Video Streaming across wide area networks

Research on technologies for content distribution, by Denis Rojo and Eleonora Oreggia

Introduction

This research aims to focus on several aspects of video streaming: compression algorithms, broadcasting daemons, encoding technologies, patenting and licensing issues, secure authentication measures and archiving.

A range of possibilities is to be examined to individuate a viable technology which will be the underlying framework for Wide Area Network (WAN) access to the Video Art Collection of Montevideo.

Main requirements are quality of the video playback, standardization and predictable longevity of the compression algorithm adopted.

Other parameters we'll take into account in our exploration of digital video streaming technologies are: efficiency, costs, compatibility, integration with adopted technologies, independence and freedom to adapt the tools to our needs.

Being the field of research a rapidly evolving ground, this research has to be referred as the current status of technology in year 2006. At the time of writing some promising technologies are still in development and, once they achieved maturity, they might slightly affect our results[1].

Streaming

When we speak of on-line distribution we must first make clear a distinction between **on-demand** and **programmed** streaming of contents: in the first case the audience can choose what to play at any time on each screen, while in the second case the broadcaster decides the sequence of videos streamed on all audience screens. The choice among these two distribution schemes is purely functional to the service that must be provided, sufficiently advanced technology is available to realize a stable setup in both cases.

On-demand streaming raises bandwidth costs: it is necessary to establish a new network stream for each player, which can rapidly fill up all available bandwidth in case of a wide audience. On-demand access to the video collection should therefore be time constrained and limited to a special part of the audience.

Programmed streaming can establish a channel for audience where content is organized in a time schedule: it is possible to select and group videos and can provide streamlined broadcasts scheduled at certain times (as, for example, in television).

---

1 Codecs like Dirac and Snow are still in development and therefore kept out of this research, while it is foreseeable their importance and incidence on our results once they are mature enough for production use, we are then planning an update of this publication.

Implementation

There are various possible solutions implementing a stable video streaming service. In case of on-demand streaming, the service to be provided is comparable to a normal web server, where clicking on the selected link can start the video; with programmed streaming it gets more complicated as we need a user-friendly tool to administer the program schedule.

For **on-demand** streaming a simple web server setup with server-side scripts[2] can easily index, provide informations and stream video on-demand to the users, while keeping the whole setup as flexible and open to future extensibility as possible. It's the current case for the Montevideo Catalogue is locally accessible in MPEG2 format (consultation in the Mediateque) while it is streamed on the internet using RealVideo. As it is an extensible system, choice can be easily provided for remote consultation to use open and non-proprietary codecs, as explored in this research.

With **programmed** streaming in most cases we are dealing with real-time encoding, which requires powerful CPUs and, in case of live events, the need to archive the videos while streamed. Such tasks can be integrated and automated with various available open source technologies: shell scripting, available command line encoders/streamers and website front-ends can be combined into a framework for ad-hoc solutions. On the other end, such setups can be very expensive in case of proprietary technologies which anyway offer less flexibility to be adapted for specific needs.

Currently the programmed streaming of events from NIM is realized using Mpeg4IP (free and open source software) streaming in Mpeg4 codec over a Darwin Streaming Server. It is a reliable solution, but still lacks an interface for scheduling and relies on MPEG4 which is a patented proprietary codec.

A more flexible solution is offered by scripting of ffmpeg2theora in simple graphical interfaces that are functional to specific tasks, encoding with the Theora codec and then broadcasting the stream over an Icecast server, which provides a versatile authentication scheme to be integrated into most existing accounting databases.

Authentication

Access to video streams can be managed on a user basis, providing accounts and passwords that can give access to different subsets of the collection, during certain periods of time and from certain geographical areas or specific buildings.

All available authentication methods provide a sufficient level of security, which is still not the highest: a key or password protection can be requested at the moment of connection, but once access is granted the stream will be transferred in clear[3] to the players, which in extreme cases makes the data transfer vulnerable to tapping, spoofing and man-in-the-middle attacks.

In order to enhance the level of security encryption should be applied on the

2 like Apache, with the help of server-side scripts (PHP) and a MySQL database
3 Not encrypted

video stream; but even there, as for example in case of satellite transmissions, several vulnerabilities have been discovered which are already exploited and nowadays widely available to consumers, see for example the DeCSS[4] case.

Currently there seems not to be a strong encryption algorithm available for video and, even if there would be one, the most vulnerable point stays uncovered: the so called analog hole[5].

Copy protection

When dealing with digital media it makes no sense to talk about copy-protection.

Digital Rights Management (DRM) systems have been developed since the computer gaming industry exists and they were all doomed by failure, as piracy reached to widespread even in the most restricted technologies.

