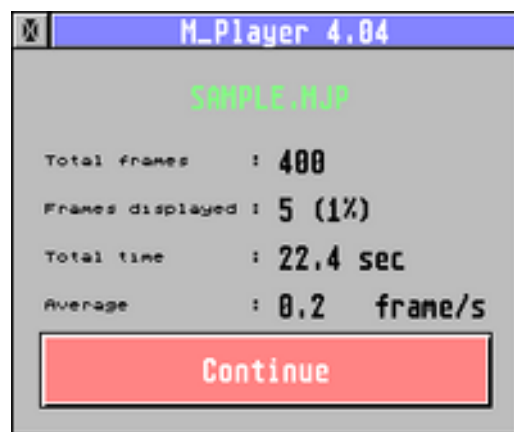
As you can see, **the sound is never supported in MPG files.**

III.3 MJPEG files/Codec in AVI-MOV

MJPEG (motion JPEG) frames can appear as:

- a CODEC into AVI or MOV files
- standalone files

and they all require the module **JPEG_68K.RIM** (written by Eric Da Cunha).



In the first case, nothing new, you can replay your AVI or MOV file as before. You'll just notice that on a real Atari, the frame decoding is slow, slower than the MPEG, those frames are time and power consuming.

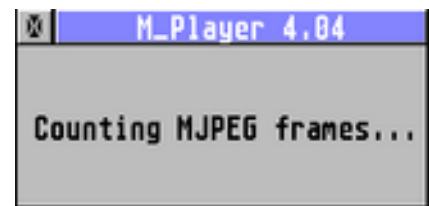
As an example, you can see the stats of a MJPEG file 640×360 on a TT that should be replayed at 20 fps.

In the second case, you have **one or more concatenated JPEG images**, typically with extension JPG for single images and MJP (short for mjpeg) for movies. They often come from cameras as there is no need to build a specific file structure, JPEG files are just written one after the other.

Two problems come there:

- the **MJPEG files don't have any timing** information
- they **don't have the number** of frames information

This is why you'll see the dialog "**Counting MJPEG frames...**" when you load one. This can take a while depending on the length of the file.



And this is why the **option panel** lets you specify a default frame rate. See at Set Options and Paths.



This value runs from 0,1 to 99,9.

Examples:

- *With 0,1 fps, every frame will last 10 seconds, a slideshow.*
- *With 10 fps, every frame will last 0,1 second.*
- *With 99,9 fps, every frame will last 0,01 second.*

Note: If you want to see **every image** with no synchro, then select "**Max Speed**" in the main dialog.

Note: if a **JPEG** file contains **one single image**, then it remains displayed on the screen until you press **Control** or one **mouse button**.

III.4 Speed up some slow animations

As mentioned before, **(M)JPEG**, **MPEG** and also **IV31/IV32** codecs are **really slow** on a standard Atari machine. There is a **hidden feature** that allows you to skip some frames at a given rate to maintain the motion.

When the video is reported to be : **mjpg, jpeg, mpg1, mpg2, mjpa, mjpgb, iv31, iv32** you can press **Control** while clicking on **Play** and you'll enter this little dialog box:

Click on **any number**, for example 3, and only one image out of three will be displayed.

If you select **Cancel**, then all images are processed as usual.



III.5 GIF files

As the MPG, you don't get the number of frames displayed unless the file contains the M_PLAYER extension. This happens only if the file was created by M_Player.

Two other extensions are supported :
DELAY : that sets the time between two frames
NETSCAPE : for a loop displayed

If a GIF has a loop extension, you can stop the display at the end of a loop using the Shift key.

III.6 Old Atari/PC animations files

➤ Some are mute

FLI, FLC, FLH

from Autodesk Animator (16, 256 or TC modes)

SEQ

from Cyberpaint (ST Low)

PI1+DLT

from Cyber, select the PI1 file, the DLT must be in the same folder with same name.
(ST Low)

FLM

From Kinetic Microsystems (ST Low)

DL

from the PC world. Support for DL1 and DL2 files is provided.

➤ Some have sound

FLM

from Lexicor. (ST Low, ST Med, ST High)

This format has been extended by M_PLAYER and MP_STE with sound. That's what I call Extended FLM. Both M_PLAYER and MP_STE can generate such files. This new format is adapted to ST Low and ST High video modes.

Sound has the DMA requirements :

- 8 bits mono/stereo
- frequencies 12,5kHz, 25kHz or 50kHz.

FLM

From Video Master, the digitizer for ST/Falcon. (160×100 in 16 or 256 colors)

Sound is always 8 bits mono with the above frequencies.

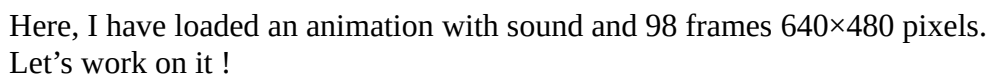
CDH, CDL, CDV

From the CD of the Stratos french magazine. Animations in mono, 16 colors or 256 colors to suit ST High, ST Low or Falcon 256 colors resolution. For the last, a TIFF image with the same name in the same folder is required for the palette.

Sound is 8 bits mono in 12,5kHz.



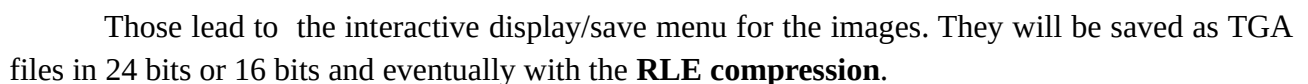
Note: *there was another way to do it before the General Menu was created. It is still supported: manually set the button "**Save As...**" when you are replaying videos. Then you select **Play!***



A video is a set of **images** and eventually a **sound**. All of this can be **taken apart** and saved separately.

Save Sound button

TGA 24b and TGA 16b buttons



If you want to modify the size of the images saved, you can fill **New L** and **New H** with your own values, else leave those fields blank. To keep the original ratio, just use :

Fill **New L** or **New H**, leaving the other empty. **Calc Ratio** will compute the blank value rounding the results to a 1, 4, 8 or 16 pixels boundary.

Interactive saving

With **TGA 24b** or **TGA 16b** button, unless you have filled the auto settings with *Set Options and Paths*, you're driven to a file selector. You have to give a generic name to your images and select the folder to save them. If you leave the filename blank, then the default **FRM00001.TGA** will be used.

If you want to change this name, just type the first letters and the digits and TGA extension will be automatically appended. For example, if you type VIDEO, then VIDEO001.TGA will be used as the first file name. Validate and you enter this screen :



For TGA 16/24 bits saving, the image must be decoded in those high color formats. If you're running on a **system with a maximum of 256 colors**, then the images are decoded internally with high color and just displayed with a **monochrome filter** so you can still use the feature and know what the current image is.



Save

Save current image and go to the next according to the value you've entered in

Options for conversion:
Use one image out of 2__ images.

Let's call this value the STEP.

All

Save all images in sequence STEP by STEP without waiting for a key.
Hold **Control** to return to interactive mode.

Quit

Leave this screen and return to the General Menu (or the file selector if you entered manually this mode).

Run

Display images in sequence without saving.
Hold **Control** to return to step by step mode.

Space

Go to next image.

IV.2 Converting a video

M_Player allows you to turn a video from one format to another **on the fly**, without saving tons of images.

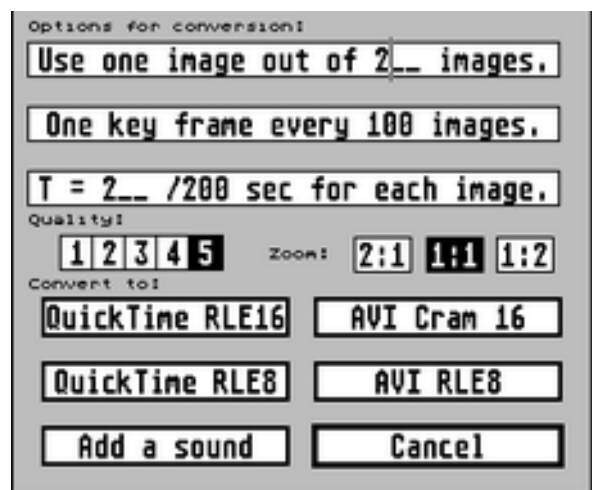
This feature has been rewritten to be accepted by multitasking environments, the memory is correctly shared.

M_PLAYER runs a second instance of itself, the first one disassembles the source file and the second one reassembles the destination file.

The **default** conversion is **without sound**. If you want to **keep the original audio track**, you have to save it first (*as explained above*). Notice that, when this is done, the button « Add a sound » automatically turns to selected.

Use one image out of 2__ images.

The default value is 1 to keep all the frames. But you can set 2, 3, or more to reduce the frame rate.



T = 2__ /200 sec for each image.

The time for each frame. Here 2/200 sec means 100 frames per second ! Too much. A value of 25 means 25/200 sec, this is 8 frames per second.

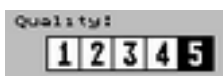
If you change the frame rate, you have to modify this timing accordingly.

If you use a **sound**, this value is **not used** as the frame rate will be computed to **fit the sound duration**.

One key frame every 1__ images.

The key frames are a complete rebuilt of the screen. Other frames are only a modification of the previous image, they are called delta frames (*they store the difference*).

If you want M_Player to be able to **skip frames** on **slow machines** to maintain the synchronization, you have to add key frames. One per second is not so bad. (*see Key-Frames use*).



The quality setting tells M_Player the threshold from which a pixel is considered as equal or changed compared to the previous image.

For 16 bits encoding :

Quality of 1 is the worst. A pixel has to notably change its color to be added to the new image.

Quality of 5 is the best. A pixel has to be exactly the same as before to be skipped, else it is added to the new image.

So you can see that, with a low quality, few pixels will be stored in the delta frames : this will lower the size of the video and speed up the display.

On the contrary, a high quality will lead to larger files and slower replay.

For 8 bits encoding :

Quality just tells to use color dithering (q=5) or grey scales (q=1).



You can keep the original size or zoom it by two (2:1) or reduce it by 2 (1:2).

When all parameters are ok, use one of these four exit buttons to start conversion :



QuickTime RLE 16 :

fast encoding and replay, TC colors but low compression that leads to large files.

AVI Cram 16 :

slow encoding, average replay, TC colors and better compression.

QuickTime/AVI RLE8 :

fast encoding and replay, 256 colors with dithering or greys, average compression.

You'll get a conversion box with bar progression and time estimation.



Note :

If the source animation has non standard width and height (odd values for example), this can lead to a bad display in the destination file.

If this happens, then save the frames as TGA with New H and New L rounded to 4 or 8, and reassemble the animation.

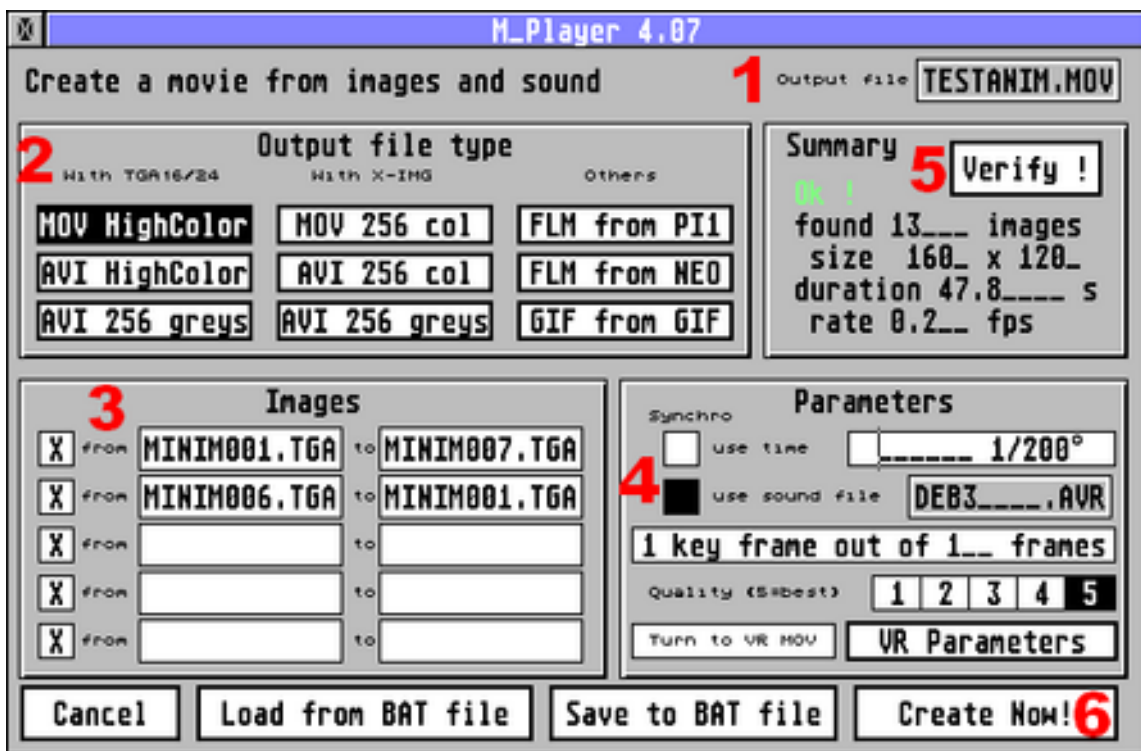


V Build a video from images/Slideshow

M_Player allows you to put together a **set of images** and eventually a **sound** into a unique video file. Several formats are supported : **QuickTime, Video for Windows, Gif and FLM**.

