This is the author accepted manuscript of a paper accepted for the 15th ACM Multimedia Systems Conference (MMSys 2024); 15-18 Apr 2024, Bari, Italy.

# Efficient viewport prediction and tiling schemes for 360 degree video streaming

Jayasingam Adhuran and Maria G. Martini
j.adhuran@kingston.ac.uk
m.martini@kingston.ac.uk
Kingston University London
London, UK

## ABSTRACT

360-degree video streaming for VR visualisation is characterised by large transmission data volume and stringent interactive latency demands; hence guaranteeing suitable transmission quality, while meeting the existing constraints, is highly challenging. This paper addresses the relevant "grand challenge" presented at MMSys 2024 on 360-degree video on-demand streaming, aiming at designing and implementing a 360-degree video on-demand streaming solution using the open-source evaluation platform E3PO. The proposed solution incorporates several strategies including viewport prediction, tiling, encoding tile selection for streaming and upsampling. The source code for the proposed solution will be publicly available on Github[1].

## CCS CONCEPTS

• **Applied computing**; • **Information systems**; • **Computing methodologies** → **Artificial intelligence**;

## KEYWORDS

360-degree video, virtual reality, viewport prediction, quality, adaptive tiling

**ACM Reference Format:**
Jayasingam Adhuran and Maria G. Martini. 2024. Efficient viewport prediction and tiling schemes for 360 degree video streaming. In *ACM Multimedia Systems Conference 2024 (MMSys '24), April 15–18, 2024, Bari, Italy.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3625468.3653425

## 1 INTRODUCTION

The emergence of Virtual Reality (VR) applications has enriched the user experience in terms of interactive and immersive media consumption [33]. VR technologies have been developed based on omnidirectional videos, which are commonly known as 360° videos. A 360° space is generated such that a sphere of uniform radius encloses the space around a source point. Assuming that the viewpoints are uniformly distributed, the observable 3-D space of such videos becomes isotropic. Thus, unlike in traditional 2-D videos, the observation space of omnidirectional videos could be defined as a spherical surface. These omnidirectional videos are known as spherical or 360° videos and can be represented by the parameters of the spherical coordinate system, latitude, longitude, and radius ($\theta \in (-\pi/2, \pi/2), \phi \in (-\pi, \pi)$, and $R$ respectively). They are often projected to 2-D space to support the existing video processing procedures.

In 360° videos, the Field of View (FOV) of the user encloses only a portion of the spherical information, also known as the viewport, which is a rectilinear image generated from the user's head position. Since the user's observation path is unknown when encoding is performed, the standard encoding procedure of a 360° video does not address the user-observed viewports, resulting in an abundance of non-observed video information being coded, hence an additional transmission cost. To exploit the fact that the FOV of the user encloses only a portion of the spherical information, state-of-the-art 360° video streaming technologies specifically address the characteristics of 360° videos and viewports of interest for the user. However, further optimization is necessary to address the bandwidth cost and improve the quality of user-observed viewports.

## 2 PROBLEM STATEMENT

The aim of this work is to maximize the objective video quality of the user's actual viewing area on the terminal device, measured by Mean Square Error (MSE), while minimizing the resources. In terms of system resources, three major costs are considered:

- the bandwidth cost $C_b$ of streaming all data from the server to user;
- the storage cost $C_s$ of storing video data on the server;
- the computation cost $C_c$ for some solutions that require real-time processing or transcoding.

.

The final metric to maximize is the following:

$$S = \frac{1}{\alpha MSE + \beta(w_1 C_b + w_2 C_s + w_3 C_c)} \tag{1}$$

where $\alpha = 0.006$, $\beta = 10$; $w_1 = 0.09$, $w_2 = 0.000015$, and $w_3 = 0.000334$. The unit for $C_b$ and $C_s$ is GB, and $C_c$ represents the duration of the video playback in seconds.

The solution was expected to be implemented in the E3PO framework (Bytedance) [11], schematized in Figure 1, in the blocks reported with blue text. The details of the implementation of the solution are reported below in the paper (Section 4).

