PERCEPTUALLY OPTIMIZED MPEG COMPRESSION OF SYNTHETIC VIDEO SEQUENCES

Enrico Masala, Davide Quaglia

Dipartimento di Automatica e Informatica
Politecnico di Torino
Corso Duca Degli Abruzzi, 24 — I-10129 Torino, Italy
Email: [enrico.masala|davide.quaglia]@polito.it

ABSTRACT

This paper addresses the problem of improving the quality performance of synthetic video sequences by means of standard frame–based coders. The proposed technique can exploit both the knowledge of the 3D model and the intermediate information computed during the rendering process. Firstly, objects are classified, either semantically or automatically, according to their importance. Then the object classification is translated into a macroblock classification, with particular attention to object boundaries. The classification influences the encoder parameters selection, for instance, the quantization parameter. In order to maximize the performance, we propose a rate–distortion formulation of the problem. Experimental results compared with model–unaware encoding show that the proposed techniques can deliver consistent visual quality improvements for different synthetic scenarios using the same bitrate or even less. Demo sequences are available at http://media.polito.it/perceptual3d.

1. INTRODUCTION

Synthetic video sequences are going to play a key role for both entertainment and training, since they are widely used in animation movies, video games and virtual reality applications (e.g., immersive collaborative environments and scientific visualization tools). In most of these applications, compression should be applied to video sequences for storage and transmission purposes. In general, the highest compression efficiency can be achieved by coding the original model description, e.g., using a standard format like the Virtual Reality Modeling Language (VRML) [1]. However, the distribution of the original model to end–users, as traditionally done in VRML applications, may not be advisable for copyright reasons and for the need of a rendering application in each client.

An alternative approach consists in distributing a compressed version [2–6]. Even if specific compression techniques for synthetic sequences have been proposed [3, 4, 7], the use of a traditional frame–based video coding standard like MPEG–1 or MPEG–2 can be an appealing approach since MPEG codecs are often embedded in many clients (e.g., DVD players) and no additional software would be required. Coding techniques in MPEG are mainly designed for natural video, not graphics; however, the knowledge of the synthetic model can contribute to reduce bitrate and computational complexity and to enhance quality. In particular, computational complexity reduction is a desirable factor in a distributed environment where encoding is performed on the server for many clients [2, 5]. The main technique to reduce bitrate in standard video coding standard is quantization which, however, may introduce visual artifacts in the representation of important objects (regions of interest, ROI) and, in particular, of their edges. For a given bitrate, the overall perceived quality can be increased if ROIs are identified since many more bits can be spent to code them while less important regions can be quantized more coarsely. The automatic determination of ROIs in video sequences is an active research field [8] but it is, in general, computational expensive.

In this work the model information provided by the 3D animation engine is exploited to quickly determine the regions of interest in synthetic video sequences. Two approaches are proposed for the assessment of the 3D object importance: 1) full automatic classification based on the distance from the point of view, and 2) manual classification of objects in the 3D space. The importance of 3D objects is then mapped on the importance of pixels in the resulting video frames. Pixel–based classification is exploited to decide the quantization stepsize for each macroblock taking into account the trade–off between overall quality and coding efficiency. Object edges are also coded with high quality. The approach of varying the quantization stepsize according to the object importance was first addressed in [5] by transmitting foreground and background as separate MPEG-4 objects.

The paper is organized as follows. Section 2 briefly introduces the system from the point of view of the animation and coding environment. Section 3 describes the details of the proposed technique. Experimental results are reported in Section 4. Finally, in Section 5 conclusions are drawn and some future works are foreseen.

2. SYSTEM OVERVIEW

Figure 1 shows the layout of a client–server system to which the proposed technique can be applied. The 3D model is stored in the server. The animation 5B engine is a computer graphics application which applies a sequence of geometrical transformations to the model obtaining an animation; the geometrical transformations can be driven by the remote user (e.g., the user can move the point of view of the scene through client's keyboard). The animation engine creates a 2D view of the 3D scene transforming the animation in a sequence of frames (e.g., arrays of luminance and chrominance samples). Frames are fed into a frame–based video encoder belonging to the widespread MPEG or H.26x families. A subset of the model information held by the animation engine is

This work was supported in part by MIUR, Project FIRB-PRIMO, http://primo.ismb.it.