Even the most recent attempts to enforce copy-protection with the so called Trusted Computing (TC) technology can't be successful unless all computers are controlled by it and unless the control is not removable: such a situation can never be totally realized and in case of video it is enough to hack one player in order to get all the materials out of protection and freely copied.

Therefore when stepping into the digital dimension of distribution a balanced approach must be kept to knowingly evaluate the possibilities offered by digital technologies as well their limits and especially their differences from earlier distributions methods bound to the physical world.

So while it is impossible to deny completely the copy of materials once they are delivered, it is still possible to establish trust with distributors and audience to agree on their usage: it is the case of cultural events and educational institutions, in which a specific contract can be agreed transparently and matching the artist's will about the distribution of artworks.

It is interesting to see how the Creative Commons licenses analytically categorize the various choices the artist has for the distribution of artworks, resuming and “iconifying” the selected distribution terms in order to be quickly understood by humans as well efficiently cataloged by machines. These licenses have been already translated and forensically matched to various law systems all around the world: artists adopting them are provided an efficient instrument to define the boundary of property on their creations[6].

---

4 **DeCSS** is a computer program capable of decrypting content on a DVD video disc encrypted using the Content-Scrambling System (CSS).

5 The **analog hole** (sometimes **analog reconversion problem** or **analog reconversion issue**) is a fundamental, and inevitable vulnerability in copy prevention schemes for noninteractive digital content which is intended to be played back using analog means. When the information is converted to a perceptible analog form, there are no restrictions on the resulting analog signal, and the content can be captured back into digital form with no restrictions.

6 As they evolved quickly on the wave of the Free Software Movement, it must be mentioned that there are current critiques to the way CC licenses ended up enforcing property rights in the digital domain, while initially being recognized as an effort towards liberty of knowledge exchange. The editor notes here there is a difficulty in such a philosophical debate and it is distinguishing between importance of sharing artworks and the importance of sharing algorithms.

CODECs

What are commonly called codecs are algorithms for de/compression (**co**ding/**dec**oding) of digital data, in our case video.

We are moving into a minefield: low bandwidth video streaming technology is interesting for different kinds of applications, mostly related to video displacement and analysis, and rapidly became an hot spot for business companies as well. The codec is the core part of this technology: it's what really makes the difference as well what is being patented by commercial producers in order to establish control on the market. The choice for a proper codec is crucial for the realization of any video streaming project, while their adoption in media production is being exploited by large patent-holding corporations.

Since the bandwidth and the computing power of the providers/listeners is not infinite, it is very important for the codec to realize an efficient trade-off between bandwidth and quality, compression and computational costs, accessibility and support of modern standards.

With WAN streaming we can care less about the long-term survival and compatibility of the codec we choose: we'll not convert the video collection in that format to preserve it, but only to provide it outside, then it will always be possible to re-encode all the materials from the local video collection into a new streaming format. Montevideo collection uses extreme care in conservation, keeping originals in uncompressed digital form[7] as well encoded at high quality in MPEG2. This means that choosing a codec for WAN streaming is not a definitive decision and in case enough facilities are available one might even choose to provide multiple formats for WAN video streaming, automatically re-encoding at a lower quality from files instead of tapes.

Now we'll proceed with an overview of every single codec we are examining, followed by an evaluation of their benchmarks. Further are attached tables with the results of encoding and playback benchmarks.

We have chosen four codecs for their maturity, stability and wide adoption: **Mpeg4**, **Theora**, **H.264/AVC** and **XviD**. We will continue describing each codec with an overview on licensing policies, portability and software implementations to stream and play it; comparative benchmark tables reporting the results of our tests are then included in the appendix section.

MPEG4

Description

The popularity of Mpeg algorithms is also due to their early appearance in both audio and visual compression fields. Mpeg4 is the ancestor of video streaming codecs for low-bandwidth streaming: introduced in late 1998, is the designation for a group of audio and video coding standards and related technology agreed upon by the ISO/IEC Moving Picture Experts Group (MPEG). The primary uses for the MPEG-4 standard are web (streaming media), conversational

7 BETA

(videophone), and broadcast television.

Features

MPEG-4 absorbs many of the features of MPEG-1 and MPEG-2 and other related standards, claims to add new features such as (extended) VRML support for 3D rendering, object-oriented composite files (including audio, video and VRML objects), support for externally-specified Digital Rights Management and various types of interactivity, but **as of today there are no complete implementations of the entire MPEG-4 set of standards**. It's audio counterpart AAC (Advanced Audio Codec) was standardized as an adjunct to MPEG-2 (as Part 7) before MPEG-4 was issued.

Licensing

MPEG-4 is **patented proprietary technology**.

