File Size Comparisons of Modeled and Pixel-Based Video in Five Scenarios

Juergen Wuenschmann, Christian Feller, and Albrecht Rothermel
Universität Ulm
Institute of Microelectronics, Albert-Einstein-Allee 43, 89081 Ulm, Germany
Email: {juergen.wuenschmann, christian.feller, albrecht.rothermel}@uni-ulm.de

***Abstract*—File sizes have been compared for object modeled video versus pixel-based video with respect to different factors, which influence the data rate. The goal is to simplify the choice, which representation is favorable in terms of file size for varied content. Traditional video compression is very sophisticated and to achieve even lower data rates than the state-of-the-art video codecs while preserving the visual quality is difficult and increases the complexity. Using the object-based representation, possibilities to reduce the amount of data to be stored or transferred are investigated. With our improved object-based encoder a data rate reduction of 67,6-90 percent compared to an uncompressed object-based representation and up to 90.5 percent compared to high quality, variable bitrate, pixel-based video is achieved for the investigated scenarios.**

***Index Terms*—Video signal processing; Video compression; High definition video; Object-based video coding.**

I. Introduction

The reduction of redundancy and irrelevancy has been a major part of digital signal processing since its beginning [1]. With a growing amount of data in video signal processing, due to rising resolution, frame rate, and quality demand, a highly effective compression is essential. The possibility to achieve higher compression rates while preserving a certain quality level becomes harder and the complexity of the implemented algorithms is rising. Most of them are based on the traditional, pixel-based video processing scheme [2].

For modeled and animated material, another path can be chosen. The material can be compressed and transmitted or stored directly, without rendering it to a pixel-based representation. A big advantage of this modus operandi is that it is possible to adapt to any display device property, as, e.g. resolution, frame rate, and number of views, through adjusting a built-in renderer. A basic object-based compression scheme was proposed in [3] and standardized as MPEG-4 pt. 25. This reference implementation has been improved and a comparison to the original implementation is given in [4].

To compare the traditional pixel-based video representation and the object-based video representation, five test scenarios are created. Each isolates the influence of a certain video property on the amount of data and therefore the compression efficiency. With these test scenarios, it is possible to distinguish, which representation is favorable to use for certain video material.

The paper is organized as follows: In Section II, the four pixel-based and four object-based parameter sets are described. The five test scenarios and their influence on the amount of data are explained in Section III. Section IV contains the results and a discussion and Section V summarizes and concludes the paper.

II. Video Representations

The comparisons have been performed between object-based and pixel-based video representations. Both representations have been tested using various parameters. For object-based material these formats have been used:

- *Collada* - XML representation of the scene.
- *Collada rared* - Win Rar compressed Collada.
- *MP4 ref encoded* - encoded using the standard (non-improved) version of the MPEG-4 pt. 25 encoder.
- *MP4 enc xyz OneStream* - encoded using our improved MPEG-4 pt. 25 encoder.

These four formats incorporate settings of the whole efficiency range for object-based material. The most ineffective one is the textual representation, which can be compressed very effectively with the state-of-the-art rar compression. A Collada file has to be decompressed completely to be used. With the object-based video compression MPEG-4 pt. 25, streaming of video clips is possible. Two versions of this encoder are used. The one used as reference is the version proposed in [3], while many improvements are integrated in our updated version. The pixel-based representation is produced rendering a scene in Blender and using the H.264 encoder *x264* to encode the video with a state-of-the-art video encoder. Parameters used were high profile at level 4.1 and two reference frames. Clips were rendered with different resolutions. Due to space limitations, the only resolution shown in this paper is $1920x1080$ Pixel, which is Full HD. The quality settings used are:

- *crf18 1080p* - variable bit rate encoding using constant rate factor 18.
- *crf24 1080p* - as above using constant rate factor 24.
- *10MBit 1080p* - constant bit rate encoding of $10MBit/s$.
- *20MBit 1080p* - as above with $20MBit/s$.

These settings are based on the settings used typically for Blu-Ray discs and represent high quality video encoding.