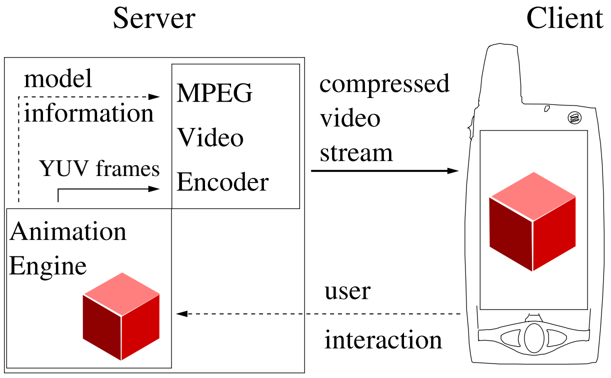

Fig. 1. Overview of the system.

also transferred to the video encoder to improve compression as described in Section 3; in this work for each frame pixel we consider its depth (i.e., its z–buffer value) and the object to which it belongs. The client receives the compressed stream, decodes it, and displays frames. Many clients such as mobile devices and set–top boxes (e.g., DVD players) are provided with DSP capabilities oriented to standard video decoding.

MPEG and earlier H.26x video compression standards divide each frame into a set of non–overlapped blocks of 16×16 pixels called *macroblocks*. A macroblock (MB) consists of four blocks of luminance samples and up to eight blocks of chrominance depending on the chrominance sub–sampling schema; each block consists of 8×8 pixels referring either to the actual frame content or to the temporal prediction residue. The discrete cosine transform (DCT) is then applied to each block to determine the energy corresponding to spatial frequencies. The value of high frequency coefficients is reduced through scalar quantization to decrease the bitrate of the resulting compressed bitstream. However, the quantization noise can be perceived if there is a considerable amount of energy at the higher frequencies as in case of strong edges. Since the quantization stepsize can be chosen on a MB basis and remains the same for all blocks within the MB, the quality degradation leads to blocking artifacts as in well–known low–quality JPEG images. These artifacts are even worse in most synthetic video sequences where there are many strong edges in very smooth regions.

An arbitrary number of consecutive macroblocks (in raster–scan order) is coded into a *slice*. In the coded bitstream the quantization stepsize is specified at the beginning of each slice for all its macroblocks; optionally, the quantization stepsize can be changed for a given MB provided that its value is specified for that MB at the expense of extra bit cost. For instance, in H.263 the quantization stepsize is differentially encoded between adjacent macroblocks of the same slice; for this reason its value can only change in the range ± 2 with respect to that of the previous MB. Therefore, if higher variations are desired a new slice have to be created at the expense of extra bit cost. In the H.264 video coding standard [9] 8×8 DCT is replaced by a purely integer 4×4 transform and 52 different quantization step sizes can be chosen on a macroblock basis; with respect to prior standards, the stepsizes are increased at a compounding rate of approximately 12.5%, rather than increasing them by a constant increment.

The quantization stepsize can be adjusted to preserve the quality of edges and important objects but new quantization values have to be coded in the bitstream; in general, there is a trade–off between the quality gain and the coding efficiency; in Section 3.4 this trade–off will be formulated as a rate–distortion optimization problem.

3. COMPRESSION OPTIMIZATION

3.1. Object Perceptual Importance

In video sequences, both natural and synthetic, some elements are more important than others. For example, objects in the foreground are usually more significant than those in the background. Moreover, artifacts near to strong edges are easily perceived. For these reasons, important groups of pixels should be encoded with high quality. While different objects are easily separated by the human visual system, their automatic segmentation is an open research problem and many solutions have been proposed [8].

In case of synthetic video sequences, object segmentation is straightforward since the frame is the result of the projection of the 3D space on a plane; in fact, the animation engine knows the position of each object and the correspondence between pixels and objects. The importance of each object can be assessed either automatically or manually. Assuming that the foreground objects are more important than background objects, automatic object classification can be performed considering their distance from the point of view. This information is usually held by the animation engine in a data structure called *z–buffer* and, therefore, this kind of segmentation can be performed automatically with low computational power. Since the animation engine knows the exact correspondence between objects in the 3D space and pixels in the frame, important pixel areas can be easily identified if the designer manually defines the importance of each object in the 3D scene. Boundary regions can be identified as pixel areas between objects.