[1] https://github.com/Adhuran/GC_MMSys_Challenge


*MMSys '24, April 15–18, 2024, Bari, Italy*

ACM ISBN 979-8-4007-0412-3/24/04.
https://doi.org/10.1145/3625468.3653425

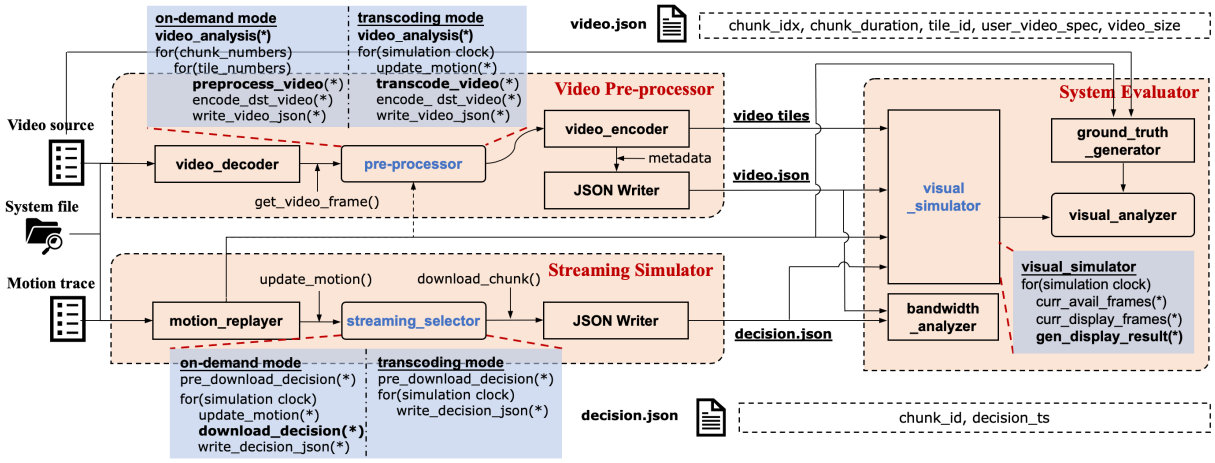



**Figure 1: E3PO Framework [11]**

# 3 RELATED WORK

## 3.1 Projection of spherical video

In order to process and encode appropriately the relevant data, 360° videos are usually projected into 2-D space, and rearranged in rectilinear formats. The respective inverse operations at the decoder are also performed to retrieve the original representation. Furthermore, padding may also be applied as part of the content projection process, in order to reduce the discontinuity along the boundaries of the video frame [13]. The most commonly used technique is a single face rectilinear mapping projection called the EquiRectangular projection (ERP). Although areas do not match between a sphere and rectangular surface, ERP involves one-to-one mapping of pixel values from the spherical image in the 3-D domain to the rectilinear image in the 2-D image, causing sample density discrepancies. Other projection and packaging techniques include rhombus dodecahedron projection [18], CubeMap (CMP) [41], octahedron projection [22], Truncated Square Pyramid (TSP) [34], icosahedron projection [7], adjusted cubemap [15], rotated sphere projection [1] and segmented sphere projection [40].

## 3.2 360-degrees video compression

360° video compression involves several stages. A 360° video is captured using an omnidirectional camera and the several images obtained are stitched to create a spherical video frame [31]. Following projection into 2-D space, video is encoded using a conventional 2-D video codec. In the post-processing stage, video is decoded and the respective inverse operations to packaging and projections are performed. Finally, viewports are generated as per the user's head movements to deliver the 360° video content to the user.

Several solutions have been proposed recently for 360° video coding, in particular to boost compression efficiency [2, 4, 6, 12, 20, 27, 38]. In general, these research works on 360° video coding can be classified into three categories, namely: pre- and post-coding, context adaptive coding, perceptual coding. Pre- and post-coding primarily refers to the re-projection techniques of the existing 2-D projected 360° video contents as well as the packaging mechanisms of the video frames. In contrast, in context adaptive coding, the spherical properties of the 360° videos are exploited and used with the video compression tools such as quantization and motion compensation in order to improve the coding efficiency. Finally, perceptual coding addresses the deployment of user perceptual models, specifically viewport dependent encoding schemes in the video coding processes.

Perceptual video coding has been a vastly researched area in the video coding domain which focuses on enhancing the user perceived visual quality by improving the fidelity of the regions of interest (ROI) [14]. While in the context of conventional videos ROI detection and ROI based video coding is a vastly investigated research topic [23, 29, 36], in 360° videos the identification of the regions of human interest is very challenging because the viewports that represent user observed regions are constructed instantly according to the user's head movement. Viewport centric coding methodologies can be generally categorized as tiled and non-tiled approaches. Benefiting from parallel processing features, tiled approaches are mainly applicable for streaming of the 360° video content. As such, viewport based ROI video coding research works either follow a scalable coding approach [24, 30] or assign high bitrates to the tiles that represent primary viewports [5, 17, 28]. In the scalable coding approach multiple layers are encoded at different qualities, with the base layer being encoded at the lowest quality. However, the major drawback in this approach is the requirement to transmit the base layer at all times which can increase the bandwidth.

