Immersive Video Streaming with 6 Degree-of-Freedom Interactions

組別：A227　　指導教授：徐正炘　　組員名稱：孫元駿、王婧庭

Abstract

Recently, Virtual Reality (VR) becomes increasingly more popular. It enables a wide array of novel applications in many domains, such as video streaming, computer games, occupational training, healthcare, manufacturing, etc. We opt Depth Image-Based Render (DIBR) as the method to synthesize remoted scenes.

We address the problem of quantifying the perceived quality in 6DoF (Degree-of-Freedom) live immersive video in two steps. First, we develop a set of tools to generate (or collect) datasets in a photorealistic simulator, AirSim. Using these tools, we get to change diverse settings of live immersive videos, such as scenes, trajectories, camera placements, and encoding parameters. Second, we develop objective and subject evaluation procedures, and carry out evaluations on a sample immersive video codec, MPEG MIV, using our own dataset. Several insights were found in our experiments, e.g., Quantization Parameter (QP) is a good control knob to exercise target view quality and bitrate, but camera placements (or trajectories) also impose significant impacts. These findings shed some light on the key research problems for the development of merging applications relying on immersive interactions. According to the results of this paper, we can utilize it to solve the camera placement problem or more deeply discuss the user experience of immersive video in the future.

Introduction

Virtual Reality (VR) refers to 3D simulations of virtual scenes consisting of objects and persons that offer immersive interactions. The VR market is expected to reach 15.5 billion euros by 2022 [1]. As a killer application of VR technologies, virtual worlds, a.k.a. metaverse, are online environments that allow multiple users at diverse geographical locations to interact with one another *instantaneously* in virtual scenes. The metaverse market is projected to grow at a staggering 43.3% annual growth rate until 2028 [2]. Dionisio et al.[3] pointed out that the success of metaverse rely on the following four features: *realism*, *ubiquity*, *interoperability*, and *scalability*. One of the key enablers for these four features is *live immersive video streaming*, which allows multiple users to depart from their physical locations and meet one another at a remote scene, which can be an artificial or natural one [4].

We consider live immersive video streaming of a *dynamic remote scene* to multiple Head Mounted Display (HMD) users, who will interact with the scene in 6 Degree-of-Freedom (6DoF). 6DoF includes: (i) yaw, roll, and pitch, which specify the user *orientation* and (ii) surge, heave, and sway, which specify the user *position*. One way to enable 6DoF interactions is through capturing or generating 3D scenes, using volumetric media, like point clouds or meshes [5]. Doing so, however, is computational expensive and thus is less suitable for streaming *dynamic* and *live* scenes. Therefore, we opt for low-complexity Depth Image-Based Rendering (DIBR) to bring remote scenes, objects, and persons to HMD users over the Internet. DIBR synthesizes new videos from multiple RGBD videos at different positions and orientations.

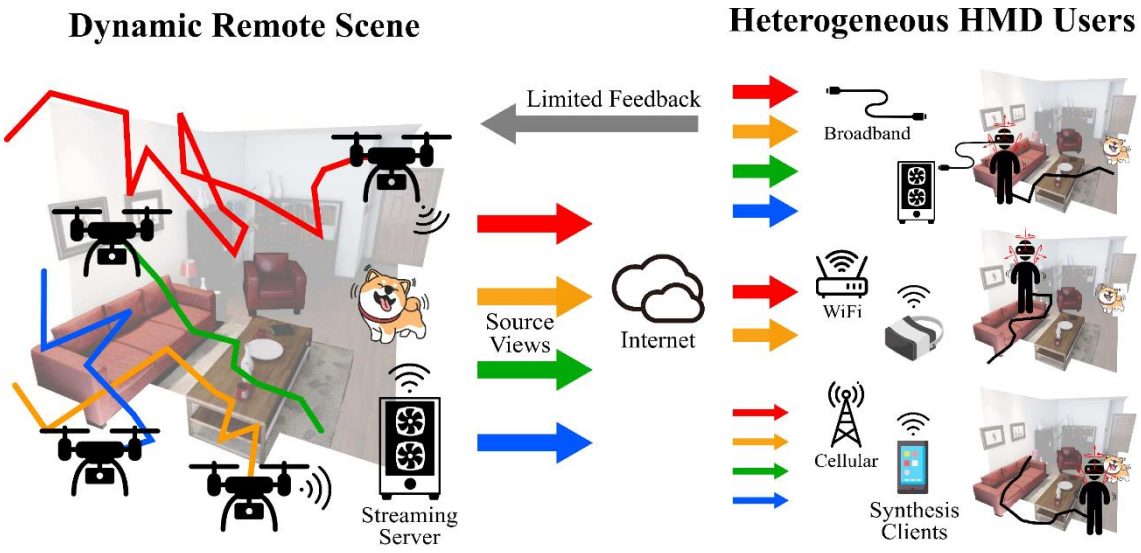


Fig. 1: Sample 6DoF live immersive video scenario.

Fig. 1 illustrates a sample live immersive video streaming session from a remote dynamic scene to multiple HMD users; more general bi-directional streaming scenarios are also possible. We envision that multiple HMD users (on the right of the figure) freely explore the dynamic remote scene, which may contain moving objects, such as a running dog, and environment changes, such as dimming ambient light due to sunset. Each HMD user sees a unique *target view*, which depends on his/her positions and orientations over time, as described by a *target view trajectory*. The target views are synthesized by the *source views* captured at the dynamic remote scene (on the left of the figure). For example, multiple cameras can be carried by drones to capture RGBD videos with camera parameters following *source view trajectories*. The captured source views are encoded at the *streaming server* and streamed to *synthesis clients* with diverse computational devices and communication networks. HMD users (top in the figure) who are tethered with powerful workstation and broadband access networks can receive all source views for better synthesized target view quality. However, users constrained by all-in-one HMDs (with entry-level GPUs and limited battery capacity) or cellular networks (with limited bandwidth) may receive fewer or reduced-quality source views to ensure realtimeness. It

is not hard to see that the source view (drones) trajectories should be carefully computed based on the target view (HMD users) trajectories. Nonetheless, such computation needs to consider various constraints, including aerodynamics of drones, computation power of the streaming server, and bandwidth of the Internet, and thus is an open and challenging problem.

