MPEG-4 Tools and Applications: An Overview[1]

B. Lehane, N. O'Connor, N. Murphy
Centre for Digital Video Processing
Dublin City University
lehaneb@eeng.dcu.ie

Abstract

In this paper we present an overview of the software tools currently available for the creation and display of MPEG-4 content. We first describe tools for encoding raw video into MPEG-4 compliant bitstreams. We then describe how this content may be used to create a complete MPEG-4 scene containing both graphical and interactive elements in addition to the more usual video and audio elements. Clearly, MPEG-4 content cannot be viewed without appropriate decoders and players and these are addressed in the third section of this paper. Finally, we demonstrate how these tools may be combined together to create MPEG-4 applications by presenting the details of two sample applications we have developed.

Keywords: MPEG-4, BIFS, interactive content.

1 Introduction

As standardisation activity on the ISO/IEC MPEG-4 audio-visual compression standard [1] draws to a close, it remains to be seen whether or not the standard will be successful. Previous MPEG work items, MPEG-1 targeting video from a CD-ROM and MPEG-2 targeting digital television, have been extremely successful in the market place. Both have been adopted world-wide in many different variations of the originally targeted applications. However, in the case of MPEG-4, the niche markets for the standard are not yet clear. It is thought that one such niche market will be delivery of multimedia content to a new generation of mobile devices over next generation mobile networks. It is thought that another niche market will be content production for next generation interactive audio-visual services.

One crucial factor in the successful take-up of any audio-visual compression standard is the ready availability of hardware/software for content authoring (i.e. creation) and subsequent consumption (i.e. playback). Unlike previous standards, which often required dedicated hardware encoders, it appears that MPEG-4 (whatever its use) will be mainly implemented by software tools for both encoding and decoding. Such tools are now starting to become available and in this paper we present an overview of some of these. We demonstrate the usefulness of these tools in building MPEG-4 applications. This also serves to illustrate the types of applications where MPEG-4 may ultimately find a home for itself.

The remainder of this paper is organised as follows: Section 2 presents an overview of some popular MPEG-4 encoding tools; Section 3 discusses MPEG-4 systems issues and describes some system-level tools; Section 4 presents a list of the various decoders/players available; in Section 5 we describe two sample MPEG-4 applications we have developed using a subset of these tools.

2 Encoding Tools

MPEG-4 claims to be a "Create once, play anywhere" standard, and thus the creation of MPEG-4 content must be robust yet effective. It is vital that encoders can support a large range of options so that content creators can target specific delivery devices/mechanisms whilst still ensuring that the content can be played 'anywhere'. For example, if the target device were a Personal Digital Assistant (PDA) with the content delivered over a mobile network, then a low bit-rate and a small frame size would be appropriate. However, the content should not be limited to consumption on the PDA but should still be able to play on a desktop PC. Conversely, if the target device were a desktop PC, it should then be possible to take advantage of the higher processor power available and use higher bit/frame rates. Clearly, an MPEG-4 encoder should be able to create content to suit a broad range of applications. In the following, we describe some currently available software encoders.

[1] The support of the following is gratefully acknowledged: the Enterprise Ireland Informatics Initiative, the Irish Research Council for Science and Engineering Technology and the PRTLI-funded RINCE initiative.

2.1 Libfame

Libfame [2] is a Linux-based open-source library of tools that can be used for the purpose of MPEG-1 and MPEG-4 video encoding. There are two encoders that use this library, *fame* and *recmpeg*. *fame* (Fast Assembly MPEG Encoder) takes a YUV video input (from a video capture card) and, performs real-time MPEG-4 encoding. It is a flexible encoding tool that allows the user to specify the picture size, coding type (e.g. IPPI — B frames are not supported), the image quality (as a percentage), frames per second, motion estimation search range, and motion estimation method. The output is in the form of an MPEG-4 elementary stream. *fame* optionally uses the LAME MP3 (MPEG-1 Layer 3) encoder to produce audio tracks. The MPEG-4 video and MP3 audio tracks created by *fame* are produced as two separate output bitstreams. Since the output of the *fame* encoder is an elementary stream, generally it will need to be passed through the systems layer (using one of the tools described in Section 3) before it can be played. This requires multiplexing the audio and video files together to produce a synchronised MPEG-4 file. Using *fame*, it is also possible to stream the encoded content to an IP address. *recmpeg* is identical to *fame* in terms of its flexibility, but uses a local raw YUV video sequence as its input to produce an mpeg-4 video.