Assembling a video **requires a batch file**. It's a simple text file with the information needed to build an animation. You can create it by hand or use the old EASY_BAT.PRГ tool, but the easiest way is to use the **integrated panel**.

V.1 The integrated panel



Here is the dialog in which you'll **define your animation**. The steps are:

- (1) Define the **output name** by clicking here. *The default path points to the "CONVERT" folder that you specified in Set Options and Paths.*
- (2) Select the **output file type**. As I am using TGA files, I select MOV HighColor.
- (3) Define the **images**, here two groups to get a back and forth movement between MINI001 and MINI007. *The default path points to the "IMAGES" folder that you specified in Set Options and Paths.*
- (4) Set some **parameters**, here I don't specify a time for each image but I use a sound file. *The default path points to the "SOUNDS" folder that you specified in Set Options and Paths.* The frame rate will be computed according to the sound duration. Every image is a key-frame with the best quality.

(5) Click on **Verify!** to make sure that everything is consistent (images with same sizes matching the anim file type, etc). You can see in green **Ok!** and the specifications of the anim that will be generated. If an error is detected, you'll see **Errors** in red.

(6) Click on **Create now!**, the batch file is stored in memory and directly used by the program.

You can also **save the batch file to disk**, so you'll be able to reload it and modify some parameters to get a different video.

You can **modify the file by hand**, but **never edit** the lines **from DIAL to ENDD**. They contain every information to rebuild the panel as you left it.

Another example with Quick Time VR files is explained at QuickTime VR movies.

V.2 Limitations and links between image type and animation

- With **TGA 16b/24b** compressed or uncompressed, you can create :
 - ✓ a Quick Time RLE16 movie (MOV), High Color
 - ✓ a Video for Windows CRAM16 movie (AVI), High Color
 - ✓ a Video for Windows CRAM8 movie (AVI), 256 grey levels.
- With **GEM XIMG 256** colors images, you can create :
 - ✓ a Quick Time RLE8 movie (MOV), 256 colors
 - ✓ a Video for Windows RLE8 movie (AVI), 256 colors
 - ✓ a Video for Windows CRAM8 movie (AVI), 256 grey levels.
- With **GIF** images you can create a GIF movie.
- With **Degas PI1 or Neochrome NEO** images, you can create an Extended FLM movie.

V.3 Structure of a batch file, create it by hand.

You can get **more precise and powerful features** programming your own batch file with a simple text editor.

A batch file is an ASCII text files that contains 3 parts:

- the **identifier** with an optional DIAL/ENDD section
- the **header**
- the **datas**

The batch file can be used to display a slideshow of images. The only difference between slideshow and creation is the presence of a line **o=<file name>** that specifies the output file name.

Load this batch file into M_Player and use « **PLAY!** » to run the slideshow. You have to enable « **Save As...** » to initiate the creation if available (if **o=<file name>** is present).

Only the very first line must start on the leftmost row, for the others, spaces and tab to indent are allowed, there must be only one command per line, lines starting with ';' are considered as comments.

The size of the file is limited to 12488 bytes (I think it's enough...).

The identifier

It must be on the first line, first character and is always:

M_PLAYER or **m_player** (or **m_PLAYER** if you prefer!)

Just following the identifier, the DIAL/ENDD section can be present if the file was saved from the integrated panel. Never edit those lines, or remove them all...

The header

Starts on the second line (*or after some comment lines*) and gives general infos on the slideshow:

w=xxxx	the width of the movie
h=xxxx	the height of the movie
c=xxxx	the compression (the type of file): tga2 for TGA 2 uncompressed (24 or 16 bits) tgac for TGA 10 RLE encoded (24 or 16 bits) ximg for IMG with 'XIMG' extension and 256 colors gif8 for either gif87a or gif89a dega for Degas 320x200x16 or 640x400xmono neoc for Neochrome 320x200x16
b=xxxx	size of the buffer to load the entire largest image file

Those first 4 infos can't be omitted! The following can (for a slide show, but must appear for a MOVIE-creation):

t=xxxx	time for one frame (number of 1/200sec) With a GIF it is the time to wait between two frames (without the display time), with 0, no delay block will be added.
f=xxxx	number of frames in the whole slideshow
o=fpath	path and name of the file to create (MOV, RLE16, RLE8 or GIF, or FLM)
s=fpath	path and name of the sound file (WAV or AVR, 8/16 bits, mono/stereo). For MOV/AVI the frequencies are 11025, 22050, 44100. For FLM 12517, 25033, 50066 +/-2% and only 8 bits.

If 's=' is present, then 't=' is ignored and the frame rate is computed according to the sound duration.

q=x	quality, default value is 5 (5 bits per color), but you can reduce it to 4,3,2 or even 1 to get a more efficient compression, but with a lost of quality. This parameter is used only in AVI and MOV files in 16 bits compression. For 8 bits compression, a quality of 1 means grey levels !
k=xxxx	specifies the rate of the key frames (for example, with k=5, frames #5, 10, 15, 20 ... will be key frames). See below: "Key-Frames use". This parameter is used only in AVI and MOV files.

r=xxxx for GIF files only, fixes the number of loops for that movie, the range is 0 to 65535. This will add the NETSCAPE extension as follows:
 \$21, \$ff, \$0b, 'NETSCAPE', '2.0', \$03, \$01, Intel word=xxxx, \$00.
 A value of 0 indicated an infinite loop.

v=max_x,max_y,start_x,start_y,img_cell,loop_x
 for MOV files only, this allows you to generate a QuickTime VR file (an interactive movie).
max_x=number of columns (start at 1)
max_y=number of lines (start at 1)
start_x=horiz. index of the first cell (0 to max_x-1)
start_y=verti. index of the first cell (0 to max_y-1)
img_cell=number of images per cell
loop_x=1 for horizontal looping, 0 to stop at boundaries
 See 'QuickTime VR movies' below for more details.

m=xxxx if you call M_Player as a movie maker (MOV,AVI,FLM,GIF) from your own program and you don't want to fill your hard drive with TGAs, you can send to M_Player the images one by one using a kind of dialog through this stucture, see The "Mountain" mode.

The order of those info is not fixed, but you mustn't put spaces between the characters :

w=320 is ok
 w = 320 is wrong

For MOV and AVI creation, the lines « o= » and « q= » are used to specify the exact type of file you want.

Suppose we want to create a file named MYVIDEO with quality 4 **from TGA files** :

Output file type	Output file name	Quality
MOV RLE16	o=myvideo.mov	q=4
AVI CRAM16	o+=myvideo.avi	q=4
AVI CRAM8/grey	o+=myvideo.avi	q=-4

Suppose we want to create a file named MYVIDEO **from XIMG files** :

Output file type	Output filename
MOV RLE8	o=myvideo.mov
AVI RLE8	o=#myvideo.avi
AVI CRAM8/grey	o+=myvideo.avi

The data

Start immediately after the header with a key word:

data

Then, every line is read and:

if starting with a '.' then it's a command, executes it

else it's the name of a file, displays it and its name becomes the current name.

commands are:

.rept xxxx

.endr

repeat xxxx times what's between rept and endr. Loops can be nested.

.disp

display the file corresponding to the current name

.getp

use the palette of the next file (only applied with palette based images). If '.getp' is not specified, then the palette will be ignored (for example when all of the frames share the same palette that is loaded once), this speeds up the display when running in NOVA 256c (because the palette must be passed to the VDI).

* not used with GIF movies *

.incr

increments a number found into the current name

Example, if name was: C:\IMAGES\FRM00001.TGA

it becomes C:\IMAGES\FMR00002.TGA.

.decr

same as above with a decrementation.

.stop

end of the slideshow. This command MUST appear, else the program crashes.

.keyf

forces the next frame to be a key-frame (if k=xxxx is used, then this auto key counter will be reset).

See below at "Key-Frames use".

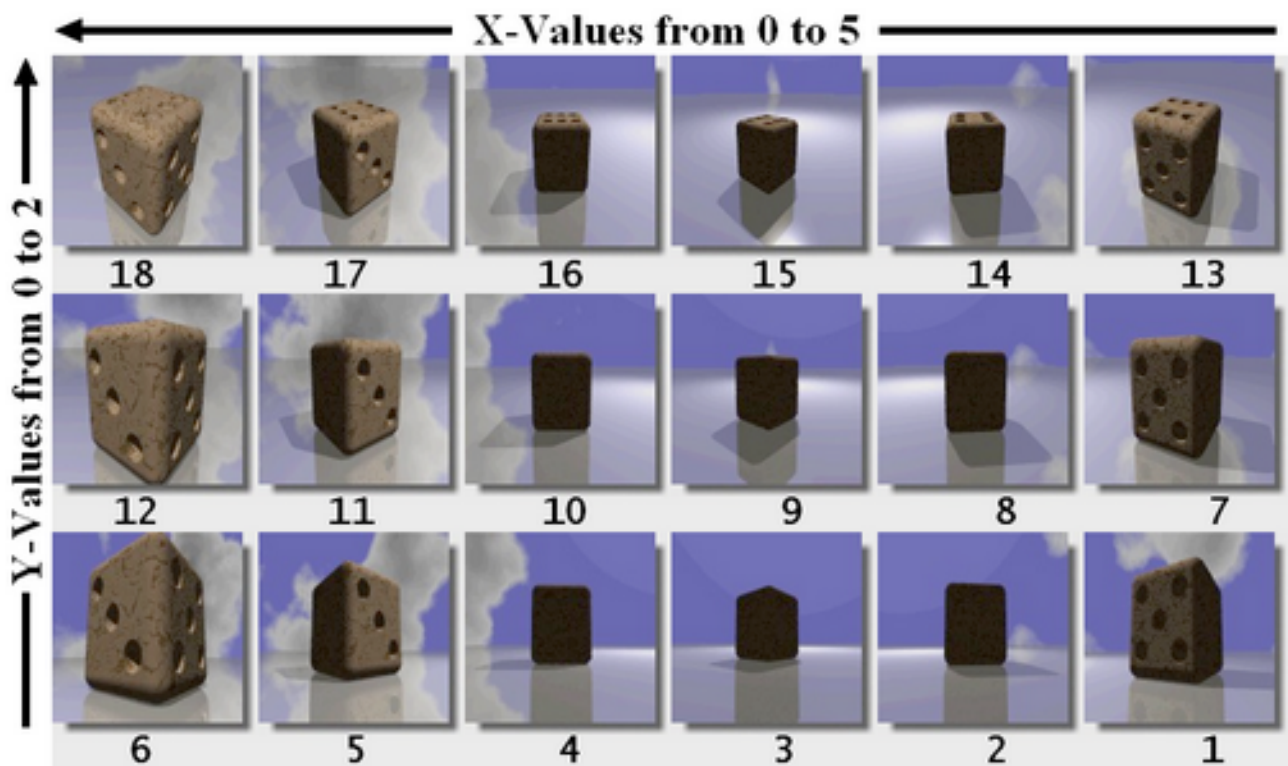
* not used with GIF,FLM movies *

V.4 QuickTime VR movies

VR stands for **Virtual Reality**. Those movies are **interactive** and allow the user to navigate within a 2D-universe by changing the point of view of a camera horizontally or vertically or both. The navigation is available with the arrow keys or with the mouse.

Let's look at an example, I have 18 images of a dice. They are divided in three groups corresponding to the elevation of the eye :

- one group of six images viewed from the ground. (images #1 to #6)
- one group of six images viewed at mid-height (images #7 to #12)
- one group of six images viewed on top of the dice (images #13 to #18).



