Video Coding Based on Low Level Spatial Description of the Frames Sequence and on Human Visual System Perceptual Characteristics

Luca Leschiutta
Dipartimento di Automatica e Informatica - Politecnico di Torino
Corso Duca degli Abruzzi, 24 I-10129 Torino, Italy
{luca.leschiutta@polito.it}

Abstract

Aim of this work is to envisage a video coding technique capable to exploit low level plots between nearby pixels inside the video sequence.

The coding technique is first developed for images via the decomposition into small rectangular regions and then is extended to the video case utilizing a motion compensation technique to find replicas of the rectangles found on the first frame into the following ones.

The results of these techniques are discussed in terms of obtained compression ratios and in terms of optimization of the results varying several parameters in the rules that lead to the composition of the shapes identified inside the video sequence.

Keywords: Data compression, Image coding, Video coding, Motion compensation, Human Visual System

1. Introduction

This work is focused on finding new video compression techniques that do not rely on traditional transforms such as DCT and Wavelet, but exploit low-level pixels patterns inside subsequent frames.

These techniques are expensive from a computational point of view, but they become appealing given the performance of today's computers and the possibility to free oneself from the patents existing on most used compression tools.

The video coding tool is based on the work presented in [3] and summarized in the second paragraph.

This technique consists in considering the first image of a GOP, in dividing it into rectangles and in looking for replicas of such rectangles in the following frames.

The search is performed via motion compensation techniques to be able to find the maximum number of base rectangle replicas.

Each GOP is described via a set of values which represent base rectangle size, number of replicas, motion vectors and relevant color component. Depending on the fact that, to each of this parallelepipeds, is associated a single color or a color interval, it is possible to have loss-less or lossy compression. It is moreover possible to perform non-linear transformations in appropriate color spaces to have perceptually uniform video coding.

The developed video coding technique is therefore entirely unrelated to MPEG; perceptually optimizable for any presentation device or viewing condition; suited for high fidelity. The main drawback consists in limited compression ratio.

2. Rectangle coding technique

This technique is based on decomposition of an image or a frame in our case in small rectangular homogeneous zones and is inspired by what is done in [4] for binary images.

The algorithm can be applied to any raster image, defined using an appropriate tristimulus color space independently for each color component (e.g. in a standard RGB image the 3 matrices R, G, and B can be processed to obtain a different description of the image as a set of 3 rectangle sequences <r>, <g> and <b>).

The following description is focused on gray scale images (where only one matrix has to be treated) but can be easily extended to the color case.

Each rectangle is described by 3 parameters: height (Dy), width (Dx) and luminance (Y). The height and width are expressed in pixels, the luminance is normalized over 8 bits.

As it will be discussed later, the main parameters that can be varied are the rectangle composition rules.

The simplest rule consists in dividing the 256 luminance levels in fixed width intervals and to search for rectangles whose pixels all fit in one of these intervals.

One particular case is to have 256 intervals of width 1 (this will lead to a loss-less transformation of the image).

A preliminary analysis showed that the maximum allowed size of the rectangles should be 8*8 pixels for two reasons:

- Larger rectangles would easily impair the image quality.
- Considering a simple case, in which the width of the color interval is such that the image quality degradation is just noticeable, 99.9% of the rectangles found are smaller then 8*8.

The algorithm starts scanning the pixels matrix from the top left corner and verifies which, of the possible rectangles originating in that point and complying with the given set of rules, is the larger in size.

Once a rectangle is found, the corresponding pixels are marked as visited and the algorithm keeps on scanning what's left of the image, always proceeding left-right and top-down. The image can be therefore described by a set of triples having the form <Yi Dxi Dyi>, where Dxi and Dyi are respectively width and height of the i-th rectangle expressed in pixels, Yi is the luminance of the rectangle (see Figure 1).

Some rectangles are considered "spurious" (i.e. too small) and can be discarded.