AT&T is suing companies such as Apple over MPEG-4 patent infringement. This AT&T action against Apple illustrates that it is hard to know which companies have patents covering MPEG-4.

This means that, although the software to create and play back MPEG-4 videos may be readily available, a license is needed to use it legally. The MPEG Licensing Authority can license patents required for MPEG-4 visual techniques from a wide range of companies. Audio is licensed separately.

Implementation and Compatibility

Encoding can be done with mencoder and transports supported are **.mov .avi .mp4**.

Decoding is supported by a vast range of applications available on all platforms.

The primary MPEG-4 audio codec, AAC is decoded/played by Apple's iPod product line. Two video codecs included in MPEG-4, Simple Profile (SP) and Advanced Video Codec AVC, are decoded/played by the 5th Generation iPod (AKA the "video iPod"). However, neither the iPod nor Apple's Quicktime Player are fully MPEG-4 compliant decoders, as they do not natively support many of the required parts of the standard.

The streaming server supporting MPEG4 is **Darwin**, which is licensed under the Apple Open Source license[8].

Theora

Description

Theora is a video codec being developed by the Xiph.org Foundation as part of their free codec collection, and is completely **patent free**. It is popular among the open source community and its the video codec advocated by the Free

8 NIM/Montevideo already adopts this server for internet streams: it runs reliably but has no possibilities for flexible authentication of users and automatic archiving of streams.

Software Foundation.

Features

Theora implements wavelet based encoding, as opposed to usual discrete cosine transformation algorithms, a new competitive approach which opens up the possibility for integer-based calculus, easier to embed in low-cost chips.

The video provided by this codec can be flexibly combined with other audio layers in a variety of ways inside the Ogg and other containers, research is ongoing to embed meta-data bound to particular points on the timeline as well multiplex more multimedia information along the video file.

Licensing

Theora adopts a BSD-style license which is a permissive license and is one of the most widely used licenses for free software. It is used most commonly for the Berkeley Software Distribution, a Unix-like operating system for which the license was named. This license has few restrictions on it compared to other licenses such as the GNU GPL or even the default restrictions provided by copyright, putting it relatively closer to the public domain. The BSD License has been referred to as copycenter, as a comparison to standard copyright and copyleft free software.

Implementation and Compatibility

The free libtheora library contains the reference implementation of both encoder and decoder while also a fast optimized implementation is already provided for x86 multimedia assembler extensions (theora-mmx).

At the moment, there are a few official stand-alone programs to actually encode and stream Theora video. While Ffmpeg2theora uses the FFmpeg engine to realize a practical tool for batch-processing large collections of files, FreeJ[9] provides advanced functionalities like titling and compositing of multiple video sources into a single Theora stream, which makes it a tool for both mixing and streaming, with even more flexibility added by its scriptability (Javascript).

Decoding is fully supported on all platforms by VideoLan, while theora plugins are provided for Windows Media Player, Apple Quicktime Player and RealPlayer/Helix. For web based streaming, the solution offered by the Cortado java applet supports all platforms and can be well integrated into a web page, without the need of any configuration from the user side.

It's foreseeable that we'll have more hardware players capable of Theora video playback: manufacturers obviously prefer this codec as they can freely include it into hardware devices to have video playback without being burdened by patent expenses.

The streaming server providing best support for Theora streaming is **Icecast**, also maintained by the Xiph.org foundation. It is a reliable and very flexible server being developed since more than 5 years and already adopted in massive

9 FreeJ development was supported by NIM in the past two years.

production environments. The recent developments on Icecast 2 open up the ground for implementing ad-hoc security authentication schemes. This permits to provide the content upon membership subscriptions and can be tightly integrated with web based applications and databases.

H.264 / AVC

Description

The intent of H.264/AVC project has been to create a standard that would be capable of providing good video quality at bandwidth levels that are substantially lower (less than half) than what previous standards would need. Surprisingly enough, it satisfies the need for a variety of applications in both low and high bit rates and low and high resolution video: for broadcast, DVD storage, HDTV, RTP/IP packet networks, remote conferencing and multimedia telephony systems.

Features

Motion compensation, SP and SI frames and some more advanced mathematical theories realize this algorithm as very efficient.

Licensing

Usage of this codec is conditioned by **patent licensing royalties**.

As with MPEG-2 Parts 1 and 2 and MPEG-4 Part 2 amongst others, the vendors of H.264/AVC products and services are expected to pay patent licensing royalties for the patented technology that their products use. The primary source of licenses for patents applying to this standard is a private organization known as MPEG-LA.

The Digital Video Broadcast (DVB) standards body in Europe approved the use of H.264/AVC for broadcast television in Europe in late 2004.