III. Test Scenarios

Scenario 1: The first analyzed scenario comprises the effect of the scene length on the data rate. To isolate the scene

 ISBN: 978-1-61208-195-3

length from other parameters influencing the amount of data, a generic test sequence was chosen. The sequence is derived from the incrementing cubes scenario described in [4]. It shows an amount of i^2 cubes arranged side by side with random color and random rotation. The difference is, that the

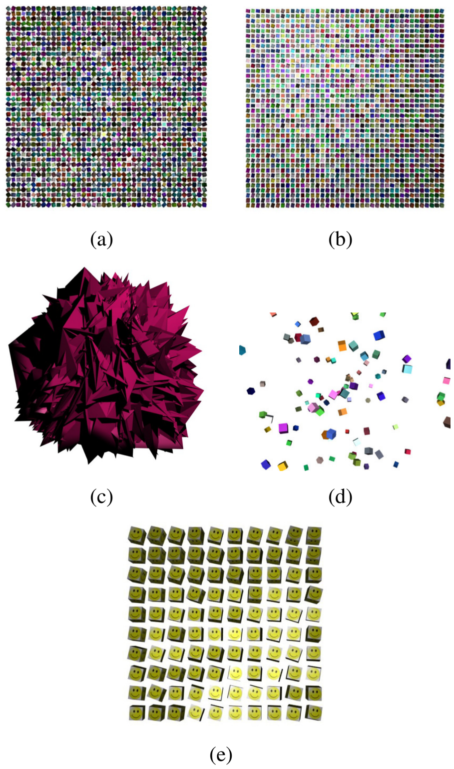

Fig. 1. As example, the first frame of (a) Scenario 1, (b) Scenario 2, (c) Scenario 3, (d) Scenario 4, and (e) Scenario 5 is shown.

number of cubes is kept constant at 1600 Cubes and instead the number of frames is altered.

Scenario 2: The second scenario, derived from the incrementing cubes test described in [4], is the evaluation of how the speed of movement influences the amount of data. It shows an amount of i^2 cubes arranged side by side with random color and random rotation. The number of cubes in the scene is again kept constant at 1600 Cubes and the rotation of all cubes is, despite a random initial orientation, identical. The rotation is defined by the number of turns and is incremented in every iteration by $5 \cdot \pi$. Using this scenario, it is possible to evaluate the quality of a temporal prediction used in video representation encoders.

Scenario 3: The following scenario reveals the effect of the complexity of objects in a scene on the data rate. To investigate this effect, a cube is subdivided with every iteration and the vertices are arranged randomly inside a spherical space. The generated object is rotating. Using simple subdivision, the number of vertices for iteration i is $v = (2^{i-1} + 1)^3 - (2^{i-1} - 1)^3$.

Scenario 4: Camera movement is the fourth evaluated effect on the amount of data. The scenario is build up of a static scene containing 100 Cubes with random orientation and color, randomly arranged in a spherical space. A camera moves around this scene with different velocity. This scenario represents the influence of panning and tilting of a video camera.

Scenario 5: The next analyzed scenario is as well derived from the incrementing cubes scenario of [4]. It has been upgraded to use textures mapped onto the objects. As in the original script, the number of cubes displayed is i^2 for iteration i. Every cube has a random rotation and a texture is mapped onto each side of a cube instead of random coloring for each cube in the original experiment. Using this test, it is possible to evaluate the influence of a more complex surface of moving objects on the data rate of the two video representations.

An example frame for each of the five scenarios can be seen in Fig. 1.