2.2 QuickTime™ Pro

Perhaps the easiest encoder to use, QuickTime™ Pro [3] accepts many different file formats, and can transcode to MPEG-4. As with other encoders, it can accept a 'raw' video file in the form of AVI (Audio-Video Interleaved), and create an MPEG-4 bitstream. The quality of the output depends on user-defined options such as bit rate, frame size and frames per second. It also includes an option to ensure compliance with as many MPEG-4 players as possible by creating a simplified MPEG-4 video bitstream. QuickTime™ Pro is extremely easy to use, and has the largest range of inputs and outputs of any of the encoders listed here. There is a huge range of formats supported including (but not restricted to): AVI, MPEG-1, MPEG-4, MOV, MP3, WAV, MIDI, JPEG, GIF, AU and BMP.

2.3 MPEG4IP

MPEG4IP [4] isn't an encoder as such; rather it is an open-source standards-based system for encoding, streaming, and playing MPEG-4 audio and video. It is based on many existing open source packages and is an attempt to combine them all together to offer a complete package that creates and plays MPEG-4 files.

MPEG4IP includes mp4encode, which converts AVI files directly to MPEG-4 video files. The format of this is similar to the *fame* encoder, as it is possible to specify the frame width/height, video and audio bit rate, and frames per second. mp4encode also comes with an MP3 encoder and an AAC (Advanced Audio Coding) encoder. It is also possible to encode audio separately to the video. Once the audio/video is created it is possible to use the other tools in the package to multiplex and encode as MPEG-4.

The Linux version of MPEG4IP comes with mp4live, this enables users to encode using a video capture card supported by the Video for Linux drivers. In a manner similar to the *fame* encoder, this means that it is possible to encode MPEG-4 content and stream it over a network.

2.4 DivX® / VirtualDub

The DivX® encoder [5] uses a different approach to the encoders we have seen so far. In order to use DivX® a processing utility such as VirtualDub is required. The processing utilities can act as a portal to many different encoders, and users can specify what encoding tool to use. For example, to encode an AVI file using DivX®, it is first required to pass the AVI file through the VirtualDub [6] program using DivX® as the encoder. Using a program like VirtualDub, it is possible to pick different encoders to use on a file and thus comparison of results is possible. It is even possible to view a sample output using the chosen encoding method before actually writing the file.

2.5 Conclusion

Due to MPEG-4`s high compression rates and broad range of applications, there are many encoders becoming available. Encoders such as *fame* and MPEG4IP are suitable for online processing as they allow encoding in real-time on the fly. On the other hand, software tools such as VirtualDub is more suited to off-line coding since they can be configured to use different encoders allowing comparison of different encoding qualities prior to final encoding.

3 Systems Tools

The capabilities of traditional video standards, such as MPEG-1 and MPEG-2, are limited to displaying synchronised audio and video streams in a 2-D rectangular plane. MPEG-4 however, adopts an object-oriented approach to creating a scene, whereby each element in the scene is defined as a node in a scene tree. This gives rise to a broad range of possibilities for MPEG-4 scene creation that simply didn't exist in previous video standards. For example, it is possible to have multiple video nodes in a single 3-D scene, thereby displaying more than one video elementary stream at the same time. This is achieved via a scene descriptor, which tells the decoder/player the structure of the scene to be reconstructed as well as defining the various components making up the scene. Backward compatibility with previous coding standards is ensured, since they correspond to a simplified MPEG-4 scene description. For example, MPEG-1 content with video and audio corresponds to a 2D MPEG-4 scene with two nodes: a video elementary stream, and an audio elementary stream.

The scene descriptor is based on Virtual Reality Mark-up Language (VRML) but the range of node types that can be used is not restricted to those supported by VRML. A wide range of nodes is supported, from arbitrary shapes (2D or 3D), to text, to still images, to natural audio/video. These nodes are designed to be multitasking, in that a text node not only displays text, but may also be interactive so that user actions can trigger events. For example, clicking on a video (or an object in a video) to go to a web page relating to the content. The MPEG-4 scene descriptor is known as BIFS (BInary Format for Scenes).

Systems tools are essential for generating complete MPEG-4 scenes. Not only do they attach timing information, as in traditional standards, but they also have the responsibility of composing all of the nodes in the scene descriptor into a meaningful display. A diagram of the entire process is shown in Figure 1.