Implementation and Compatibility

x264 is a GPL-licensed H.264 encoder used in the free VideoLAN and MEncoder applications and, as of December 2005, remains the only reasonably complete open source and free software implementation of the standard, with support for Main Profile and High Profile except interlaced video. A Video for Windows frontend is also available, but has compatibility problems, as Video for Windows can't support certain features of the AVC standard correctly. x264 is not likely to be incorporated into commercial products because of its license and patent issues surrounding the standard itself.

The LGPL-licensed libavcodec includes a H.264 decoder. It is used in many programs like in the free Videolan media player and MPlayer multimedia players, and in ffdshow and FFmpeg decoders projects. Apple Computer has integrated H.264 (claimed to be Main Profile, but actually Baseline Profile plus 1 B-frame

support) into Mac OS X version 10.4 (Tiger), as well as QuickTime version 7, which was released on April 29 2005 with Tiger. QuickTime 7 is also now available for Microsoft's Windows operating system. Apple uses H.264 in the system for video playback and use in iChat video conferences. Sorenson offers a commercial and closed-source implementation of H.264, not very popular indeed.

Several companies are producing custom chips capable of decoding H.264/AVC video while the patents are raising costs for such productions, especially on mobile phones.

There are no free streaming servers known to be supporting H.264 broadcasting, while the Apple Quicktime Streaming Server is a commercial solution for it.

XviD

Description

XviD is an open-source “MPEG4-class” video codec originally based on OpenDivX. XviD was started by a group of volunteer programmers after the OpenDivX source was closed in July 2001 and its use blossomed among file sharing communities as an efficient format to exchange DVD quality movies online. XviD was awarded as a Top Ten Open Source project in 2005.

Features

XviD features MPEG-4 Advanced Simple Profile features such as b-frames, global and quarter pixel motion compensation, lumi masking, trellis quantization, and H.263, MPEG and custom quantization matrices. XviD is a main "competitor" of DivX (XviD spelled backwards). While DivX is closed source and may only run on Windows, Mac OS and Linux, XviD is open source and can potentially run on **any** platform (embedded systems, set-top boxes, DVD players, hand-helds, mobile phones, etc).

Licensing

On January 2001, DivXNetworks founded OpenDivX as part of Project Mayo, intended to be a home for open source multimedia projects. OpenDivX was an open-source MPEG-4 video codec based on a stripped down version of the MoMuSys reference MPEG-4 encoder, however the code was placed under a restrictive license. In early 2001, DARC member Sparky wrote an improved version of the encoding core called encore2, which was updated several times before. In April, it was removed from CVS without warning. The explanation given by Sparky was "We (our bosses) decided that we are not ready to have it in public yet". In July 2001, DARC released a beta version of their closed-source commercial DivX 4 codec, which was based on encore2. Many accused DivXNetworks of starting OpenDivX for the sole purpose of harvesting other people's ideas to use in their DivX 4 codec, some were angry at the way DivXNetworks handled a so-called open source project. It was after this that a fork of OpenDivX was created, using the latest version of encore2 that a few people downloaded before it was removed. Since then all the OpenDivX code has

been replaced and XviD is free software published under the GPL.

Despite it's free and open source licensing, XviD might incorporate patented algorithms that cannot be exported to USA without complying with their patent office, although such issues have never been raised so far and would be spawning quite an engaging debate if they do.

Due mostly to concern over patents, the official homepage does not provide binary versions of XviD.

Implementation and Compatibility

XviD has been ported to various platforms and is being supported by all common media players on Windows, Apple and GNU/Linux/BSD platforms, while it is also supported in a number of embedded low-power video players (GP2x, mobile phones, etc.). Since XviD uses MPEG-4 Advanced Simple Profile (ASP) compression, video encoded with XviD can be decoded with other MPEG-4 compliant decoders as well. Codecs (DirectShow filters) for Microsoft's Windows are also available at Koepi's XviD homepage. For Linux users, many distributions provide the XviD codec for use with media player software, such as MPlayer and VLC (however, all of them use built-in MPEG-4 decoder from FFmpeg by default and therefore don't require XviD, some of them don't even offer the option to use XviD at all).

Compatibility problems arise between the transports that we can use to encapsulate XviD: we are once again caught into a war between corporate software manufacturers when we see that the **same encoded video** plays on Windows OS only if encapsulated into AVI, or else plays only on Apple OS if encapsulated in Quicktime.

The streaming server supporting XviD is Apple's Darwin Server, which is open source. Montevideo adopts this server for internet streaming of live events.

Encoder Benchmarks

The results here reported are referring to test files encoded with **ffmpeg** and **mencoder** using the **dyne:II GNU/Linux** operating system freely available on dynebolic.org, the script used for these experiments is included in the appendix of this research, all the software employed is provided as source (upon request) and in executable binary form as live-CD.