In our research, we study a fundamental research problem for solving the aforementioned optimization problem: *how to quantify the perceived quality of synthesized target views?* This is not an easy task because human perception system is highly nonlinear and hard to predict. Particularly, we tackle the problem in two steps:

- *Data collection.* We develop a suite of tools to collect source views from a photorealistic simulator based on AirSim [6], which is built upon Unreal Engine [7]. Using the tools, we get to exercise diverse *settings*, including scenes, trajectories, camera placement, and encoding parameters, when collecting a rich set of source views for experiments. More importantly, we can also gather the target views from the simulator, which serve as the ground truth for quality assessment. With the developed tools, virtually unlimited amount of dynamic remote scenes can be generated by us and the research community.
- *Quality assessment.* We carry out both objective and subjective quality assessment to understand how different settings affect the perceived quality of the synthesized target views. Our assessment results reveal the key settings that affect the perceived quality the most. Moreover, the developed quality assessment procedure can be adopted by the research community.

To the best of our knowledge, the above tasks have not been thoroughly investigated in the literature. We notice that, although our proposed solution can work with any DIBR synthesizers, for concrete discussion, we employ the recent MPEG Immersive Video (MIV) standard [8] for encoding and decoding RGBD video streams and camera parameters, as well as synthesizing and rendering target views in this paper.

Related Work

Objective quality assessment of live immersive video streaming has been conducted to understand how different settings affect the target view quality. For example, Cai et al. [9] compared the performance resulted by using different 2D video codecs to encode depth video in 6DoF streaming. They found that AVS3 codec achieves better coding efficiency at an expense of higher computational complexity. Sebastian et al. [10] conducted a similar study on depth video compression. In contrast, Szekielda et al. [11] studied the implications of 2D encoding parameters on both RGB and depth videos. Software components other than 2D video codecs have also been exercised. For example, Fachada et al. [12] focused on the impact of different synthesizers under linear versus planar camera placement. Pre- or post-processing of the RGB and depth videos for higher

coding efficiency was also investigated. For example, Jeong et al. [13] proposed to downsample the RGB video and upsample the depth video to increase the coding efficiency. Salahieh et al. [14] compared the performance of MIV reference software, called Test Model of Immersive Video (TMIV) [15] in the 3D domain. In particular, they considered different settings, including *single*-versus *multi-pass synthesis* and *MIV* versus *MIV View modes*}. Their performance compassion revealed that multi-pass synthesis leads to significant performance boost in MIV View mode. Last, the impacts of different camera (source view) densities on synthesized target view quality were evaluated in Ray et al. [16]. They concluded that higher camera densities lead to better perceived quality. Compared to our work, the aforementioned studies have two major limitation. First, almost all of them (except Ray et al. [16]) employed existing scene/trajectory datasets. The only exception is Ray et al. [16], in which two scenes created in Blender were used. Second, none of them considered *continuous* 6DoF trajectories of source *and* target views. The closest setup to continuous 6DoF trajectories was the *discrete* camera locations/orientations adopted in Fachada et al. [12] and Ray et al. [16]. *Our AirSim-based data collection tools offer the opportunity to generate large and flexible datasets with real 6DoF source/target trajectories.*

Subjective quality assessment of live immersive video streaming were mostly done on volumetric media [17] [18]. For instance, Wu et al. [17] adopted Double Stimulus Impairment Scale (DSIS) method to study the impact of point cloud compression. Nehme et al. [18] investigated the subjective quality of rendered 3D mesh models. The subjective quality of DIBR-based approach has also been recently studied. For example, Schwarz et al. [10] employed the Double Stimulus Impairment Scale (DSIS) [19] method to study the implication of Quantization Parameter (QP) of depth video on the subjective quality with two scenes. Jung and Boissonade [20] also used the DSIS method to study the subjective quality with MPEG-defined target view trajectories with different synthesizers in four scenes. *Different these two studies that only varied one parameter of the 2D codecs, our subjective quality assessment exercised many more settings, including camera placements (source view trajectories).*

Data Collection System

A. Implementations

The purpose of collecting our own dataset is to investigate the impacts of diverse settings on the synthesized target views. This dataset has to cover a wide variety of usage scenarios with: (i) different camera placement strategies, (ii) random target view trajectories mimicking real HMD users, (iii) scenes with diverse characteristics (e.g., lighting conditions, color tone, and dynamics), we capture the source views by extending AirSim [6], which is an open-source project that provides Application Programming Interface (API) for programmers. For example, we develop tools using camera control

API to capture source views.

Fig. 2: The considered scenes sorted in increasing complexity levels: (a) Light, (b) Arch, (c) Xoio, (d) Office, and (e) Real.

Target view trajectory generator. We have surveyed some random trajectory generators including (i) random waypoint, which selects the next target position randomly and go at a random velocity then pause for a short period, (ii) random walk, which randomly choose an adjacent point to go, and (iii) random direction, which go in a random direction with a random velocity for a period of time. Eventually, we decide to employ the random waypoint model because it is the most suitable one for simulating a random mobility of a user. However, the maximum velocity and the dynamic range are left specified.

To make the random data behave like normal human, we also implemented a user-trajectory collection system with Unreal Engine HMD library. Several users were recruited to tilt (roll), nod (pitch), and rotate (yaw) their heads. The head movement will be recorded and become one of the instances our user study dataset. Head trajectories will be further processed to extract the data distribution among different dimensions to generate reasonable random waypoint model. We found that the dynamic range for roll, pitch, and yaw are 80, 80, and 100 degrees respectively, and the angular velocity falls in the interval of 8 to 80. The translational components are generated with the bounding box (0.35m x 1m x 1m) which we define the bounding box area by referring to the dataset with similar camera placement in MPEG common test condition. [21]. For rotational components, we use the statistics as described in the following section. In our implementation, the two components are generated at random, and each component will try to travel to its destination independently with random but reasonable speed then immediately select the next waypoint.

