JIF - CREATING A BETTER ANIMATED GIF

Akhil Raju

MIT

ABSTRACT

GIF's are ubiquitous on the Web, but they are antiquated and cannot keep up with the growing demand of short, looping video clips. In this paper, I propose a new image format, JIF, that allows for higher compression and color-flexibility than GIF. JIF utilizes 4:2:0 chroma-subsampling, multi-channel encoding, differential pulse code modulation, and Laplacian coding, a flavor of Huffman encoding that eliminates the need to transmit the Huffman table. JIF outperforms GIF for a variety of clips, giving an average of approximately 5 percent better compression.

***Index Terms*—** GIF, Multi-channel coding, Huffman coding

1. INTRODUCTION

1.1. Motivation

The GIF file format was invented in 1987, but it still dominates the Internet today. Until just recently, it was the second most used image format on the Web [1]. The format offers an easy method for people to share short, looping "video clips," and the ubiquity of these clips have helped drive the popularity of many social media sites like Tumblr and Reddit.

However, GIF's are notorious for their inefficient compression and, thus, inherent low-quality. Bandwidth limitations, especially on mobile devices, discourage the use of high resolution GIF's, since they take a long time to transmit. The use of GIF's has evolved from its initial low frame-rate, low-resolution, simple animation usage back in the 1990's, but unfortunately the technology has not evolved with it.

There are many efforts to overcome the shortfalls of GIF. Most notable is the website gfycat.com, which converts GIF's to HTML5 video, taking advantage of the more efficient compression of video in order to drastically reduce the file size and allow for higher quality clips [2]. Essentially, the higher compression gives more work to the browser to decompress the animated clip, and for the modern Internet, with significantly more powerful browsers than previous decades, that is a valid tradeoff.

My goal for this project was to take a somewhat similar approach in order to create JIF, a new file format that yields better quality for animated clips.

1.2. GIF

GIF's coding standard is geared towards simplicity in while decoding. It uses little inter-frame compression and uses LZW coding on each frame. It restricts the color palette to 256 usable colors for each frame. GIF is regarded as a "lossless" image format, but the restricted color palette corresponds to a significant amount of loss when converting video clips to animated GIF's, which is one of their major uses today.

This project, and the proposed file format JIF, aims to attain a higher level of inter-frame compression so as to reach the same compressed size as a GIF while freeing the color palette of the frames. Though the format will be lossy, it will be closer to the ground-truth signal.

2. METHODS

I tested several approaches to attain the desired level of compression and quality, and these approaches varied in terms of their image representation, quantization, and codeword assignment. Each approach, however, served as a building block for the next. Section 3 outlines the coding tools of each these different approaches.

For a given approach, I built JIF's from a bank of JPEG images, which I used as my ground-truth signal. The JPEG images were frames from various video clips, and they represented a broad selection of typical clips that might be converted to a GIF for sharing. I used a total of 5 different clips to test each approach, and their content varied from home-made videos to cartoon animations to high-quality movie scenes.

For each JIF, its PSNR and SSIM index were calculated and compared to the GIF constructed from the same set of JPEG images.

3. APPROACHES

3.1. Approach 1: 4:2:0 Subsampling, DPCM, Fixed-Length Coding

This approach converted the frames to the YCbCr domain and subsampled the Cb and Cr channels by a factor of 2. It then used differential pulse code modulation (DPCM) to encode on the differences between frames when quantizing the im-

age channels. A fixed-length codeword was assigned to each pixel.

All the following approaches implemented the same 4:2:0 subsampling scheme.

3.2. Approach 2: 3D-DCT, Fixed-Length Coding

This approach still converted the image to the YCbCr domain and sub-sampled the chroma channels. It then, however, calculated the block-wise three-dimensional DCT (the third dimension being time), and encoded only the highest magnitude coefficients. A uniform quantization and fixed-length coding were utilized.

3.3. Approach 3: Per-pixel 2-Channel Coding, DPCM, Fixed-Length Coding

This approach applied 2-channel coding at the pixel level but in the time domain. For each pixel, its value was extracted for each frame in order to create an vector of its values. The vector was passed through a 1-D 8th-order FIR low-pass filter, whose output was subsampled by a factor of 4 (that number could change) before being quantized and coded with DPCM and a fixed-length coding scheme. The subsampled, quantized vector was interpolated and subtracted from the original image in order to give the high-pass component, which was quantized and coded similarly.

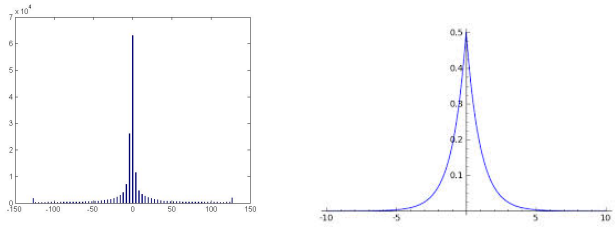
The motivation for this approach came from the low-variability in many animated GIF's. There are not many scene changes, and the value of a pixel at one time step well predicts its value at the next. In essence, the DPCM-quantized version of each pixel's time-vector is a relatively sparse vector. For this reason, the low-pass component can be significantly subsampled.