IV. Results

The results for the first scenario are shown in Fig. 2. Despite

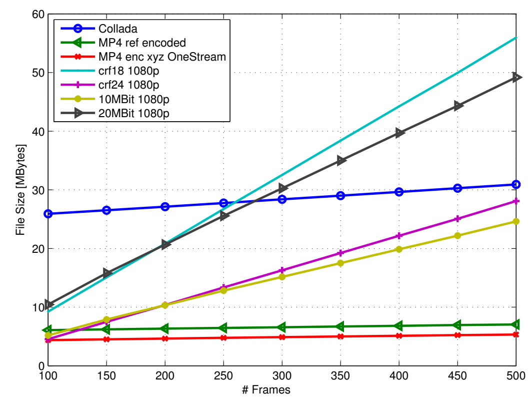

Fig. 2. Scenario 1: Effect of the scene length on the file size of object-based and pixel-based video representations.

the different parameter sets, the pixel-based video representation shows a linear increase in file size with incrementing number of frames. The object-based representation is nearly unaffected by the length of the scene. With growing length,

TABLE I
SCENARIO 1: DATA RATE SAVINGS FOR COMPRESSED OBJECT-BASED REPRESENTATIONS WITH RESPECT TO UNCOMPRESSED COLLADA.

	File Size Saving [%]
Collada rared	95.9 - 95.5
MP4 ref encoded	76.6 - 77.3
MP4 enc xyz OneStream	83.1 - 82.8

additional key frames for the animations have to be stored, which create little overhead. Consequently, an object-based representation is favorable for long scenes. In Table I the data rate savings for the different compression schemes for object-based material are shown. It is obvious, that the non-streamable *Collada rared* performs best and the MPEG-4 pt. 25 reference encoder performs worst. Our improved encoder lies in the middle, comprehending the benefit of the streaming possibility.

The effect of the velocity of animations is shown in Fig. 3. The variable bit rate video files are strongly influenced by

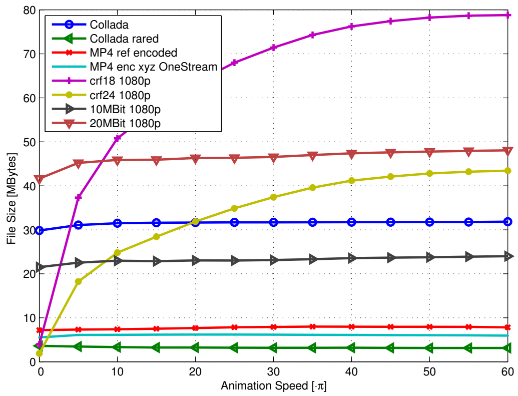

Fig. 3. Scenario 2: Growing animation velocity only influences the variable bit rate video representation.

the animation speed. The temporal prediction is working less efficient for rising speed. For the constant bit rate files, the file size is kept nearly constant by the rate control. Since the number of keyframes is not influenced by a growing animation speed, all object-based representation files have a constant file size. The data rate savings for the different object-

TABLE II
SCENARIO 2: DATA RATE SAVINGS FOR COMPRESSED OBJECT-BASED REPRESENTATIONS WITH RESPECT TO UNCOMPRESSED COLLADA.

	File Size Saving [%]
Collada rared	87.9 - 90.2
MP4 ref encoded	76.6 - 75.5
MP4 enc xyz OneStream	81.5 - 81.4

based compression versions are shown in Table II. In this scenario, the improved encoder, with up to 81.5% file size saving, performs 6% better than the reference encoder. The non-streamable *Collada rared* performs best with up to 90% saving.

The results for the third scenario are shown in Fig. 4. The

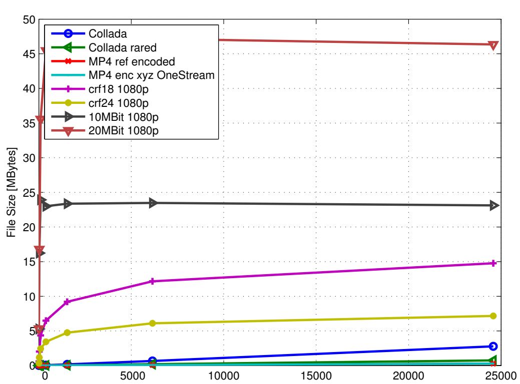

Fig. 4. Scenario 3: Effect of the number of vertices on the file size of object-based representation and pixel-based representation.

effect of the level of detail of an object on the pixel-based representation is low. The constant bit rate files are unaffected except for very small numbers of vertices. The variable bit

TABLE III
SCENARIO 3: DATA RATE SAVINGS FOR COMPRESSED OBJECT-BASED REPRESENTATIONS WITH RESPECT TO UNCOMPRESSED COLLADA.