Table 1: Scenes in Our Dataset

Scene	#Obj	#Mesh	Space	Lighting	Color Tone	TI	SI
Light	52	51.3K	Narrow	Bright	Warm	19.1	35.0
Arch	282	5.5K	Wide	Bright	Warm	27.1	57.4
Xoio	125	2.8K	Wide	Bright	Cold	26.2	57.8
Office	96	100.4K	Narrow	Dark	Cold	29.6	62.2
Real	352	221.1K	Narrow	Dark	Warm	34.7	66.7

Moreover, we adopt the random waypoint as the mobility model to generate random target view trajectories. In addition to random trajectories, we also implement tools to collect real HMD user trajectories using Unreal Engine. In a pilot test, we recruit three HMD users, play a random scene to them, and analyze their target view trajectories. We found that the dynamic ranges of roll/pitch/yaw are 80/80/100 degrees, and the angular velocity is between 8 and 80 degree/s. We also found the HMD users moves at a speed between 0.1 and 1 m/s. We use these statistics to generate random target view trajectories.

B. The Dataset

We select five scenes from Unreal Engine marketplace [22], and modify one of them into a dynamic scene. Fig. 2 gives sample video frame of these scenes (except the dynamic one). Table 1 summarizes the scenes with key characteristics. The synthesized target views from these scenes exhibit different complexities in terms of Temporal Information (TI) and Spatial Information (SI). The dynamic scene, *RealD*, is generated by adding a howling wolf and a rolling ball to *Real*. For each scene, we manually choose direction that have the richest set of visual features. We then place 36 cameras with one of the four placements: 6×6, 9×4, 12×3, and 18×2, which we shown in Fig. 3. Following MPEG's recommendations, all cameras and trajectories are confined in a $0.35 \times 1 \times 1$ m^3 bounding box. We set the resolution and Field-of-View (FoV) of each camera to be 1280×720 and 90°, respectively. We generate 10 target view trajectories: t_1 to t_{10} using the random waypoint model. We also selected a real HMD user's trajectory from our pilot test that covers the largest surface of the scene. We refer to it as u_1. Each trajectory contains 90 samples of positions and orientations at 30 Hz.

One last complication is the different coordinate systems used by: (i) Unreal Engine (North-Eastern-up, left-handed), (ii) AirSim (North-Eastern-down, right-handed), and (iii) TMIV (North-Western-up, right-handed). We have implemented scripts to convert the coordinate systems. For each combination of the scene and camera placement (trajectories), we capture RGBD videos from individual cameras, which are *source views*. We also captured the RGBD video clips following individual target view trajectories, which are *target views*, or ground truth. Both source and target views are stored as raw videos. In summary, our dataset contains: (i) source view placements (trajectories), (ii) target view (trajectories), (iii) source views, and (iv) target views. We plan to make our tools and sample dataset public.

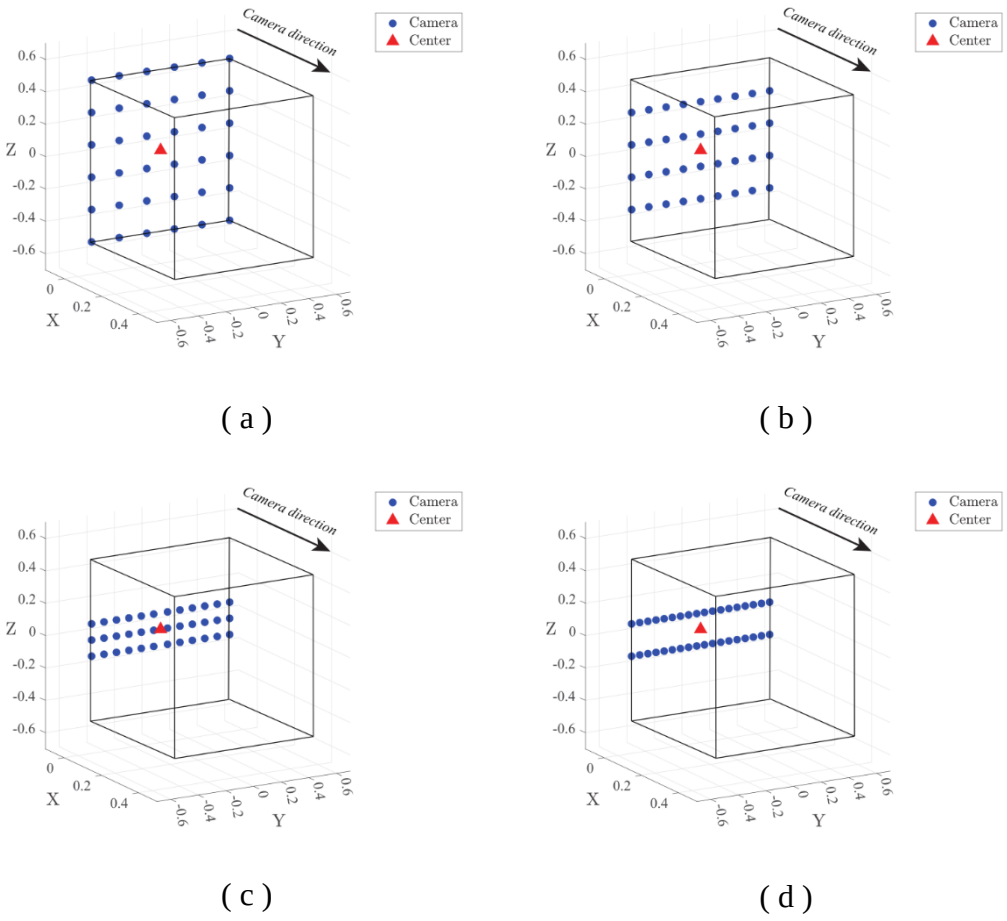


Fig. 3: The camera placement: (a) 6x6, (b) 9x4, (c)12x3, and (d) 18x2.

MPEG Immersive Video Standard

In this section, we briefly introduce the workflow and components of MIV codec [15] [8].