This approach also can be run easily in parallel. The encoding and decoding per pixel do not depend at all on other pixels, allowing the processing per pixel to happen in parallel. This leads to significant gains in encode and decode times.

Unfortunately, this approach yielded strange artifacts in the encoded images. The time-based coding left residual information of previous or future frames in the current one. For instance, the "ghosts" of past objects in a clip would remain in the subsequent frames. Also, the final frame of a clip tended to amass a lot of residual information and be unrecognizable in terms of the original frame.

3.4. Approach 4: Per-frame 2-channel Coding, Fixed-length Coding

This approach applied the same 2-channel compression scheme as the approach in 3.3, but does so in the spacial domain, per frame, rather than the time domain. It utilizes a 2-D low-pass Gaussian filter and subsamples the low-pass component by a factor of 4 (that number could change) in both the horizontal and vertical directions. It used uniform quantization and fixed-length coding.



(a) Quantized Pixel Distribution (b) Laplacian Distribution

Fig. 1. The quantized pixel distribution typically fits a Laplacian distribution.

Many GIF's are relatively simple with not many details, making this approach capable of significant compression since it can significantly subsample the low-pass component and devote few bits for the high-pass components.

3.5. Approach 5: DPCM, Laplacian Coding

This approach is analogous to the approach outline in 3.1, except it uses a flavor of Huffman coding called Laplacian coding, which I designed for this project.

Laplacian coding differs from general Huffman coding only in that the distribution of values is known beforehand, so the dictionary does not need to be transmitted along with the encoded signal. The distribution in this case is the Laplacian distribution. The DPCM-quantized pixel values in the Y, Cb, and Cr channels tended to fit a Laplacian distribution, as shown in Figure 1(most likely due to the relatively low-variability from frame to frame in many animated clips), whose PDF is given by

$$p(x) = \frac{1}{2b} \exp(-\frac{|x-\mu|}{b}).$$

The different channels and different images simply vary in terms of which Laplacian parameter b is used for the distribution, and $\mu = 0$ for most cases. Once the probabilities are known, either the encoder or the decoder can build the Huffman table. Thus, the encoder needs to only transmit the necessary Laplacian parameters, rather than transmitting the entire table. A search through some fixed values can yield the optimal Laplacian parameter for a given set of pixels.

This approach used Laplacian coding on each entire channel, but a natural extension is a more adaptive coding, in which a new distribution is fit for each frame. The encoder needs to find the optimal Laplacian parameter for each frame, and it simply transmits that single parameter for each frame. A better fit distribution yields better coding efficiency.

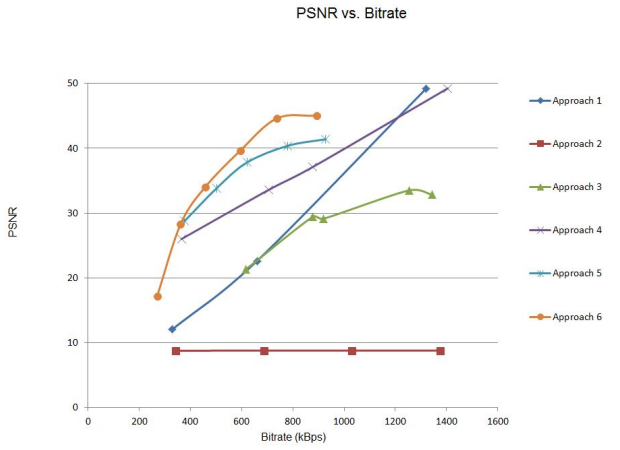


Fig. 2. The PSNR vs. Bitrate for the various JIF coding approaches.

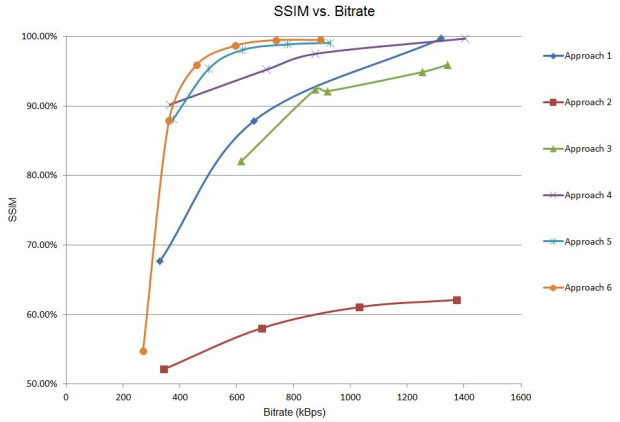


Fig. 3. The SSIM vs. Bitrate for the various JIF coding approaches.

