COMPLEXITY EVALUATION OF RANDOM ACCESS TO CODED MULTI-VIEW VIDEO DATA

Ulrich Fecker and André Kaup

Chair of Multimedia Communications and Signal Processing
University of Erlangen-Nuremberg
Cauerstr. 7, 91058 Erlangen, Germany
email: {fecker, kaup}@LNT.de, web: www.LNT.de

ABSTRACT

Significant progress has recently been made in the coding of video data captured with several cameras. Applying image-based rendering algorithms to this data, the user can watch the scene from any desired viewpoint. For that, the renderer may need arbitrary images from different camera views. That is why random access to the coded data is crucial. However, in efficient multi-view coding schemes, random access is complicated by the complex relations between the different frames. In this paper, the Joint Multi-View Video Model (JMVM) is exemplarily used to analyse the complexity of random access to multi-view video sequences. It is shown how the effort to access a certain frame depends on its position in the sequence, and the average and maximum complexity to access a randomly selected frame is quantified. The influence of the number of camera views and the length of each group of pictures is illustrated and discussed.

1. INTRODUCTION

Traditionally, objects or scenes are represented by three-dimensional geometrical models when they shall be displayed on a computer screen. Recently, more and more interest can be observed in image-based rendering, where real images are used to render views of the object. For that, multiple images of the object are captured by cameras from different positions. If the object or the scene is moving, a video stream needs to be recorded at each camera. The result is then a so-called multi-view video sequence. From this data, intermediate views can be interpolated, so that the scene can be watched from any desired viewpoint and viewing angle. Image-based rendering can for example be applied in three-dimensional television (3D TV) and free-viewpoint television (FTV) [1], but can also be used in medical visualisations (see e. g. [2]).

As the data rates involved in multi-view video are very high, efficient compression is required. A straightforward method to compress multi-view video data is simulcast coding. This means that the video stream from each camera is coded separately using a classical hybrid video coder, e. g. H.264/AVC. The advantage is that off-the-shelf video coders and decoders can be used. Furthermore, the data of each camera can be encoded without the data from other camera views. In a practical capturing setup, this allows the data to be compressed directly at each camera, and the transmission to e. g. a storage device requires much less data rate. This method is for example used in the light field video camera at Stanford University [3].

However, the amount of data resulting from multi-view video capturing is huge. Therefore, it is desirable to compress the data as efficiently as possible. The data rates after applying classical video coding to each video stream might still be too high to realise practical multi-view video systems. Because the different video streams show similar content — even though there is a certain disparity between them —, they are highly correlated. It is therefore desirable to make use of this cross-view correlation to improve the coding efficiency.

In the recent past, several coding schemes have been proposed for multi-view data. They share the common idea to modify the motion compensation step of classical video coders. For the prediction of blocks in the current frame, not only temporally preceding frames from the same video stream are searched, but also frames from other camera views. In [4], it could be shown that for a significant percentage of blocks in typical multi-view sequences, the prediction efficiency can be improved when spatial prediction across the camera views is introduced.

Proper multi-view coding schemes can therefore lead to substantial coding gains compared to simulcast coding. However, the improvement in the compression efficiency comes at the price that multiple relationships between the different images of the multi-view sequence are introduced. This approach makes the decoding process much more time- and memory-consuming. If a certain frame from the sequence is needed for rendering, several other frames need to be decoded first before the actually desired frame can be accessed.

For practical systems using image-based rendering, this can lead to severe difficulties: To generate intermediate views, a renderer needs images from different cameras. If the viewpoint or viewing direction changes — which may occur very often in systems such as free-viewpoint television —, images from other camera views are needed. As the user shall be able to change his viewpoint freely, it is virtually impossible to predict which images are needed in the next time step.

In the following, the problem of random access to multi-view video data is therefore analysed. First, the Joint Multi-View Video Model (JMVM), which has been used for the analysis, is explained. After that, the complexity of decoding a certain frame within a multi-view sequence is investigated. In the next step, the results are used to compute the average and maximum decoding complexity for random access to multi-view data. Finally, the effect of the length of each group of pictures (GOP) on the decoding complexity as well as on the coding efficiency is addressed.