Fig. 4 (a) shows the high-level workflow of MIV encoder. The inputs of MIV encoder are *source views*. Each source view is composed of attribute (texture) videos, geometric (depth) videos, and camera parameters. MIV encoder do the following process to compress source views:

- **Automatic parameter selection.** MIV encoder automatically calculate the parameters for compression, e.g., assessing geometric video quality, splitting source views into multiple group according to configuration, and labeling source views in each group.
- **Single-group encoders.** MIV encoder encodes each group of source views separately. In each group, the encoder chooses several views as the basic view according to the label of source view, and remove the duplicate area in other source views. The basic view and remaining area of other views are packed into rectangle video frames, which are called *atlases*. Fig. 5 show the example of atlases.

The outputs of MIV encoder are attribute atlases, geometric atlases, and metadata

bitstream. The atlases are further compressed by video codec, and multiplexed with metadata bitstream as a single bitstream.

Fig. 4 (b) shows the high-level workflow of MIV decoder. The inputs pf MIV decoder is the bitstream contains atlases bitstream, and metadata bitstream. The video decoder first be employed to decompress attribute atlases and geometric atlases. After that MIV decoder do the following process to decompress atlases and synthesize the user's

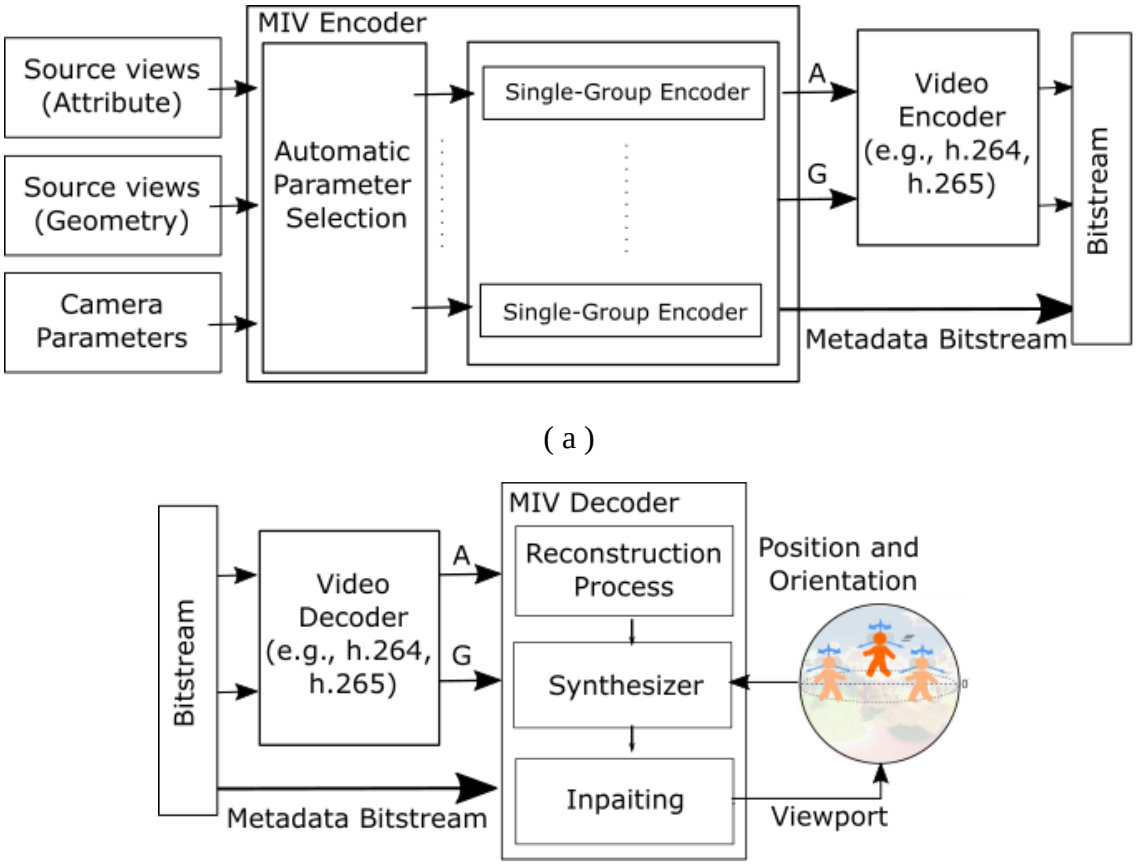


Fig. 4: The high-level overview of process flow of TMIV: (a) Encoder and (b) Decoder.

viewport.

- **Reconstruction process.** The MIV decoder reconstruct the source view by using the data in atlases.
- **Synthesizer.** The MIV decoder employ view synthesis techniques to synthesize the user's viewport according to user's position and orientation. Specifically, the synthesizer warp the pixel of each source view to user's viewport according to depth information, and blending the pixel values from each source views.
- **Inpainting.** After synthesis, the synthesized result may contain holes without information. The inpainting process uses the information from neighbor pixels to calculate pixel value for holes.

The outputs of MIV decoder are user's viewport synthesized according to user's position and orientation.

Fig. 5: The example of atlases. The left picture is attribute atlas, and the right picture is geometric atlas.

Objective Quality Assessment

A. Setup

We carry out experiments using our collected dataset to quantify the objective quality of the synthesized target views. TMIV [23] version 10 is employed, which takes a few groups of source views and exploits the redundancy among source views in each group. We put all source views in a group. TMIV encoder selects a most representative view (called *basis* view), removes duplicated areas from other source views (called *additional* views), and packs them into rectangle video frames (called *atlases*).

These RGBD video frames are encoded by a traditional 2D codec. We employ x265 codec [24] version 4.2.4 in our experiments and combine the x265 bitstreams and TMIV metadata into the final TMIV bitstream for immersive video streaming. TMIV decoder decodes RGBD video frames using x265 and synthesizes target views. We vary the following parameters of TMIV and x265 in our experiments:

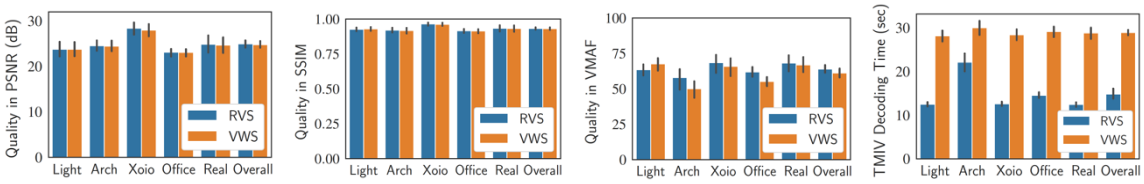
- **QP** trades off the bitrate and RGBD video quality. MPEG group suggests using a smaller QP for depth video (80% of the corresponding RGB videos) because depth imposes significant impacts on synthesized quality [25]. We set (RGB video) QP $\in \{ 20, 36, 44, 48, 50 \}$, targeting 5 to 50 Mbps total bitrates.
- **Synthesizer** of TMIV can be either Reference View Synthesizer (RVS) or View Weighting Synthesizer (VWS).
- **Camera placement**$\in \{ 6 \times 6, 9 \times 4, 12 \times 3, 18 \times 2 \}$ for the five scenes in Table 1.