## 3.3 Viewport prediction for 360° video streaming

In order to stream the appropriate portion of the 360° video, avoiding wasting resources to stream regions the user is not interested in, future viewport prediction techniques can be adopted. For example, the authors in [24] propose Weighted Linear Regression (WLR) based future viewport prediction to identify the tiles which need to be transmitted. This methodology reportedly achieves over 28% coding efficiency vs. scalable HEVC codec at the enhancement layers. This method benefits from increased Quality of Experience (QoE) performance compared to non-scalable tiled approaches; predicting user observed viewports to select the tiles brings higher coding gains

compared to a tiled approach that does not apply viewport prediction technique. For example, the authors of the research work in [17] deploy two viewport predictions which are used in adaptively selecting Quantization Parameters (QPs) for different tiles. This approach results in an increased coding gain of 45% over viewport-independent approach when Viewport PSNR (VPSNR) [39] is used as the quality metric in assessing the coding gain. The literature has also addressed the combination of the viewport dependent tiles with scalable solutions to ensure the QoE of users is enhanced [16, 32]. Recent works focus on semantic-aware prediction [35] [8], saliency [37], real-time prediction [19], and deep-learning and bidirectional optical flow based prediction [3].

Although tiled based coding systems are useful for streaming purposes as they reduce transmission delays, provide higher flexibility and improve the QoE, the associated coding losses remain an issue.

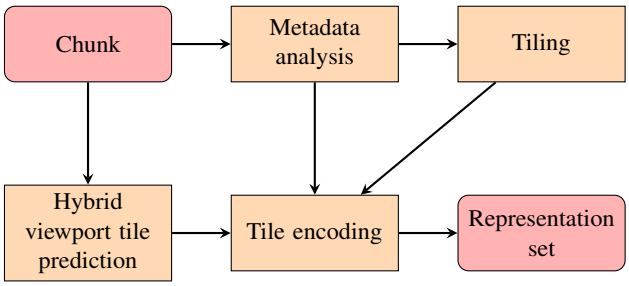



**Figure 2: Overview of the pre-processing proposed strategy.**

## 4 PROPOSED STRATEGY

This work aims at delivering a solution for the on-demand streaming scenario of 360° videos. In this context, as shown in Figure 1, E3P0 breaks the total process into three phases: pre-processing, decision making and evaluation. In the pre-processing phase, potential tiles of the video chunks are retrieved, encoded and stored. In the decision making stage, tiles with potential match for a particular user are selected. Finally, during the evaluation, the selected tiles are used to retrieve the user observed viewports based on realtime head movement data and network conditions, including bandwidth limitations and rendering and transmission delays.

### 4.1 Pre-processing

*4.1.1 Projection.* We adopted equirectangular projection (ERP) for the main tiles and Equi-Angular Cubemap (EAC) for the background/low resolution version of the video, as reported in the example in Figure 3 for the *Release_video_1* video sequence that was used in the test phase (reported in Section 5). We define width and height of the projected source video as $W$ and $H$. A block diagram of the subsequent pre-processing steps is reported in Figure 2 and the different steps are described in more detail below.

*4.1.2 Metadata analysis.* Here, we analyze the metadata of the source video, such as frame rate, colour space, chroma subsampling, which are subsequently used during hybrid viewport tile prediction and tile encoding, explained in Section 4.1.4 and Section 4.2 respectively.

*4.1.3 Tiling.* Four tiling modalities, corresponding to different bitrates, are considered and reported in Figure 4. In all of them, the top and bottom "large tiles" are the same, while different tiling methods are used for the central portion of the video.

The rationale for these different tiling modalities is to address the compromise between:

- need to select for streaming only the portions relevant to the viewer;
- higher coding efficiency obtained for larger portions of the scene due to better exploitation of redundancy.

For instance, if tiles 1 and 2 from the first representation are selected by the streaming selector, these are directly streamed. However, if tiles 1, 2, 3, 4 from representation 1 are selected by the streaming selector, tile 1 in representation 2 is streamed, since this would achieve better compression efficiency.

The four tiling modalities are described in detail below.