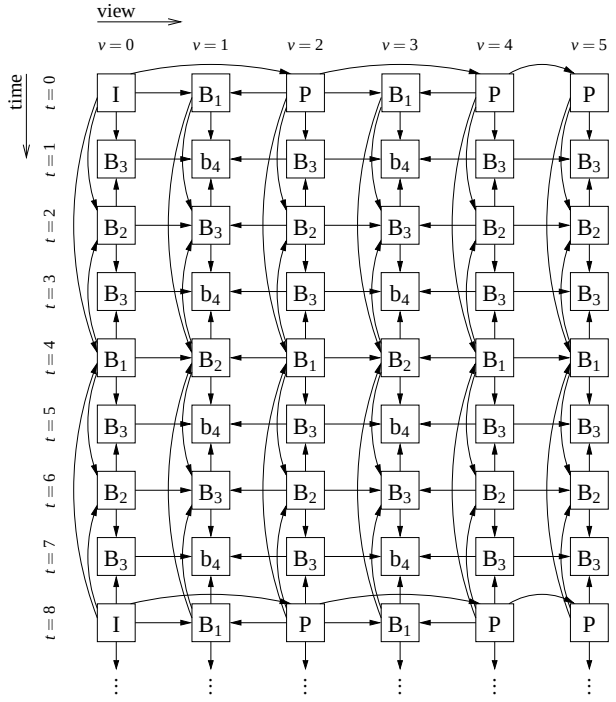

Figure 1: Prediction structure of the JMVM coding scheme (6 views, GOP length: 8) [5]

2. JOINT MULTI-VIEW VIDEO MODEL (JMVM)

In [5], Mueller et. al. suggested a multi-view video coding scheme based on hierarchical B pictures. Due to its good coding efficiency, this scheme was chosen as the basis for a future multi-view video coding standard by the Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEQ [6]. As the problem of random access to multi-view video data strongly depends on the specific prediction structure, this scheme is exemplarily used for the analysis.

The prediction structure is illustrated in Fig. 1. To facilitate random access in the temporal direction, the video sequence is divided into groups of pictures (GOPs) of a certain length. In Fig. 1, a GOP consists of eight frames, and six camera views are used. For other GOP lengths, such as e. g. 12 or 15, the structure is slightly varied. If l_{GOP} denotes the length of the GOP in the temporal direction and n_{views} the number of camera views, the number of frames in one GOP is:

$$n_{\mathrm{GOP}} = l_{\mathrm{GOP}} \cdot n_{\mathrm{views}}$$

For the example of six views and a GOP length of eight, 48 frames are contained in one GOP. However, camera arrays with more than 100 cameras have already been demonstrated, and GOP lengths of 12 or 15 have mostly been used in the JMVM coding experiments. In the case of 100 cameras and a GOP length of 12, a single GOP would contain 1 200 frames. One can easily imagine that random access to each of these frames implies an immense effort. This effort shall be quantified in the following analysis.

3. ACCESS TO A SINGLE FRAME

In this section, it is assumed that a certain frame of a multi-view sequence is requested by e. g. a renderer. The decoder must then decode this frame from the coded bitstream and deliver the resulting image to the renderer. As all frames except the I-frames depend on other frames within the same GOP, several frames must be decoded before the desired frame can be accessed. Of course, not all n_{GOP} frames in the GOP must be decoded, but only the frames actually serving as references to the desired frame.

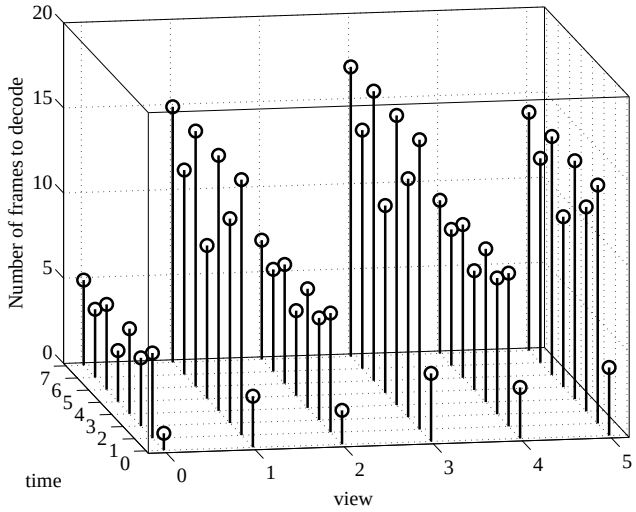

Figure 2: Number of decoded frames needed to access a certain frame depending on its position within the GOP (6 views, GOP length: 8)

In Fig. 2, the number of frames to be decoded for accessing a desired frame is plotted depending on the position of the desired frame in the GOP. This plot has been generated in the following way: For the desired frame, a list of possible reference frames is generated. For each of the frames in the list, their possible references are added. Frames appearing twice in the list are deleted. This is applied recursively as long as the length of the list does not increase any longer. The final length of the list then equals the number of frames which need to be decoded to access the desired frame.