As a result the image would be described by a set of quadruples having the form <Dpi Yi Dxi Dyi>, where Dxi and Dyi are respectively width and height of the i-th rectangle expressed in pixels, Yi is the luminance of the rectangle and Dpi is the distance from the rectangle (i-1)-th expressed in pixels.

2.1. Rectangles decomposition rules

Three main methods have been followed to perform rectangle decomposition:

1) Fixed width intervals
2) Perceptually variant intervals.
3) Transformation in a perceptually uniform color space

In the first case the spectrum of each color is divided into a given number of intervals with a fixed size.

In the second case the width of each interval is varied according to a logarithmic law that simulates the response of the Human Visual System (HVS) to the various luminance levels.

In the third case, an evolution of the second one, instead of conceiving ad-hoc rules to segment the three different RGB color components, the image is first transformed into the perceptually uniform CIE Lab color space and then processed using the rectangle coding technique with fixed width intervals.

The main drawback of the third methodology is the computational complexity of CIE Lab non-linear transforms, however, given the constant increase of CPU performances, this is not considered as a major blocking point.

The transform–antitransform operation introduces on its own approximately 45 dB of PSNR that are completely unnoticeable and not worth accounting for.

In addition, another HVS characteristic we can use to discard some, less perceptually relevant, information, is the masking of high spatial frequencies. For this reason the acceptance of rectangles in which a small amount of pixels does not fit into the given interval it is deemed profitable; the result will be to have fewer, but bigger, rectangles in which some high spatial frequency information has been lost.

3. Video coding based on rectangle decomposition of first frame

This phase consists into an extension to the third dimension of what has been done on the first frame.

As said before algorithm starts scanning the pixels matrix of the first frame from the top left corner and verifying which, of the possible rectangles originating in that point and complying with the given set of rules, is the larger in size.

Once a rectangle is found, the corresponding pixels are marked as visited and the algorithm keeps on scanning what's left of the image, always proceeding left-right and top-down.

In this case it is not worth discarding single pixels because we can hope to find anyway larger shapes exploiting the third dimension.

We end up with a simplified description of the first frame via a set of triples which represent the color component and size of every rectangle (the distance from the previous rectangle is not needed anymore).

The size of the rectangle is as usual limited to 8 * 8 and the rectangle composition rule can be the same described in the previous paragraph.

Figure 1 First frame description as a set of rectangles, each of whom is identified by the triple < Yi Dxi Dyi>.

3.1. Search for replicas of the rectangles found on the first frame

The next step consists in searching in the following frames to see if the rectangles found on the first one have replicas such as shown in Figure 2.

Figure 2 Repetition of rectangles in subsequent frames. A video sequence can be described as a set of parallelepipeds, each of whom is identified by the quadruple < Yi Dxi Dyi Dzi >.

Let's work on a three frame video sequence for simplicity sake.

On the second frame in the place of a given rectangle found on the first frames two cases are possible:

1) If the rectangle is still present the correspondent pixels are marked as visited ant the height of the parallelepiped based on the first frame is increased by one.
2) If the rectangle is not present then nothing is done and the corresponding pixels are available for a later codification that will search for rectangles on the unmarked pixels of the second frame

Again the search of first frame rectangles replicas will be performed on the third frame, some parallelepipeds height will be increased and the corresponding pixels will be marked as visited.

In this way the video sequence will be coded via a set of parallelepiped that can be expressed in the following way < Yi Dxi Dyi Dzi >, where Y is the associated color component and Dx, Dy, Dz are width depth and height.

After the first frame each parallelepiped can originate into any frame of the sequence. For this reason the group of pictures concept is superseded.

In Figure 3 is shown a graphical representation of the video codification.

3.2. Motion compensated parallelepipeds

The technique described in the previous paragraph works well only on the sections of video which correspond to backgrounds shot with still camera.

In large sections of the video, for definition, either the subject will be moving or it will be used a moving camera and all the sections of the video will be moving possibly on different directions. For this reason it is likely that the rectangles found on the first frame will shift a little in the subsequent frames as it is shown with the chin of the subject of Figure 4.

Figure 3 Parallelepipeds found inside a video sequence.