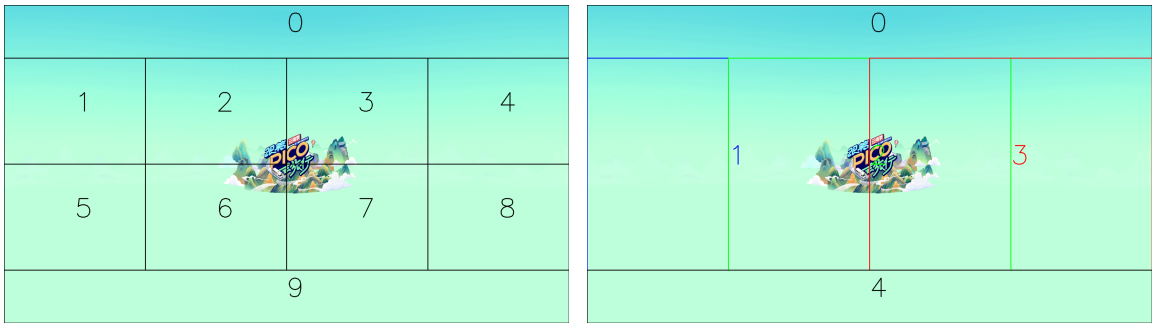
- Representation 1 ($R1$) (Figure 4(a))
  The scene is divided in 10 tiles.
  1st tile: width=$W$, height = $\frac{1}{6}H$, starting at (0,0);
  10th tile: width=$W$, height $\frac{1}{6}H$, starting at (0, $\frac{5}{6}H$);
  tiles 2-9 have equal size, each of width = $W/4$ , height = $\frac{1}{3}H$.
- Representation 2 ($R2$) (Figure 4(b))
  The scene is divided in 5 tiles.
  1st tile: width= $W$, height = $\frac{1}{6}H$, starting at (0,0);
  5th tile: width=$W$, height = $\frac{1}{6}H$, starting at (0, $\frac{5}{6}H$);
  overlapping tiles
  3rd tile: width = W/2 , height = 2/3 H, starting at (0, 1/6 H)
  4th tile: width = W/2 , height = 2/3 H, starting at (1/4 W, 1/6 H)
  5th tile: width = W/2 , height = 2/3 H, starting at (2/4 W, 1/6 H)
  (only one out of the 3rd, 4th, or 5th tile will be selected).
- Representation 3 ($R3$) (Figure 4(c))
  The scene is divided in 6 tiles.
  1st tile: width = $W$, height = $\frac{1}{6}H$, starting at (0,0);
  6th tile: width =$W$, height = $\frac{1}{6}H$, starting at (0, $\frac{5}{6}H$);
  remaining 4 tiles equally split with width = 1/4 W and height = 2/3 H.
- Representation 4 ($R4$) (Figure 4(d))
  The scene is divided in 3 tiles.
  1st tile width=$W$, height = $\frac{1}{6}H$, starting at (0,0);
  2nd tile: width=$W$, and height = 2/3 $H$;
  3rd tile: width=$W$, height = $\frac{1}{6}H$, starting at (0, $\frac{5}{6}$ H).

*4.1.4 Hybrid viewport tile prediction.* Hybrid viewport tile calculation is performed on the 2-second video chunk to predict the tiles for representations that are needed to be encoded. This prediction process includes two independent techniques: Salience calculation and VPredSCNN [3] based tile estimation. Salience calculation is performed every 5 frames (1/6 s), based on an algorithm inspired by the work in [26] and the code in [25]. We also use viewport-centric VPredSCNN network to predict the tiles. This is performed every 32 frames in order to reduce the computational complexity.

(a) EquiRectangular Projection (ERP) representation

(b) Equal Area Cubemap (EAC) representation

**Figure 3: Projections of omnidirectional videos (*Release_video_1*) used in the proposed strategy.**




(a) Representation 1

(b) Representation 2

(c) Representation 3

(d) Representation 4

**Figure 4: Tiling schemes used in the proposed strategy.**

## 4.2 Video encoding

The 2s chunks for the different representations are encoded via H.264 as per E3PO default parameters (that could not be modified). Only tiles predicted using the Hybrid viewport tile prediction techniques are encoded. Moreover, tiles {0,9}, {0, 4}, {0, 5} and {0, 2} from representations $R1$, $R1$, $R3$ and $R4$ are dropped from the encoding process as these areas have higher pixel redundancies due to the projection methodologies.

The tiling and coding processes determine the storage cost $C_s$.

## 4.3 Streaming selector

*4.3.1 Gaze prediction.* The gaze prediction algorithm adopted is a slightly modified version of the algorithm in E3PO [11].

The tiles delivered to the client are only those matching the predicted gaze and hybrid viewport tile prediction data. The tile selection algorithm is described in detail in Algorithm 1.

This determines the bandwidth cost $C_b$ of streaming data from the server to the user.

