Ogg Theora a Free Video Codec and Multimedia Platform

Ralph Giles

Xiph.org Foundation, Vancouver Canada
`giles@xiph.org`
`http://theora.org/` and `http://xiph.org/`

Abstract. $\hat{p}$
Ogg Theora is a compression format for digital distribution of film and video. It is technically competitive with other available formats, builds on existing open source software, and provides a truly free alternative to the established non-free and proprietary codecs. This is a technical overview of the format and how the compression works. Interleaving Theora with audio and other codec data is covered. Issues around seeking and playback synchronization of multiple data streams are also discussed. The author hopes this will provide insight on some of the practical issues associated with video software in general as well a useful introduction to the Ogg Theora format.

Keywords: compression, video, digital distribution, multimedia.

1 Video Compression

1.1 Introduction

Ogg Theora is a format for distribution of digital video. It offers compression, good bit rate scalability, support for both seeking and streaming playback, and detection of packet loss. Not only is an open source reference implementation available, but Theora is the only modern video codec that is freely-implementable without licence or royalty fees on implementations, encoded content, or the specification itself. As such it is the best option for multimedia content in the context of Free Software, and the best option for commercial distribution of multimedia content. It levels the playing field by providing commodity technology, allowing anyone to distribute film and video in digital form, and by creating a broader market for art.

This paper will serve as a technical overview of the Ogg Theora format. Before we discuss the format in general, however, it is helpful to first understand how digital video works, and how it can be compressed.

1.2 Video

Ever since the invention of the motion pictures at the end of the 19th century, the moving image has held a fascination for us, creating in film and television an entirely new art form. The development of analog television in the mid-20th century allowed the electronic transmission of moving images for the first time. Now, at the beginning of the 21st century, digital technology allows us to capture, edit, process and distribute moving images more easily than ever before. But how exactly do we go about representing such a thing in a computer?

All devices for recording moving images have relied on a trick of perception. Rather than capture motion itself, the motion is sampled by taking a succession of still images. When those images are played back in sufficiently rapid succession, the brain interpolates between the changes, recreating the perception of motion. Thus even in analog motion picture technology, the temporal data is already sampled digitally.

So we need only represent a sequence of still images within the computer. Fortunately, we've been handling still images digitally for some time. Typically digital representation involves sampling the colour of the image discretely, generating a two dimensional array. Each element of the array can be a single intensity in the case of black-and-white images, or a vector of three or more values for colour images, each value representing the intensity of the sample within a range of wavelengths, like red, green, and blue. Such an array representation is called a raster image.

Thus, we can represent a motion picture within the computer as a temporal array of raster images, and play it back by displaying them one after another at the appropriate rate.

Representing a motion picture as a sequence of raster images seems straightforward, and it is. Unfortunately, digital film and video is at the top of a hierarchy of media formats in terms of storage requirements. Text is the easiest medium to represent digitally, followed by still images, audio, and then finally video and film, in terms of raw storage required. And while current computers can store an almost inconceivable amount of text, or thousands of still images, they are only now reaching a point where uncompressed audio is something that can be stored any context beyond 'elements of the current project'.

Let's look at the storage requirements of a motion picture. A novel stored as text is about 1 MB. Uncompressed CD-quality audio is 600 MB per hour. A single still image at PAL resolution (768x576) stored in typical 24-bit RGB colour is 1.27 MB. The minimum playback rate to create the effect of smooth motion for most content is around 24 frames per second; PAL uses 25. That's 32 MB/s for video, or 111 GB per hour. And while one can certainly reduce the dimensions and rate of the image frames, PAL (standard definition video) is at the low end of motion picture formats. High definition video at 1920x1080x60fps would be 1.2 TB per hour. Analog film formats are capable of 10 times that resolution.

1.3 Compression

So, there's plenty of room at the top. Fortunately, computers don't just have storage, they have, well, computational ability, and we can trade some of that computational ability for storage space by compressing the data. And, since hard drives haven't always been the size they are today, the need to compress digital media is not new.

Compression is itself a wide field, and it is useful to understand the basics if one is interested is working on media compression.

All compression algorithms work by exploiting redundancy in the uncompressed bit stream. General purpose compression like the deflate[5] algorithm employed by `gzip`, generally work by replacing repeating elements with a shorthand, and coding those shorthands efficiently, using a number of bits inversely proportional to their frequency in the compressed bit stream. The original file can be recovered by just replacing all the shorthands with their full recorded values.

This technique works fairly well, and is the basis for more advanced efforts. The next most obvious trick is to try and re-arrange the data so that there are more repeated elements, or longer runs of fewer elements. This will result in a smaller file after general purpose compression. For example, instead of compressing the colour of each sampled element in an image, one might compress the difference from the previous element, either horizontally or vertically. The PNG[6] image format uses techniques like this to achieve up to a factor of 5 reduction in file size over generic compression. In general, one can use knowledge about the type of data being compressed to optimize its compression, whether it's audio, video, 3d models, or whatever.