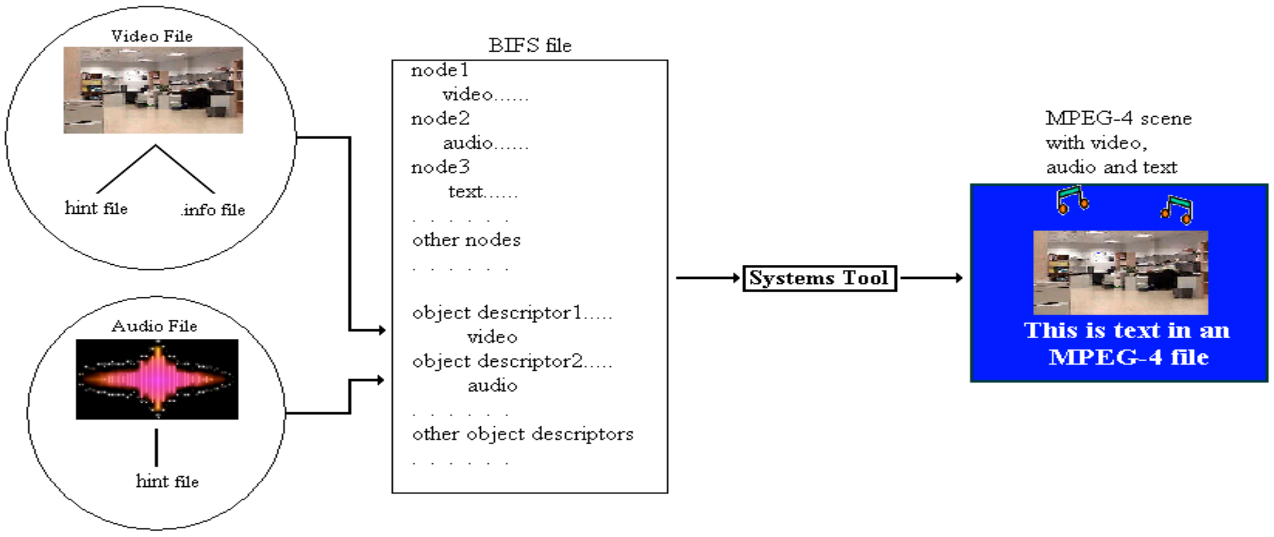

Figure 1: Process of creating an MPEG-4 scene using BIFS.

3.1 mp4tool

The mp4tool [7] takes its input in the form of a text file. This file contains both the scene descriptor information and the object descriptor information. This information is then encoded into an MPEG-4 systems file. This file is a way of linking all of the nodes in a scene together and combining them into a single scene. For example, it tells the decoder when to play a video stream, what size it is, and where it is located in the scene.

If the scene includes a video or audio stream, a hint file is also required. This hint file contains information regarding the temporal position of the access units[2] in a file, specifying the composition time stamp and decoding time stamp. It also specifies the distance in bytes between each access unit. This hint file is not part of the standard, but is an accepted file format used when the official MPEG-4 systems reference software was implemented.

[2] An access unit is the smallest data element in a stream to which timing information can be attached. For video and audio it corresponds to a frame.

As mentioned previously, MPEG-4 scenes provide more than simple audio/video presentation. The scene descriptor section of the input file contains information relating to other nodes in the scene. This is the method whereby a programmer/content creator designs the overall scene, specifying the spatio-temporal position of the nodes. Nodes consist of primitive components called fields, which contain all of the information related to the node (e.g. a text node will have *string* and *fontstyle* fields).

The object descriptors contained in the input file specify the properties of the various objects (video/audio) present in the scene. For example, a *MovieTexture* node will link to an object descriptor (OD), this OD will specify (among other things) the stream type (e.g. video), the resolution of the time stamps, and the filename. In this way the mp4tool can integrate the video into the complete scene. The output of this program is a single MPEG-4 BIFS bitstream which includes any required elementary streams.

mp4tool has many other options that can be very useful in the creation of MPEG-4 content. Typical usage takes an input file and produces an MPEG-4 file, but it is also possible to work in the opposite direction and extract the input file from an MPEG-4 file. It is also possible to obtain information from an MPEG-4 file, such as the Object type, size, duration etc. Extraction of elementary streams is also possible, so that a user could extract an original video stream from a scene involving multiple video/audio streams and other nodes. Apart from the fact that this can be extremely useful for data manipulation, it is also useful for creating new content as it is straightforward to re-use existing scene descriptors and/or objects for other similar scenes. Another useful feature is the JAVA-based GUI which has all of the features of the command line version. Finally, mp4edit is a light version of mp4tool, which contains some of the common features in mp4tool such as stream export and information retrieval on the scene.