Figure 4 Repetition of rectangles in subsequent frames with and without motion compensation.

It is therefore necessary to introduce a motion compensation technique similar to the one described in [2] The rectangle replicas are searched not only on the exact position occupied in the previous frame but also on the surrounding zones. The amount of motion compensation (i.e. how large is the zone in which the search is performed) is selectable. The video coding result is depicted in Figure 6; at this stage it is not enough anymore to use a quadruple to code each parallelepiped because for each frame it is necessary also a motion vector i.e. a representation of the relative position of a given rectangle in the new frame with respect to the previous one (see Figure 5).

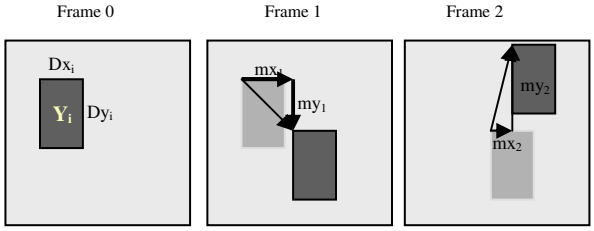

Figure 5 Motion-vectors resulting from rectangle shifting in subsequent frames.

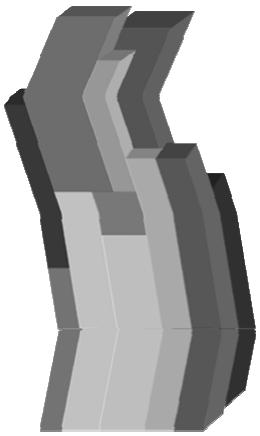

Figure 6 Pseudo-parallelepipeds found inside a video sequence when motion compensation is adopted.

It is therefore necessary to represent the video via a set of pseudo-parallelepipeds each of whom can be expressed in the following way:

<Yi,Dxi,Dyi,mx1, … , mxz, my1, … , myz, >

Where:

- Yi = color component associated with the pseudo-parallelepiped i;
- Dxi = width of pseudo-parallelepiped i;
- Dyi = depth of pseudo-parallelepiped i;
- mx1, … , mxz = component along x axis of motion vectors;
- my1, … , myz, = component along y axis of motion vectors;
- z = number of motion vectors, i.e. height of the pseudo-parallelepiped.

The above notation is much more onerous than the one foreseen on the previous paragraph, but the introduction of motion compensation is anyway profitable from a compression ratio point of view.

3.3. Maximization of parallelepipeds volume

The video coding technique described so far relays heavily on the 2D algorithm developed for image compression in [3] . In fact, given a certain pixels, first it is found the biggest rectangle that can be built on that pixels, then it is sought the number of replicas of that rectangle.

From a compression point of view it would be better to search directly the biggest rectangle that can be built on a given pixel. The drawback of doing this is that the computation time will be much increased.

From a procedural point of view the algorithm is fairly simple.

First it is necessary to select several parameters:

1) The sought parallelepipeds maximum size. Experience driven considerations lead to look for parallelepiped having base 8*8 and height 10.
2) The parallelepipeds composition rules that will be discussed on the next paragraph about lossy video coding.
3) The amount of motion compensation (usually 1 or 2 pixels horizontally and vertically)

At this point it is necessary to order all the possible parallelepipeds from the largest to the smallest for what concerns the volume and to scan them until one that complies with the above mentioned rules is found.

Moreover the above must be done on all the possible pseudo-parallelepipeds that can be built on the given pixel.

3.4. Lossy video coding

Similarly to what is done for images, also in this case the algorithm is scalable, from lossless to lossy, changing several parameters in the parallelepipeds composition rules.

Color interval: If the color interval is set to 1 then we have lossless coding. To have lossy coding the color interval size must be increased uniformly or adopting some sort of perceptually uniform coding similar to the one described in paragraph 2.1 and based on variable intervals or transformation on CIE Lab color space.

Out of range pixels: The amount of out of range pixels that can be accepted in a parallelepiped, from a perceptual point of view, is higher than the one that can be accepted in a rectangle. As usual the selection allows.