All mentioned codecs have been benchmarked using a test movie provided by Montevideo and including excerpts from most characteristic video-artworks in the collection, encoded in Mpeg2 8000Kbit/s in two different versions: **direct** and **prefixer**.

The **Prefixer** is an analogue filter specific for cleaning video signal from a tape before it gets digitized. From the encoding results we can clearly see that, starting from two mpeg presenting identical size (441 Mb), the size of the encoded clips is smaller when the video was prefixed, and that the higher the bitrate, the higher the size difference between 'direct' and 'prefixer'. This means that the more noisy and disturbed is the signal, the more information is needed

to describe it. And that on a higher bitrate, more resources will be used in describing details that are not giving a significant difference on a lower bitrate level.

In other words the differences between 'direct' and 'prefixer' become more sensible on a high bitrate.

The test movies are a collage of fragments of video-art presenting a wide range of possibilities and problems that can be encountered in preservation, digitizing and encoding of variate material. This includes, for example, original old film transported to video, both color and b/w, graphics, black and white, very fast animations, monotonic fields, colorful frames, different types of movement, the human skin, and very noisy images. The length of the files is 0:07:13

We have tested the compressed results of codecs with 8 different bandwidth settings affecting the quality and the traffic generated when streaming: 200, 300, 600, 800, 1200, 4000, 8000 Kbit/s, all test results have been commented and reviewed and the effective bandwidth usage has been verified and noted.

Among all four codecs tested, Theora and H264 gave the best results, both on the size and the quality side. In general a decent quality is reached only when the bitrate is above 1000 kbp/s, while H264 is the only codec really supporting bitrates lower than that, all other codecs are adjusting over 1000 kbp/s even when the settings are stating lower values.
What is remarkable of Theora is that a clip encoded at 4000 kbp/s presents a quality comparable to the original Mpeg2 in less than a half size. When compared with Xvid, Theora clips are lightly smaller, but significantly superior in quality. Especially on animations definition Theora showed to be the best algorithm.

On the other side H264 is performing well on small resolutions, but if you increase the bitrate the size will be much bigger than the equivalent in other codecs. Another problem found in H264 is that it requires a lot of CPU usage to be uncompressed.

Benchmark tables follow, below is the reference for the original test movie:

FORMAT	MPG
VIDEO CODEC	**MPEG2**
AUDIO SETTINGS	mp3lib, 48000 Hz, 2 ch, s16le, 128.0 kbit

	bitrate (kbp/s)	**size** (Mb)	**quality**	**colors**	**graphix**	**b/w noise**	**synch**	**notes**
Testtape Direct	8000.0	441	good	ok	ok	lightly pixelate on b/w anim	yes	noise on old film
Testtape Prefixer	8000.0	441	better	ok	ok	lightly pixelate on b/w anim	yes	image more clean, less noise in general

Mpeg4

FORMAT AVI
AUDIO SETTINGS mp3, 48000 Hz, 2 ch, s16le, 128.0 kbit

	Enc settings	**bitrate** (kbp/s)	**size** (Mb)	**quality**	**colors**	**graphix**	**b/w noise**	**synch**	**notes**
Testtape Direct									
	200	893.6	54	very pixelate	ok	borders non well delineated	square pixels and shadows	yes	
	300	1039.5	62	very pixelate	not altered	ok	square pixels	yes	
	600	1232.2	72	pixelate	not altered	ok	pixels	yes	bad res on old films
	800	1230.8	72	lightly pixelate	not altered but shadows	ok	pixels and horizontals lines (non de-interlaced aspect)	yes	Not good enough in b/w and animations
	1200	1629.1	92		not altered	ok	pixels on b/w movements	yes	
	4000	3998.7	215		not altered	ok		yes	
	8000	7388.7	391	good	not altered	good	pixels in b/w animations	yes	
Testtape Prefixer									
	200	873.1	53	low	ok	borders and shadows	square pixels	yes	
	300	1020.5	61	pixelate and shadows	not altered but pixelated	not bad	bad	yes	
	600	1230.7	72	lightly pixelate	not altered	ok	bad in noisy parts	yes	
	800	1213.6	71	not bad	not altered	ok	square pixels	yes	problems on b/w and anime
	1200	1633.2	92	good, lightly pixelate on bw	not altered	ok		yes	bad on old noisy films
	4000	3903.3	210	good	not altered	ok	lightly dirty on b/w animations	yes	
	8000	6651.7	353	good	not altered	good	b/w movements still dirty	yes	