For the dynamic *RealD*, we vary the camera placements (source view trajectories) every 30-frames in the following order: 6X6, 9X4, and 12X3.

- **Target view trajectory** $\in \{t_1, t_2, \ldots, t_9, t_{10}, u_1\}$.

We run our experiments on a workstation with Xeon E5-2678 CPUs at 2.5 GHz. We configure the x265 encoder to use default coding parameters with a Group-of-Picture (GoP) of 30, corresponding to the frame rate. We measure the following objective metrics, and report the average results with 95% confidence interval whenever applicable:

- **Video quality** of the target views in Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), and Video Multi-Method Assessment Fusion (VMAF) [26], which are full reference quality metrics.
- **Bitrate** of the TMIV bitstream.
- **Running times** of the per-frame TMIV encoder, x265 encoder, TMIV decoder, and x265 decoder.



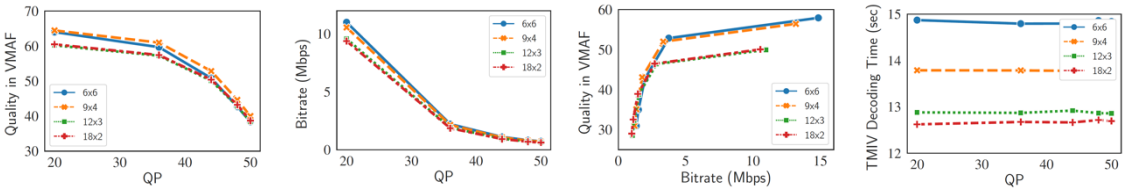
(d) (d) (d) (d)

Fig. 6: The video quality achieved by different synthesizers in: (a) PSNR, (b) SSIM, (c) VMAF; and required: (d) TMIV decoding time.

B. Results

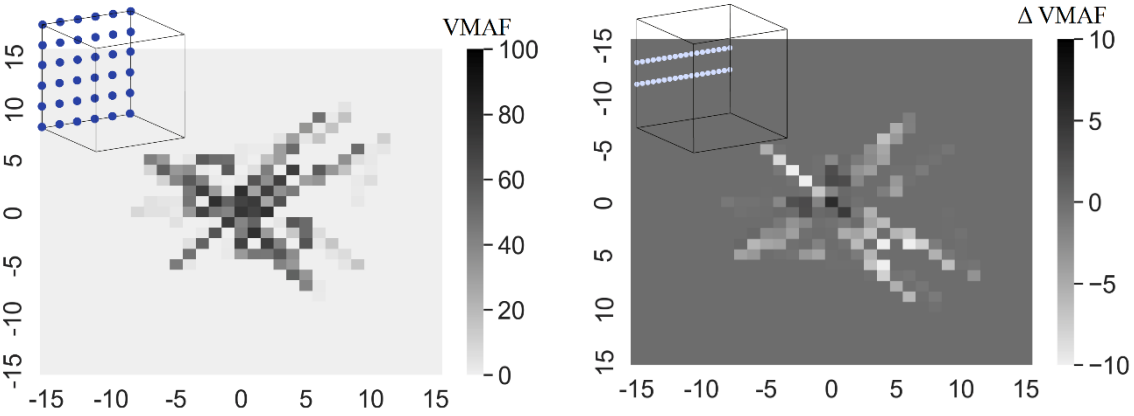
Comparisons between synthesizers. We compare two synthesizers using $t_1 - -t_{10}$ in different scenes. We report sample results from the QP of 20 and the camera placement of 6X6 in Fig. 6. Other combinations lead to similar results. This figure reveals that all three video quality metrics show similar trends, although the difference in VMAF is more clear. Hence, in the rest of the paper, we report sample video quality results in VMAF. Furthermore, RVS leads to higher video quality in most scenes, except the low-complexity *Light* (with the lowest TI and SI as indicated in Table 1). In contrast, VWS works better in more complex scenes and under higher motion. Furthermore, we observe no statistical significance in VMAF (and PSNR/SSIM) of the overall video quality: p-values are all above 0.18. Last, Fig. 6(d) shows that VWS results in longer TMIV decoding time than RVS. Because RVS achieves comparable video quality with about half of the TMIV decoding time than VWS, we give the sample results from RVS in the rest of the paper.

Implications of QP values. Next, we vary QPs under different camera placements and present the results from individual scenes in Fig. 7. Fig. 7(a) shows that the video quality decreases when QP is increased. Besides, Fig. 7(b) reveals that the bitrate decreases even faster when the QP is increased. For instance, when increasing QP from 20 to 50, the bitrate is divided by fourteen. We also plot the Rate-Distortion (RD) curves in Fig 7(c). We observed that 6X6 and 9X4 camera placements give better video quality compared to 12X3 and 18X2 ones. We take a step further and compute the BD-rates [27] , which are omitted due to space limitation. Take *Arch* as an example, the BD-rates of 12X3



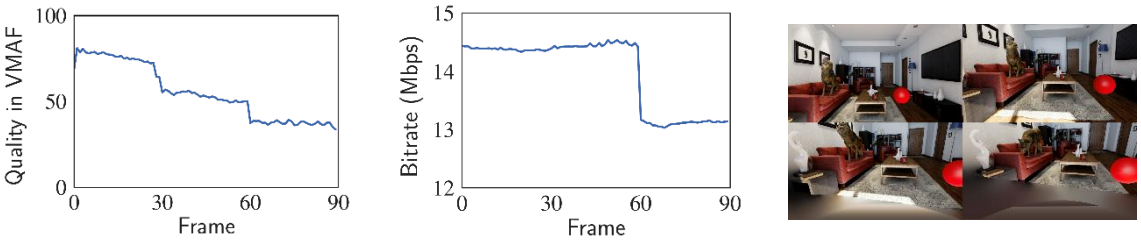
(a) (b) (c) (d)

Fig. 7: The objective results from different QPs and camera placements: (a) video quality, (b) bitrate, (c) RD-curves, and (d) TMIV decoding time.



