

EFFICIENT AND SCALABLE VIDEO COMPRESSION BY AUTOMATIC 3D MODEL BUILDING USING COMPUTER VISION *

Franck Galpin^{1†}, Raphaële Balter^{1,2}, Luce Morin¹ and Stéphane Pateux^{1‡}

¹ IRISA-INRIA, Université de Rennes 1, Campus de Beaulieu, 35042 Rennes, France

² France Telecom RD, 4 rue du Clos Courtel, 35512 Cesson-Sévigné, France

galpin@laposte.net, raphael.e.balter@irisa.fr,

luce.morin@irisa.fr, stephane.pateux@francetelecom.com

ABSTRACT

This article deals with video coding and compression using 3D modelling. Contrary to classical 3D model-based coding where 3D models have to be known, 3D models are here automatically computed from the original video sequence using computer vision methods. Camera parameters and scene contents are supposed unknown and we only assume a static scene viewed by a moving monocular camera. In this context, we have proposed a video coding scheme based on a stream of 3D models extracted from the video and scalably encoded. This method proves to be competitive and visually superior to the state of the art H264 video coder, especially in the field of very low bitrate coding. An other benefit of this approach is to provide advanced 3D functionalities on the reconstructed video sequence such as virtual object insertion, enlightenment modification, stereoscopic visualization, camera path stabilization, virtual path generation or interactive navigation.

1. INTRODUCTION

Computer vision aims at extracting 3D information from 2D images. Once relying on sets of local features (corners and edges) matched in a few images, computer vision techniques now use large sets of images or video sequences and take into account dense pixel information. Proposed algorithms tend to be more and more fast, robust and generic to image content and acquisition conditions, thus providing video analysis techniques more compatible with video compression requirements.

On the other side, video compression schemes based on

pixel motion compensation are reaching their efficiency limits with the MPEG-ITU H264 video coder. Research now focuses on adding new functionalities (such as scalability or video content manipulation) or proposing content-adapted methods for further increasing coding efficiency.

Using 3D modelling of scene contents instead of 2D motion modelling is used in 3D-model-based coding, where the video sequence is represented with one or several textured 3D models and a set of camera/objects parameters. This topic has been addressed for many years, particularly in the field of visio-conference where a 3D model of the human face is used to represent the video sequence of the speaker [1]. 3D-based coding has several advantages compared to image-based coding: first it allows very low bitrate coding since the only data to transmit are camera parameters (position, orientation, focal length), the 3D model of the scene (if it is unknown on the decoder side) and the corresponding texture image mapped on this 3D model. Moreover, it allows 3D functionalities such as viewpoint transformation, illumination changes or synthetic 3D object insertion. However many works [2] [3] are based on known 3D models, and few results are available concerning 3D-model-based video compression with unknown 3D-models.

In this paper we address the problem of extending the 3D-model-based coding approach to unknown models based on computer vision methods, and to assess the feasibility and efficiency of such an approach.

As an *a priori* known 3D model is unavailable, it should be automatically extracted from the input video sequence using shape-from-motion methods developed in computer vision [4] [5] [6]. However, shape-from-motion is still a difficult problem without simplifying assumptions on scene contents or using a specific camera motion [7] [8].

In the case of video compression, the aim of 3D reconstruction is not to recover an exact geometric model of the scene but to obtain a compact description of the video sequence. It may not be a faithful 3D model of the actual scene, as long as it enables to reconstruct the original images. Ba-

*This study has been achieved within the French national project RNRT-V2NET.

[†]now with Toyota RD, Japan

[‡]now with France Telecom RD, France

sed on this idea, we propose to estimate a set of local 3D models, instead of a unique one containing all the information viewed in the entire video sequence. Each model is a valid representation for a small portion of the sequence, and we thus keep the GOP structure of the classical coding schemes. This choice has several advantages: -global consistency of extracted 3D information is not required, thus allowing to use inaccurate camera parameters, -sequences of arbitrary size can be processed with on the fly estimation and streaming of the 3D models, -camera motion is unconstrained as long as it is not degenerated. In the following, we first briefly describe the extraction of the 3D models and then pay more attention on the compression stage. Results of compression performance on real video sequences are finally shown and discussed.