As can be seen from the plot, the decoding complexity strongly depends on the position of the frame within the GOP. Frames with higher hierarchy levels (such as e. g. B_3-frames) are more complex to decode than I- and P-frames. Furthermore, frames in odd views (e. g. $v = 1$ and $v = 3$) are more complex to decode than frames in even views. This is due to the fact that in even views, only the first frame ($t = 0$) depends on other camera views while all other frames only use temporal references. In odd views, however, each frame uses references from the spatial as well as the temporal direction.

4. RANDOM ACCESS TO ARBITRARY FRAMES

In a practical image-based rendering application, a user may navigate freely through the depicted scene. To generate intermediate views, the renderer might need the images of two or even more camera views. In contrast to classical video, where the playback order of the frames is fixed, it is virtually impossible to predict which frame is needed for rendering at a certain point in time. Scenarios such as free-viewpoint television suggest that the different time steps are still played

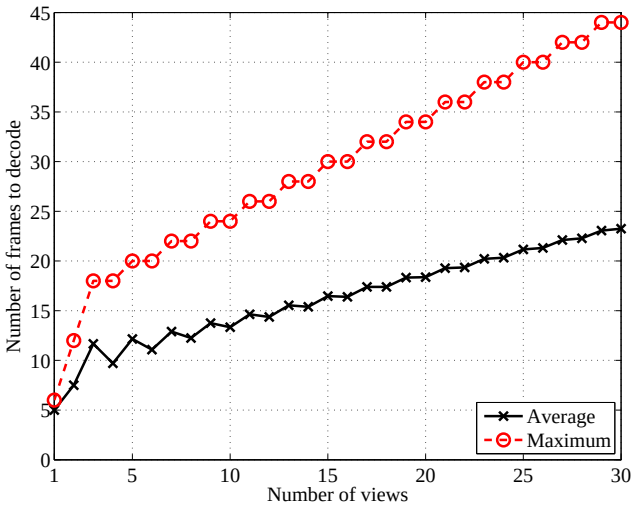

Figure 3: Average and maximum number of frames which need to be decoded to access a frame (GOP length: 12)

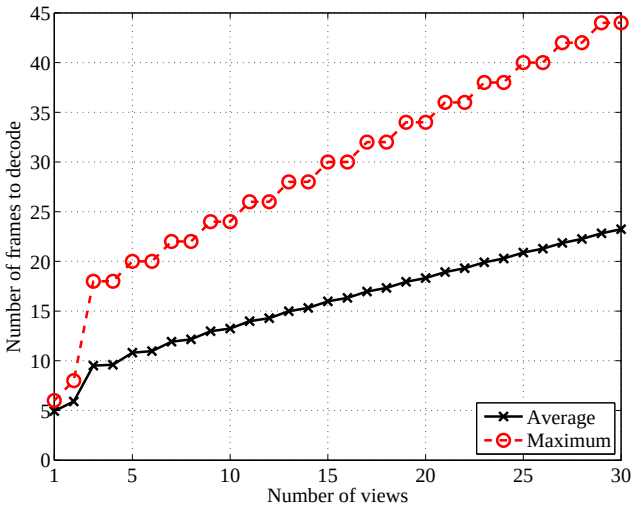

Figure 4: Average and maximum number of frames which need to be decoded to access a frame (GOP length: 15)

back in their sequential order. However, at each point in time, any of the camera views or even more than one view can be requested by the renderer. Other scenarios such as the visualisation of medical data might even switch freely between the different time steps.

To analyse the complexity of the decoding process, it is therefore assumed that a frame is chosen at random from all possible frames and requested for decoding. As it was shown in the last section, the decoding complexity strongly depends on the position of the frame within the GOP it belongs to.

Fig. 3 and 4 show the number of frames which are needed on average to decode a randomly selected frame from a multi-view sequence. This number depends on the number of views of the sequence and also on the GOP length used for encoding. The plots also show the maximum number of frames which need to be decoded in the worst case. This number serves as an upper bound on the decoding complexity.