A more complex scenario can be seen in Figure 2 in which foreground objects project their shadows both on the background and on other foreground objects. In this case only the portion of the shadow which lays on the background can be coded with lower quality.

3.2. Macroblock Classification

When the sequence of frames is encoded with the MPEG or H.26x video coding standards, the subdivision of pixels into 16×16 macroblocks needs to be addressed. Moreover, the quantization stepsize can be adjusted at MB level. Figure 2 shows a frame taken from a computer graphics animation; the overlaid grid represents MB subdivision. Assuming an object classification based on the distance from the point of view, the bottle, the knife and fruits are foreground objects and should be coded with higher quality. In the background, the table is more important than the wall. The first remark is that some macroblocks are entirely composed of pixels of the same importance while others contain pixels of mixed importance. The importance of a MB can be evaluated in two ways: 1) by using the importance assigned to the majority of its pixels, and 2) by using an importance value computed as the average of the importance of its pixels. The former method limits the overall number of importance levels in the frame and simplifies the subsequent determination of the quantization stepsize.

Another issue to be addressed is the determination of the macroblocks containing edges. Two solutions are possible: 1) choosing macroblocks that contain pixels corresponding to object boundaries, and 2) choosing important macroblocks that are adjacent to at least one less important MB. The former method requires

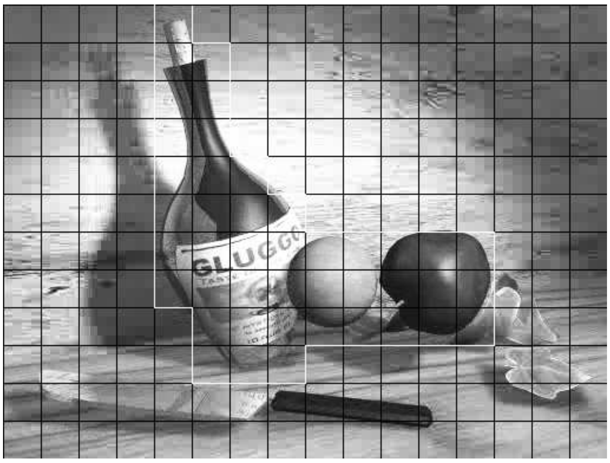

Fig. 2. Example of a frame with MB subdivision and identification of the region of interest (white borders).

the support of the 3D engine while the latter is simpler to implement.

3.3. Quantization Strategies

The amount of visual artifacts in reconstructed macroblocks mainly depends on the quantization stepsize (QP). A good strategy to enhance perceived quality is to reduce the QP value for important macroblocks and vice versa (Strategy 1). However, the optimal QP value for each macroblock is difficult to determine and depends on macroblock content. An excessively high value could easily lead to blockiness effects, while a low one could result in poor compression performance.

Besides, quantization produces visible artifacts (e.g., ringing) near object boundaries, and their impact on perceived quality is often large. We propose a computationally efficient strategy, that consists in determining the border macroblocks between important and less important areas and to adjust their QP to a low value (Strategy 2). The originality of this approach, compared to classical edge detection techniques, resides in the computationally efficient discrimination between important and less important borders, as well as between object boundaries and simple variations of luminance by taking advantage of the information coming from the 3D model.

Every video coding standard provides mechanisms to save bits if the QP is constant (e.g., in MPEG-2, the QP specified in the slice header is valid for all the macroblocks in that slice). However, changing the QP leads to bitrate increase because all variations need to be coded. Therefore, determining the optimal frame subdivision into slices and the optimal QP variation points is not a trivial task. The next subsection addresses this task using a rate–distortion optimized approach.