2. 3D MODELS STREAM GENERATION

We first present the structure of the representation: each 3D model is extracted and used for a small portion of the video sequence called a GOP (Group Of Picture). The GOPs overlap themselves, that is the last image of a GOP is the same as the first image of the next one (see fig. 1). We call these images *keyframes*. Keyframes are automatically selected adaptively to video contents, based on several criteria depending on motion, percentage of outgoing points in images and 3D reconstruction stability [9].

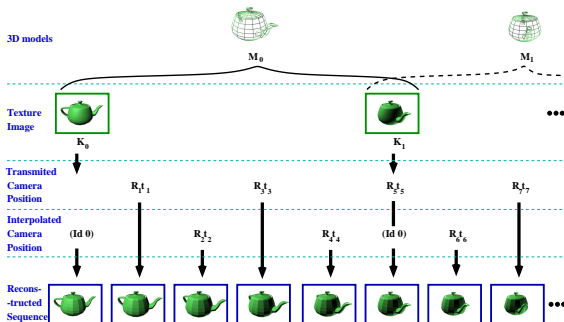


Fig. 1. Reconstruction of the original sequence

For each GOP, a 3D model is automatically extracted on the fly. The principle of extraction is based on shape-from-motion method: we use a dense mesh-based motion estimator using multi-grid and multi-resolution approaches [10]. Camera intrinsic parameters are estimated using a self-calibration algorithm or fixed to approximate values. Accurate intrinsic parameters are not required since we design our representation to be robust to inaccurate intrinsic parameters. Extrinsic parameters are computed using classical

calibration method and an adapted bundle adjustment method [9]. The dense motion field from the first to the last image of the GOP and camera parameters for these two images allow to reconstruct a dense depth map of the first image of the GOP. Figure 2 shows an example of such a depth map extracted from the *Street* video sequence (see results section).



Fig. 2. An example of a depth map (bottom) extracted from the *Street* video sequence. The top figure is the corresponding image in the sequence.

A uniform triangular mesh is then applied on each first image of each GOP which gives a 3D model and its associated texture image (the first image of the GOP). Camera extrinsic parameters are then retrieved for each image in the video sequence using a pose estimation algorithm.

3. STREAM COMPRESSION

Once the original video sequence has been encoded as a stream of 3D models, it is compressed using adapted compression techniques for the different parts of the representation (3D model, texture images and camera parameters).

3.1. Texture images

An IPP scheme is used, as in MPEG for low bitrate, that is: one *Intra* (I) image at the beginning of the video stream followed by *Predicted* (P) images. *Intra* images do not depend of other images, whereas *Predicted* images depend on the previously decoded image.

The first image of the sequence T_0 (*Intra* type) is compressed using a method similar to JPEG2000 standard [12], that is wavelet transform, adaptive quantization (using *EBCOT*) and entropic coding.

The following texture images (*Predicted* type) are predicted using the previously decoded texture image (of the previous GOP) T_n and the previous 3D model M_n :

$$\tilde{T}_{n+1} = Pr(M_n, T_n, C_{n+1})$$

where C_{n+1} is the camera associated with keyframe T_{n+1} and $Pr(M, T, C)$ denotes the perspective projection of model M textured with image T onto camera C .

It produces a prediction of the texture image \tilde{T}_{n+1} similar to a forward motion compensated image in classical video coding scheme (see fig. 3). We then encode the difference image E_{n+1} using the same method as for the *Intra* image.

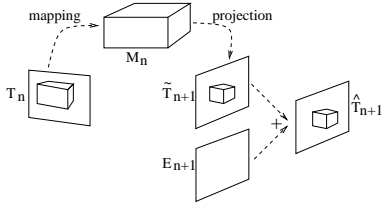


Fig. 3. Principle of predictive coding method of the texture images. The predicted texture image \tilde{T}_{n+1} is computed by mapping and projection of the previous keyframe T_n using the previous 3D model M_n .

3.2. 3D models

3D models can be seen as being extruded from the uniformly meshed depth maps. Each 3D model, that is the decimated depth map, is compressed using wavelet transform and an adapted quantization law: the depth of each vertex of the 3D model is quantized using a normalized inverse law, which provides a linear dependence between 2D reprojection error and quantization step [11].

3.3. Bitrate allocation and adaptation

The representation is effectively compressed, allowing to compute the real coding cost. Figure 4 shows a typical result on the first 3D model extracted from a real video sequence (see figures 9 and 10). It displays 2D reprojection error for the 3D model, depending on the coding cost per triangle in bits. It shows that the 3D model cost is between 2 and 3 bits per triangle for a reprojection error less than 0.5 pixel, which is very low compared to the texture cost. This is confirmed with figure 5: the texture image cost is the major coding cost in the proposed representation.