**Algorithm 1** Proposed Tile Selection Algorithm

**Input:** Tile set $T_p$, $T_g$ and $T_c$
**Output:** Selected tile set $T$

Obtain predicted tiles $T_p$ from Hybrid viewport tile prediction
Obtain predicted tile $T_g$ from E3PO gaze prediction
Obtain predicted tile $T_c$ from E3PO cache for selected tiles
Obtain tiles $T^i_{r1}$, $T^i_{r2}$, $T^i_{r3}$ and $T^i_{r4}$ from representations $R1$, $R2$, $R3$ and $R4$ where $i$ is the tile number. $i = [1:8]$ for $R1$, $i = [1:3]$ for $R2$, $i = [1:4]$ for $R3$ and $i = 1$ for $R4$.
$T_s = T_g \cap T_p$
Set $t = t_s, \forall\ t_s \in T_s$, and $t \in T$ where $T$ is the selected tiles set.
**if** $(t \in T^1_{r4}) \vee (T^1_{r4} \notin T_c)$ **then**
    Set $T \leftarrow \{\}$
    Insert $T^1_{r4}$ into $T$
**else**
    **if** $T^1_{r2} \subset T_c$ **then**
        **if** $T^1_{r1}, T^2_{r1}, T^5_{r1}, T^6_{r1}, T^1_{r3}, T^2_{r3} \subset T$ **then**
            Remove from $T$.
        **end if**
    **end if**
    **if** $T^2_{r2} \subset T_c$ **then**
        **if** $T^2_{r1}, T^3_{r1}, T^6_{r1}, T^7_{r1}, T^2_{r3}, T^3_{r3} \subset T$ **then**
            Remove from $T$
        **end if**
    **end if**
    **if** $T^3_{r2} \subset T_c$ **then**
        **if** $T^3_{r1}, T^4_{r1}, T^7_{r1}, T^8_{r1}, T^3_{r3}, T^4_{r3} \subset T$ **then**
            Remove from $T$
        **end if**
    **end if**
    **if** $\{T^1_{r1}, T^2_{r1}, T^5_{r1}, T^6_{r1} \subset T\} \wedge t_t \in T_c$ where $t_t \in \{T^1_{r3}, T^2_{r3}, T^1_{r1}, T^2_{r1}, T^5_{r1}, T^6_{r1}\}$ **then**
        Remove $T^1_{r1}, T^2_{r1}, T^5_{r1}, T^6_{r1}$ from $T$
        Insert $T^1_{r2}$ into $T$
    **else if** $\{T^2_{r1}, T^3_{r3}, T^6_{r1}, T^7_{r1} \subset T\} \wedge t_t \in T_c$ where $t_t \in \{T^2_{r3}, T^3_{r3}, T^2_{r1}, T^3_{r1}, T^6_{r1}, T^7_{r1}\}$ **then**
        Remove $T^2_{r1}, T^3_{r1}, T^6_{r1}, T^7_{r1}$ from $T$
        Insert $T^2_{r2}$ into $T$
    **else if** $\{T^3_{r1}, T^4_{r1}, T^7_{r1}, T^8_{r1} \subset T\} \wedge t_t \in T_c$ where $t_t \in \{T^3_{r3}, T^4_{r3}, T^3_{r1}, T^4_{r1}, T^7_{r1}, T^8_{r1}\}$ **then**
        Remove $T^3_{r1}, T^4_{r1}, T^7_{r1}, T^8_{r1}$ from $T$
        Insert $T^3_{r2}$ into $T$
    **end if**
    **if** $\{T^j_{r1}, T^{j+4}_{r1}\} \in T$ where $j \in \{1, 2, 3, 4\}$ **then**
        Remove $T^j_{r1}, T^{j+4}_{r1}$ from $T$
        Insert $T^j_{r3}$ into $T$
    **end if**
    **if** $\{(T^j_{r1} \in T_c \vee T^{j+4}_{r1} \in T_c)\ T^j_{r3} \in T\} \in T$ where $j \in \{1, 2, 3, 4\}$ **then**
        Insert $T^j_{r1}, T^{j+4}_{r1}$ into $T$
        Remove $T^j_{r3}$ from $T$
    **end if**
**end if**
Insert $t_g$ into $T$, where $t_g$ is the background encode
**return** tile set $T$

## 4.4 Video sequence synthesis and display

We considered optional upscaling of the background portion of the video to improve the quality in the case of mismatch in the viewport prediction. We selected the Lanczos filter for upscaling since, based on our previous works [10][9][21], we have identified it represents a good trade-off between performance and complexity.

# 5 TESTING CONDITIONS

In the test phase (before submission of the solution) a panoramic video of a natural environment *Release_video_1* and the associated head motion traces from two subjects were provided.

The final evaluation was performed on six segments from three videos, with head motion traces from one subject per video segment, namely *v1_s1*, *v1_s2*, *v2_s1*, *v2_s2*, *v3_s1* and *v3_s2*. The video segments represent three different video categories (i.e., natural landscapes, computer-generated animations and outdoor sports). Additionally, *v4_s1* and *v4_s2* were used for testing the first three solutions in the ranking. The thumbnails of the video sequences used in the final evaluation are shown in Figure 5.

All sequences used during both test phase and final evaluation were in ERP format with a resolution of $7680 \times 3840$ and frame rate of $30fps$.