(b) (b)

Fig. 8: Sample video quality distribution of individual grids: (a) absolute VMAF from 6X6 and (b) delta VMAF of 18X2, compared to that of 6X6. Dots represent cameras in the bounding box.



(a) (b) (c)

Fig. 9: Per-frame: (a) video quality and (b) bitrate; and (c) sample synthesized target view frames from dynamic scene RealD.

and 18X2 camera placements are 7.92% and 2.47% compared to the 6X6 one. This shows that camera placements (source view trajectories) affect the coding efficiency of live immersive video streaming.

To better understand how target view trajectories affect the synthesized view quality, we take a deeper look at QP=20 with *Arch* under camera placements 6X6 and 18X2. We first divide the wall with the most visual features into 30×30 grids. We then project $t_1 - -t_{10}$ to these grids and compute the average video quality of each grid. We use the results from 6X6 as the baseline and plot its absolute per-grid VMAF as a heatmap in Fig. 8(a). We also report the delta VMAF of 18X2 compared to 6X6 in Fig. 8(b). These two figures depict that wider camera placement results in better video quality at the center of the wall. This can be attributed to the unequal coverage *areas* (by at least a camera) and *density* (number of covered cameras) across different camera placements (source view trajectories). In particular, 18X2 camera placement focuses more on the vertical center as indicated in Fig. 8(b). Last, we observe that the x265 encoding and decoding times decrease when QP is increased. This can be attributed to the reduced amount of data passing the quantizer of x265. However, different QPs do not affect the TMIV decoding time too much as reported in Fig. 7(d). This is because the TMIV decoder takes source views in YUV as inputs.

Dynamic scene. We report results from QP=20 with *RealD*, in which camera placements are dynamic. Particularly, the camera placements are: 6X6 (first 30 frames), 9X4 (next 30), and 12X3 (last 30). We report the video quality and bitrate over time in Fig. 9(a) and Fig. 9(b), respectively; and give sample synthesized target views in Fig. 9(c). We observe that the video quality changes when cameras (drones) move. Similarly, the bitrate also varies over time. This experiment demonstrates the importance of camera placements, especially in dynamic scenes. *We envision that an algorithm to compute the source view trajectories given the scene, target view trajectories, network bandwidth, etc., is critical to optimize 6DoF immersive video streaming.*

Subjective Quality Assessment

A. Design

In our subjective quality experiment, we consider 2 synthesizers, 5 camera placements, and 5 QPs. We randomly select a target view trajectory t_9 and loop the 90-sample trajectory to get 10-sec target videos of a sample scene *Real*. We then play 50 videos with different settings in a random order to subjects using a 27-inch 2K monitor. We adopt the DSIS [19] method and play the reference and synthesized videos side-by-side. We recruited 17 subjects between 19 and 33 years old; 6 of them are males. Each subject first answers the demographic questions. We then explain the procedure, followed by a training session of 10 videos from a totally different scene (not in Table 1). We adopt the ACR (Absolute Category Rating) scale, where 5 indicates excellent and 1 indicates

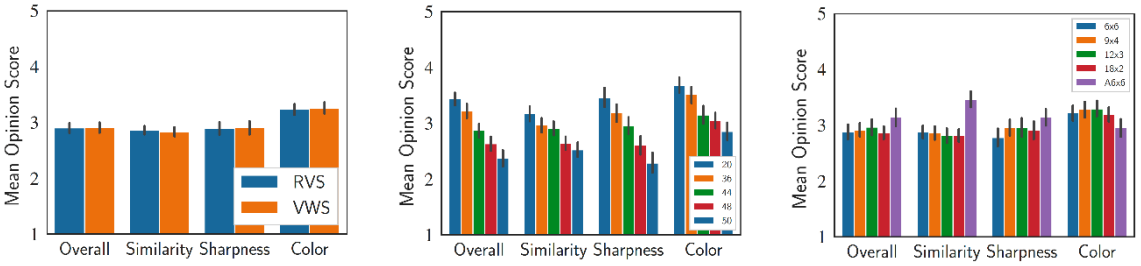
bad. For each synthesized target view video, a subject says his/her scores aloud, which are recorded by an assistant. We consider the following four questions:

- **Overall:** How was the overall quality?
- **Similarity:** How similar was the target video to reference one?
- **Sharpness:** How was the image sharpness?
- **Color:** How was the color naturalness?

Subjects are also given a chance to provide text-based feedback. It takes each subject at most 50 mins to complete all the videos. They can take a 10-min break in the middle. We report the Mean Opinion Scores (MOS), with 95% confidence intervals if applicable.

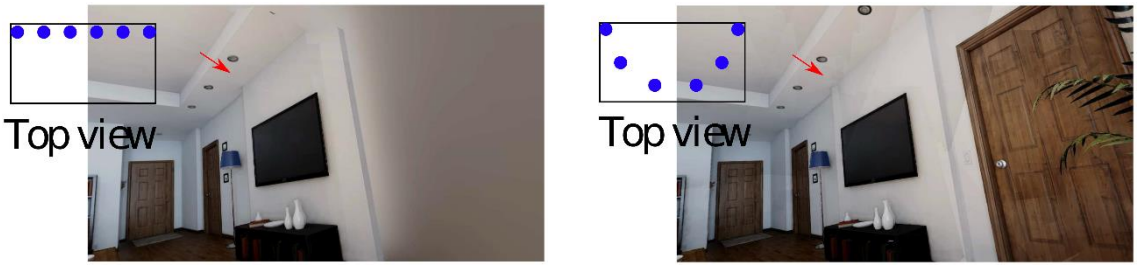
Table 2: Correlation Between Subjective Metrics: PLCC/SROCC

Correlation	Overall PLCC	Overall PLCC
Overall	1.00	1.00
Similarity	0.94	0.96
Sharpness	0.95	0.95
Color	0.92	0.95



(a) (b) (c)

Fig. 10: Subjective quality assessment results, the implications of: (a) synthesizer, (b) QPs, and (c) camera placements.



