International Conference on Artificial Reality and Telexistence
Eurographics Symposium on Virtual Environments (2018)
G. Bruder, S. Cobb, and S. Yoshimoto (Editors)

Compression Of 16K Video For Mobile VR Playback Over 4K Streams.

I. Vazquez and S. Cutchin
{ikervazquezlopez}{stevencutchin}@boisestate.edu
Boise State University
1910 University Drive, Boise ID, USA.

Abstract

Mobile virtual reality headset devices are currently constrained to playing back 4K video streams for hardware, network, and performance reasons. This strongly limits the quality of 360° videos over 4K streams; which in turn translates to insufficient resolution for virtual reality video playback. Spherical stereo virtual reality videos can be currently captured at 8K and 16K resolutions, with 8K being the minimal resolution for an acceptable quality video playback experience. In this paper, we present a novel technique that uses object tracking to compress 16K spherical stereo videos captured by a still camera into a format that can be streamed over 4K channels while maintaining the 16K video resolution for typical video captures.

CCS Concepts

• ***Computing methodologies*** → *Computer graphics; Image compression;*

1. Introduction

Mobile Virtual Reality (VR) devices are currently constrained to playing back 4K video streams because of hardware, network, and performance reasons. Virtual Reality requires real-time video playback in order to maintain the users immerssive experience. Playback with lag, buffering or jitter will end the users emotional VR experience and the realism of the VR experience. Most existing mobile VR devices have hardware restrictions that impede the decoding of videos at resolutions higher than 4K [BPS18]. Unfortunately, 4K video streams provide insufficient resolution for quality VR video playback experiences [RAAT*17]. This occurs for a few reasons. First, 4K VR video is spherical and while an entire frame is 4K only at most 1/4 of the full frame is presented to the end user at a time. This reduces the apparent resolution to 1024k at any given moment. Second, VR devices place the device very close to the human eye and this yields a strong screen door effect, which exacerbates the impact of the reduction in resolution to 1024K, producing particularly large and fat pixels. Third, since VR playback is stereoscopic the 4K video stream is typically shared between both left and right images. This leads to a resolution of 512K for each eye, or a halving of the frame rate while maintaining 1024K frames. Given the screen door effect it is necessary for VR video to have a resolution of the full device resolution at any given moment in time for a pleasant VR viewing experience. This requires a resolution of 2960x1440 per frame for an effective experience. As such it is necessary to stream video at an effective resolution of 8k or 16k for a quality VR video experience.

Standard compression algorithms such as MPEG [MWS06], achieve high compression rates while maintaining good quality for general videos. However, for the reasons above, MPEG compressed 4K VR video streaming is insufficient. Spherical stereo VR videos can currently be captured at 8K and 16K resolutions, with 8K being the minimal resolution for an acceptable quality video playback experience but, as previously stated, mobile VR headsets cannot handle these resolutions.

To address the mobile VR headset streaming constraint, we propose a solution that segments the videos into multiple distinct objects based on object identification and tracking with the background identified as its own independent object. Identified objects are then compressed as separate video streams using standard MPEG compression and streamed interleaved to a receiving playback headset. For this early work, we limited the problem to the consideration of still camera videos where the background remains still and objects move relative to the camera. Objects may begin and stop moving at any time in the video sequence. New objects may appear and old objects disappear as well as objects crossing over the paths of other objects. In our problem space background pixels do not change their color or intensity and therefore, streaming them repeatedly is inefficient. Segmenting and streaming pixels that belong to objects in motion reduces the overall required bandwidth relatively to standard MPEG compression techniques applied to entire frames. As a direct result, in our test cases the video quality increases with a simultaneous reduction in overall stream bandwidth.

DOI: 10.2312/egve.20181320

In the rest of this paper, we present our initial testing harness that generates individual videos for each of the segmented objects contained in the original 16K video as well as our initial quality metrics and compression comparisons.

The presented approach reduces the required amount of data for streaming and increases the video playback quality compared to standard methods such as MPEG. In some cases achieving compression improvements of ninety percent.

Our implemented technique depends upon the existence of a dedicated GPU on the playback device with support for hardware texture mapping with programmable u, v coordinates and polygonal geometry.

2. Related work

Current work for VR video streaming focus on *Field Of View* (FOV) streaming where an interaction between the content provider and the remote user is required. In [SSHS16], authors present a FOV tile based approach for High Efficiency Video Coding (HEVC) for HMD and state the constraints of FOV streaming, such as encoding overhead when each user's FOV is encoded. Following this methodology, *Bassbouss* generates in [BPS18] 16K content which uses server-side transformations to stream the current FOV content to the end user. This content is used in [BSB17] where they pre-render relevant video FOVs to ensure an efficient streaming of high quality videos. One of the limitations of this approach is that the FOV is not a rectangular area in the source equirectangular video, therefore in [RAAT*17] they use an improved tiling method to stream the content that address this.