3.2 MP4ENC/BIFSENC

The MP4ENC/BIFSENC combination of tools is very similar to the mp4tool described above. Instead of having a single text file input, these two programs take two text inputs corresponding to a scene file and a script file. The scene file contains all of the scene descriptor information, whereas the script file contains the object descriptor information. These two tools do not support as many options as mp4tool, although it is possible to extract video/audio tracks and their timing information from files containing complete scenes.

3.3 Conclusion

The systems tools outlined here can be used to author complete MPEG-4 content. The two tools described are quite similar in that both take input in the form of a scene descriptor and object descriptor. The scene descriptor is used to create the spatio-temporal representation of the scene, and the object descriptors are used to integrate the required objects.

4 Decoders and Players

Of all the MPEG-4 tools listed here, decoders and players are the most plentiful. Typically decoders simply decode the MPEG-4 file into an uncompressed format (e.g. YUV), whereas players decode the file and render it on the display device. Since MPEG-4 is such an all-encompassing standard, no decoder/player currently supports all features. For example, some players simply play a simple MPEG-4 scene containing only audio and video, and do not support any 2D or 3D graphical nodes. Conversely, a player targeting VRML-like 3D virtual worlds may not support any video/audio features at all. In this section we do not cover the entire set of MPEG-4 players/decoders available. Rather, we present a representative selection reflecting the capabilities of MPEG-4.

MPEG-4 specifies various profiles (and levels) that are used to categorize players and decoders. In order to claim compliance to a certain profile, the decoder/player must support certain visual tools. These range from supporting the decoding of an I-frame, to supporting 3D face body animation. We present only a subset of the most popular profiles here. Many of the profiles not shown contain tools related to advanced BIFS features, and very few (if any) players currently implement these.

Table 1 shows a subset of the profiles used to classify MPEG-4 decoders and players. It can be seen that if a player is compliant with the Core profile, then it is automatically compliant with the Simple profile, since Simple is a subset of Core. The full list of visual profiles and tools can be viewed in the MPEG-4 Visual document: ISO/IEC 14496-2 [8]. Of course, some players, which are not aiming to be compliant to a complete profile, may decide to implement some, but not all, of the specified tools in a certain profile. For example a player may support B-VOP`s[3] without implementing the rest of the tools in the core profile.

Visual Tools	**Simple**	**Core**	**Main**	**Advanced Simple**	**Basic Animation**
Basic[4]	X	X	X	X	
Error Resiliance[5]	X	X	X	X	
Short Header	X	X	X	X	
B-VOP		X	X	X	
Method 1 & 2 Quantisation		X	X	X	
P-VOP Scalability[6]		X	X	X	
Binary Shape		X	X		X
Grey Shape			X		
Interlace			X	X	
Sprite			X		
Scalable Still Texture					X
2D Dynamic Mesh					X
Global Motion Compensation				X	
Quarter-pel M-C				X	

Table 1: Subset of MPEG-4 visual profiles and tools.

4.1 Platform4

The Platform4 player from Philips supports the Simple profile from the visual standard. This means that it is capable of playing basic MPEG-4 Video, with error resilience and support for short headers. It also supports AAC audio and CELP. It is a well-integrated easy-to-use player that has all the typical controls one would associate with a video player (play, pause, loop etc) as well as options to display information about the file being played such as Bit rate, size, and type of video. It also comes with a version for a PDA running windows CE and supports streaming of content. [9]

4.2 DivX®

The DivX® [5] encoder mentioned in Section 2.4 is supported with its own player. DivX® supports the MPEG-4 simple profile, while DivX® pro supports the Advanced Simple profile. Since there is an associated encoder, the player can be used to play any content created using this (which may not be strictly MPEG-4). The player has many presentation options incorporated, so that it is possible to set the size of the display area, the brightness, the aspect area etc, as well as the usual timeline controls.