	File Size Saving [%]
Collada rared	73.3 - 67.6
MP4 ref encoded	90.0 - 64.0
MP4 enc xyz OneStream	90.0 - 67.6

rate files show a steep incline for a small number of vertices and are reaching saturation for a higher level of detail. The object-based representation shows a linear behavior for an increasing number of vertices. Consequently, for an increasing level of detail a break-even is expected, where the pixel-based representation is favorable depending on the desired quality. In Table III the data rate savings for the different compression schemes for object-based material are shown. The MPEG-4 pt. 25 reference encoder and our improved encoder perform similar with data rate savings up to 90%, comprehending the benefit of the streaming possibility. The reason for the similar performance is that the improvements mainly address the animation encoding, which is a negligible factor in this scenario.

In Fig. 5 the file size for the different representations is shown with respect to the camera velocity. The constant bit

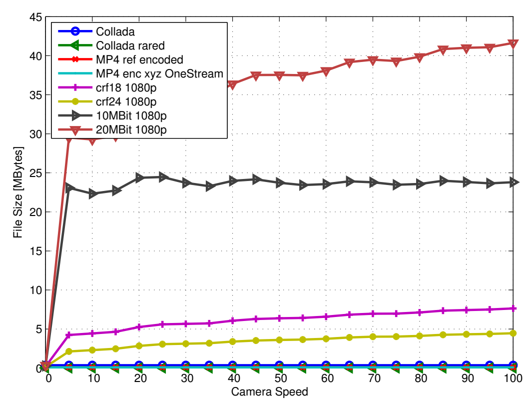

Fig. 5. Scenario 4: The object-based representation is not influenced by camera movement, whereas the pixel-based representation shows growing file size for increasing camera speed.

rate video with $20MBit/s$ limit has not reached the bit rate limit, but the file size is increasing with the camera speed as the variable bit rate videos do. In contrast, the object-based representation files are unaffected by increasing camera movement speeds, since the scene and the number of keyframes in the animation does not change. Table IV reveals, that rar compression is the most effective one for the object-based representation. The two MPEG-4 encoded parts have nearly the same compression ratio, because the main improvement is in the animation coding. For the current

 ISBN: 978-1-61208-195-3

TABLE IV
SCENARIO 4: DATA RATE SAVINGS FOR COMPRESSED OBJECT-BASED REPRESENTATIONS WITH RESPECT TO UNCOMPRESSED COLLADA.

	File Size Saving [%]
Collada rared	95.9 - 95.0
MP4 ref encoded	75.8 - 75.0
MP4 enc xyz OneStream	75.9 - 75.2

TABLE V
SCENARIO 5: DATA RATE SAVINGS FOR COMPRESSED OBJECT-BASED REPRESENTATIONS WITH RESPECT TO UNCOMPRESSED COLLADA.

	File Size Saving [%]
Collada rared	94.5 - 85.9
MP4 ref encoded	89.0 - -458.6
MP4 enc xyz OneStream	88.0 - 80.1

scenario, only the camera and the according scene illumination is animated, which represents only a small part of the whole scene graph.

The results for the last scenario described are shown in Fig. 6. The experiment is comparable to the one shown in

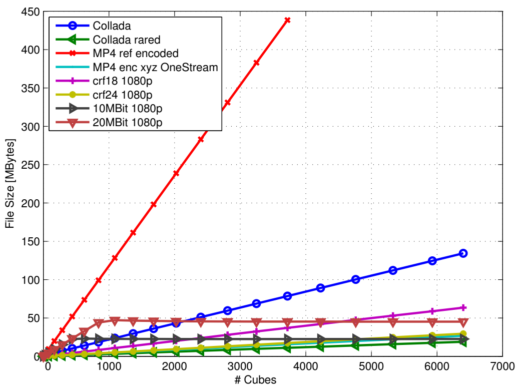

Fig. 6. Scenario 5: Using Textures, the file size grows for all representations. *MP4 ref encoded* is affected the most, because a texture is encoded once for every object it is used on.