So far, all of the compression techniques we've discussed allow perfect reconstruction. They are known as *lossless* algorithms. Normally, that's what you want; you don't want to lose your original file when you compress it, or lose the infinite replicability for which digital media are known. However, lossless compression rarely achieves more than a factor of 10 with natural still images, so we're still a long way from being able to stream live video over a home broadband connection, or fit a film on an optical disc. It's time to exploit human perception again.

It turns out that while a marvel of engineering, and often exquisitely sensitive, the human eye is not equally sensitive to all elements of an image. It is more sensitive to information at low spacial frequencies, and that sensitivity is lower for intensity information than it is for colour. If we're are willing to discard some of the original information, we can find *lossy* compression algorithms that result in a reconstruction that is an acceptable reproduction, or even perceptually identical to the original, but using far fewer bits. And there are many cases where this is a reasonable approach, such as distribution of a finished work for viewing, either on line or on disc.

The widespread JPEG[7] image compression format works this way. Unlike PNG it is a *lossy* format, that is it does not allow perfect reconstruction of the original image data. However, while PNG rarely provides compression factors over 10x for natural images, JPEG routinely provides 10-20x with no obvious loss of quality.

Now, since a motion picture is just a sequence of still images, we could compress it by, for example, just applying JPEG compression to each frame. And indeed people do. This format, generally known as *motion jpeg*, is what the so-called movie mode of many digital still cameras produce.

However, compression is about exploiting redundancy in the input data, and compressing each frame of a motion picture ignores one of the major correlations: often successive frames are very similar. Only some of the picture may change from frame to frame, or when things move, they move coherently, like the background during a pan. Even a foreground object moving across the frame will be substantially similar from frame to frame, even if it occurs in a different place. By exploiting the redundancy between frames, current lossy video codecs can provide compression ratios from 50:1 to 400:1 with useful quality.

2 The Theora Video codec

Theora is a format for lossy compression of motion pictures[1], and it takes advantage of all the techniques we've discussed above, and more.

Like the MPEG-1 and MPEG-2 video formats, Theora is based on the block discrete cosine transform (DCT) method. In fact, it is substantially similar in approach to those codecs, however many of the details are different, particularly in how the information is ordered in the compressed bit stream. These changes account for both the efficiency improvements Theora offers and the differentiation that avoids the patent encumbrances associated with the MPEG codecs.

The DCT approach was originally pioneered in the JPEG image format. It works by dividing the image frame into blocks, and then applying the DCT, a discrete analogue of the Fourier Transform, to each block. High spatial frequency information, which the eye is less sensitive to, can then be discarded by just not encoding the higher-frequency DCT coefficients.

In practice, rather than actually discarding coefficients, Theora uses a variable-resolution *quantizer* to record the higher frequency coefficient values at lower resolution. This helps compression both because lower resolution means fewer bits to store, and because it increases the likelihood of runs of the same value, which are also stored as such.

Most video codecs, including Theora, also borrow a trick from JPEG, which it borrows from analog broadcast television. Because the eye is more sensitive to intensity changes than changes in colour, analog television uses less bandwidth to transmit the colour information. Intensity information was already transmitted separately from colour for compatibility with black and white televisions.

The digital version of this technique is to multiply each RGB colour vector element in the image by a matrix, rotating to a similar set of so-called YUV coordinates, where the intensity information is concentrated in a single component. The two remaining colour components can be stored at a lower resolution, for example by discarding 3 out of every 4. Except where there are sharp changes in intensity and colour, the difference is hardly noticeable. Because of its history in analog television, many consumer cameras perform this step in hardware. This is a primary difference between consumer and professional video cameras.

Coherence between frames is exploited using what are called *motion vectors.* For each block in the image, in addition to providing a (truncated) set of DCT coefficients, directions can be included to copy the contents of the block from so offset location in either the previous frame, or the last frame that was encoded in its entirety without motion vectors. Finally, blocks can be left completely unchanged from the previous frame. In this way, objects can be replicated from one frame to the next, even if they're moving, and only the changes need to be explicitly coded.

This approach to inter-frame coherency is also similar to what the MPEG-1 and MPEG-2 video codecs do. However it is simpler in that there are no forward motion vector references ('B' frames).