3.4. Rate–Distortion Optimization

Let N be the number of macroblocks in a frame, and let m_i denote the actual values for the source coding parameters assigned to the i-th macroblock, e.g., the quantization parameter and the macroblock coding mode, including subdivision into smaller blocks (available in recent standards such as H.264). We define a perceptually–weighted quality metric for each frame as:

$$D = \sum_{i=1}^{N} w_i d(m_i), \tag{1}$$

where, for the i-th macroblock, $d(m_i)$ is the macroblock encoding distortion and w_i is the weighting parameter whose value is proportional to the macroblock importance as determined above. Higher values of w_i are assigned to more important macroblocks. The distortion values $d(m_i)$ depend on the encoding parameters m_i and they can be computed using a conventional measure, e.g., MSE. The number of bits R needed to encode the frame is

$$R = \sum_{i=1}^{N} r(m_i, m_{i-1}) \tag{2}$$

where $r(m_i, m_{i-1})$ is the number of bits required by the i-th macroblock, including the slice header if needed. Note that in general this value depends not only on m_i, but also on the choice for the previous macroblock m_{i-1}, due to the differential encoding of the quantization parameter and the motion vector predictors.

The ability to compute the previous rate and distortion values makes possible to apply a rate–distortion optimized encoding algorithm, minimizing the well–known Lagrangian

$$J = D + \lambda R. \tag{3}$$

This problem can be effectively solved using dynamic programming techniques.

4. EXPERIMENTAL RESULTS

We modified the standard MPEG-2 encoder to implement some of the previous techniques. We developed an automatic object classifier that consider an object as important if its distance from the point of view is less than one fourth of the farthest object in the scene. For Strategy 1, we assigned a different QP value to important (QP_i) and less important (QP_{ni}) macroblocks. For Strategy 2, macroblocks containing the edges of important objects are quantized with a QP value (QP_b) which is lower than QP_i and QP_{ni}.

The rendering process was performed using 3DSmax v.5, visual results are shown for two synthetic demo sequences included in the software package. The *Complex* sequence consists of many moving objects on a table. All objects are textured, including background. The *Anibal* sequence shows the head of a cartoon character animated on a uniform background.

The proposed strategies are compared with a quantization approach that keeps QP constant for the entire frame. Three different values of QP have been used: 5, 13, 31. Table 1 shows the bitrate

Table 1. Compression performance of different encoding strategies.

	Bitrate (kbit/s)	
Technique	*Complex*	*Anibal*
Fixed QP=13	650.0	288.7
Fixed QP=31	267.3	178.0
Fixed QP=5	1817.1	691.0
Str. 1, QP_i=13, QP_{ni}=31	556.1	289.1
Str. 1+2, QP_i=13, QP_{ni}=31, QP_b=5	904.7	433.0

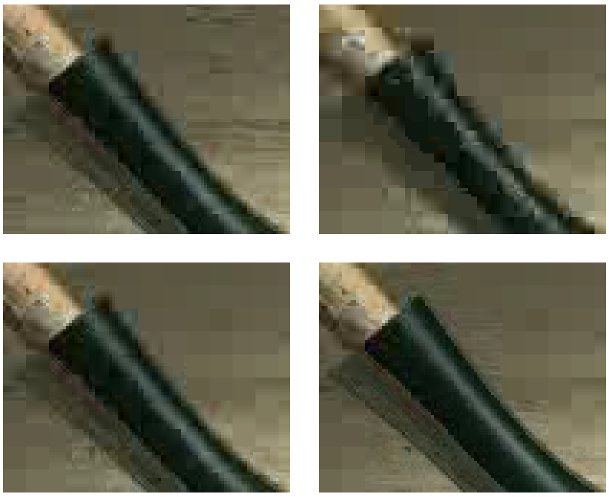

Fig. 3. Comparison of a detail of the *Complex* sequence. From top to bottom and left to right: fixed QP=13, fixed QP=31, Strategy 1, Strategy 1+2.

required to encode the sequences as a function of the quantization strategy. The fixed QP=31 strategy presents the lower bitrate but its visual performance is clearly unacceptable, as shown in Figure 3 and 4 (top right). Compared to the fixed quantization technique (QP=13), Strategy 1 saves 15% of the bitrate for the *Complex* sequence, while delivering the same visual quality, as shown by Figure 3. In the video encoded with Strategy 1+2 the edges, which are one of the most sensitive elements for the human visual system, exhibit about the same visual quality obtained with the fixed QP strategy with QP=5, but with half of the bitrate.