4.3 mpegable™

In a manner similar to the DivX® player, the mpegable™ player supports the visual Simple and Advanced Simple profiles. Its main target application is playback of video/audio scenes. Again, like DivX®, there is an encoder produced by the same company. It includes the usual timeline options as well as the ability to enable de-blocking and de-ringing post-processing filters. [10]

4.4 Osmo4

Osmo4, previously called the Osmose player, is a systems player based on the MPEG-4 Systems reference software. It is based on many open source packages including XviD for MPEG-4 video Simple profile. Just as the visual standard comes with a set of visual profiles, the systems standard comes with its own set of profiles, usually in the form of Graphics profiles and Scene Graph profiles. These specify what nodes in an MPEG-4 scene the player/decoder in question supports. Osmo4 is compliant with the Complete2D Scene Graph and Graphics profiles. This means that it supports all of the 2D nodes possible in a scene, but none of the 3D nodes, i.e. it supports

[3] B-VOP: Bi Directional Video Object Plane. Much like a B-Video Frame in traditional compression standards [8].
[4] Basic: This includes I-VOP, P-VOP, AC/DC prediction, 4-MV, and unrestricted MV.
[5] Error Resilience: This includes Slice resynchronisation, Data partitioning, and Reversible VLC.
[6] Scalability for both Rectangular and Arbitrary shape.

rectangle, circle, text etc, but not cylinder, cone, face etc. Perhaps one of the most interesting facets of this player is that it supports tools such as Touchsensor, which detect when the mouse has touched a certain part of the scene. This means that interactivity with the MPEG-4 scene is possible. In addition, the player has the usual array of timeline options associated with video players. [7]

4.5 EnvivioTV™

The EnvivioTV™ [11] video decoder is a plug-in to RealOne™ or QuickTime™. As such, any timeline controls depend on the chosen parent player. In a manner similar to the Osmo4 player, it supports the video Advanced Simple profile, and the Advanced2D graphics and Scene Graph profile. Using these profiles it is possible to play interactive video and graphical content.

4.6 Octagon

Compared to the first three players mentioned which concentrate on audio and video playback, the Octagaon player is at the opposite end of the MPEG-4 spectrum. It does not support any MPEG-4 audio/video codecs, but its main use is in VRML-based applications. It supports all VRML nodes, where the VRML node set is a subset of the MPEG-4 systems node set. It has also implemented a number of MPEG-4 nodes that did not exist in VRML that were deemed useful to content creators (e.g. some of the 2D nodes). Its strength lies primarily in the creation of MPEG-4 3D scenes, however the obvious drawback of this player is the lack of MPEG-4 video/audio support. [12]

4.7 Conclusion

This section briefly described six different MPEG-4 decoders/players. Three of these (Platform4, DivX® and mpegable™) are quite similar as their target application is simply to view compressed video and audio content. They offer no support for any graphical or scene graph profiles, and thus only take advantage of the efficient compression functionality of MPEG-4 and not additional MPEG-4 innovations (such as scene interactivity or graphics). Two players (Osmo4 and EnvivioTV™) facilitate the broader capabilities of MPEG-4 systems by supporting graphical and scene graph profiles as well as visual and audio content. These players provide support for all 2D graphical nodes implemented by MPEG-4, and make viewing of interactive MPEG-4 content possible. The final player, Octagon, is an MPEG-4 graphical player that does not support any MPEG-4 video or audio playback.

5 Example Applications

In order to demonstrate the capabilities of the MPEG-4 standard and illustrate the types of applications possible, in this section we describe two example application that we have developed using some of the tools described earlier. The first application demonstrates MPEG-4's suitability for streaming of audio-visual content to mobile devices. The second illustrates the new forms of interactive multimedia content made possible by MPEG-4.

5.1 MPEG4 streaming to a mobile device

Since in this application the content is to be streamed to a PDA, the bit-rate, frame rate, and frame size must be quite small, while still maintaining a good level of picture quality. In this sample application the PDA used is a Compaq iPAQ hand-held PC and a mobile connection is simulated with a wireless LAN (WLAN). The system is illustrated in Figure 2.