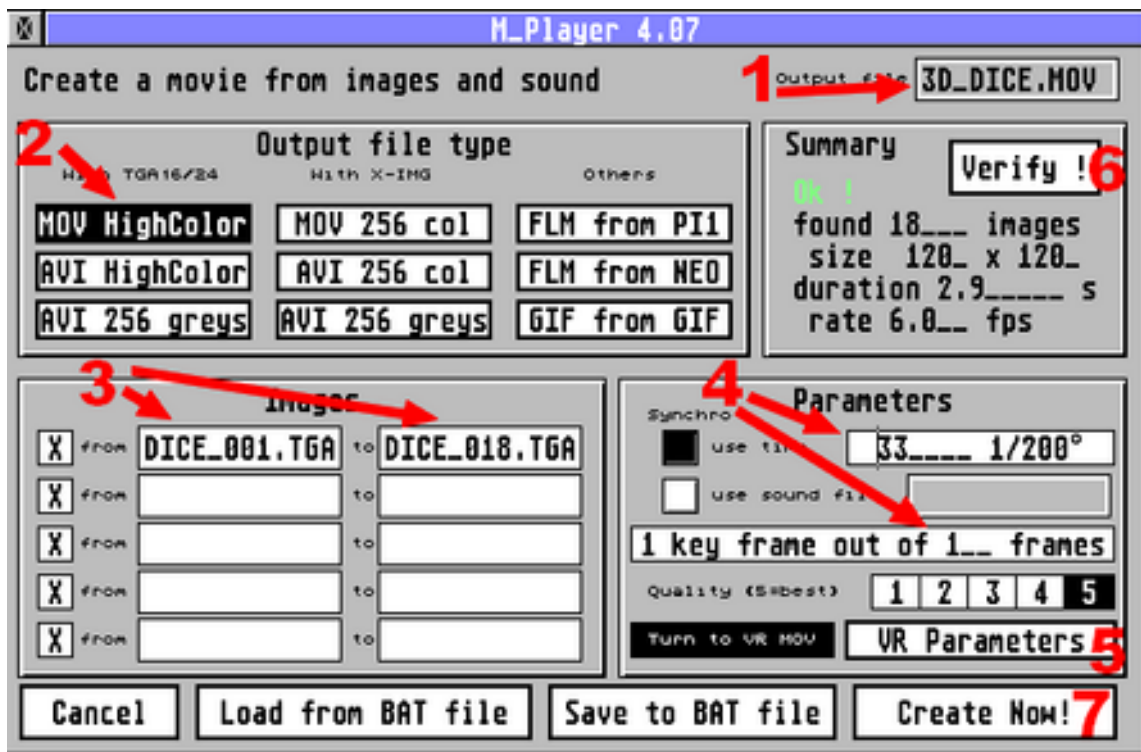
Fixed cells

When the movie starts, M_PLAYER displays the initial image, let's say that this one is at coordinates (X=3 ; Y=1) : this is image #10. Then, using the arrow keys :

- ➔ UP will display image #16
- ➔ DOWN will display image #4
- ➔ RIGHT will display image #9
- ➔ LEFT will display image #11

If the **horizontal loop** is set, then you can move from image #7 to image #12 for example.

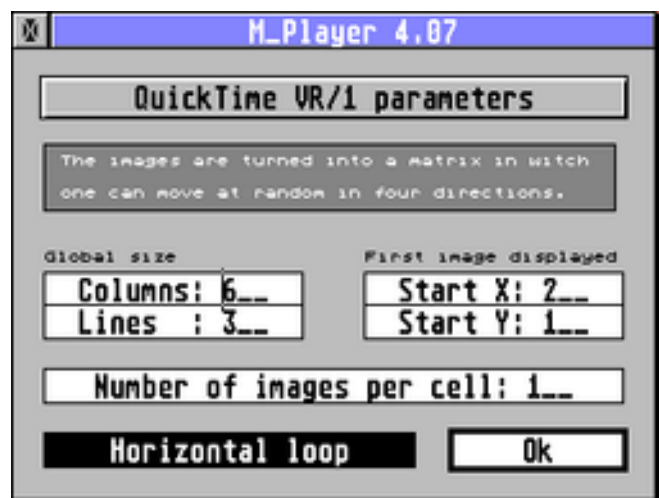
These are the **settings** you have to make in the dialog:



Don't forget to click on **VR Parameters** and fill in the following information:

With a value of **one image per cell**, this VR will display fixed images that you'll animate with the mouse or the keyboard.

Then, back on the main dialog, click on **Verify !** to ensure that everything is correctly set.



You can save the BATCH file for future modifications.

Finally, click on **Create now !** to prepare the video encoding.

The dialog on the left summarizes your work, an internal BATCH file has been created (memory.bat) and will be used to generate 3D_DICE.MOV.

Click on **Create !** and the work starts.

A progression bar appears and tells you how far you are.



A stats box appears at the end.

Click on **Continue** and you go back to the General Menu where you'll click on "**Load and Play**" to load your 3D_DICE.MOV new animation.



A new dialog appears, specific for VR files:

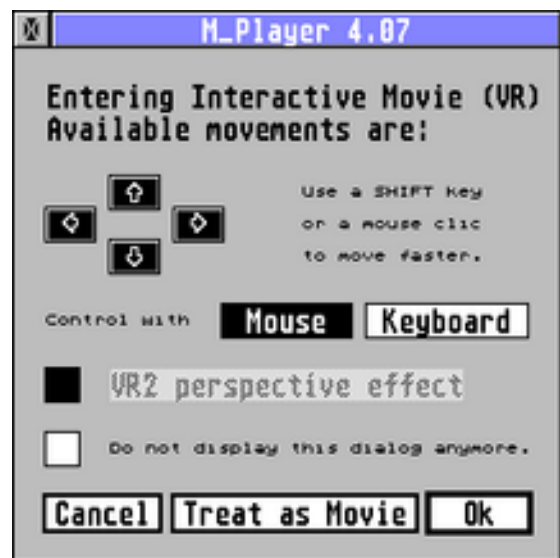
You can select the **control method**, either using the **mouse** or the **keyboard**.

You can treat the file as a standard movie (the 18 images will be displayed in sequence), but if you click on **OK**, you enter the VR mode.



As you can notice, it is detected as a Quick Time VR animation.

Click on **PLAY!**.



Now you can observe your dice up and down, and all around!

If the speed of 6 frames/sec is not enough, just **press SHIFT** while you are using the mouse or keyboard and the frames will be displayed **with no delay**.

The corresponding BAT file to create the Virtual Reality movie is :

```
M_PLAYER          ** identifier
c=tgac            ** TGA compressed images
w=120             ** width
h=120             ** height
b=18944           ** buffer for max image
o=K:\VR\3D_DICE.MOV ** output file
q=5               ** best quality
k=1               ** all are key frames
t=33              ** 6 image / sec for one turn around
f=18              ** 18 frames
v=6,3,2,1,1,1    ** 6 columns, 3 lines, start in (2;1)
                  ** 1 image per cell and loop allowed
data              ** start data section
K:\VR\DICE_001.TGA ** first image
.rept 17
.incr              ** add the 17 following
.disp
.endr
.stop             ** end.
```

Important note :

Every frame must be a key frame as the order of the display is not predictable. See at Key-Frames use.

See EASY_BAT.PRГ usage for a complete description of the BAT creation.

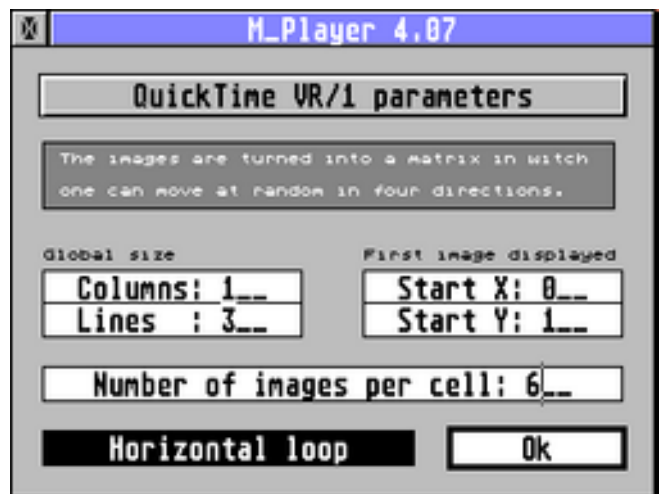
Animated cells

One **cell** can contain an animation instead of a fixed image.

For example, see the images above, you can imagine only three vertical cells and each one contains six views of the dice that are automatically animated. So you can just move up and down and the horizontal movement is automatic.

Just modify the VR Parameters as shown on the right. Or, if you work with a text editor, just change one line :

```
v=1,3,0,1,6,0    ** 1 column, 3 lines, start in (0;1)
                  ** 6 images per cell no loop allowed
```



Panoramas

These are the second form of VR files where things are inverted. You are not looking at an object around it, but you are inside a place and you can rotate around this center point to explore all the place.

M_PLAYER can display such images but has not the ability to render the correct proportions of what is seen.

V.5 GIF movies

A **GIF movie** can only be created **from GIF images**, those images must share the same global size and the same global palette (if a local palette is present, it will be stored).

Every other bloc than \$2C (the images) is skipped (*for example copyrights, infos, etc...*). Only one bloc is always added, the one defined by myself that indicates the number of frames:

\$21, \$ff, \$0b, 'M_PLAYER', 'FRM', \$04, Intel Long Frames, \$00

This bloc must be located just after the local palette (*and, if present, after the Netscape extension concerning the looping*).

The **delay time** between two frames is stored into an unsigned word, so it is limited to 65535, this means no more than 655.35 seconds (*and so 131070 for 't=' as it is a number of 1/200 of second*). If the value is greater than this limit, 0 will be used instead (meaning no delay!).

M_Player can be **used to modify an existing GIF-Animation** (*for example to add the M_PLAYER extension, the Netscape extension or to modify the delays between two frames,...*).

Example :

I have an animation NUKE01.GIF that has no M_Player extension, with a infinite loop and that is played too fast, I run it once to get the info:

size 64x64, 9 frames displayed (in the stats box at the end. When an anim has a loop, SHIFT stops it!).

Then I create the following batch file:

```
m_pLaYeR
w=64                the size I noted
h=64
c=gif8
b=16000             the file is only 15xxx bytes long
t=11                11/200 = 0,055s of delay
f=9                 9 frames into my initial anim
r=3                 will repeat only 3 times (instead of infinite)
o=C:\TGA\nuke.GIF  the output file
data
C:\TGA\nuke01.GIF  only one file!
.stop
```

Upon exit, I'll have a **new anim** NUKE.GIF with the **M_PLAYER extension**, with different delays and with 3 loops only. The output file is shorter, because M_Player has skipped every info bloc!

V.6 FLM Movies

As they are created from ST-Low images (c=dega or c=neoc), you can omit 'w=', 'h=' and 'b=', they will be filled with the following default values:

w=320
h=200
b=33000 (PI1 32034 or 32066, NEO 32128)

A FLM file **hasn't the 'key frame' capability**, and so, be sure that the frame rate you specify will be possible on your machine. Else, as no frame is skipped, the images could last longer than the sound.

They are designed for the Atari Video system. The encoding of the images is exactly the same as the screen. So, the decoding is done directly in video RAM with a high speed even on a simple Atari ST. I extended this format with sound for the Atari ST/STE, it is largely used in MP_STE.PRG, the video player for Atari 68000.

V.7 Example of a BATCH FILE

Example of slideshow :

(The comments starting with ** don't appear in the file!)

```
** start of file
M_Player                                ** Id
; comes from a FLI                      ** comment
w=320                                   ** width
h=200                                   ** height
; 67/200 means 3 f/s                   ** comment
t=67                                    ** time for one frame
f=29                                    ** number of frames
c=tga2                                 ** files are TGA 2 uncompressed
b=192018                                ** size of one file (320x200x3 + 18)
data                                    ** end of header/start of data
E:\imprimer\mou00001.tga               ** first frame, display it
.rept 2                                 ** twice the whole
    .rept 7                             ** 7 times
        .incr                           ** increment name of file (00002 to 00008)
        .disp                           ** and display it
    .endr                               ** loop
    .rept 7                             ** another 7 times
        .decr                           ** dec name of file (00007 to 00001)
        .disp                           ** and display it
    .endr                               ** loop
.endr                                   ** twice!
.stop                                  ** end of slideshow
```

Well, this example has displayed:

mou00001.tga
mou00002.tga

...

mou00007.tga
mou00008.tga
mou00007.tga

...

mou00001.tga

And this twice! The images came from a FLI that I saved using the Step by Step mode.