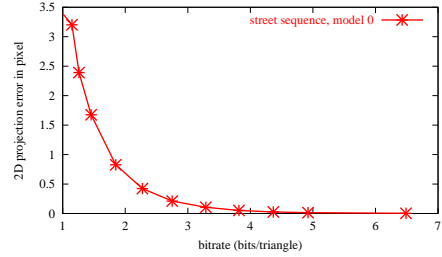


Fig. 4. Typical bitrate obtained with the adapted quantization.

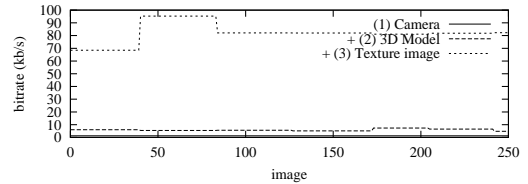


Fig. 5. Coding cost of each part of the representation: (1) camera cost, (2) adding the 3D model cost, (3) adding the texture image cost.

An arbitrary target bitrate R_c can thus be exactly reached thanks to *EBCOT* coding of texture images, by choosing the coding cost of the current texture image R_{text} as:

$$R_{text} = \frac{R_c \cdot (t_{n+1} - t_n)}{fr} - (R_{cam} + R_M) \quad (1)$$

with fr the framerate of the video sequence, R_{cam} the coding cost of the camera parameters, R_M the coding cost of the 3D model and t_n the time at GOP n .

4. VIDEO SEQUENCE RECONSTRUCTION

On the decoder side, the original video sequence has to be reconstructed based on transmitted information.

4.1. Principle

All images inside a GOP are reconstructed by projecting the associated textured 3D model on each camera viewpoint in the GOP, as illustrated in figure 1. More precisely, for reconstructing current image I_c in GOP n , the 3D model M_n textured with keyframe image T_n is projected onto current viewpoint C_c :

$$\hat{I}_c = Pr(M_n, T_n, C_c) \quad (2)$$

where $Pr(M, T, C)$ denotes perspective projection of model M textured with image T onto camera C . As the 3D model is a 3D triangular mesh associated with texture coordinates in texture image T_n , classical rendering techniques and dedicated hardware can be used for real time decoding.

However, because this method uses only the texture of the first image in the GOP, the reconstructed image quality decreases along the GOP, and a visual abrupt change occurs at the transition between two successive 3D models (see Figure 8).

4.2. Models fading and morphing



Fig. 6. Effect of texture fading on *Street* sequence (image 240) (a) without fading (b) with texture fading.



Fig. 7. Effect of geometric morphing on *Stairway* sequence: to visualise the geometric shift, two successive models are projected and superimposed for a viewpoint outside the camera path, without geometric morphing (left) and with geometric morphing(right).

5. RESULTS

The compression scheme performance is first compared to the state of the art standard coder (H264/MPEG4-AVC¹) and we show new results in very low bitrate coding area. Results obtained on the *Street* video sequence² are first presented: format is CIF 4:2:0 at 25Hz, global motion is a translation along z-axis, neither camera parameters nor scene content are known. Internal camera parameters are fixed to typical values and focal length is fixed to an approximate value. The video sequence is compressed with H26L video encoder at 82kb/s which is the minimum bitrate this coder can achieve on this sequence. The sequence

¹used version is H26L-TML coder v.6. which overcomes previous MPEG and ITU standards [13]

²video sequence from Thomson Multimedia and FTRD

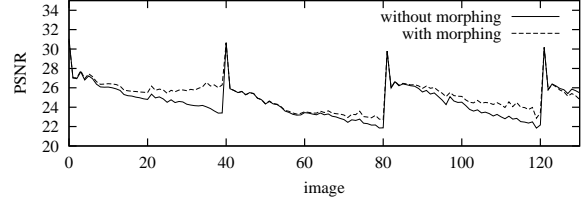


Fig. 8. PSNR score on *Street* video sequence, with and without the fading implementation.