(a) (b)

Fig. 11: Sample target views: (top) 6X6 and (bottom) A6X6.

B. Results

Implications of synthesizers and QPs. We first plot the MOS of different subjective metrics produced by different synthesizers in Fig. 10(a). This figure reveals that the subjective difference between the two synthesizers is rather small, which is consistent with our objective assessment results. We also plot the MOS under different QPs in Fig.

10(b), which shows that QPs affect the sharpness the most, as indicated by the steepest slope. This can be attributed to the blocking artifacts due to quantization with large QPs. In contrast, QPs affect the similarity the least. This could be explained by the difficulty of judging the similarity (of side-by-side) target views. Last, the color and overall quality fall somewhere in between.

Correlation between subjective quality metrics. Next, we give the correlation between subjective metrics in PLCC (Pearson Linear Correlation Coefficient) and SROCC (Spearman Rank Order Correlation Coefficient) in Table 2. PLCC captures the linear relationship between two random variables, and SROCC captures the rank relationship [28]. *We observe that the overall quality has positive correlation to all three other subjective metrics; however, none of these three subjective metrics dominates the overall quality*.

Implications of camera placements. In our pilot user study, multiple subjects mention that the door of the scene is missing, as shown in Fig. 11. We take a closer look and find that the door is not covered by any camera in the (planar) placements. Hence, we augment camera placement 6X6 into A6X6, in which the cameras are horizontally rotated into an arc placement. Fig. 10(c) gives the MOS resulted from diverse camera placements. We make three observations. First, different (planar) camera placements do not affect the subjective quality too much. We believe this is because subjects are attracted by the missing door, and are less sensitive to quality difference. Second, A6X6 leads to higher MOS on overall quality, similarity, and sharpness. This can be attributed to the existence of the door. Third, A6X6 results in lower MOS on color. This can be explained by lower camera density of A6X6. Several subjects mention diamond-shape artifacts, which negatively affect the color naturalness. Last, we note that the quality improvement of A6X6 comes at a price of higher bitrate: on average a BD-Rate of 8.75 Mbps is observed. *The user study shows the importance of more flexible camera placements on user perceived subjective quality.*

Conclusions

In this paper, we developed tools upon open-source AirSim to generate/capture diverse scenes and source/target view trajectories. Using this dataset, we evaluated TMIV and x265 through detailed objective and subjective evaluations. Several insights were found, for example:

- The two synthesizers in TMIV produce comparable target view quality, but RVS runs 2 times faster.
- QP is a good control knob to exercise target view quality and bitrate, but camera placements (or trajectories) also impose significant impact.
- Subjective overall quality has strong linear/rank correlation with subjective similarity, sharpness, and color.

Future Works

We plan to extend this work in multiple directions. For instance, we are developing a parameterized perceived quality model for 6DoF live immersive video streaming. Guided by the resulting models, a wide array of optimization problems can be formulated, solved, and implemented for enhancing immersive experience. For example, existing optimal camera placement algorithms considered 3D view coverage rather than synthesized target view quality [29].

References

[1] SoftwareTestingHelp. (2021) Future of virtual reality market trends and challenges. https://www.softwaretestinghelp.com/future-of-virtual-reality/

[2] Emergen Research. (2021) Metaverse market shared. https://www.emergen-research.com/industry-report/metaverse-market

[3] J. Dionisio, W. Burns, and R. Gilbert, "3D virtual worlds and the metaverse: Current status and future possibilities," ACM Computing Survey, 2013, Vol. 45, No. 3, pp. 1–38.

[4] K. Nevelsteen, "Virtual world, defined from a technological perspective and applied to video games, mixed reality, and the metaverse," Computer Animation and Virtual Worlds, 2018 Vol. 29, No. 1, p. e1752.

[5] J. Hooft, M. Vega, T. Wauters, C. Timmerer, A. Begen, F. Turck, and R. Schatz, "From capturing to rendering: Volumetric media delivery with six degrees of freedom," IEEE Communications Magazine, 2020, Vol. 58, No. 10, pp. 49–55.

[6] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in Field and Service Robotics. Springer, November 2018, pp. 621–635.

[7] Epic Games. (2019) Unreal engine. https://www.unrealengine.com

[8] J. Boyce, R. Dore, A. Dziembowski, J. Fleureau, J. Jung, B. Kroon, B. Salahieh, V. Vadakital, and L. Yu, "MPEG immersive video coding standard," Proceedings of the IEEE, 2021.

[9] Y. Cai, R. Wang, K. Qiu, R. Peng, Z. Cheng, and Q. Wang, "Depthmap video compression performance evaluation for ieee 1857.9," in Proc. of IEEE International Conference on Multimedia Expo Workshops (ICMEW'21), July 2021, pp. 1–6.

[10] S. Schwarz and M. Hannuksela, “Perceptual quality assessment of hevc main profile depth map compression for six degrees of freedom virtual reality video,” in Proc. of IEEE International Conference on Image Processing (ICIP’17), September 2017, pp. 181–185.

[11] J. Szekiełda, A. Dziembowski, and D. Mieloch, “The influence of coding tools on immersive video coding,” in Proc. of WSCG International Conference on Computer Graphics, Visualization and Computer Vision, May 2021.

[12] S. Fachada, D. Bonatto, A. Schenkel, and G. Lafruit, “Free navigation in natural scenery with DIBR: RVS and VSRS in MPEG-I standardization,” in Proc. of IEEE International Conference on 3D Immersion (IC3D), December 2018, pp. 1–6.

[13] J. Jeong, S. Lee, I. Ryu, T. Le, and E. Ryu, “Towards viewport-dependent 6DoF 360 video tiled streaming for virtual reality systems,” in Proc. of the ACM International Conference on Multimedia (MM ’20), October 2020, p. 3687–3695.