INCR and DECR commands can only work correctly if the names of the files END with digits! (for example 00001BB.TGA won't work because it ends with BB).

Example of creation :

You can specify each file if you don't want to use commands:

```
M_PLAYER
w=184
h=240
b=50000
c=ximg          ** t= omitted because there's a sound
f=5
o=#C:\ANIM\US.AVI    ** will create an AVI ('#')
s=C:\SOUNDS\KISS.AVR ** will add this sound
data
.getp           ** supposing that all have the same
C:\IMAGES\I.IMG  ** palette, we load it once!
C:\IMAGES\YOU.IMG
C:\IMAGES\I.IMG
C:\IMAGES\YOU.IMG
C:\IMAGES\YOU_N_I.IMG
.stop
```

'stop' is the only command that must appear.

Common errors (those I made...)

- x The header of the output file is built BEFORE adding the frames, so the **number of frames** that you specify (f=xxxx) **must be exact!** Else you'll get a totally corrupted file. To ensure that it's the right value, launch a slide-show before encoding, if total_frames=frames_displayed, it's Ok.
- x This is the same for 'w=' and 'h='. Be sure that **all of the images have the same size**, the one you specified.
- x The 'supported' message into the graphic box is based upon your 'c=xxxx'. The program hasn't read a file to verify that, ensure that it's the good format.
- x The **sound frequencies** accepted are those from the PC world: **11025, 22050 and 44100** KHz, or the closest possible to them. Else, you get an 'Unsupported' message in the sound box.
- x If using IMG images you get the RLE8 compression that uses a palette. But, the output file can contain only ONE palette, so you must be sure that **all the images share the same colors**, else, it's the last palette read (with .getp) that will be saved.

V.8 Key-Frames use

Here is a brief introduction to the relevance of **inserting key-frames** into a movie. There are two cases:

- your **machine is fast** and no frame will be skipped when playing, then you don't need to use the key frames.
- your machine **isn't so fast** and there may be some frames skipped, the use of key-frames is highly recommended!

What happens when a frame is skipped?

In order to **increase the compression**, only the **differences between two frames** are stored, this means that one frame is only a **partial display**. If one frame is skipped (*because of the synchronization*) the next one won't have the expected background! You'll get a wrong display.

A **key frame** is used to solve this problem, it is a complete rebuilt of the screen. So, for synchronization needs, if M_PLAYER has to skip frames, it jumps to the next key-frame to go on displaying.

An example:

Suppose you have used $k=5$ and $t=40$ (*5 frames per second*) into the header and that your machine can only display 3 images /second. Here is what will happen:

second #1: frames 0,1,2
second #2: frames 5,6,7
second #3: frames 10,11,12
etc...

Every key frame will be at its exact position (*in time*), and between 2 key-frames, the player does as much as it can. If you play this same movie with a faster machine, you'll get maybe:

second #1: frames 0,1,2,3
second #2: frames 5,6,7,8
second #3: frames 10,11,12,13
etc...

And with a very fast machine, you'll get every frame.

The best is to put 1 or 2 key-frames per second, this way, the display will be synchro with the sound once or twice a second.

You must know that a key frame **is larger** than a normal one, because it rebuilds every pixel, and that inserting too few key-frames will lead to a poor synchro quality. It's a **question of balance**. Make some tests to find the best solution.

Idea: *it's a good idea to choose the frames where the background changes suddenly, because anyway most of the pixels will be rebuilt!*



VI Albums

With this feature you can **group up to nine videos** into one album with a **short description** for each. For example, this can be a **thematic album**: songs, cartoons, family, etc.

Or a group corresponding to a **physical support** such as a CD-ROM. This way, you don't have to browse the disk looking for hidden places with animations, they will be easily reached.

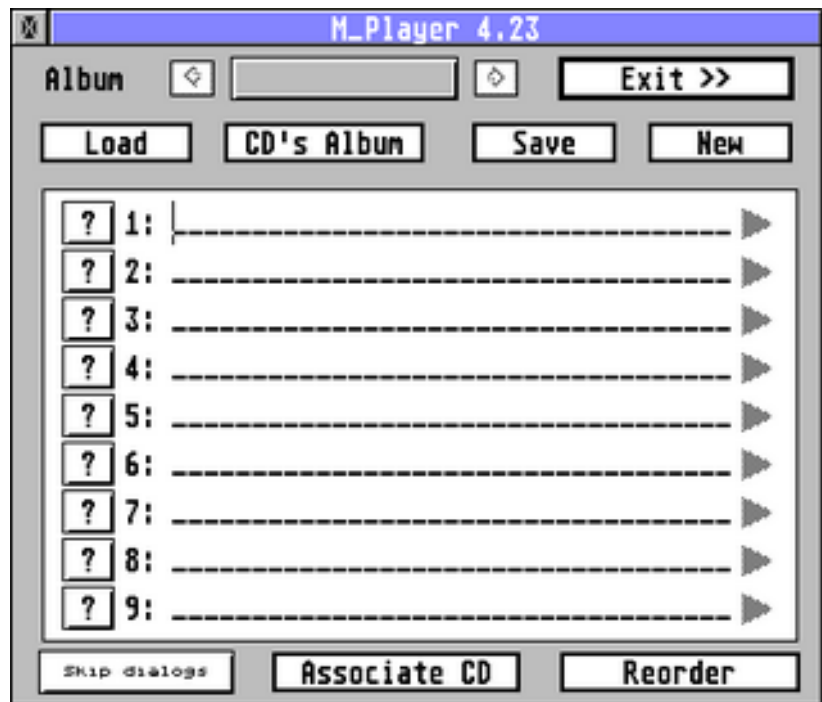
VI.1 Create an album by hand

If it's the first time you enter Albums, an empty dialog appears, else click on "New" to get a brand new page.

To define a video entry, you must click on one of the nine "?" buttons.

This opens a file selector and you can define your video.

If you validate this, then the **"▶"** button is enabled and the cursor is placed on the description line, so you can put a short comment.



Notice that that "?" is changed into **"☑"** when a video exists.

Load, Save

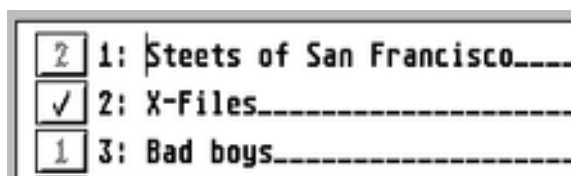
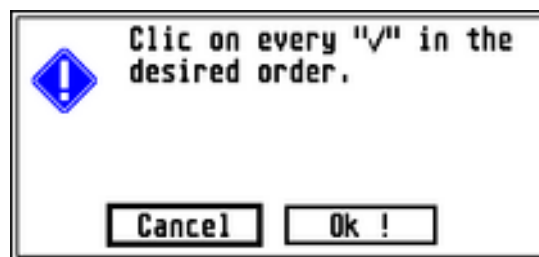
allows you to load and save your album. In this case its name appears in the upper box.

Reorder

If you have more than one video in your list, you can reorder then. If not, you'll hear a bell sound meaning reordering is not available.

An alert box tells you how to reorder.

Suppose I want the previous list to appear in alphabetical order. Then I have to click first on "☑" for "Bad boys", notice that it turns to "1" and becomes disabled. So you can't click on it anymore.



Then on "☑" for "Streets of San Francisco", it turns to "2" and becomes disabled.

The last one will be X-Files. The list automatically appears in the order you wanted.

Don't forget to save the Album if you wish to keep it reordered.

VI.2 Create an album automatically

You can use the Search a disk for animations facility to create albums. If more than nine files are found, then more than one album are generated and they are automatically chained the one to the other.

That's the goal of the two arrow buttons you can see around the Album's name.

In the example on the right, the file TEST.000 is chained to another file, click on the right arrow to display the content of TEST.001.



VI.3 Linking an album to a CD-Rom

Associate CD / Dissociate CD

Whether you created it by hand or automatically, if your set of videos come from a CD-Rom, you can mark this album with the CD signature using "Associate CD". M_PLAYER will automatically find the CD drive unit, and take some information to sign your album.

Note that when an album is associated with a CD disk, a black spot appears in front of its name:



If you want to remove this link, just use "Dissociate CD".

CD's Album

That's the way to use the link ! Just insert the CD in your drive and click on CD's Album. M_PLAYER opens a file selector, just go to the album folder and validate. No file name is required as the program will find automatically the album that was associated to this CD.

VI.4 Using an album

► (play)

Plays the video. The default setting is to show the dialog boxes. The main one before playing with the CODECS information and the statistics at the end. After a replay, the program goes back to the album. To exit to the General Menu, use "Exit >>".

Skip Dialog

If you click on this, then the replay will be done without any dialog box.

This preference is saved inside the album file.



VII Capture the screen to an animation

You can use M_PLAYER to grab the screen of another application (either as an accessory under TOS or as an application under Multitasking) to build an animation on the fly.

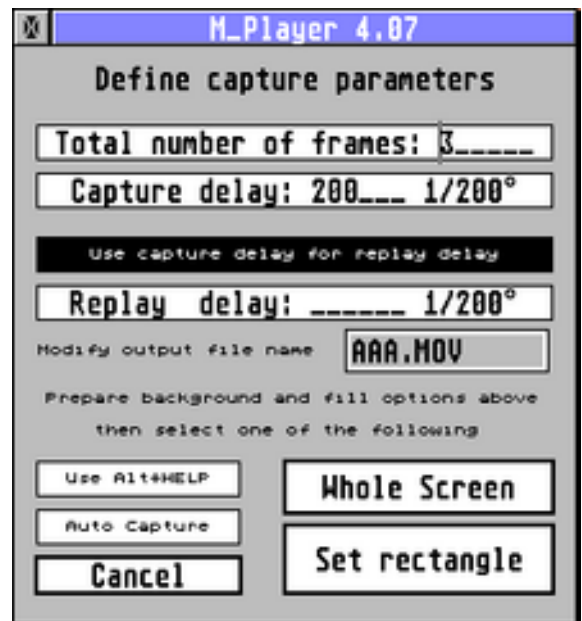
The type of animation depends on the current resolution:

- ◆ if your screen is in **high-color or true color**, you'll get a **16 bits compression** (MOV RLE16 or AVI CRAM16)
- ◆ if your screen is in **256 colors** or less, you'll get a **palette based compression** (MOV RLE8 or AVI RLE8)

The dialog appears to let you set your parameters.

Be sure to **fill the upper part before** using "Whole Screen" or "Set rectangle".

- ◆ The **number of frames** will be the number of screen captures that will be added to the animation.
- ◆ The **capture delay** is the time in 1/200°sec between each capture when you use **Auto Mode**, else it is ignored.
- ◆ The **replay delay** is the time in 1/200°sec that defines the frame rate of the animation.
- ◆ The **output file name** is the name of the created movie.



VII.1 The manual mode

There are two ways to grab the screen:

- a hot key **Alt+Help** (Use Alt+HELP enabled)
- a **dialog** (Use Alt+HELP disabled)

The **Auto Capture** is discussed below.

Alt+Help method should be reserved for applications that don't give access to the menu and use the whole screen, in this case you can't reach your button to capture.

Another problem: the *Alt+Help* hot key is not managed anymore on newer systems. It runs under TOS, again with Mint 1.12 but **has been abandoned somewhere between 1.12 and 1.19.**

Once you have selected the method click on:

- ✓ **Whole Screen**, the capture will be performed on the whole screen including the menu.
- ✓ **Set Rectangle**, the cursor turns to a cross and you can click and drag the rectangle to define your zone of capture.

When the rectangle is defined, you get the standard information panel with the chosen dimensions and the number of frames.

Note that **Save as...** is forced to validate the creation of an animation.