To evaluate the quality of the streamed videos, most of the works [BPS18, BSB17] reviewed use the Peak Signal-to-Noise Ratio (PSNR) [HTG12] metric due to its standardization. However, PSNR shows some problems when dealing with non-rectangular FOV content and because of that, authors in [XHYV17] present a new evaluation framework including and extending a Spherical PSNR [YLG15] metric to an Area-Weighted PSNR (AW-PSNR). For each possible FOV in a VR video, the respective PSNR score is computed and then AW-PSNR weights the score using the area that FOV captures from the equirectangular VR video frame.

3. Object based compression

In still camera VR videos, pixels belonging to moving objects have color and intensity changes while background pixels typically remain relatively unchanged. Streaming background pixels that don't change is clearly inefficient and wastes bandwidth. In order to avoid sending unchanged pixels, we segment the video frames into individual objects that have some motion in the video, track them, and generate an individual video for each object. Each of these videos captures a single moving object through the entire video and the size of the video is directly related to the pixel size of the captured object. By applying this technique, we get a set of smaller videos capturing each of the objects and their surrounding area together with their respective motion information.

We identify moving objects' using a simple thresholded pixel color/intensity method. When an object moves, pixels that belong to the object have different color and intensity values with respect to the previous frame. Similar to [TOB*97] and as shown in Figure 1b, we compute image differences from two contiguous video frames to detect which pixels are part of an object. We can, by setting a threshold for this operation, discard noise in videos that could lead to incorrect object segmentation. We empirically verified that a threshold value of 20 shows good performance for motion detection by removing most of the noise.

We traverse all frames within the video and identify and number all objects across the video.

There exist cases where simple thresholding is not enough to separate objects from noise and, in order to discard the noise, further processing is required. We use image processing morphological operations such as erosion, dilation, open and, close to generate object blobs and, at the same time, remove noisy pixels. Based on texture composition of the object, our motion detection method fails to detect objects of certain types (e.g. smooth surface objects, such as in the video used for testing, Figure 1a and 1e). As seen in Figure 1b, only the boundaries of the ball are detected as motion pixels and the object is split in two. By applying a combination of image morphological operations, we connect pixels from disjoint regions that are part of a single object, Figure 1c. From this, for every video frame, we generate **blobs** for each of the objects to segment them from the background.

Blobs represent where objects with motion are in the frame but do not provide any information about which pixels belong to which object. Connected Components (CC) assign the same unique id to pixels belonging to a single tracked object. Using those ids, we segment the pixels, and generate an individual binary mask per connected pixels. These masks locate objects in each of the frames. Objects have diverse shapes and sizes and instead of coding each shape individually as in [TOB*97] we use a bounding box approach. Based on generated masks, as seen in Figure 1d, we compute the bounding box of each object at each frame. The size of the object could change over time and thus the size of the bounding box can also change.

To address this, we fix the size of the bounding box of an object to the maximum size the bounding box across all frames. In order to do that, first we track each object's bounding box searching for the closest one in the previous frame, and then, we assign the unique id of that object. The tracking algorithm at this time is quite naive and we are working on more sophisticated solutions based on existing techniques from the literature. The initial naive technique however, shows promising compression and quality results.

We collect initial position and all movement data for every object across all frames. Based on this movement information, the bounding box and the pixels for each object we generate individual videos for the changing data within each individual object. This produces for each object a video where its shape, color and surroundings are captured for every frame in the original video.

Object videos capture a small area of the input video, and when these videos are packed together with the tracking information for streaming purposes, the background pixels are missing. We infer the video background by keeping track of pixel values that do not belong to objects. Once we process the entire video, each pixel has

(a) *Input video.* **(b)** *Consecutive frame difference.* **(c)** *Mask generation* **(d)** *Bounding box generation and tracking.* **(e)** *Object video.*

Figure 1: *Pipeline of the object based compression method.*

a set of candidate values from individual frames that can qualify as the background value. We use a voting technique to determine the final value of pixels that are part of the background image. This background selection method will fail if a moving object sits still throughout the majority of the video. We are developing an extended tracking and masking technique to deal with this case and others.

The last step of the method, involves aggregating all the computed components from the video (video background image, individual object videos and tracking information) which are packed and ready to stream them to the remote user. This format allows different types of streaming that we discuss as future work in Section 8

4. Video reconstruction

To reconstruct the original video from the compressed format, we use the streamed object videos and their motion tracking information. In order to re-generate the original video frames, we use the segmented video background inferred from background voting process. The voted background image becomes the background of each frame. We know when (in which video frames) an object appears and, making use of the object's tracking information, we replace the corresponding background region with frames of the object video. This simple compositing technique reconstructs the original video.

5. Testing environment

For this initial method, we limited the scope of the video compression problem and created ideal world synthetic videos using still cameras and constantly moving objects. Still camera videos have stationary backgrounds which have small changes or do not change at all. We assumed that lighting conditions do not have smooth transitions because our moving object detection method does not consider small color/intensity changes. If a pixel's value changes smoothly over time, the motion detection fails to detect it and we do not create a video for that object.

Tracked objects that stop at a certain frame for a period of time are not detected due to their lack of movement. Therefore, we only consider constantly moving objects around the camera to avoid those cases and we generated synthetic videos at 16K, 8K, 4K and 2K frame sizes.