As can be seen from the graphs, this upper bound does not increase when the number of views is raised from an odd number to the next even number, e. g. from five to six. This can be explained by the special arrangement made to code the last view when the number of views is even (see Fig. 1). In the figure, the last view ($v = 5$) is coded using less hiararchy levels than the view with $v = 3$.

For the same reason, the average decoding complexity may even decrease in some cases when the number of views is raised from an odd number to the next even number. This is however only true for small numbers of views. For large numbers, the average decoding complexity increases linearly with the number of cameras. A good approximation is given by:

$$\bar{c} \approx 0{,}479 \cdot n_{\text{views}} + 9{,}02 \qquad \text{(GOP length: 12)}$$
$$\bar{c} \approx 0{,}484 \cdot n_{\text{views}} + 8{,}80 \qquad \text{(GOP length: 15)}$$

In this equation, $\bar{c}$ denotes the average number of frames which need to be decoded to access a frame at any position within the multi-view sequence. For both GOP lengths, 12 and 15, and $n_{\text{views}} \geq 3$, the maximum number of frames which need to be decoded to access a single frame is given by:

$$c_{\max} = \begin{cases} n_{\text{views}} + 15 & \text{if } n_{\text{views}} \text{ is odd} \\ n_{\text{views}} + 14 & \text{if } n_{\text{views}} \text{ is even} \end{cases} \qquad (1)$$

5. DECODING COMPLEXITY VERSUS CODING EFFICIENCY

The JMVM prediction scheme allows to choose different GOP lengths. Choosing a higher GOP length will reduce the number of I- and P-frames and increase the number of hierarchy levels. This allows for more efficient prediction and leads to a better coding efficiency. This effect is shown in Fig. 5, where the coding efficiency is shown for three possible GOP lengths. The plot shows that the efficiency slightly increases when the GOP length is higher.

One might assume that a higher GOP length will always lead to a larger decoding complexity. This is however not true in all cases. Fig. 6 and 7 show the average and maximum decoding complexity depending on the GOP length for sequences with 8 and 9 views, respectively. Let us focus on the GOP lengths 12 and 15, which have most commonly been used in coding experiments. The maximum number of frames to decode is exactly the same for both GOP lengths, as can also be read off from (1). As can be seen from the plot, the average decoding complexity is even smaller when a GOP length of 15 is used. This is especially the case for small numbers of views.

In this case, a GOP length of 15 therefore is a better choice compared to a length of 12 in terms of coding efficiency as well as in terms of decoding complexity.

6. SUMMARY

For image-based rendering applications, the problem of random access to the coded data is crucial. Accessing the desired frame becomes much more complex when prediction across the different camera views is introduced to improve the coding efficiency. The problem of random access to multi-view video data was therefore analysed using the JMVM prediction scheme as a highly relevant example.

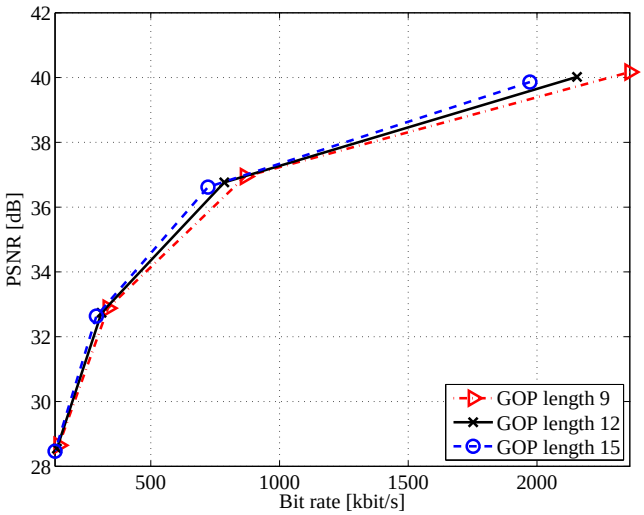

Figure 5: Coding efficiency of the JMVM encoder (version 1.0) depending on the GOP length (Ballroom sequence, 8 views)