Theora

FORMAT AVI
AUDIO SETTINGS vorbis, 48000 Hz, 2 ch, s16le, 128.0 kbit

	Enc settings	**bitrate** (kbp/s)	**size** (Mb)	**quality**	**colors**	**graphix**	**b/w noise**	**synch**	**notes**
Testtape Direct									
	200	200	43	quite good	lightly saturated	ok	pixelate on human skin and fast mov	yes	noise b/w is good
	300	300	43	quite good	lightly yellow	ok	good	yes	lightly pixelate on b/w human skin
	600	600	49	quite good	a bit saturated but not altered	good	good also on anime	yes	pixels on b/w human skin
	800	800	54						
	1200	1200	70	good	not altered	good	good	yes	a bit pixelate on b/w human skin
	4000	4000	210	very good	not altered	good	very good	yes	
	8000	8000	349	very good	not altered	very good	very good	yes	perfect on fast anim
Testtape Prefixer									
	200	200	37	ok	lightly saturated	ok	pixelate on human skin and fast movemen t	yes	quite good in animation
	300	300	38	quite good	not altered	ok	good	yes	lightly pixelate on b/w human skin
	600	600	45	quite good	not altered	good	good	yes	pixel on b/w human skin
	800	800	52	quite good	not altered	good	good	yes	pixel on b/w human skin
	1200	1200	69	good	not altered	good	good	yes	a bit pixelate on b/w human skin
	4000	4000	157	very good	not altered	good	very good	yes	
	8000	8000	220	very good	not altered	very good	very good	yes	

H.264 / AVC

FORMAT AVI
AUDIO SETTINGS mp3, 48000 Hz, 2 ch, s16le, 128.0 kbit

	Enc settings	**bitrate** (kbp/s)	**size** (Mb)	**quality**	**colors**	**graphix**	**b/w noise**	**synch**	**notes**
Testtape Direct									
	200	192.8	18	not too bad, but some shadows, ghosts and approximation	saturated	ok	not uniform, lightly pixelate	yes	
	300	290.7	23	not bad, shadows and ghosts	saturated, blue especially	ok	not bad	yes	unstable image on color grey scale and anim
	600	587.3	38	not bad but motion estimation problems	lightly saturated but not altered	good	good	yes	unstable image, shadows and ghosts on anim and grey scales
	800	786.0	49	quite good	ok but very light	good	good	yes	Still some ghosts on grey scales and noise field
	1200	1181.5	69	good	not altered	good	good	yes	
	4000	3955.2	213	good	not altered	good	good	yes	
	8000	7885.3	417	good	not altered	good	very good	yes	
Testtape Prefixer									
	200	192.6	18	not too bad, but some shadows, ghosts and approximation	saturated	ok	ok but grey scales can make ghosts	yes	
	300	290.5	23	not bad but shadows and ghosts	a bit saturated	ok	ok	yes	unstable image on grey scale and anim
	600	585.1	38	not bad	not altered	ok	ok	yes	motion est shadows and ghosts
	800	782.1	48	good	ok	good	good	yes	
	1200	1176.5	69	good	not altered	good	good	yes	
	4000	3928.5	211	good	not altered	good	good	yes	very good on old films
	8000	7856.3	415	good	not altered	very good	very good	yes	

XviD

FORMAT AVI
AUDIO SETTINGS mp3, 48000 Hz, 2 ch, s16le, 128.0 kbit

	Enc settings	**bitrate** (kbp/s)	**size** (Mb)	**quality**	**colors**	**graphix**	**b/w noise**	**synch**	**notes**
Testtape Direct									
	200	643.2	41	bad, image often confused	saturated		pixelate and dirty	yes	shadows, ghosts
	300	1039.5	43	very pixelate and noisy, shadows	lightly too yellow	shadows around text elements if text is over video	pixelate and dirty	yes	
	600	1232.2	52	quite ok on colored parts	not altered		problems on old film res and noisy animation s	yes	bad res on b/w and grey-scales
	800	961.3	58	quite ok but there are imperfections	ok	ok	ok but ghosts on grey scales	yes	anim and noise fields pixelated
	1200	1629.1	70	ok, a bit pixelate on b/w fast movements	not altered		lightly pixelate	yes	quite undefined on old films
	4000	3998.7	215	ok	not altered	ok	lightly pixelate, good res on old films	yes	shadows in colored grey parts
	8000	7388.7	347	good	not altered	good	ok	yes	pixels on fast anim
Testtape Prefixer									
	200	614.7	40	image confused	too lighty and pixelated	ok	pixelated, shadows and ghosts	yes	
	300	653.8	42	pixelate	pixelated and lightly yellow	shadows around text elements	dirty and pixelate	yes	
	600	827.4	51	ok but pixelated on b/w fast movement	not altered		shadows and pixels, more on fast move	yes	
	800	944.9	57	ok but pixelated on b/w	ok	ok	pixelated	yes	shadows and ghosts
	1200	1202.2	70	ok	not altered			yes	pixels on b/w fast movement
	4000	3352.2	182	quite good	not altered	ok	pixels on noisy anim	yes	shadows in colored grey
	8000	4458.6	244	good	not altered	good	ok	yes	pixels on anim

Player benchmarks

The players considered within this table are **QuickTime**, **RealPlayer, Videolan**, **WindowsMedia** and **Mplayer**.