# 6 RESULTS

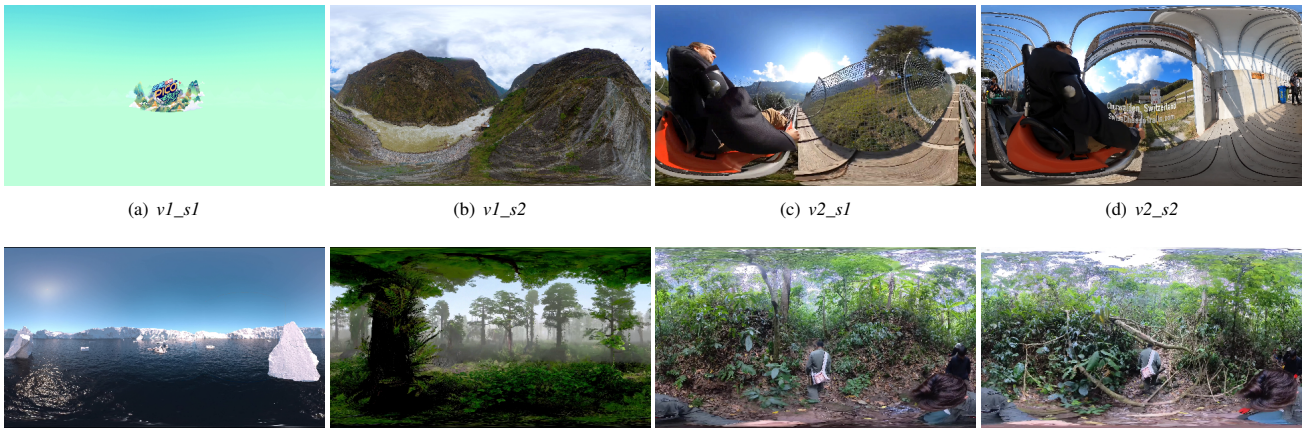
The first two rows of Table 1 report the results summary of the test phase. The performance is measured according to Equation (1) provided. The table also provides the breakdown of cost-related components that are associated with the grand-challenge score $S$. The computational cost associated with transcoding scenarios does not apply to us. On the content available for the test phase, our solution outperformed the other solutions submitted for the challenge at the time of the challenge deadline (source: challenge leaderboard). The results of the evaluation phase for our solution are also reported in the following rows of Table 1.

# 7 CONCLUSION

This paper presented the design and implementation of a 360-degree video on-demand streaming solution using the open-source evaluation platform E3PO. Results highlight a good performance in terms of MSE with limited costs in terms of data storage and transmission bandwidth.

# REFERENCES

[1] Adeel Abbas and David Newman. 2017. Ahg8: rotated sphere projection for 360 video. *JVET-F0036, Hobart, AU* 31 (2017).
[2] Jayasingam Adhuran, Chathura Galkandage, Gosala Kulupana, and Anil Fernando. 2020. Efficient VVC Intra Coding for 360° Video with Residual Weighting and Adaptive Quantization. In *2020 IEEE International Conference on Consumer Electronics (ICCE)*. 1–5. https://doi.org/10.1109/ICCE46568.2020.9043002
[3] Jayasingam Adhuran, Gosala Kulupana, and Anil Fernando. 2022. Deep learning and bidirectional optical flow based viewport predictions for 360° video coding. *IEEE Access* 10 (2022), 118380–118396.
[4] Jayasingam Adhuran, Gosala Kulupana, and Anil Fernando. 2022. Regions Of Interest Aware Versatile Video Coding for 360° Videos. In *2022 IEEE 11th Global Conference on Consumer Electronics (GCCE)*. 832–833. https://doi.org/10.1109/GCCE56475.2022.10014034
[5] Jayasingam Adhuran, Gosala Kulupana, Chathura Galkandage, and Anil Fernando. 2020. Multiple quantization parameter optimization in versatile video coding for 360° videos. *IEEE Transactions on Consumer Electronics* 66, 3 (2020), 213–222.
[6] Jayasingam Adhuran, Gosala Kulupana, Chathura Galkandage, and Anil Fernando. 2020. Optimal Distortion Minimization for 360° Video Compression with VVC. In *2020 IEEE International Conference on Consumer Electronics (ICCE)*. 1–3. https://doi.org/10.1109/ICCE46568.2020.9043034
[7] SN Akula, A Singh, R KK, RN Gadde, V Zakharchenko, E Alshina, and KP Choi. 2017. AhG8: Efficient Frame packing method for Icosahedral projection (ISP). *Document, JVET-G0156* (2017).