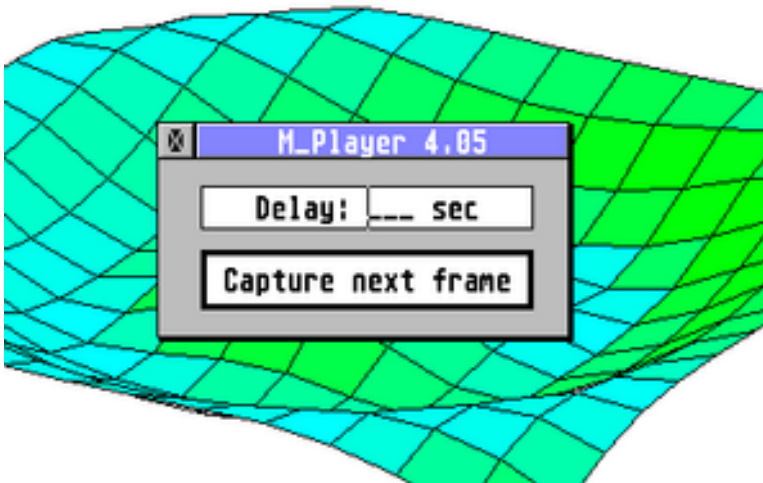
You can check the dimensions of your rectangle. *Important: the width is aligned on a 16 pixels boundary and that the height is aligned on a 4 pixels boundary.*

Just click on **Capture!** to start capturing.



→ If you are **using the Alt+HELP** method, a bell sound indicates when M_PLAYER is ready to grab the screen. Use your hot key when you are ready, and wait for the next bell. And so on.

→ If you are **not using the Alt+HELP** method, then a little dialog appears to capture the screen. It includes a delay parameter, so if you think that the application will require 10 seconds before displaying the screen you want to capture, then set Delay = 10 seconds and click on **Capture next frame**. And so on. You can specify a new delay for each frame.



When all the planned frames are grabbed, then you get the final statistics dialog:

So you can load your animation from where it was saved and see your work!

With a simple TT with no graphic card, the animation is correctly saved but remember that it is replayed using the TT Low resolution (320x480x256). Then if your animation exceeds the 320 pixels, it will be displayed as an approximation.

But as soon as you get a better display, the images will appear fully as they were captured.

When the screen is captured you can see a shrinking box appear. The time between this little effect and the next bell sound is the time taken by M_PLAYER to compress this screen and add it to the animation.



VII.2 The auto capture mode

In the dialog "**Define capture rectangle**", you can enable "**Auto Capture**". This doesn't change the way you start your capture: either with Alt+Help or with the dialog using an optional delay.

But then, **as soon as the first frame is captured**, M_PLAYER enters in **automatic mode** and captures periodically the rectangle until the last frame.

Be sure to set a **Capture Delay** that **matches** your **machine power**. If it is too short, then M_PLAYER will still be encoding a frame when the new one is grabbed. This can lead to a total mess in the system...

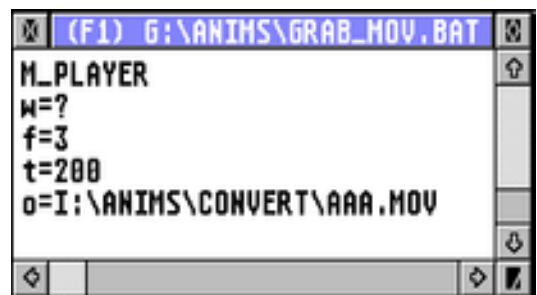
VII.3 Capturing using a batch file

Again, a BATCH file can be used to capture the screen! That's what is done internally when you use the standard dialog.

Anyway, you have to prepare a batch file. In this case, very few parameters are required:

- The identifier
- The number of frames (f=)
- The time for one image in 1/200 (t=)
- The output file name (o=)

The key parameter to run a screen capture based animation is "**w=?**" meaning that you'll define the rectangle at run time with the mouse.



As soon as you have loaded this BAT file, you'll be driven to the previous dialog but the parameters will be filled for you according to what is found in the file.

If your BAT file contains the line:

```
t=val
```

Then the **same timing** is used for both **capturing and replaying** the animation. But you can separate those two behaviors with:

```
t=val1, val2
```

The **first** value will be the **replay** timing and the **second** the **capture** timing.

Example:

```
t=80, 400
```

The screen will be captured every $400/200$ sec = every two seconds, but the animation will be replayed with $80/200 = 0,4$ sec, this is 2,5 frames per second.

VIII Enhanced files

You can enhance the display by using a ***.ENH file** containing **subtitles or comments** and that will be displayed **under your video**.

The file must be located in the **same folder**, with the **same name** and **ENH extension**.

For example, to enhance MY_VIDEO.AVI, use MY_VIDEO.ENH.

If such a file is detected, then it is **loaded and compiled**. If everything is okay, then the button "Enhanced" appears selected into the main dialog box. You can uncheck it if you don't want the comments. See Enh button.



If the button appears as Record, this means that no ENH file was found or that it contains errors (*in this case, an alert box should tell you what's wrong with it*) and that **you can record one ENH file**. See at Recording an Enhanced file

If the button appears **disabled**, then no window mode is available (*for example when the computer needs to change the resolution to display the video*) and you can't use the enhanced features.

VIII.1 The header

The ENH file is a plain text file with one command on each line. The header contains one to three lines:

ENHANCED	-> the very first word of the very first line.
l=(value)	-> <u>number of text lines</u> under the video, if omitted, then l=1.
s=(value)	-> <u>font size</u> from 0 (small) to 3 (large), if omitted, then s=2.
c=(value)	-> <u>color index</u> from 0 to max colors (1, 15, 255), default is 1. Set a negative value to use this index only in 256 colors mode with palette change.

VIII.2 The different versions

You can build a **multi language file** that will be automatically adapted to the current system language. If no version matches the system, then the very first version is used. So it is recommended to put something that most people will be able to read.

Each version start with:

v=(value)	-> the value is the <u>language code</u> (0=English, 1=German, 2=French...)
------------------	---

If two sections share the same **v=n** command, then the last one is used. This is useful to test your file, suppose your system is french and that your french version is the first one. Then you can test every other version by writing v=2 before it.

If you're not sure of your **country code system**, then have a look at the **credits dialog** box, the code appears at the end of the line in small characters:



VIII.3 The events during replay

They are a mix of time markers and text functions.

A time marker is a simple number in 1/100 sec. The time markers must appear in chronological order as the file is read sequentially.

840 -> what follows will be performed after 8,40 seconds.

d=(text) -> the text til the end of the line is displayed. If you selected more than one line, then you can use more than one d=(text) command. At the next time marker, the display starts again at the first line.

cl -> clear the text zone

be -> output a bell sound

a=(code) -> set horizontal text alignment. Available codes are:

a=l for left a=c for center a=r for right

e=(codes) -> set text effects. You can combine several codes, available are:

n = normal b = bold i = italic

u = underlined o = outlined l = light

s = shadowed (not always available...)

pa -> enter pause mode. See Pausing.

VIII.4 An example

Here is an example with comments. The comments can appear in the file itself as after every command, the remaining characters of the line are ignored.

An exception is the **d=** command that displays everything til the end of line, so don't put comments on them!

ENHANCED	<i>the magic word</i>
l=2	<i>two lines</i>
s=2	<i>font size = 2</i>
c=1	<i>text color = 1 (black)</i>
v=0	English version
100	<i>1st event at 1,00 sec</i>
a=c	<i>center</i>
d>Hello !	<i>display those two lines</i>
d=This is a second line.	

200	<i>then at 2,00 sec...</i>
c1	<i>... clear the messages!</i>
350	<i>then at 3,50 sec...</i>
d=How are you?	<i>... display this line</i>
e=i	<i>change the effects to italic</i>
d=Fine thanks!	<i>and display this second line</i>
500	<i>finally at 5,00 sec...</i>
c1	<i>... clear the messages. No more events.</i>
v=2	<i>French</i> version starts here.
100	<i>all the same with french text.</i>
a=c	
d=Salut !	
d=Voici la 2e ligne.	
200	
c1	
350	
d=Comment allez-vous?	
e=i	
d=Bien merci!	
500	
c1	

VIII.5 Pausing

You can enter the **pause mode** either with the "**pa**" command of an Enhanced file or using the **Alt+Control** keys with any other animation (*be sure to press Alt first, else you quit the replay!*).

If the display is done **without window** ("**W**" button unselected or after a resolution switch or full screen required), the only thing you can do is resume the replay by **pressing a key**.

If a **window is present**, then you can **interact with the desktop**. You can move the window, select menu entries, work on other applications. To **resume the replay**, just click on the "Closer" button of the window.



In **256 colors modes**, the colors must be swapped between desktop settings and the animation requirements. On newer systems (as AES 4.00) M_PLAYER is notified if its window is topped or untopped, so it can swap the palettes. On older systems, you'll notice that the "Fuller" button appears and allows you to swap manually the palettes if they don't...



If you use the TT Low resolution (256 colors, 320x480), then moving the window is automatically forced on a 16 pixels boundary because of the planar encoding.

VIII.6 Recording an Enhanced file

The Record button allows you to **create automatically your ENH file** without having to write it by hand. Just after clicking on the Play ! button, this dialog appears and lets you define your general options.

Don't forget that if you're not satisfied with this, you can then edit the text file to meet your needs.

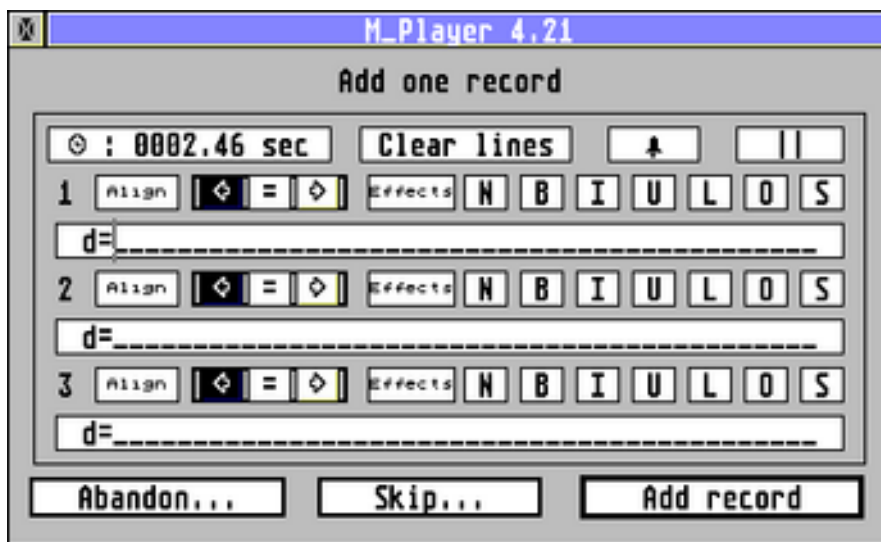
For example, you may want more than three lines.

Or you'll want to add another language by a copy paste operation.



Click on Ok ! and the replay starts.

Use **Alt+Control** to enter pause mode. This new dialog appears.



On the very first line, the time appears and you can select to:

- ◆ clear lines (**cl**)
- ◆ add a bell sound (**be**)
- ◆ add a pause (**pa**)

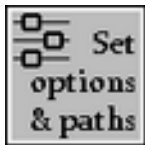
Then, **three lines of text can be edited**.

- If you want to change the **alignment**, select Align and left, center or right.
- If you want to change the **text effects**, select Effects and some of the seven options.
- Fill the **text line** If a new line covers the previous one, you don't have to clear the lines, this can save some time at replay!

Remember that once you have set an alignment or an effect, the settings remain the same until you change it again.

At then of the video, your ENH file is ready and you can reload the animation to view it.

Note: as long as a **valid Enhanced file exist**, you **can't reach the Record button**.



IX Set Options and Paths

When in the main dialog box of an animation, or from the **General Menu**, you can enter the Options panel.

This one will ease your work as you'll be able to define several paths to avoid the use of many file selectors while working.

IX.1 Replay Options

Default path

You can set here the default path. When M_PLAYER is run, this is the first path that will be used in the file selector.

In this example it points to my anim collection on drive G :



Always count MPG frames

If your system is fast enough, you can set this option and frames will be counted when loading and MPEG video.

If it is not set, you let M_PLAYER decide, according to the power of your machine, the limit above which it won't count automatically.

Sound system

M_PLAYER detects your sound hardware through the cookies.

- _SND :** if bit 0 is set, then **Yamaha** available
 if bit 1 is set, then **DMA** available
PSND : if present, a **Pcard** is available.

According to this, the different options appear enabled or disabled.

When loading M_PLAYER.OPT, if the saved value doesn't match the current hardware, or if no option file is present, then M_PLAYER selects for you the best hardware :

- ✓ DMA/SAGA is preferred (*no time consuming and good quality*)
- ✓ Psound is the second one (*an interrupt with fast routine, medium quality*)
- ✓ Yamaha is the last one (*an interrupt with large routine, low quality*)

If you change your sound setting, you'll be warned to reload the animation (if any is loaded in the main dialog).