While it shares many techniques with the MPEG family of codecs, Theora also has some new ideas that both improve compression features, and avoid patent restrictions. For example, the DCT transform blocks in the image are accessed along the recursive (fractal) *Hilbert curve* instead of the standard raster row order. This keeps more blocks that are adjacent in the image together in the coded bit stream, which again makes it easier to exploit redundancy and encode their contents more efficiently. Also, when the list of DCT coefficients ends (which is often before the maximum length of 64 since high frequency information is being discarded) the extra zeros on the end can be skipped. Better still, runs of DCT blocks that terminate at the same number of coefficients can be implicitly marked as done in a group, saving bits.

Finally, Theora includes some advanced features, like per-block selection of quantization matrices that, while not exercised by the beta reference encoder, will allow for significant improvements in efficiency in the future, allowing the format to stay competitive with newer codecs.

3 Ogg and Multiplexing

Multimedia formats are often divided into two layers. There is the actual compression format for a type of data, such as is done by the Theora codec discussed in this paper. But there is also a container format. This wraps the raw compressed bit stream into a file format suitable for storage or streaming and may offer various features such as seeking signposts, or interleaving with other compressed streams, such as audio tracks, or even alternate video tracks.

The Xiph.org Foundation has developed the Ogg container. format[2] While compressed data from Theora and our other free codecs can be embedded in other containers, such as Quicktime or even an MPEG program stream, we feel Ogg provides a nice balance between the complexity of a general format like Quicktime and the severe limitations of formats like AVI. Ogg is best known as the native format ('.ogg') for Ogg Vorbis[3] audio files, but it was intended from the beginning to be a generic multimedia container. Thus we have not just the Theora video codec, but the Ogg Theora file and stream format.

While Ogg makes a fine format for file system storage of multimedia data, its real design goal was streaming. To that end, it is designed so that it can always be produced in a single pass, suitable for live production. Another nice feature is that different streams can be *chained* together by simple concatenation. This makes dealing the Ogg in the context of network streams or classical unix filter tools quite straightforward.

It also means that features like seek tables which require multiple write passes are not included as a basic feature of the format. Some people balk at this, but Ogg does provide frequent seeking signposts, and binary search using those timestamps is of similar code complexity to a seek table implementation, and for arbitrary seeks implies only a small performance penalty. (Unless a seek table is very large, it only saves the first one or two steps of a binary search for the seek point in a variable bit rate stream.) We feel this is a worthwhile trade off for the simplification it offers.

Seek tables (such as for video 'chapters') can of course be included as a separate data format in the container when 2 pass encoding is possible.

Multiple codec streams can be interleaved in Ogg with a bounded overhead in both bit rate and latency, typically a 1-2in size over raw packet data. Corruption and dropouts are detected as part of the framing process, and truncated files generally play correctly up to the end of available data.

4 A complete video format

So, there are really three levels of Theora. There is Theora, the codec, which handles compression of raw video frames. There is a mapping of compressed Theora data packets into a container, such as Ogg or RTP or Quicktime. And there is the 'Ogg Theora' file/stream format, which is a profile for decoders including at its simplest an Theora compressed video interleaved with Vorbis compressed audio, offering parity with most digital video formats.

However, the flexibility of Ogg lets us extend Ogg Theora with the inclusion of multiple audio and video tracks, for multiple language and commentary track support like DVD-Video. Voice-oriented audio can be included more efficiently by using the free Speex codec instead of Vorbis. Subtitles can be included in multiple languages in text format, or as graphical overlays using an Ogg encapsulation of the MNG animation format.[4] This provides everything the DVD-Video format supports (MNG animation is good for karaoke too!) except for menus in a format that is both more efficient, and implementable without royalty or license fee, or even having to pay for the documentation.

We hope you'll adopt Ogg Theora as the format of choice for your work, and help us build a comprehensive set of tools to support its myriad features.

References

1. Xiph.org Foundation, *Theora I Specification*, **http://theora.org/doc/Theora_I_spec.pdf** (2004)
2. S. Pfeiffer, *The Ogg Encapsulation Format Version 0*, RFC **3533** (May 2003)
3. Christopher Montgomery, *Vorbis I specification*, **http://xiph.org/ogg/vorbis/doc/** (2002-2004)
4. Glenn Randers-Pehrson, ed., *MNG (Multiple-image Network Graphics) Format Version 1.0*, **http://www.libpng.org/pub/mng/spec/** (2001)
5. L. Peter Deutsch, *DEFLATE Compressed Data Format Specification version 1.3* RFC **1951** (May 1996)
6. Glenn Randers-Pehrson and Thomas Boutell, ed., *PNG (Portable Network Graphics) Specification, Version 1.2*, **http://libpng.org/pub/png/spec/1.2/** (1999)
7. JPEG Committee. The JPEG spec is variously known as ISO/IEC IS 10918-1 and ITU Recommendation T.81. It is not publicly available, but the Independent JPEG Group provides a reference library for the freely-implementable baseline format. **http://jpeg.org/jpeg/ http://ijg.org/** (1986/1994)