[14] B. Salahieh, S. Bhatia, and J. Boyce, “Multi-pass renderer in mpeg test model for immersive video,” in Proc. of IEEE International Conference on Picture Coding Symposium (PCS), November 2019, pp. 1–5.

[15] B. Salahieh, J. Jung, and A. Dziembowski, “Test Model 10 for MPEG Immersive Video,” International Organization for Standardization Meeting Document ISO/IEC JTC1/SC29/WG04 N0112, 2021.

[16] B. Ray, J. Jung, and M. Larabi, “On the possibility to achieve 6-DoF for 360 video using divergent multi-view content,” in Proc. of European Signal Processing Conference (EUSIPCO’18), September 2018, pp. 211–215.

[17] X. Wu, Y. Zhang, C. Fan, J. Hou, and S. Kwong, “Subjective quality database and objective study of compressed point clouds with 6dof headmounted display,” IEEE Transactions on Circuits and Systems for Video Technology, 2021, Vol. 31, No. 12, pp. 4630–4644.

[18] Y. Nehme, F. Dupont, J. Farrugia, P. Le Callet, and G. Lavoue, “Visual quality of 3d meshes with diffuse colors in virtual reality: Subjective and objective evaluation,” IEEE Transactions on Visualization and Computer Graphics, 2021, Vol. 27, No. 3, pp. 2202–2219.

[19] B. Series, “Methodology for the subjective assessment of the quality of television pictures,” Recommendation ITU-R BT, 2012, pp. 500–13.

[20] J. Jung and P. Boissonade, “VVS: Versatile View Synthesizer for 6-DoF Immersive Video,” working paper or preprint, April 2020.

[21] J. Jung and B. Kroon, “Common Test Conditions for Immersive Video,” International Organization for Standardization Meeting Document ISO/IEC JTC1/SC29/WG04 N0113, 2019.

[22] Epic Games. (2021) Unreal engine marketplace. https://www.unrealengine.com/marketplace/en-US/store

[23] MPEG. (2019) The gitlab of mpeg test model for immersive video. https://gitlab.com/mpeg-i-visual/tmiv/-/tree/v10.0.1

[24] FFmpeg. (2021) Ffmpeg. https://www.ffmpeg.org/

[25] J. Jung, B. Kroon, and J. Boyce, “Common test conditions for MPEG immersive video,” ISO/IEC JTC 1/SC 29/WG 11 N19484, 2020.

[26] Netflix. (2021) Vmaf - video multi-method assessment fusion. https://github.com/Netflix/vmaf

[27] G. Bjontegaard, “Calculation of average PSNR differences between rdcurves,” VCEG-M33, 2001.

[28] J. Benesty, J. Chen, Y. Huang, and I. Cohen, “Pearson correlation coefficient,” in Noise reduction in speech processing. Springer, 2009, pp. 1–4.

M. Roberts, D. Dey, A. Truong, S. Sinha, S. Shah, A. Kapoor, P. Hanrahan, and N. Joshi, “Submodular trajectory optimization for aerial 3D scanning,” in Proc. of IEEE International Conference on Computer Vision (ICCV’17), 2017, pp. 5324–5333.

心得

孫元駿：

這兩個學期的專題研究，我覺得受益良多，也非常的扎實。在專題研究的過程中參與也學習到很多學術研究的方法和技術。起初對於多媒體領域的一知半解，在經過大量的閱讀論文，以及和老師以及學長姐的指導之下，開始對網路以及多媒體中的基本知識以及專有名詞都有更多的了解，甚至可以發現論文的不足，或是提出自己的見解。在這兩個學期，我覺得我受到老師以及學長姐的幫助很大，不僅僅是針對研究本身，也包含了很多研究以外的細節，像是如何做好簡報、應該在報告的時候呈現什麼、有效率的提出問題、如何設計實驗以及管理實驗流程.....。每週我們都會參與自己項目的會議，以及實驗室的團體會議，我覺得這些對我來說都是很珍貴的體驗，也都可以從過程中學習到很多，也可以得到很多反饋。

這次的專題前半部主要是針對 Immersive video 合成進行研究，利用 DIBR 的技術去合成影像，並且針對 TMIV 的參數做調整，並且測量和分析。其中也用到了很多的工具，像是使用 Unreal Engine 和 AirSim API 去抓取 texture information 和 depth information，或是 PSNR、SSIM、VMAF 等 quality metric 去做影像品質的測量，也利用到 BD-Rate、BD-PSNR、PLCC、SROCC 等工具來幫助我們分析結果。

在專題的後半段主要是針對前面的測量進行延伸，針對相機擺放的問題更深入的去探討演算法，期望可以找到最佳化的相機擺放位置，以提供多人的高品質、低頻寬、實時的影像串流。同時，也針對 5G 環境的 VR Cloud Gaming 進行研究，希望可以利用 5G 的特性，去有效的利用在雲端遊戲串流之上，以降低終端設備的運算要求，使 VR 遊戲更加普及。

非常感謝指導教授、學長姐以及我的組員在這兩學期的幫助和指導，在讓我學習之外，也讓我更明確自己未來的規劃和學習方向。

王婧庭

在剛專題開始之前，我一度擔心自己的先輩知識不夠，專業能力不足，沒辦法有好的成果，但是教授告訴我每個人一開始都是這樣，慢慢學就可以了。在這兩學期的專題研究中，我從什麼都不會，一直問學長姊，到現在漸漸地有自己的想法，能夠與其他人討論，而不是只是單方面吸收，真的很感謝教授和學長姐的幫助，不僅僅只是研究方向上的，研究以外的各方面也是獲益良多，例如做實驗的方法、讀論文以及報告的技巧，還有各種需要注意的小細節，我也同時體會到在多媒體和網路領域方面還有很多值得研究的議題，以及自己在這兩個領域上有多麼不足，如同教授所說的，做研究沒有正確答案，只有一直去找更佳解。當然做研究遇到瓶頸，或是失敗也是常有的事，跑錯實驗，或是結果不如預期，都是家常便飯。所有研究成果都是失敗的累積，沒有什麼是一蹴可幾的，也因此我們才能有所成長。