- a small number of highly out of range pixels;
- a higher number of lightly out of range pixels;
- to skip one frame.

3.5. Video compression results

The parallelepiped coding technique performances, at the moment, are not comparable to the one of mainstream video coding techniques such as MPEG or H.264.

It is nevertheless interesting to compare, especially in terms of PSNR, the result obtained varying the parameters selectable in the coding phase.

Hereafter these parameters are listed and their influence on the coded video is summarized:

Color Interval: This parameter indicates the depth of the color intervals used to build the parallelepipeds in the lossy coding.

Every color component is dealt with separately so there are 256 color levels that can be divided into intervals.

The tests have been done with 3 value of Color Interval: 3, 6 and 9 which for image coding would correspond to video quality degradation not perceivable, video quality degradation acceptable and sensible video quality degradation.

Max. Width: This parameter indicates the maximum width of the parallelepipeds.

The higher this parameter the more onerous is the computation, because the algorithm starts scanning the video fragment looking for biggest parallelepipeds first.

If this parameter is too little, on the other end, the found parallelepipeds are smaller and the resulting compression ratio is decreased.

Max. Depth: This parameter indicates the maximum depth of the parallelepipeds and undergoes the same conditions of the previous one.

Max. Height: This parameter indicates the maximum height of the parallelepipeds and undergoes the same conditions of the previous one.

It is worth noting that the parallelepipeds are much more extended in height (time axis) then in the other two directions.

Quality Selection: This parameter indicates the amount of out of range pixels that can be accepted into a parallelepiped.

The tests have been performed on the following standard sequences: Container, Kitchgrass and Foreman.

The metrics used to evaluate the results are compression ratio (or bitrate) and PSNR.

To summarize the results it is possible to say that:

- The achieved compression ratio is roughly 3.2 with 50 dB of PSNR and 7.0 with 36 dB of PSNR.
- The most influential parameters for what concerns the resulting quality are Quality Selection and Color Interval (see Table 1).
- Generally speaking increasing the parallelepiped maximum size has a good effect on the compression ratio, but also the PSNR is degraded (see Figure 7).

Table 1 Variation of the PSNR and the Compression Ratio with respect to parallelepiped size in pixels.

Max. Width [pixels]	Max. Depth [pixels]	Max. height [pixels]	Mean width [pixels]	Mean depth [pixels]	Mean height [pixels]	PSNR	Compression ratio
3	3	3	1,26	1,26	2,91	49,88	3,21
5	3	3	1,27	1,26	2,91	49,88	3,22
3	5	3	1,26	1,28	2,91	49,88	3,23
7	3	3	1,28	1,26	2,91	49,88	3,22
5	5	3	1,29	1,27	2,91	49,88	3,25
3	7	3	1,26	1,29	2,91	49,88	3,24
7	5	3	1,29	1,27	2,91	49,88	3,26
5	7	3	1,27	1,28	2,91	49,88	3,25
7	7	3	1,29	1,27	2,91	49,88	3,26
3	3	5	1,26	1,27	4,82	47,66	3,89
5	3	5	1,28	1,26	4,82	47,66	3,91
3	5	5	1,28	1,27	4,82	47,66	3,93
7	3	5	1,28	1,26	4,82	47,66	3,91
5	5	5	1,29	1,27	4,82	47,66	3,95
3	7	5	1,28	1,28	4,81	47,66	3,94
7	5	5	1,29	1,27	4,81	47,66	3,95
5	7	5	1,27	1,28	4,82	47,66	3,95
7	7	5	1,27	1,29	4,82	47,66	3,95
3	3	7	1,28	1,25	6,72	46,20	4,42
5	3	7	1,29	1,25	6,72	46,20	4,44
3	5	7	1,26	1,28	6,72	46,20	4,44
7	3	7	1,28	1,26	6,72	46,20	4,44
5	5	7	1,27	1,28	6,72	46,20	4,46
3	7	7	1,26	1,29	6,72	46,20	4,45
7	5	7	1,29	1,27	6,72	46,20	4,48
5	7	7	1,27	1,28	6,72	46,20	4,47
7	7	7	1,29	1,27	6,72	46,20	4,49

4. Perceptually uniform data losses

Similarly to what has been done in [3] for images, coding technique, biased by a Human Visual System (HVS) perceptual model, are under investigation.