You can ignore this if the animation has no sound or if you won't replay it (for example if you just disassemble the file in TGA images).

MJPEG frames per second

The MJPEG files don't include any timing information. On this line you can specify the default frame rate from 0,1 to 99,9.

Example:

0,1 frame per second means 10 seconds for each frame. This is a slideshow.

Max button (for MJpeg)

Again on slow machines, you may not want to respect a frame rate with MJPEG files, but instead watch all frames one by one. In this case, select this button that validates "Max speed" as default setting in the main dialog instead of "xx.x frames/sec".

Enh button

When an Enhanced file is detected and valid, the Enhanced button will be selected (*if Enh is turned on*) or unselected (*if Enh is turned off*). So this lets you define the default setting.

Win button

When replay in a window is available, the "W" button will be selected (*if Win is turned on*) or unselected (*if Win is turned off*). So this lets you define the default setting for the very first time only. Then the setting of the "W" remains what you selected before.

IX.2 Save and convert options

These are very important and will ease your work. Here is the organization of my video working drive I :

I want every sound to be loaded or saved into the SOUNDS folder, every TGA saved into IMAGES and every conversion video saved into CONVERT.

These are the paths I have selected.

Create Folder with Anim name

If this is set, it applies to the TGA saving. A sub folder will be created with the anim name and there your images will be saved.

For example, when I disassembled BDBY87.AVI, the folder BDBY87__ was created and the images were saved inside it.



Auto Name

If this is set, then no file selector is open and the names are computed automatically.

For a sound :

The name is the same as the video with AVR extension.

For TGA images :

The name takes the first characters of the video name and the last ones are the image indice. The number of digits is set according to the number of frames available.

If this number is unknown (large MPG), 7 digits are used.

For example, if MYVIDEO.MOV has 1520 frames, then the image will start at MYVI0001.TGA.

For Video conversions :

The last three characters and the extension summarize all the settings.

With xxxxxQZB.EXT

Q is the quality from 1 to 5.

Z is the zoom factor (H for half, D for double and S for same)

B is the number of bits (8 for 8 bits and 6 for 16 bits compression)

EXT will be either MOV or AVI.

For example : HULA2S6.MOV is a conversion from HULAHOOP.MPG with [2] quality=2, [S] same size and [6] 16 bits MOV.



Disable

If this is set, all save and convert options are temporally disabled but not deleted. So you get the file selectors and have to fill the names by yourself.

Stats after conversion

If this is set a stats box opens after a conversion with the performance results.

Warnings

If for you some warnings are annoying and that you know how to use M_PLAYER, you can disable this button. You won't get them anymore.

Set Sound

If you own a TT, then the Microwire Interface is available to control the sound and this button becomes available, it leads you to this dialog.

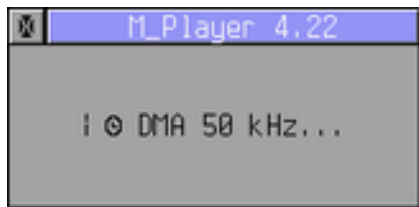
If you have a machine with the XBIOS routine "Soundcmd()" (such



as **the Falcon**), the panel is limited to the left/right volume controls. The same for the SAGA of the **Apollo Vampire**.

To use "Test Sound", your application folder must contain the file M_PLAYER.AVR witch contains data for a 25 KHz, stereo sound, 8bits signed.

On the right side, you can **test the real DMA frequencies** of your machine. Compared to the original definition of the DMA STE sound, **newer machines** have a slightly **different tuning** that can lead to **inaccurate synchronization** between images and sound.



M_PLAYER includes the frequencies for the **TT**, the **Falcon** and **Arany** (*limited to what I observed on my own computer*). If you notice a sound lag, then click on Run test.

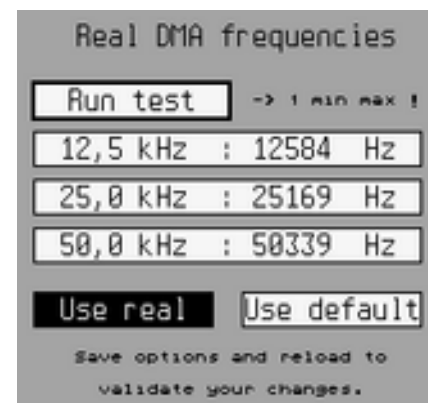
Three passes will be performed, each one lasting 20 seconds or less, for the 50, 25 and 12.5 kHz sound.

Then, the display will show you the results. Here you can see **what is expected on a TT**.

Notice that the sound is **a bit faster** than on a STE.

You can click on Use real to use those values.

You can click on Use default to keep what's built inside M_PLAYER.



Using Accept only sets the Microwire, if you want to use the new frequencies, you have to **save** the options **and reload** the program.

SAGA: For now, real frequencies calculations are not available on the **Apollo Vampire** as there is no way to know if a sound is stopped. Then, the program can't compute the duration of a sound.

The parameter **SA : 0** is the audio channel used by M_PLAYER on the Vampire. If you experience some conflicts with other programs or drivers, you can change this knowing that:

- channels 0 and 1 can replay both mono or stereo sounds.
- channels 2 to 7 can only replay mono sounds.

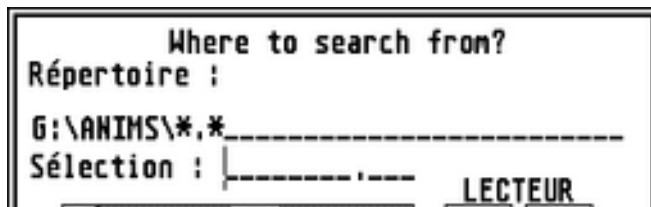
In the case you have to use a mono channel, then stereo sounds are mixed to a mono track.



X Search a disk for animations

You can create a **batch file** or a set of **albums** that will contain every animation found into a disk or a folder.

The file selector opens and ask you for the place where it should start to search. As an example, I select G:\ANIMS\ and validate my choice.



Note: you can also start a search when you're in the file selector for replaying videos. To do this, select the path you want in the file selector, the filename field won't be used. **Then press the Alternate key and validate.** This was the old way to do it before the General Menu was created.

A dialog opens and lets you select the type of files you want to search for.

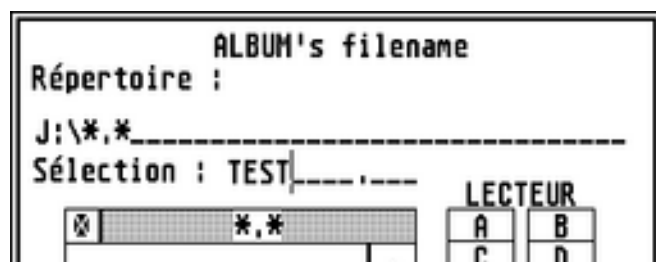
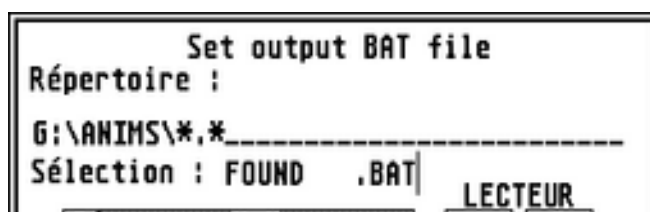
Here an example with MPG, JPG and MJP files wanted.

You have to decide whether you want **Albums** or a **Slideshow**.

Then press **Start !**

If you selected **Slideshow**, you are asked for a name to save the batch file. Then the search begins.

This **includes all sub-directories** of the one selected.



If you selected **Albums**, you are asked the album's name. In this case the file extension will be ignored as it is provided by the program.

In the example here, the first album will be named TEST.000, the second TEST.001 and so on according to the number of files found.



At the end, M_Player tells you how many directories were searched and how many files were found.

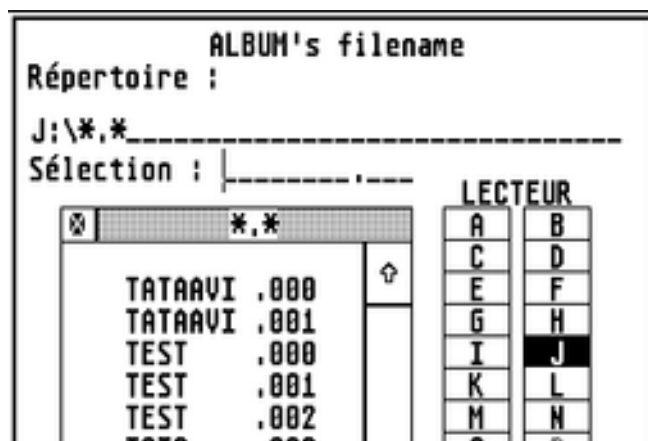
For a **slideshow**, you can validate and **automatically**, **the next file selector points to your batch file** if you want to view all the anims that were found.

They are all replayed in sequence with a black background, no window.

If you selected **Albums**, you can enter the corresponding section via the **General Menu** and load them. The example above leads you to three albums created:

TEST.000 to TEST.002.

More info here: Albums



X.1 Slideshow of animations

A batch file for a slideshow of animations is a text file organized this way :

```
SLIDEANI  on the very first line and column
path+name of first animation
path+name of second animation
...
.stop          period stop on the last line.
```

The batch file can be edited and expanded with other features such as loops or numbered file names.

Each command must start with a period. Valid commands are:

.rept xxx, .endr, .incr, .decr, .disp, .stop

Example:

```
SLIDEANI
e:\anim\intro.avi
.rept 5
e:\anim\me.avi
e:\anim\you.avi
.endr
e:\anim\end.avi
.stop
```


This file will start with '**intro.avi**', then will display 5 times **me.avi** and **you.avi**, finally it will display **end.avi**.

Numbered files example:

```
SLIDEANI
e:\anim\intro.avi
e:\anim\scene000.avi
.rept 50
.incr
.disp
.endr
e:\anim\end.avi
.stop
```

This file will start with '**intro.avi**', then it will display 51 scenes named '**scene000.avi**' to '**scene050.avi**', and it will end with '**end.avi**'.

IMPORTANT NOTE

No resolution switch is made between two animations, no 'clear screen' either. So, you must be sure that all of your animations will be displayed into the same resolution and that every new one will cover the previous one to avoid garbage on the screen and crashes of the system.

XI EASY_BAT.PRG (discontinued)

EASY_BAT is a *semi-friendly* **tool to create BATCH files**.

The sequence of dialog boxes can't be altered, so if you miss something, then you have to do it all again. That's why it is semi-friendly...

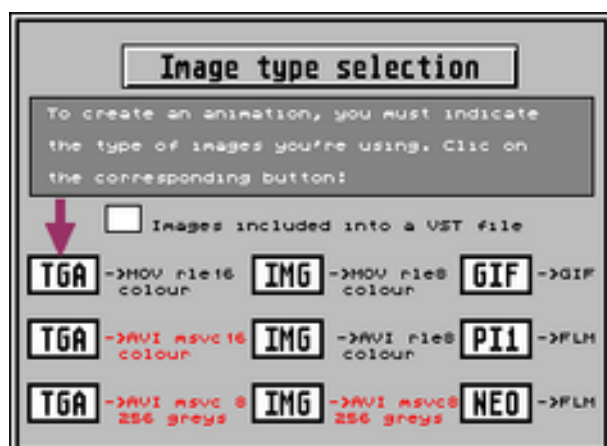
The program is discontinued but still available. You should prefer the integrated panel or directly modify the batch file for specific features.

Example: let's go back to the VR-movie creation with the 18 views of the dice as seen in QuickTime VR movies. Let's run EASY_BAT, after the copyright screen, you get this :

XI.1 Image type selection

Here I select **TGA** to create a MOV with RLE16 colors.