(a) *v1_s1* (b) *v1_s2* (c) *v2_s1* (d) *v2_s2*

(e) *v3_s1* (f) *v3_s2* (g) *v4_s1* (h) *v4_s2*

**Figure 5: Thumbnails of video sequences used in the final evaluation.**

**Table 1: Results summary**

| Sequence | $S$ | MSE | Bandwidth Cost ($w_1 \times C_b$) | Storage Cost ($w_2 \times C_s$) |
|---|---|---|---|---|
| Test Phase | | | | |
| *Release_video_1* | 8.593 | 13.074 | 3.794E-03 | 2.772E-06 |
| Final Evaluation | | | | |
| *v1_s1* | 8.534 | 12.419 | 4.264E-03 | 2.77E-06 |
| *v1_s2* | 6.356 | 17.2 | 5.410E-03 | 3.78E-06 |
| *v2_s1* | 8.826 | 10.663 | 4.928E-03 | 3.33E-06 |
| *v2_s2* | 6.493 | 12.733 | 7.757E-03 | 4.76E-06 |
| *v3_s1* | 9.217 | 6.222 | 7.112E-03 | 4.97E-06 |
| *v3_s2* | 6.303 | 8.904 | 1.052E-02 | 6.75E-06 |
| **Average** | 7.621 | | | |
| Additional Tests | | | | |
| *v4_s1* | 3.997 | 25.336 | 9.810E-03 | 6.62E-06 |
| *v4_s2* | 3.541 | 27.089 | 1.198E-02 | 7.69E-06 |


[8] Tamay Aykut, Basak Gülezyüz, Bernd Girod, and Eckehard Steinbach. 2020. Hsmf-net: Semantic viewport prediction for immersive telepresence and on-demand 360-degree video. *arXiv preprint arXiv:2009.04015* (2020).

[9] Nabajeet Barman, Yuriy Reznik, and Maria Martini. 2023. On the performance of video super-resolution algorithms for HTTP-based adaptive streaming applications. In *Applications of Digital Image Processing XLVI*, Vol. 12674. SPIE, 255–264.

[10] Nabajeet Barman, Steven Schmidt, Saman Zadtootaghaj, Maria G Martini, and Sebastian Möller. 2018. An evaluation of video quality assessment metrics for passive gaming video streaming. In *Proceedings of the 23rd packet video workshop*. 7–12.

[11] Bytedance. 2023. E3PO. https://github.com/bytedance/E3PO.

[12] J. Carreira, Sergio M. M. de Faria, Luis M. N. Tavora, Antonio Navarro, and Pedro A. Assuncao. 2020. Versatile Video Coding Of 360° Video Using Adaptive Resolution Change. In *Proc. IEEE International Conference on Image Processing (ICIP)*. 3398–3402. https://doi.org/10.1109/ICIP40778.2020.9190732

[13] Zhenzhong Chen, Yiming Li, and Yingxue Zhang. 2018. Recent advances in omnidirectional video coding for virtual reality: Projection and evaluation. *Signal Processing* 146 (2018), 66–78.

[14] Zhenzhong Chen, Weisi Lin, and King Ngi Ngan. 2010. Perceptual video coding: Challenges and approaches. In *Proc. IEEE International Conference on Multimedia and Expo*. 784–789.

[15] M Coban, G Van der Auwera, and M Karczewicz. 2017. AHG8: Adjusted cubemap projection for 360-degree video. *Joint Video Exploration Team of ITU-T SG16 WP3 and ISO/IEC JTC1/SC* 29 (2017).

[16] Xavier Corbillon, Gwendal Simon, Alisa Devlic, and Jacob Chakareski. 2017. Viewport-adaptive navigable 360-degree video delivery. In *Proc. IEEE international conference on communications (ICC)*. 1–7.

[17] Yago Sanchez de la Fuente, Gurdeep Singh Bhullar, Robert Skupin, Cornelius Hellge, and Thomas Schierl. 2019. Delay impact on mpeg OMAF's tile-based viewport-dependent 360 video streaming. 9, 1 (2019), 18–28.

[18] Chi-Wing Fu, Liang Wan, Tien-Tsin Wong, and Chi-Sing Leung. 2009. The rhombic dodecahedron map: An efficient scheme for encoding panoramic video. 11, 4 (2009), 634–644.

[19] Gazi Karam Illahi, Matti Siekkinen, Teemu Kämäräinen, and Antti Ylä-Jääski. 2022. Real-time gaze prediction in virtual reality. In *Proceedings of the 14th International Workshop on Immersive Mixed and Virtual Environment Systems*. 12–18.

[20] Sami Jaballah, Amegh Bhavsar, and Mohamed-Chaker Larabi. 2020. Perceptual Versus Latitude-Based 360-Deg Video Coding Optimization. In *Proc. IEEE International Conference on Image Processing (ICIP)*. 3423–3427. https://doi.org/10.1109/ICIP40778.2020.9191257

[21] Peter A Kara, Werner Robitza, Nikolett Pinter, Maria G Martini, Alexander Raake, and Aniko Simon. 2019. Comparison of HD and UHD video quality with and without the influence of the labeling effect. *Quality and User Experience* 4 (2019), 1–29.