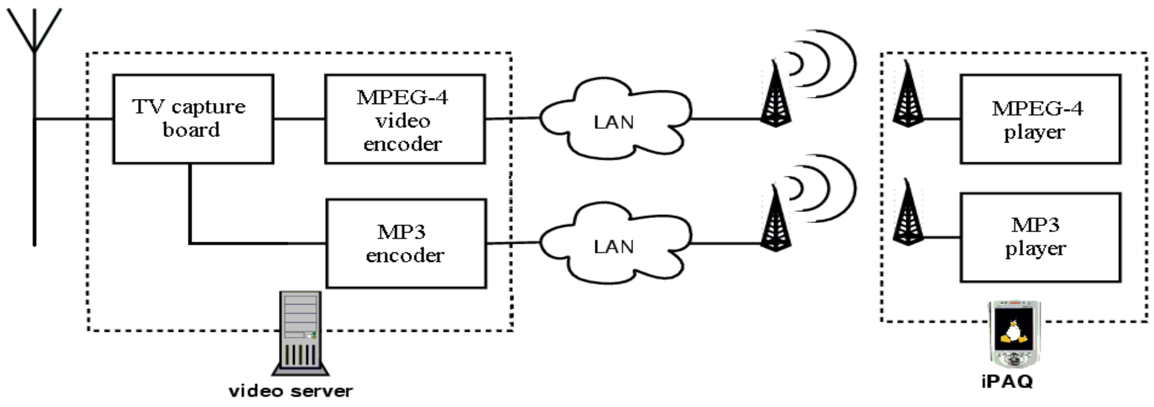

Figure 2: setup of MPEG-4 streaming to iPAQ.

In order to create the video content, the *fame* encoder is used, as it supports a wide range of bit-rates and subsequent streaming. A frame size of 320x240 was chosen (full screen for the iPAQ), with a frame rate of 12.5Hz, a quality of 70%, and a coding sequence of IPPPP. Video input is supplied by a TV capture card. The LAME plug-in to *fame* was used to encode the audio in mp3 format.

Once the content is created it is sent, via a wireless LAN, in two separate streams (one audio and one video) to the iPAQ, which is running the Linux operating system. For decoding purposes, the decode library from the OpenDivX project is used in conjunction with the madplay audio decoder. All encoding/decoding is carried out in real time, thus it is possible to watch live video content (e.g. from a broadcaster or a surveillance camera) on the iPAQ. Since in this sample application we use separate audio-video streams with no system-level multiplexing, the system is prone to synchronisation errors between the video and audio.

5.2 Interactive audio-visual content

In a second sample application, we create a 2D environment with many different nodes, including video, audio, JPEG, and 2D graphics. The objective is to demonstrate the interactive capabilities of MPEG-4 by implementing 'clickable' nodes. For this purpose, a web-page style application was created whereby users can visit different 'pages' simply by clicking on links. Whilst this appears as a simple web page to users (that could be implemented with straightforward HTML), in reality it corresponds to an MPEG-4 scene with many different nodes.

Figure 3 shows one of the pages in this application. The Osmo4 player is used to display the content as it supports 2D scene nodes, and audio/visual content. Once opened it is possible to click on any of the links on the left hand side of the screen to go to other MPEG-4 files, which in this case are stored locally. These links are actually a rectangle node with a text node overlaid. Linking to other MPEG-4 files is achieved by means of the Anchor node. At the top of the page there is a JPEG image of the CDVP logo which if selected opens a web browser that directs users to the CDVP webpage. Similarly, the colour-changing graphic on the left opens the CDVP web page. Although it is impossible to show in a still image, this graphic gradually changes colour – the background fades from red to blue to green, whilst the text in the graphic fades in the opposite direction. The MPEG-4 videos show people walking around the CDVP lab. These videos pan from left to right and are looped throughout the scene. It is also possible to access the CDVP website by clicking on the videos.