To ensure smooth transition between GOPs, we proposed an adapted texture fading and geometric morphing procedure. Texture fading is based on a linear combination of current texture T_n and the next texture T_{n+1} . Thus, current image I_c is reconstructed as

$$\hat{I}_c = Pr(M_n, T_c, C_c) \quad (3)$$

$$\text{with texture image } T_c = (1 - \gamma) T_n + \gamma \tilde{T}_{n+1} \quad (4)$$

To ensure a correct superposition of textures T_n and T_{n+1} during summation, texture image T_{n+1} is first registered into $\tilde{T}_{n+1} = Pr(M_n, T_{n+1}, C_n)$, which is 3D model M_n textured with T_{n+1} and viewed from camera C_n . The fading factor γ reflects the current camera position, relatively to the total translation from first to last image in the GOP :

$$\gamma = \frac{\|t_c - t_{t_n}\|}{\|t_{t_{n+1}} - t_{t_n}\|} \quad (5)$$

where t_t denotes camera translation vector at time t . To ensure a correct superposition of textures T_n and T_{n+1} during summation, texture image T_{n+1} is first registered into $\tilde{T}_{n+1} = Pr(M_n, T_{n+1}, C_n)$, which is 3D model M_n textured with T_{n+1} and viewed from camera C_n .

Similarly, a current 3D model M_c is computed for each current viewpoint C_c in the GOP by linear interpolation between models M_n and M_{n+1} :

$$M_c = (1 - \gamma) M_n + \gamma M_{n+1}^i \quad (6)$$

with γ previously defined, and M_{n+1}^i obtained by remeshing M_{n+1} so that each vertex in M_n has a correspondent in M_{n+1}^i .

Texture fading and geometric morphing ensure smooth transition between GOPs on original camera path or virtual paths. They take into account global illumination changes along the sequence and reduce texture distortion (see figures 6-7). They also improve PSNR of the reconstructed sequence (see figure 8).

is also encoded with the proposed method denoted *Rec3D* at same target bitrate.

Figure (13) shows *PSNR* scores along the sequence. The *Image* and *H26L* curves denote the *PSNR* of the reconstructed sequence with the *Rec3D* method and the H26L encoder respectively. *PSNR* scores are globally similar for both methods, and better for the *Rec3D* method when the 3D model is refresh (peaks on the second curve). The presented fading and morphing techniques provide quality continuity : *PSNR* increases progressively at the end of GOPs, thanks to using the next GOP's 3D model and texture which provide a this time a better reconstruction than the current GOP's model and texture. An abrupt quality change is thus avoided when changing GOPs.

Distortion on reconstructed images as shown by the *Image* curve is produced both by texture distortion (texture image compression artifacts) and geometric distortion (3D model estimation errors and depth compression artifacts). The *Texture* curve shows the *PSNR* of the decoded texture image, i.e. the quality obtained with texture distortion alone, without geometric distortion. It is dramatically much better than the final obtained quality shown by the *Image* curve. The low *PSNR* value of reconstructed images is thus essentially due to geometric distortion.

On figures (9-12), we compare visual quality of the reconstructed video sequence: the images clearly show that *Rec3D* provides better visual quality. For such a bitrate, H26L coding introduces classical block artifacts and texture degradation whereas *Rec3D* still provides high quality texture images resulting in very good visual quality of the reconstructed video sequence. Note that the better visual quality is not reflected by *PSNR* curves presented above. This is due to geometric distortion which greatly decreases *PSNR* when it may have little impact on visual quality (think of a geometric distortion equal to one pixel translation for a demonstrative example). Indeed visual quality of reconstructed images with *Rec3D* is better reflected by *PSNR* on texture images (*Texture* curve) than *PSNR* on reconstructed images (*Image* curve).

Moreover, the proposed method allows coding at very low bitrates (up to 16kb/s for CIF, 25Hz format) which are not reachable by classical image based coders. Figures (11-15) show results on the *Street* video sequence for a target bitrate of 16kb/s. Figure 15 shows the *PSNR* score along the video sequence: *Texture* curve shows that visual quality remains good for such a bitrate. Figures 11 and 14 show an image extracted from the reconstructed video sequence: despite compression artifacts, global visual quality is still good. One must notice that such a bitrate is considered as extremely low for CIF/25Hz format video sequence.

6. CONCLUSION

We have proposed a vision computer based approach for video compression of large fixed scenes viewed by a moving camera. Results on real video sequences demonstrate that this approach is very efficient for low bitrate coding and also enables very low bitrate coding not reachable by current coders. We show that the proposed scheme allows better performance than classical schemes like H26L in the case of static scene video sequences. One must also notice that the proposed representation intrinsically allows 3D manipulations like stereo-visualization or scene manipulation [9]. In future work, we plan to extend the proposed scheme to scenes with moving objects, making the approach more general for video coding purpose.