[22] HC Lin, CY Li, JL Lin, SK Chang, and CC Ju. 2016. An efficient compact layout for octahedron format, JVET Doc. *D0142* (2016).

[23] Holger Meuel, Marco Munderloh, Matthias Reso, and Jörn Ostermann. 2013. Optical flow cluster filtering for ROI coding. In *Proc. IEEE Picture Coding Symposium (PCS)*. 129–132.

[24] Afshin Taghavi Nasrabadi, Anahita Mahzari, Joseph D Beshay, and Ravi Prakash. 2017. Adaptive 360-degree video streaming using scalable video coding. In *Proc. ACM International Conference on Multimedia (ICM)*. 1689–1697.

[25] Massimiliano Patacchiola. 2017 (updated 2020, accessed Feb 2024). DeepGaze. https://github.com/mpatacchiola/deepgaze/tree/master.

[26] Massimiliano Patacchiola and Angelo Cangelosi. 2017. Head pose estimation in the wild using convolutional neural networks and adaptive gradient methods. *Pattern Recognition* 71 (2017), 132–143.

[27] Fabien Racapé, F Galpin, G Rath, and E Francois. 2017. AHG8: adaptive QP for 360° video coding. *JVET-F0038* (2017).

[28] Silvia Rossi and Laura Toni. 2017. Navigation-aware adaptive streaming strategies for omnidirectional video. In *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 1–6.

[29] Kalpana Seshadrinathan and Alan Conrad Bovik. 2009. Motion tuned spatio-temporal quality assessment of natural videos. 19, 2 (2009), 335–350.

[30] Robert Skupin, Yago Sanchez, Cornelius Hellge, and Thomas Schierl. 2016. Tile based HEVC video for head mounted displays. In *Proc. IEEE International Symposium on Multimedia (ISM)*. 399–400.

[31] Aljoscha Smolic and Peter Kauff. 2005. Interactive 3-D video representation and coding technologies. *Proc. IEEE* 93, 1 (2005), 98–110.

[32] Afshin TaghaviNasrabadi, Anahita Mahzari, Joseph D Beshay, and Ravi Prakash. 2017. Adaptive 360-degree video streaming using layered video coding. In *Proc. IEEE Virtual Reality (VR)*. 347–348.

[33] Truong Cong Thang, Quang-Dung Ho, Jung Won Kang, and Anh T Pham. 2012. Adaptive streaming of audiovisual content using MPEG DASH. 58, 1 (2012), 78–85.

[34] Geert Van der Auwera, Muhammed Coban, M Karczewicz Hendry, and M Karczewicz. 2016. AHG8: Truncated square pyramid projection (tsp) for 360 video. In *Joint Video Exploration Team of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, JVET-D0071, 4th Meeting*.

[35] Shivi Vats, Jounsup Park, Klara Nahrstedt, Michael Zink, Ramesh Sitaraman, and Hermann Hellwagner. 2022. Semantic-Aware View Prediction for 360-Degree Videos at the 5G Edge. In *2022 IEEE International Symposium on Multimedia (ISM)*. IEEE, 121–128.

[36] Minghui Wang, Tianruo Zhang, Chen Liu, and Satoshi Goto. 2009. Region-of-interest based dynamical parameter allocation for H. 264/AVC encoder. In *Proc. IEEE Picture Coding Symposium (PCS)*. IEEE, 1–4.

[37] Shibo Wang, Shusen Yang, Hailiang Li, Xiaodan Zhang, Chen Zhou, Chenren Xu, Feng Qian, Nanbin Wang, and Zongben Xu. 2022. SalientVR: saliency-driven mobile 360-degree video streaming with gaze information. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*. 542–555.

[38] Xiaoyu Xiu, Yuwen He, and Yan Ye. 2018. An adaptive quantization method for 360-degree video coding. In *Proc. Applications of Digital Image Processing XLI*, Vol. 10752. 107520X.

[39] Matt Yu, Haricharan Lakshman, and Bernd Girod. 2015. A framework to evaluate omnidirectional video coding schemes. In *Proc. IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 31–36.

[40] C Zhang, Y Lu, J Li, and Z Wen. 2017. AHG8: Segmented Sphere Projection for 360-degree video. *Joint Video Exploration Team of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, JVET-E0025* (2017).

[41] Minhua Zhou. 2016. AHG8: A study on compression efficiency of cube projection. *Document JVET-D0022, Chengdu, CN* (2016).