Playback tests were performed on Windows 2000 and Mac Os X 4.5.6.

Unfortunately the relation between codecs, formats, players and systems is still a mined field: compatibility is very difficult to achieve because of the combination of these four variables. Many open source multimedia playback applications have been specifically tuned to play all manner of multimedia files, while proprietary multimedia applications, such as Microsoft Windows Media Player and Apple's QuickTime Player, are more rigid in the types of files they will accept.

These results should not be read as a final general statement on codecs and players within the digital domain, but as a specific test case and a collection of informations in a certain determined environment.

Technical notes:

Xvid can be played by QuickTime on Mac OSX after installing a Plugin. This is not automatically found by the software updates, but it has to be specifically searched by the user.

Mplayer on Mac OSX is very easy to install, both the graphical version and the command line version.

The command line version can be installed using the command 'sudo port mplayer' if DarwinPort is installed and well configured, or it can be both 'fink' and 'apt-get' when fink is running on the system. The Fink Project works but it is lately not very up to date.

The ported versions of mplayer are not supporting Theora. Theora support is disabled in the configuration file flags, so it would be very easy to fix. Though it is possible to have this support installing specific versions like, for example, Mplayer-DEV-CVS-060409.

Ogg/Theora files can be played by QuickTime on MacOsx and by WMP on Windows after installing a plugin. (http://xiph.org/quicktime/download.html, http://www.illiminable.com/ogg/).

Player benchmark table:

system		PC				MAC			
player		**Qt**	**Real**	**vlc**	**wmp**	**Qt**	**Real**	**vlc**	**mplayer**
codec	format								
mpeg4	**avi**	N	N	Y	N	Y	Y	Y	Y
	mov	Y*	Y*	Y	N	Y*	Y*	Y	Y
	ogg	N	N	Y	N	N	N	Y	Y
theora	**avi**	N	N	Y	N	N	N	Y	Y
	mov	N	N	Y	N	N	N	Y	Y
	ogg	N	N	Y	Y#	Y#	N	Y	Y
h264	**avi**	N	N	Y	N	Y	Y	Y	Y
	mov	N	N	Y	N	N	N	Y	Y
	ogg	N	N	N	N	N	N	Y	Y
xvid	**avi**	N	Y+	Y	Y	Y	Y	Y	Y
	mov	N	N	Y	N	N	N	Y	Y
	ogg	N	N	Y	N	N	N	Y	Y

LEGENDA:

***** =The files encoded with mencoder are not compatible. This problem can be solved using ffmpeg, already included into the dyne distribution. Here the example command:

ffmpeg -i mpeg4_encoded_with_mencoder.mov mpeg4_passed_in_ffmpeg.mov

+ = Xvid on Real and WMP (Windows) requires 16 bits screen resolution (+). Videolan on the same platform can do it automatically.

 # = After installing a plugin only

Credits

This document is written by **Denis “Jaromil” Rojo**, benchmarks conducted and analyzed by **Eleonora Oreggia** in the Montevideo Artlab, with reviews and contributions by **Gaby Wijers**, **Ramon Coelho**, **Robert de Geus** and **Wiel Seuskens**.

This research is contributed by Montevideo to Medien Gestaltung / Kompetenz Netzwerk media.coop.

The operating system employed for encoding is entirely based on open source technology and freely redistributable (GNU GPL), developed by the dyne.org foundation.

This research would have never been possible without the existence of the **Free Sofware Movement**, a wide network of programmers distributed all across the world sharing knowledge and tools by the principles that the Free Software Foundation has set. Thanks to: Xiph Foundation, the FFmpeg and the Mplayer projects and all the open-source video communities for their help and support, the Piksel, Transmission.cc and Giss networks. Personal thanks to Silvano “kysucix” Galliani, Sami Kallinen, Dirk Stoop, Lluis Gomez y Bigorda and Valentina Messeri.

Encoder script

The script used in benchmarks is included below, it is written in shell script and can run on any GNU/Linux system. Encoders adopted are Mencoder and FFMpeg2Theora, codec libraries used are Lame, Ogg/Vorbis/Theora, X264, Xvid and FFMpeg.