For the *Anibal* sequence, Strategy 1 presents the same bitrate as the fixed QP=13 strategy. For this particular sequence, in fact, due to the uniform background, when the QP for this area is modified from 13 to 31 the bitrate is only minimally affected. Strategy 1+2 performs well also on this sequence, using 37% less bitrate compared to the fixed QP strategy with QP=5. Visual comparison is shown in Figure 4. More visual results are available at http://media.polito.it/perceptual3d.

5. CONCLUSIONS AND FUTURE WORK

We presented an approach to improve the quality performance of synthetic video sequences compressed with a standard frame–based coder. The proposed techniques exploit the knowledge of the 3D model to classify objects. Given a certain classification, the encoder parameters (e.g., the quantization stepsize) can be optimally selected for each macroblock, using a rate-distortion formulation of the problem. Experimental results compared with model-unaware encoding show that the proposed techniques can deliver consistent visual quality improvements for different synthetic scenarios using the same bitrate or even less. Future work will be devoted to further improve the presented algorithms exploiting the new coding tools provided by the H.264 video encoding standard.

6. ACKNOWLEDGMENTS

The authors would like to thank Angelo Gattuso for implementing and running some of the algorithms presented here.

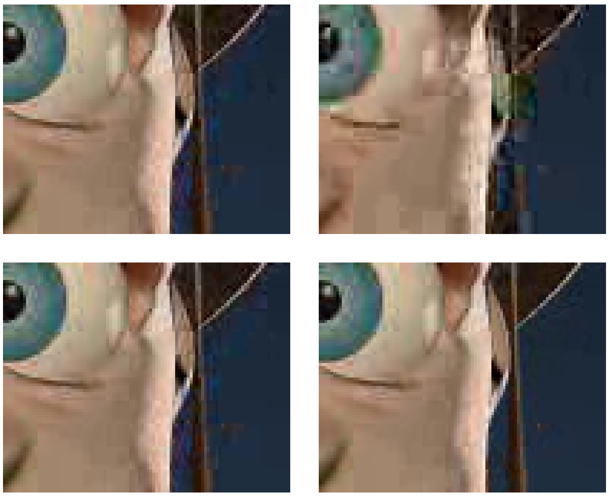

Fig. 4. Comparison of a detail of the *Anibal* sequence. From top to bottom and left to right: fixed QP=13, fixed QP=31, Strategy 1, Strategy 1+2.

7. REFERENCES

[1] ISO/IEC, "Information technology – Computer graphics and image processing – The Virtual Reality Modeling Language (VRML) – Part 1: Functional specification and UTF-8 encoding," *ISO/IEC 14772-1*, 1997.

[2] M. Levoy, "Polygon-assisted JPEG and MPEG compression of synthetic images," in *Proc. ACM SIGGRAPH*, 1995, pp. 21–28.

[3] D. Cohen-Or, "Model-based view-extrapolation for interactive VR web-systems," in *Proc. IEEE Int. Conf. Computer Graphics*, June 1997, pp. 104–112.

[4] I. Yoon and U. Neumann, "Compression of computer graphics images with image-based rendering," in *Proc. of SPIE Multimedia Computing and Networking*, 1999, pp. 66–75.

[5] Y. Noimark and D. Cohen-Or, "Streaming scenes to MPEG-4 video-enabled devices," *IEEE Computer Graphics and Applications*, vol. 23, no. 1, pp. 58–64, Jan–Feb 2003.

[6] D. Quaglia and A. Gattuso, "Model-based MPEG compression of synthetic video sequences," in *Proc. IEEE Int. Conf. on Image Processing*, Singapore, Oct. 2004, pp. 1109–1112.

[7] B. Guenter, H. Yun, and R. Mersereau, "Motion compensated compression of computer animation frames," in *Proc. ACM SIGGRAPH*, 1993, pp. 297–304.

[8] P. L. Correia and F. Pereira, "Classification of video segmentation application scenarios," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 5, pp. 735–741, May 2004.

[9] ITU-T Rec. H.264 & ISO/IEC 14496-10 AVC, "Advanced video coding for generic audiovisual services," *ITU-T*, May 2003.