3.6. Approach 6: Per-frame 2-channel Coding, DPCM, Laplacian Coding

This approach is a combination of the approaches found in 3.4 and 3.5. The different channels are multi-channel encoded, but instead of uniform quantization and fixed-length coding, it utilizes DPCM and Laplacian coding. Once again, the optimal parameters are found with a linear search, and they are transmitted instead of the entire table. The decoder uses the Laplacian parameter to calculate the probabilities of each value and construct a Huffman table.

This final approach yielded the best results: highest level of compression without any strange time-related artifacts. More discussion on its success follow in Section 4;

4. ANALYSIS

4.1. Analysis of Approaches

4.1.1. Successes

Figures 2 and 3 show the performance of the different JIF coding standards. These measurements were taken specifically for a home-made clip of a boy dunking a basketball. This clips short length and relatively low-quality make it a great example of the typical video clips that GIF's are used for. As shown, Approach 6, which is detailed in Section 3.6, outperforms the rest of the approaches.

Interestingly, though, its performance is only marginally better that Approach 5, which is detailed in Section 3.5. However, Approach 5 significantly outperforms Approach 1, which is what Approach 5 was based on. Thus, we can see that Laplacian coding makes a very significant difference and a much bigger impact than the multi-channel coding, though that yields gain as well. At a fixed bit-rate, the switch from fixed-length coding to Laplacian coding increased the quality, as measured by the SSIM index, by about 10 percent (calculated by averaging the performance gains from Approach 1 to 5 and from Approach 4 to 6). On the other hand, at a fixed-bit rate, the switch to two-channel coding only gave approximately a 6 percent gain in quality.

Still, the combination of Laplacian coding and two-channel coding in the spatial domain allow for the best efficiency. This is due to the relatively low variability of the scenes from frame to frame. Each frame well predicts the content of the next frame, so performing DPCM allows the distribution of the quantized values to tightly fit around 0.

4.1.2. Failures

Certain approaches did not work very well, on the other hand. The two lowest performing approaches, Approach 2 and 3, detailed in Sections 3.2 and 3.3, both tried to do significant compression in the time domain, making use of the fact that the frames do not change much from frame to frame. Unfortunately, this approach did not work. For the case of Approach 2, which attempted to do DCT compression in three dimensions, both the quantization step and the filtering of low-magnitude DCT coefficients caused very blocky, time-related artifacts throughout the JIF. Quantizing and throwing away time-related DCT coefficients proved to be a futile approach.

As mentioned in Section 3.3, the two-channel coding applied to the time domain suffered similar, albeit less drastic, issues. Time-related ghost artifacts popped up throughout the JIF's, and sometimes, perhaps due to the coding in the YCbCr domain, the colors would not catch up with moving objects in the clip.

Image Sequence	GIF size (kB)	JIF size (kB)	Compression Gain
Joker	3721	3037	18.3%
Dunking	1783	1721	3.5%
Scientist	2455	2431	0.9%
Segway	1977	2114	-6.9%
Drive	16614	15321	7.8%

Fig. 4. The compression gains of JIF vs. GIF for several test clips, held at a constant quality.

4.2. Comparison to GIF

The JIF format provided some increased compression. Figure 4 shows the gains in compression for the various test clips at a fixed level of quality. As shown, most of the clips are better compressed using the JIF format than the GIF format. Added to this benefit is the fact that all of the JIF images are better compressed with a higher flexibility in terms of colors. This is especially captured with the two clips that come from movie scenes. The ”Joker” clip and the clip from the movie ”Drive” both contain a wide-range of colors that GIF simply cannot support, so its quality declines. JIF, on the other hand, can support that breadth of colors. This is why we see large gains in both of those clips.

On the flip side, for an image that contains few colors and very minimal animations, the GIF format outperforms JIF. This is shown in the ”Segway” clip, a cartoon clip with minimalistic animations and a limited color palette. This clip represents a sequence that was designed for GIF, i.e. it was not pulled from a pre-existing video clip. The ”Scientist” clip is similar in that it is mostly comprised of a plain white background and few movements. It makes sense that in such instances, GIF can still provide as good or better compression than JIF.

5. CONCLUSION

The JIF format was a partial success. The encoding pipeline that included compression tools like 4:2:0 subsampling, multi-channel coding, DPCM, and Laplacian coding yielded better compression than GIF for a variety of clips and offers the freedom to no restricted color palette, allowing it to perform very well for some varieties of clips.

On the flip side, for some other clips it provided little to no gain. It’s coding standard, while not as complex as video coding, is more complex than GIF’s standard, which might make it harder to adopt or use.

Overall however, it provides a new-found flexibility with clips, and it does not have a quality ceiling, like GIF, which cannot have more than 256 colors no matter the bandwidth. The JIF format provides a new and efficient way of sharing exciting, funny, and cute clips online.

6. REFERENCES

[1] Matthias Gelbmann, “The png image file format is now more popular than gif,” *W3 Techs*, January 2013.

[2] Gfycat, “How gfycat works,” *gfycat.com/about*, 2013.