To evaluate video quality, the existing works [MWS06] use the standard metric Peak Signal-to-Noise Ratio (PSNR) [HTG12]. We use this metric to compare the resulting quality of MPEG compression algorithm and our initial methodology. We compressed raw videos using MPEG compression algorithm and our methodology, and then we compared the reconstructed video quality, i.e. the PSNR score, and the compressed file size of both videos.

	Size (Mb)		PSNR (db)	
Video	**MPEG**	**Ours**	**MPEG**	**Ours**
16K	180	82	31.62	77.45
2K	22.5	5.2	30.33	75.83

Table 1: *Video size and PSNR score comparison.*

6. Results

Our early testing results demonstrate an improved compression rate compared against direct MPEG compression. However initial tests are currently limited. Additionally our compression ratio for the 16K files is not as good as for our 2k files which appears counter-intuitive to the core method.

The PSNR values for our method were notably superior to the MPEG compression methodology with PSNR results being similar for both testing sizes. It is possible that there is an opportunity for improvement in our compositing method that would improve our PSNR scores overall, particularly for the 16K version.

7. Conclusions

In this paper, we presented an initial method that shows promising results for 16K video compression and streaming. We use naive methods to get initial results which need improvement, but establish a simple modular framework that supports extension via plugin for the core elements. This should enable rapid expansion and experimentation with alternate choices for improved performance and enhanced capabilities. The achieved file sizes and PSNR scores demonstrate that, compared to standard MPEG, our object based compression approach could be used for streaming 16K videos using less bandwidth while maintaining high visual quality.

8. Future work

This initial compression method works well for a set of test cases with application to VR viewing. However, a long list of improve-

ments are necessary for it to be useful in a general manner. The following items are planned for improving its function.

Replacing our initial naive methods with machine learning and better rule based solutions will increase the overall video quality and the PSNR score compared to traditional methods such as MPEG.

Extension of our ideal world from Section 5 to work with all varieties of moving objects such as: Objects can stop, start moving or stop and restart their movement, and we have to take those cases into consideration when generating the compressed object videos.

It is crucial for video quality to identify still objects that move at some point of the video because if not, they will not be displayed when they remain stationary due to the naive movement detection method.

Additionally, in the longer term creating methods to deal with a moving camera and integration with MPEG streams would allow the technique to function as a drop in for all video compression systems.

Finally, our object compression approach can utilize a sprite based playback system where videos are directly projected into sprites in front of the background. This would allow us to study replacing the bounding box objects with more complex polygonal objects potentially increasing our compression ratio.

Packing frames of individual objects into a single frame and transmitting their u and v coordinates, permits a direct GPU based video reconstruction and playback.

If this long list of problems can be overcome the proposed method could be utilized as a general technique for video compression without limitation of video content.

References

[BPS18] BASSBOUSS L., PHAM S., STEGLICH S.: Streaming and playback of 16k 360° videos on the web. In *Communications Conference (MENACOMM), IEEE Middle East and North Africa* (2018), IEEE, pp. 1–5. 1, 2

[BSB17] BASSBOUSS L., STEGLICH S., BRAUN S.: Towards a high efficient 360° video processing and streaming solution in a multiscreen environment. In *Multimedia & Expo Workshops (ICMEW), 2017 IEEE International Conference on* (2017), IEEE, pp. 417–422. 2

[HTG12] HUYNH-THU Q., GHANBARI M.: The accuracy of psnr in predicting video quality for different video scenes and frame rates. *Telecommunication Systems 49*, 1 (2012), 35–48. 2, 3

[MWS06] MARPE D., WIEGAND T., SULLIVAN G. J.: The h. 264/mpeg4 advanced video coding standard and its applications. *IEEE communications magazine 44*, 8 (2006), 134–143. 1, 3

[RAAT*17] RONDAO ALFACE P., AERTS M., TYTGAT D., LIEVENS S., STEVENS C., VERZIJP N., MACQ J.-F.: 16k cinematic vr streaming. In *Proceedings of the 2017 ACM on Multimedia Conference* (2017), ACM, pp. 1105–1112. 1, 2

[SSHS16] SKUPIN R., SANCHEZ Y., HELLGE C., SCHIERL T.: Tile based hevc video for head mounted displays. In *Multimedia (ISM), 2016 IEEE International Symposium on* (2016), IEEE, pp. 399–400. 2

[TOB*97] TALLURI R., OEHLER K., BARMON T., COURTNEY J. D., DAS A., LIAO J.: A robust, scalable, object-based video compression technique for very low bit-rate coding. *IEEE Transactions on Circuits and Systems for Video Technology 7*, 1 (1997), 221–233. 2

[XHYV17] XIU X., HE Y., YE Y., VISHWANATH B.: An evaluation framework for 360-degree video compression. In *Visual Communications and Image Processing (VCIP), 2017 IEEE* (2017), IEEE, pp. 1–4. 2

[YLG15] YU M., LAKSHMAN H., GIROD B.: A framework to evaluate omnidirectional video coding schemes. In *Mixed and Augmented Reality (ISMAR), 2015 IEEE International Symposium on* (2015), IEEE, pp. 31–36. 2