[4] despite that textures are used on the cubes. With textures on the surfaces, the complexity of an object is risen compared to simple coloring. No effect can be observed for the constant bit rate video files due to the rate control. The variable bit rate files of the pixel-based representation show higher file sizes than the ones reported in [4]. The increase for *crf18 1080p* is up to 55%. The reason for the increase is the more complex picture content, which degrades the outcome of the temporal prediction. In contrast, the object-based files have a rise of the file size with the size of the texture and a certain overhead for texture mapping and the texture library. A texture itself can be seen as an offset to the object-based representation. For *Collada*, the texture is uncompressed and has a size of $1.02MB$. The other object-based representations use their native compression scheme or known picture compression schemes as JPEG2000 to encode the texture. It is remarkable, that the reference encoder does not recognize the multiple use of one texture. It encodes and writes it to the MPEG-4 stream as often as it is used. Instead of lowering the amount of data to transmit or store, it gets multiplied with that precedure. Our MPEG-4 pt. 25 encoder, however, recognizes multiple use of textures and is able to work effectively when using textures. Comparing the different object-based compressions to *Collada* (Table V), *Collada rared* shows the highest saving. The efficiency of *MP4 ref encoded* is highly dependent on how often a texture is used, as mentioned, and is therefore not comparable to a real encoder. The encoder stopped working at 3969 cubes and the last iteration generated a file, which has 5.586 times the size of the uncompressed Collada file including the texture. If the experiment had run to the last iteration of 6400 cubes, the file size would have been ≈ 9 times the size of *Collada*. *MP4 enc xyz OneStream* shows a remarkable saving between 88.0% and 80.1%.

V. CONCLUSION

Five scenarios have been evaluated to show the effect of different parameters on pixel-based representation and object-based representation. The first scenario shows, that the pixel-based representation has a linear relation to scene length. This is also true for the object-based representation, but the slope is negligible. It has been proven that the two scenarios, increasing animation and camera velocity, show no effect on the object-based representation schemes, as expected, whereas the effect on the variable bit rate video is obvious. A linear relation between the number of vertices and the object-based representation is observed for the third scenario, whereas the pixel-based representation is unaffected or reaches saturation for higher levels of detail. The last scenario shows, that the use of textures does not change the general behavior for a rising number of objects, but the texture itself and its representation inside the object-based file introduces an offset to the file size. Thus, the effect of the discussed factors can be easily predicted for both video representations, when having to choose one of them. Moreover, the compression gain for the object-based representation has been evaluated. Compression ratio improvements of up to 6.5% could be reached for our version of the MPEG-4 pt. 25 codec compared to the reference encoder, not mentioning the data-generating texture compression of the reference encoder. Compared to uncompressed Collada, $67,6\% - 90\%$ file size saving is possible for the investigated scenarios. Up to 90.5% file size saving were observed for long scenes compared to high quality, variable bit rate video.

REFERENCES

[1] R. Steinmetz, *Multimedia Technologie*, Springer, 2nd edition, 1999.
[2] G. J. Sullivan and T. Wiegand, “Video compression - from concepts to the h.264/avc standard,” vol. 93, no. 1, pp. 18–31, Jan. 2005.
[3] B. Jovanova, M. Preda, and F. Preteux, “Mpeg-4 part 25: A graphics compression framework for xml-based scene graph formats,” *Signal Processing: Image Communication*, vol. 24, pp. 101–114, 2009.
[4] J. Wuenschmann, T. Roll, C. Feller, and A. Rothermel, “Analysis and improvements to the object based video encoder mpeg 4 part 25,” in *Proc. IEEE 1st International Conference on Consumer Electronics - Berlin*, September 2011.