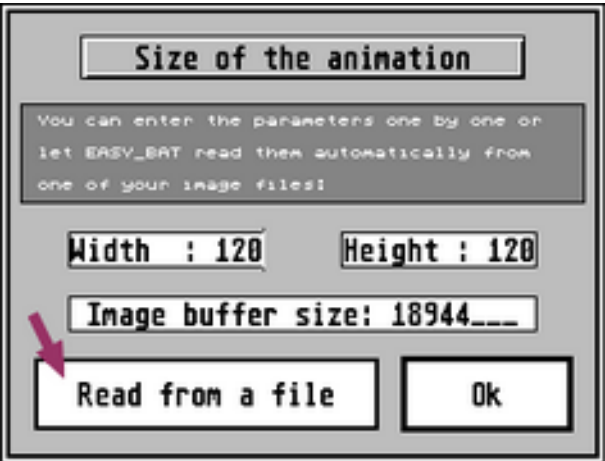
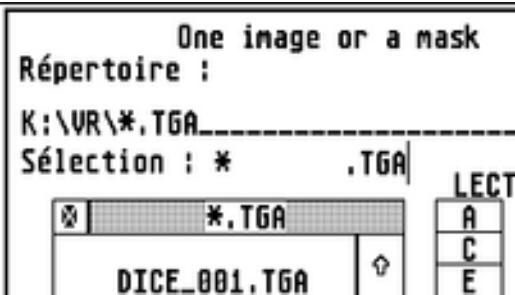
Only MOV files can handle the VR extension.



XI.2 Size of the animation

Here I click in **Read from a file** and in the file selector, I enter the folder containing my images and just write *.TGA.

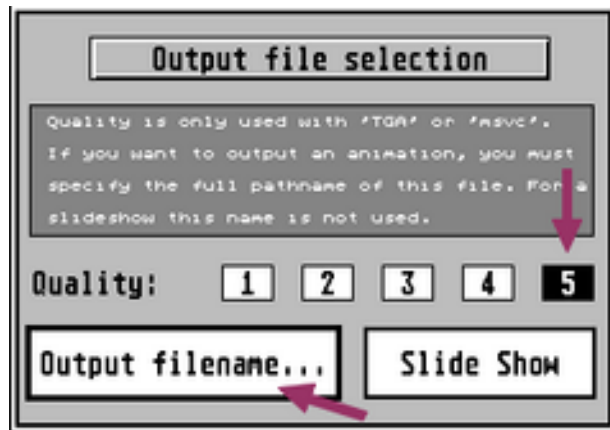
It fills automatically the three fields.



XI.3 Output file selection

Here I select **quality 5** (*the best*) and then I click on **Output filename** and give a name to the movie that will be created.

```
Name of the animation:
Répertoire :
K:\VR\*.MOV
Sélection : 3D_DICE .MOV
```

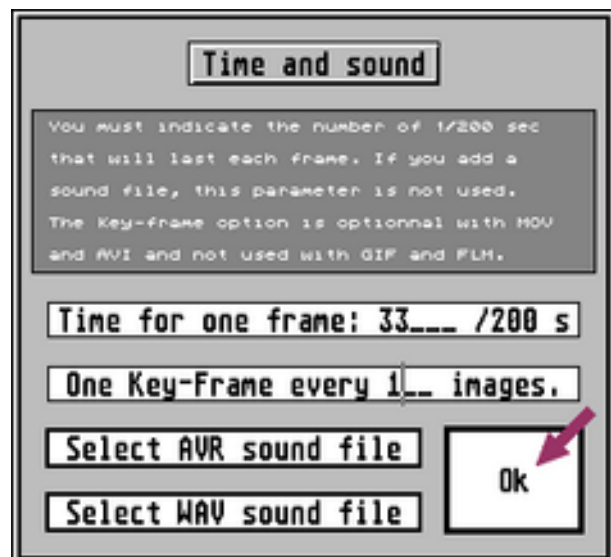


If I just want a slideshow of images, I can ignore everything and click on **Slide Show**. In this case, the BAT file won't allow you to create an animation, just to view the images in sequence.

XI.4 Time and sound

If an animation contains a sound, then the frame rate is automatically computed to fit the sound duration.

Here, no sound, so I write 33/200 sec for one image (6 fps) and I put **every frame** as a key frame.



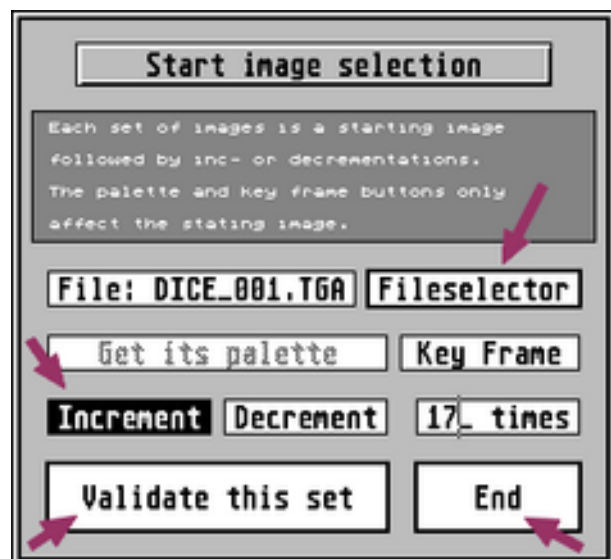
XI.5 Start image selection

In this case, the images have all contiguous numbered names. So I just have to select the first one with **file selector**.

Then I indicate that this name will be **incremented 17 times** (up to DICE_018.TGA).

I validate this set.

Then **End**, no more images required.



XI.6 Global repetition

In this case, it is not applicable.
So I let a blank field and OK.

This is mostly used with GIF movies.



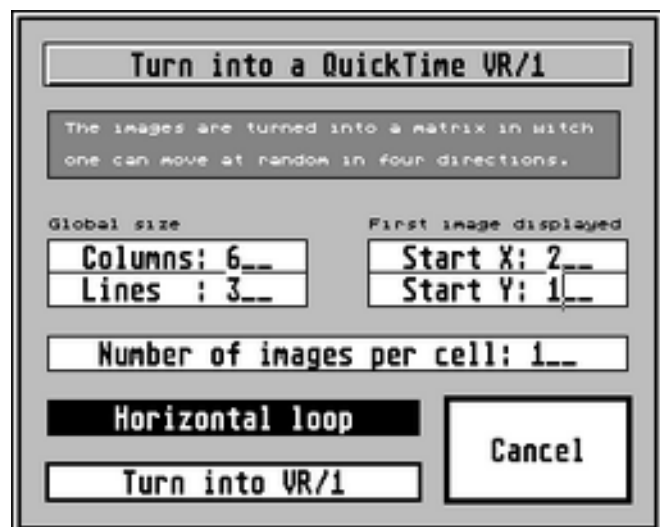
XI.7 Turn into a QuickTime VR/1

The special box for **VR information**.
I set the **number of columns and lines**.
I set the **initial image coordinates**.

There is **only 1 image per cell**, this means that they are fixed views.

I enable the **horizontal loop**.

Finally, I click on **Turn into VR/1**.



XI.8 Save the batch file

There we are, the BAT file is ready.

I click on **save**, and I give it the name 3d_DICEF.BAT (F for fixed images).



XI.9 Create the animation

You have to load the BAT file in M_PLAYER.

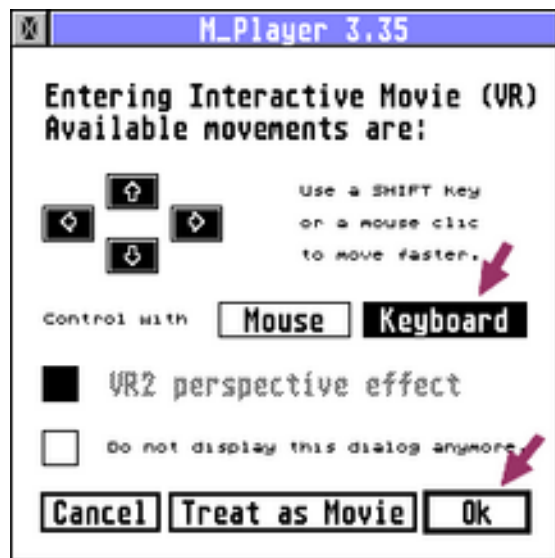
Ensure that you have checked the « Save As... » button to **enable the creation**.

Click on **Play !** The creation starts



XI.10 Replay the animation

Load the 3D_DICE.MOV animation in M_PLAYER, notice that under the name the QuickTime VR format is detected.



I select **Keyboard** to move the dice, then **Ok**.

Finally, the animation appears and I can **move around** the dice left, right, up and down.



XII Command line options

The command line is:

`<options> path\file_name`

The options must appear before the file name! Options can be separated with spaces or not.

Every option is:

`+LETTER` to set this option
`-LETTER` to clear this option

Available options:

+d, -d, -D Display Dialogs

if set will display the dialogs, if cleared won't display them. With -D (upper case), the Loading sound box won't appear, whereas with -d (lowercase) it remains.
Default value: +d

+p, -p Play Sound

(only active when '-d' is specified), then if set will play the sound (if available), if cleared won't play the sound
Default value: +p

+s, -s Synchronize

(only active when '-d' is specified), then if set will synchronize the images and sound, if cleared will display the images at the max speed.
Default value: +s

+a, -a Save As...

(only active when '-d' is specified), then set « Save As... » to enter step by step mode automatically or to create a movie with a batch file.
Default value: -a

+e, -e Error Messages

(only active when '-d' is specified), then if set will display every error message (bad file, etc...), else if cleared, won't display any error message.
Default value: -e

+i, -i Repeat

(only active when '-d' is specified), then if set will play in loop mode (on the mask or single file passed through the command line) until an event occurs (keyboard, mouse, joystick), else will play normally.

Default value: -i

This is for being used with a screen saver for example.

+r nnnn Return infos

(+r and -r are equivalent)

specifies the address of a buffer where M_Player will return the infos about a movie, in this mode the anim is not played, you'll get no dialog, only the buffer filled (72 bytes are needed):

WORD status	1=OK 0=File not found (nothing else filled) -1:unknown type (only file_name filled)
CHAR file_name	(14 bytes: max 12 + null + even address)
CHAR file_type	(28 bytes: max 26 + null + even address) such as 'Video for Windows (AVI)'
WORD graph_status	1: supported 0: no graphics (info's filled with garbage) -1:unsupported (info's filled)
WORD width	
WORD height	
LONG number of frames	
LONG compression	(4 bytes such as 'cram', 'cvid'...)
WORD sound_status	1: supported 0: no sound (info's filled with garbage) -1: unsupported (info's filled)
WORD sound bits	(more often 8 or 16)
WORD channels	(1:mono, 2:stereo)
LONG frequency	
LONG version	4 ASCII bytes representing the version of the player.

+xnnn, +ynnn Coordinates

(only active when '-d' is specified), then fixes the position where to display the anim. If a resolution switch is necessary, then this position will be ignored and the anim will be centered as usual.

In this mode, the background is not cleared, so you can display an anim into one of your windows. There are no checking upon these coordinates, you must be sure that the anim fits into the screen.

Notes:

- *when using a TT without graphic card, the TT Low mode is often used, and aspect correction can double video height.*
- *+x100 or -x100 are equivalent, both x and y must be specified, else the coordinate is ignored.*
- *when using a TT without graphic card, the X value is automatically aligned to a 16 pixel boundary.*

+w, -w Set window use

+w force the usage of a window. If it's on a standard TT, then no resolution switch is performed and you may get an animation with a monochrome dithering, unless you run in 256 colors.

-w disables the usage of a window. If it's a standard TT, then a resolution switch is performed if required to enhance the display with more colors/grey levels.

Default depends on your choice in the option box (WIN button to tell what is your preference) and the current state of the W button in the RSC file.

Remarks:

- **Options p, s, a, i, w and e are only used when '-d' is specified**, else, the player uses the returned values from the dialogs to set the options.
- When +x/+y are used, then the program checks if the display fits into the screen. If not, then the program halts and returns a value of -1.

Examples:

m_player -d *.AVI

will display every AVI from the current folder without any dialog and using the default values (+p, +s, -a).

m_player -d +a c:\batch*.bat

will create MOV files from every BAT file found into the [c:\batch](#) folder.

m_player -d-p+s+i C:\anim*.*

will display every file found into the anims folder without sound but at their normal speed (synchro is on). Will also loop into this folder until an event occurs (keyboard, mouse, joystick), it means that when the end of the folder is reached, the research starts again.

m_player +r81500 C:\anim*.DL

will fill the buffer with the info's corresponding to the first DL anim found into C:\ANIM. If no DL file is found, the first word will contain -1 and garbage for the other bytes.

Note: for options using a number as parameter (+x, +y, +r...), you can write:

+x200 decimal value

+x\$C8 hexadecimal value (with \$)

+xw?? w indicates that the two bytes ?? represent a WORD value

+xl???? l indicates that the 4 bytes ???? represent a LONG value

The '**w**' and '**l**' facilities allow you to pass parameters without the Integer to string conversion, just 'poke' them into the command line.