7. REFERENCES

- [1] F. Prêteux and M. Malciu, "Model-based head tracking and 3d pose estimation," in *Visual Conference on Image Processing*, San Jose, California, pp. 94–110, 1998.
- [2] B. Girod and al., "3d image models and compression - synthetic hybrid or natural fit?," in *Proc. ICIP*, 1999.
- [3] G. Martìnez, "Joint position estimation for object-based analysis-synthesis coding," in *Proc. VCIP*, 2000.
- [4] M. Pollefeys, R. Koch, M. Vergauwen, B. Dknuydt, and L. Van Gool, "Three-dimensional scene reconstruction from images," in *Proc. SPIE Electronic Imaging, Three-Dimensional Image Capture and Applications III*, 2000.
- [5] R. Kochand M. Pollefeys and L. Van Gool, "Realistic 3d scene modeling from uncalibrated image sequences," in *Proc. ICIP*, 1999.
- [6] M. Magnor and B. Girod, "Model-based coding of multi-viewpoint imagery," in *Proc. VCIP*, 2000.
- [7] A. Fitzgibbon and A. Zisserman, "Automatic camera recovery for closed or open image sequences," in *Proc. ECCV (1)*, 1998, pp. 311–326, 1998.
- [8] D. Nistèr, "Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors," in *Proc. ECCV*, Dublin, Ireland, pp. 649–663, 2000.
- [9] F. Galpin and L. Morin, "Sliding adjustment for 3D video representation," in *Eurasip Journal on Applied Signal Processing*, vol.10, 2002.
- [10] S. Pateux, G. Marquant, and D. Chavira-Martinez, "Object mosaicking via meshes and crack-lines technique. Application to low bit-rate video coding", in *Proc. PCS*, Seoul, Korea, pp.417–420, 2001.
- [11] F. Galpin and R. Balter and L. Morin and K. Deguchi, "3D Models Coding and Morphing for Efficient Video Compression," in *Pro. CVPR*, Washington, 2004.
- [12] Information Technology, "Jpeg2000 image coding system, iso/iec fdis 15444-1," 2000.
- [13] H. Schwarz and T. Wiegand, "The emerging JVT/H.26L video coding standard", in *Proc. IBC*, Amsterdam, 2002.



Fig. 9. *Street* video sequence at 82kb/s: image 100 reconstructed by H26L coder.



Fig. 10. *Street* video sequence at 82kb/s: image 100 reconstructed by Rec3D coder.



Fig. 11. *Street* video sequence at 16kb/s: image 100 reconstructed by Rec3D coder.



Fig. 12. *Street* video sequence at 82kb/s: zoom on image 100 reconstructed by H26L (left) and Rec3D (right) coder.

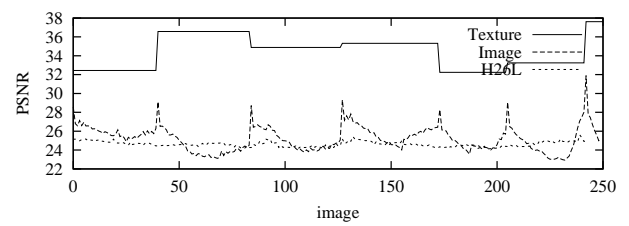


Fig. 13. *PSNR* of *Street* video sequence at 82kb/s: comparison between H26L and Rec3d method.

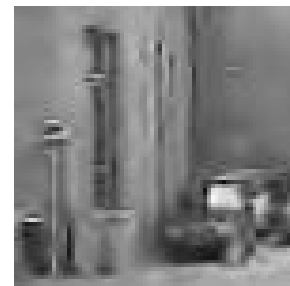


Fig. 14. *Street* video sequence at 16kb/s: zoom on image 100 reconstructed by Rec3D coder.

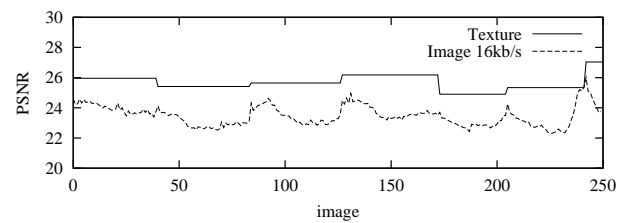


Fig. 15. *PSNR* of *Street* video sequence at 16kb/s compressed with Rec3D method.