The software used to realize the benchmarks are Mencoder and FFMpeg, all encoding has been done on a little-endian x86 computer taking advantage of MMX and SSE CPU instructions. The operating system used for encoding is dyne:II GNU/Linux, freely available online (GNU GPL) and optimized for multimedia tasks. An encoding script was programmed to simplify the usage of command line software employed, its source code can be found in the appendix section of this research.

```
#!/bin/sh
#
# Example script to encode video in various test formats for streaming
# by jaromil and xname
#

# commands used:

#### H.264 + MP3
# mencoder -oac mp3lame -lameopts    vbr=0:br=128:mode=0
#          -ovc x264    -x264encopts bitrate=$bitrate
#### XVID + MP3
# mencoder -oac mp3lame -lameopts    vbr=0:br=128:mode=0
#          -ovc xvid -xvidencopts bitrate=$bitrate
#### THEORA + VORBIS
# ffmpeg2theora --audiobitrate 128 --samplerate 44 -a 8
#               --videobitrate $bitrate --videoquality 8
#### MPEG4
# mencoder -oac mp3lame -lameopts  vbr=0:br=128:mode=0
#     -ovc lavc -lavcopts vcodec=mpeg4:vbitrate=$bitrate

if [ -z $1 ]; then
   echo "usage: $0 [ h264 | xvid | theora | mpeg4 ]"
   exit 1
fi

ORIGS=/Streamtestfiles/Originaltestfiles/

DEST_H264=/Streamtestfiles/h264/mp3
DEST_XVID=/Streamtestfiles/xvid/mp3
DEST_THEORA=/Streamtestfiles/theora/vorbis
DEST_MPEG4=/Streamtestfiles/mpeg4/mp3

mkdir -p $DEST_H264 $DEST_XVID $DEST_THEORA $DEST_MPEG4

BITRATES="200,300,600,800,1200,4000,8000"

# small useful function to iterate comma separated arrays:
iterate() {
    echo "$1" | awk '
    BEGIN { RS = "," }
          { print $0 }';
}

########################################################
## main ()

for bitrate in `iterate $BITRATES`; do

   for f in `ls $ORIGS`; do
```

```
      if [ `echo $@ | grep -i h264` ]; then

      ########################################################
      ## H264
      encoding="`echo ${f} | cut -d. -f1`_h264_$bitrate.avi"

      echo "(`date`) encoding $encoding"
      echo "using h.264/mp3 at bitrate $bitrate"
      echo "starting up in 3 seconds (press ctrl-c to cancel)"

      sleep 3

      mencoder -oac mp3lame -lameopts    vbr=0:br=128:mode=0  \
                        -ovc x264    -x264encopts bitrate=$bitrate     \
                        ${ORIGS}/${f} -o ${DEST_H264}/${encoding}
      ########################################################

      fi

      if [ `echo $@ | grep -i xvid` ]; then

      ########################################################
      ## XVID
      encoding="`echo ${f} | cut -d. -f1`_xvid_$bitrate.avi"

      echo "(`date`) encoding $encoding"
      echo "using xvid/mp3 at bitrate $bitrate"
      echo "starting up in 3 seconds (press ctrl-c to cancel)"

      sleep 3

      mencoder -oac mp3lame -lameopts    vbr=0:br=128:mode=0  \
               -ovc xvid    -xvidencopts bitrate=$bitrate     \
               ${ORIGS}/${f} -o ${DEST_XVID}/${encoding}
      ########################################################

      fi

      if [ `echo $@ | grep -i theora` ]; then

      ########################################################
      ## THEORA
      encoding="`echo ${f} | cut -d. -f1`_theora_$bitrate.ogg"

      echo "(`date`) encoding $encoding"
      echo "using theora/vorbis at bitrate $bitrate"
      echo "starting up in 3 seconds (press ctrl-c to cancel)"

      sleep 3

      ffmpeg2theora --audiobitrate 128 \
                    --videobitrate $bitrate \
                    ${ORIGS}/${f} -o ${DEST_THEORA}/${encoding}
      ########################################################
      fi

      if [ `echo $@ | grep -i mpeg4` ]; then

      ########################################################
      ## MPEG4
      encoding="`echo ${f} | cut -d. -f1`_mpeg4_$bitrate.avi"

      echo "(`date`) encoding $encoding"
      echo "using mpeg4/mp3 at bitrate $bitrate"
      echo "starting up in 3 seconds (press ctrl-c to cancel)"

      sleep 3


      mencoder -oac mp3lame -lameopts    vbr=0:br=128:mode=0  \
                        -ovc lavc    -lavcopts  vcodec=mpeg4:vbitrate=$bitrate     \
                        ${ORIGS}/${f} -o ${DEST_MPEG4}/${encoding}
      ########################################################

      fi

   done

done

## end
#########################################################################
```