Two techniques to discard information in a perceptually uniform way in the space domain can be applied:

- In the first one the width of each interval can be varied according to a logarithmic law that simulates the response of the HVS to the various luminance levels of the Y component.
- In the second one it is possible to work on all color components; the frames are first transformed into the CIE Lab perceptually

uniform color space and then the video coding described in paragraph 3 is applied to the resulting sequence.

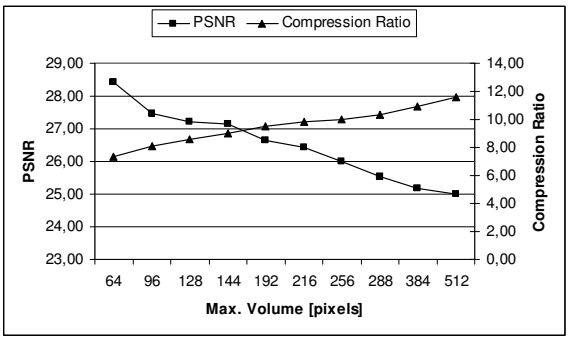

Figure 7 Variation of the PSNR and the Compression Ratio with respect to parallelepiped size in pixels.

These techniques are still under test, but there is no reason to believe that the results will be different from the ones of image coding.

The next step will make use of the HVS poor discrimination for colors viewed successively and compared in memory. This could lead to a high reduction of the color needed to represent a video sequence.

Conclusions

The founding motivations for the research have been, on one hand, the possibility to use simple processing into the space and time domain to exploit low level plots between nearby pixels and high computational speed and, on the other hand, the need for image and video coding tools not relaying on patented technologies.

It was soon evident that the simple construction of parallelepipeds over rectangles found on the first frame of a group of pictures did not work well for the characteristics peculiar of moving pictures. For this reason a motion compensation technique that allows for a few pixels shift of the rectangles between the various frames was introduced. Moreover some algorithms to maximize the found pseudo-parallelepipeds were implemented.

At the moment this video coding techniques still needs improvements because of the low compression ratios and of the high computational costs.

References

[1] J W Woods, T S Huang, “Picture Bandwidth Compression by Linear Transformation and Block Quantization”, Symposium on Picture bandwidth Compression, MIT, Cambridge Mass., Apr 1969.

[2] S. Dubiusson, F. Davoine, “Motion compensation using adaptive rectangular partitions”, Proceedings. of International Conference on Image Processing ICIP99, Volume 1, pp. 56 – 60, 1999.

[3] L. Leschiutta, “Perceptually Optimal Image Coding Based on Human Visual System Characteristics and Uniform Color Space”, Proceedings of CIE Midterm Meeting and International Lighting Congress, Leon, Spain, pp. 78-88, May 2005.

[4] Mohamed, S.A.; Fahmy, M.M, “Binary image compression using efficient partitioning into rectangular regions”, IEEE Transactions on Communications, Volume: 43, Issue: 5, pp:1888 – 1893, May 1995.

[5] L.W. Chang, C.Y. Wang; S.M. Lee, “Designing JPEG quantization tables based on human visual system”, International Conference on Image Processing ICIP 99 Proceedings, Volume 2, 24-28 Oct., pp. 376 - 380, 1999.

[6] T. Eude, A. Mayache, “An evaluation of quality metrics for compressed images based on human visual sensitivity”, Fourth International Conference on Signal Processing Proceedings ICSP 98., 12-16 Oct., pp. 779 - 782 vol.1, 1998.

[7] Rec. ITU-R BT.500-11, “Methodology for the subjective assessment of the quality of television pictures”, 2002