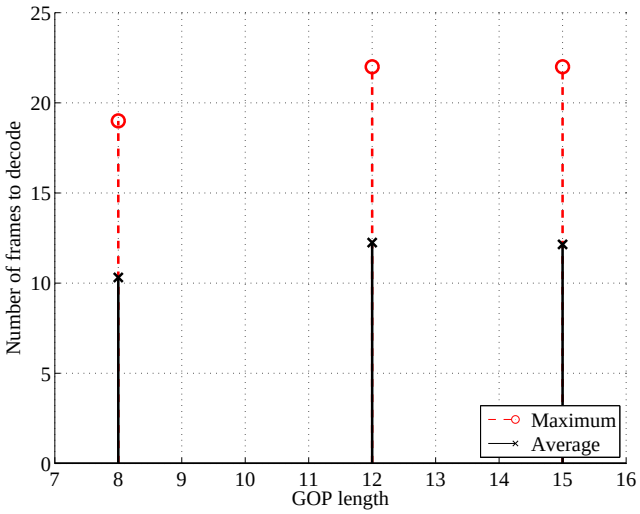

Figure 6: Average and maximum number of frames which need to be decoded to access a frame (8 views)

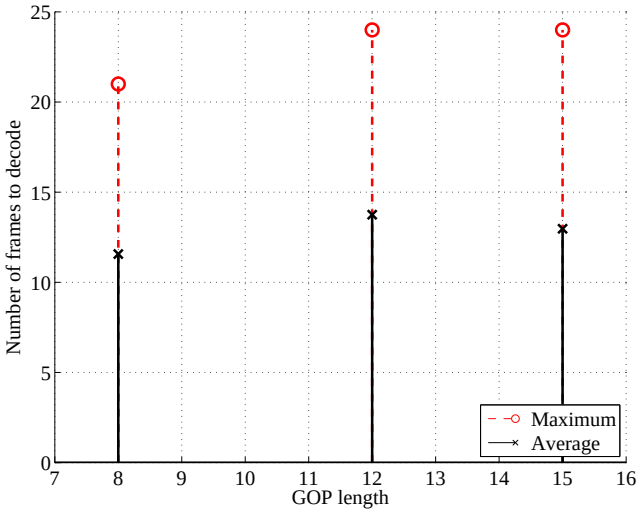

Figure 7: Average and maximum number of frames which need to be decoded to access a frame (9 views)

The number of frames which need to be decoded to access a certain frame was derived, and it was shown that this number strongly depends on the position of the frame within its GOP. Based on the results, the average and maximum decoding complexity could be calculated. The complexity depends on the number of views as well as on the GOP length. If the number of views is not too small, a linear relationship between the decoding complexity and the number of views could be given.

Finally, it was illustrated how the decoding complexity depends on the GOP length. It could be shown that longer GOP lengths — leading to slightly better coding efficiencies — do not necessarily increase the decoding complexity in any case.

7. ACKNOWLEDGEMENTS

This work was funded by the German Research Foundation (DFG) within the Collaborative Research Centre "Model-based analysis and visualisation of complex scenes and sensor data" under grant SFB 603/TP C8. Only the authors are responsible for the content.

The authors would like to thank Felix Rudert for his valuable assistance with the programming work involved in the analysis.

REFERENCES

[1] M. Tanimoto, "Free viewpoint television — FTV," *Picture Coding Symposium (PCS 2004)*, San Francisco, CA, USA, Dec. 2004.

[2] F. Vogt, S. Krüger, J. Schmidt, D. Paulus, H. Niemann, W. Hohenberger, C. H. Schick, "Light fields for minimal invasive surgery using an endoscope positioning robot," *Methods of Information in Medicine*, vol. 43, no. 4, pp. 403–408, 2004.

[3] B. Wilburn, M. Smulski, K. Lee, and M. A. Horowitz, "The light field video camera," *Proc. Media Processors 2002, SPIE Electronic Imaging 2002*, San Jose, CA, USA, Jan. 2002.

[4] A. Kaup and U. Fecker, "Analysis of multi-reference block matching for multi-view video coding," *Proc. 7th Workshop Digital Broadcasting*, Erlangen, Germany, pp. 33–39, Sep. 2006.

[5] K. Mueller, P. Merkle, H, Schwarz, T. Hinz, A. Smolic, T. Oelbaum, and T. Wiegand, "Multi-view video coding based on H.264/MPEG4-AVC using hierarchical B pictures," *Picture Coding Symposium (PCS 2006)*, Beijing, China, Apr. 2006.

[6] A. Vetro, Y. Su, H. Kimata, and A. Smolic, "Joint multi-view video model (JMVM) 1.0," *Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG, Document JVT-T208*, Klagenfurt, Austria, Jul. 2006.