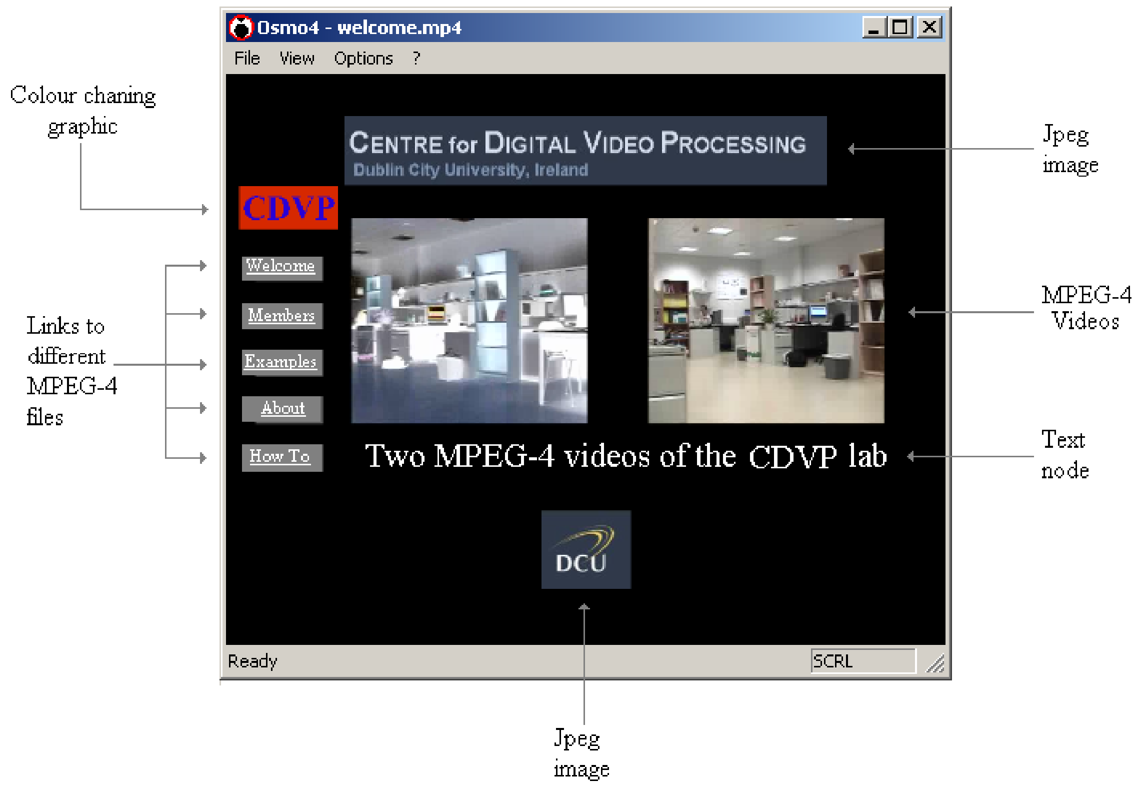

Figure 3: Welcome page of MPEG-4 interactive content.

This scene (which is only one of the scenes in this 'web-page') demonstrate the flexibility of MPEG-4; it contains two video nodes, two JPEG images, various text and rectangle nodes, as well as (among others) ColorInterpolator, timeSensor, touchsensor, and anchor nodes. All of this content is stored in a single MPEG-4 file. Note that for an MPEG-4 file like this, there is no start-time or finish-time as such, since the behaviour of the scene depends on user input. Other scenes in the application demonstrate different MPEG-4 functionality. For example users can reveal 'hidden' videos or images by moving the mouse over certain areas of the screen, or can click on a video so that it opens in its own player.

In order to create this application, elementary video and audio streams were created using the *fame* and LAME encoders. We then manually created a scene descriptor text file and used mp4tool to create the final MPEG-4 file.

6 Conclusion

In this paper, we presented a selection of the software tools currently available for creating and viewing MPEG-4 content and presented some example applications developed using these tools. This illustrates the broad scope of the MPEG-4 standard since the target applications have very different objectives. It also serves to illustrate the usefulness of the currently available MPEG-4 software tools. For example, the *fame* encoder is used for both applications, even though the first involves creating low bit rate content with a small frame size whilst the second requires high bit rate with large frame sizes. We also illustrated that using MPEG-4 it is possible to create interactive audio-visual content containing different types of content – e.g. mixed natural/synthetic visual content.

7 References

[1] ISO/IEC 14496-1:2001.1 Information technology coding of audio-visual objects: Part 1: Systems.

[2] Overview of the *fame* library, viewable at: http://fame.sourceforge.net/

[3] QuickTime™ data sheet L19086B_QT6_DS.pdf, viewable at: http://www.apple.com/quicktime/

[4] MPEG4ip guide, viewable at: http://mpeg4ip.sourceforge.net/

[5] The official DivX® guide, viewable at: http://www.divx.com/

[6] VirtualDub home Page: http://www.virtualdub.org/

[7] Website of MPEG-4 @ ENST, http://www.comelec.enst.fr/~dufourd/mpeg-4/index.html

[8] ISO/IEC 14496-2:2000.1 Information technology coding of audio-visual objects: Part 2: Visual.

[9] Website of Philips digital networks, http://www.digitalnetworks.philips.com

[10] Website of mpegable™, http://www.mpegable.com

[11] etv_021217.pdf, EnvivioTV™ data sheet available from: http://www.envivio.com

[12] Website of Octagon player, http://www.octaga.com/