XIII Calling M_PLAYER as an accessory

When **M_Player** is an **accessory**, it waits for two messages:

AC_OPEN (40)

when called from the menu bar of the desktop

M_PLAYER ('MP' or \$4D50)

when called by another application, the AES buffer must be like this:

word(0) : \$4D50
word(1) : appl_id of the calling process
word(2) : 0 (not more than 16 bytes)
word(3) and
word(4) : LONG address of the command line
word(5) : return message or zero.
word(6) ; not used
word(7) : not used

If the **LONG address** is 0, then no command line is used and you'll get the normal dialogs and file selector. If an address is specified, it should point to a null-terminated string corresponding to the command line as described above.

Unlike the Gemdos command line, there is no first byte indicating the length of the string.

For example: "-d+x100+y100 d:\anim\test.flm"

to display the anim TEST.FLM at (100,100) without clearing the background and without displaying the dialogs.

return_message, if set to 0, is unused. But, if set to anything else will cause M_PLAYER to send back to your application the corresponding message when the replay is finished.

For example, if return_message='ND', your application will receive the message 'ND' (with evnt_mesag) when M_Player has finished.

XIII.1 The "Mountain" mode

This mode was created in association with Rémi Vanel, the author of Mountain, a GEM video maker software that initially could output FLI/FLC/FLH animations. With this new system, its capabilities were expanded to MOV/AVI output with sound.

The main feature of this mode is to be able to **build an animation** without saving images to disk, they are **created on the fly** and directly assembled to the movie. So, the calling program and M_PLAYER **work together** in parallel.

With a basic TOS system, it is recommended to install M_PLAYER as an accessory, but you can still call it with the GEMDOS Pexec() function.

On a Mint system, you can call it with the AES Shel_Write() function. Take care that all shared memory blocks must be declared as global, or you can simply specify this in the PRGFLAGS of your program header.

So, how does it work?

Your program must prepare four things:

- ✓ a routine that will create an image when called by M_PLAYER
- ✓ a 16 bytes structure to dialog with the player
- ✓ a buffer that will contain the pixels you create
- ✓ a standard BATCH file using dummy file names and including the "Mountain" parameter:

Examples:

For a 160×120 MOV animation with sound in RLE16 and 150 frames:

```
M_PLAYER
c=tga2
w=160
h=120
b=38418
o=F:\EXAMPLE.MOV
q=5
s=F:\SOUND.AVR
f=150
m=xxxx
data
DUMMY000.TGA
.rept 149
.incr
.disp
.endr
.stop
```

For a 640×400 FLM monochrome animation with 25 frames and 10 fps:

```
M_PLAYER
c=dega
w=640
h=400
b=32034
o=F:\EXAMPLE.FLM
t=20
f=25
m=xxxx
data
DUMMY000.PI3
.rept 24
.incr
.disp
.endr
.stop
```

Then you program saves this text file, let's say MOUNTAIN.BAT, and calls M_PLAYER with -d and +a options on the command line:

Command line is `-d+a mountain.bat`

Either into Pexec() or with the 'MP' AES message.

The dialog starts ! M_PLAYER creates the animation and, every time it needs a new image, it will call your routine.

`m=xxxx` (xxxx in decimal)

xxxx is an address that points to this structure:

```
LONG your_routine
LONG frame number required by M_player
LONG the address where your pixels are
WORD flag
WORD extension
```

For the first call, you must fill your **routine address** and the flag and extension.

FLAG='EX' then an extension is available

extension = \$0201 zoom×2 or \$0102 zoom÷2, \$0101 no zoom.

If you add \$8000 to the extension, then you ask for the **progression dialog box** to be displayed, else the player is mute.

extension = \$8201 zoom×2 or \$8102 zoom÷2, \$8101 no zoom.

Then, each time an image is needed, M_PLAYER fills the **frame number** and calls your routine. Upon return, your routine must fill the **buffer address** and the **flag** format.

Supported values are (for now):

FLAG = 0	16 bits INTEL: gggbbbbb xrrrrrgg
= -1	16 bits Falcon : rrrrrggg gggbbbbb
= 4	a complete DEGAS image 320×200×16 with palette
= -4	a complete NEOchrome image 320×200×16 with palette
= 1	a complete DEGAS image 640×400×2

Values 0 and -1 are used to output MOV/AVI files with an eventual dithering to 256 colors.

Values 4, -4 and 1 are used to output FLM Extended files.

XIV Supported file formats

Here is the list of what M_PLAYER can read.

QuickTime *.MOV (VR1 Objects & Vr2 Panoramas supported)

Video codecs : CVID, RLE1, RLE2 (grey and color), RLE4 (grey and color),
RLE8 (grey and color), RLE16, RLE24, RLE32, SMC8 (grey and color)
RAW1, RAW2 (grey and color), RAW4 (grey and color),
RAW8 (grey and color), RAW16, RAW24, RAW32, RPZA (15 bits)
WRLE (256 colors), MSVC8 (grey and color), MSVC16
YUV2, YUV9, YVU9, **JPEG, MJPA, MJPB**
(For **JPEG, MJPA** and **MJPB**, the external module *JPEG_68K.RIM* must be
located in the application folder).

Audio codecs : TWOS, RAW (8/16 bits, mono/stereo), IMA4

Video for Windows *.AVI

Video codecs : CVID, CRAM16, CRAM8, MSVC16, MSVC8
RLE8, RGB8, YUV9, YVU9, IV32, **JPEG, MJPG**
(For **JPEG, MJPG**, the external module *JPEG_68K.RIM* must be located in
the application folder).

Audio codecs : TWOS, RAW (8/16 bits, mono/stereo)

JPEG *.JPG

Unique images. (*JPEG_68K.RIM* required)

MJPEG *.MJP

Video codec : JPEG (*JPEG_68K.RIM* required), sequences of concatenated images.

MPEG *.MPG

Video codec : mpeg1, mpeg2
No support for audio.

Extended Lexicor Film *.FLM

Video codecs : ST Low, ST High, TT Med
Audio codecs: DMA 8 bits mono/stereo, 12,5kHz, 25kHz, 50kHz.

Compuserve *.GIF

Video codecs : GIF87, GIF89

Video Master *.FLM

Video codecs : VMAS, VFAL

Audio codecs : 8 bits mono, any frequency

CD Stratos Magazine *.CDH, *.CDL, *.CDV

Video codecs : ST Low, ST High, Falcon 256

Audio codecs : DMA sound 8 bits mono 12,5kHz

Old ST Low Atari animations

- *.SEQ (Cyber Paint)
- *.PI1 + *.DLT (Cyber)
- *.FLM (Kinetic Microsystems)
- *.FLM (Lexicor)

Old PC animations

- *.FLI/FLC/FLH (Autodesk Animator)
- *.DL (types 1 and 2)

Batch Files *.BAT

For slide shows or creation of animations.

Images : TGA2, TGA10, 16 or 24 bits, uncompressed or RLE

GemXIMG 256c, GIF, PI1, NEO

Sound : AVR or WAV, uncompressed.

XV Translating the interface

As of version 4.09, every single text has been removed from the executable file to be stored into the RSC file. This way, one can easily translate the whole interface including the alert messages.

XV.1 File selector titles

The file selector can display a title with a maximum of 30 characters to guide the user.

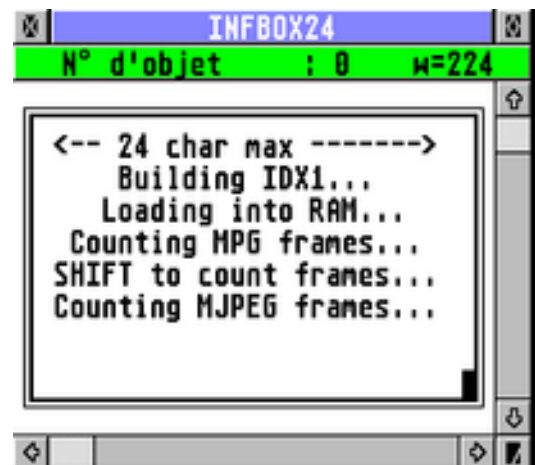
You'll find all those titles into the FSEL30 tree. A first line remembers you the limit you can't bypass.



XV.2 General info box

One general information box is used and is currently limited to 24 characters.

If you can't fit your text into this limit, you must modify the corresponding tree GENINFO.



XV.3 Step by step box

This special box reminds you the keys you can use when disassembling a video. They appear enclosed in parenthesis.

This string is parsed by M_PLAYER to adapt the key to every language.



Be sure not to use twice the same key! In english, keys S, A, R and Q are used.

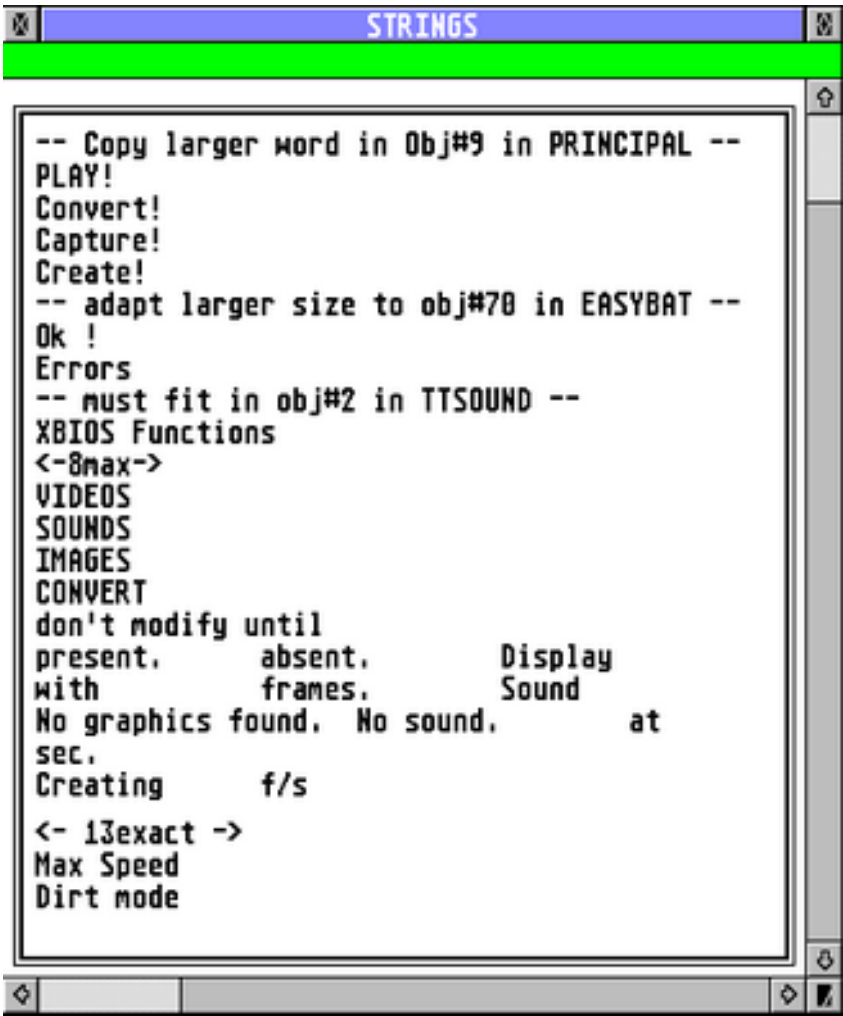
XV.4 Free strings

Lots of other strings are grouped into the STRING tree.

Some are just information to guide you (telling you to copy one string into a specific tree or to limit your text to a certain number of characters).

Respect the strings that end with a period "." and those who don't.

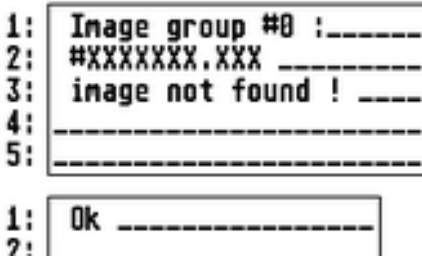
Respect the uppercase letters (start of a sentence) and the lower case.



XV.5 Alert strings

When translating alerts, you'll find some strange fields starting with "#". You must leave them as they appear including the spaces before and after if present.

M_PLAYER will look for those fields to display specific information according to the context.



XV.6 General menu

For the general menu, there is no other way than redrawing the icons to change the text..!

I can help.

