CaV3: Cache-assisted Viewport Adaptive Volumetric Video Streaming

Junhua Liu [1, 2] Boxiang Zhu [1, 2] Fangxin Wang [2, 1, 3, 4 *] Yili Jin [1, 2] Wenyi Zhang [1, 2]

Zihan Xu [1, 2] Shuguang Cui [2, 1, 3, 4, 5]

[1] The Future Network of Intelligence Institute, The Chinese University of Hong Kong, Shenzhen
[2] School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen
[3] The Guangdong Provincial Key Laboratory of Future Networks of Intelligence
[4] Peng Cheng Laboratory
[5] Shenzhen Research Institute of Big Data

2023 IEEE Conference Virtual Reality and 3D User Interfaces (VR) | 979-8-3503-4815-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/VR55154.2023.00033

ABSTRACT

Volumetric video (VV) recently emerges as a new form of video application providing a photorealistic immersive 3D viewing experience with 6 degree-of-freedom (DoF), which empowers many applications such as VR, AR, and Metaverse. A key problem therein is how to stream the enormous size VV through the network with limited bandwidth. Existing works mostly focused on predicting the viewport for a tiling-based adaptive VV streaming, which however only has quite a limited effect on resource saving. We argue that the content *repeatability* in the viewport can be further leveraged, and for the first time, propose a client-side cache-assisted strategy that aims to buffer the repeatedly appearing VV tiles in the near future so as to reduce the redundant VV content transmission. The key challenges exist in three aspects, including (1) feature extraction and mining in 6 DoF VV context, (2) accurate long-term viewing pattern estimation and (3) optimal caching scheduling with limited capacity.

In this paper, we propose `CaV3`, an integrated cache-assisted viewport adaptive VV streaming framework to address the challenges. `CaV3` employs a Long-short term Sequential prediction model (`LSTSP`) that achieves accurate short-term, mid-term and long-term viewing pattern prediction with a multi-modal fusion model by capturing the viewer's behavior inertia, current attention, and subjective intention. Besides, `CaV3` also contains a contextual MAB-based caching adaptation algorithm (`CCA`) to fully utilize the viewing pattern and solve the optimal caching problem with a proved upper bound regret. Compared to existing VV datasets only containing single or co-located objects, we for the first time collect a comprehensive dataset with sufficient practical unbounded 360° scenes. The extensive evaluation of the dataset confirms the superiority of `CaV3`, which outperforms the SOTA algorithm by 15.6%-43% in viewport prediction and 13%-40% in system utility.

1 INTRODUCTION

Volumetric video (VV) is an emerging new form of video application providing unbound 360° watching experiences in photorealistic dynamic 3D scenes. Wearing a head-mounted display device, people can fully immerse themselves into a virtual 3D video scene and freely change their positions as well as rotate their heads to watch the video content from any angle. Compared to traditional 2D videos and 360° videos [16], VV provides interactivity with 6 degree-of-freedom (DoF), including three dimensions of position (X, Y, Z) and three dimensions of orientation (yaw, pitch, roll). With such experience of full immersion and interactivity, VV is envisioned as a killer application of the next generation of videos in the 5G/6G era, and will empower various multimedia services such as VR, AR, MR, and Metaverse [14]. The global VV market for industrial applications is expected to reach 22.5 billion USD by 2024 [26].

Among the several VV data formats such as point cloud [13, 20], voxel [53], mesh [41], or V-PCC [11], the most popular one is point cloud given its simplicity, which is composed of vast quantities of points with 3D coordinates and RGB attributes in a 3D space. Although VV brings a fantastic viewing experience, a key problem lies in how to stream VV with high quality of experience (QoE) given the huge gap between the enormous volumetric video size and the limited network bandwidth capacity. Taking a common 30 frames per second video as an example, when the number of points per frame is near 760,000, the bandwidth requirement for a VV is up to 2.9 Gbps [20].

Pioneer works have made efforts towards this problem in multiple aspects, i.e., codec [22,44], compression [11], bitrate adaptation [20], etc. A major direction is viewport prediction and tiling-based adaptive VV streaming to reduce the transmitted content. For example, ViVo encoded tiles into different qualities according to their relationship with users' viewport, distance, and occlusion and designed a simple adaptive streaming scheme. Li et al. [20] proposed a saliency-based algorithm for more accurate tiling. However, though such viewport adaptive solutions to some extent reduce the transmission volume, the state-of-the-art (SOTA) streaming systems can only save 40% of data usage [13] on average. The content within the viewport still requires large bandwidth resources, leading to poor QoE, especially under unstable network conditions.

In this paper, we further investigate the VV content features and watching behaviors with a comprehensive measurement and obtain a key observation: *quite a portion of video tiles will appear in the viewport repeatedly only with slightly different angle changes, even across a long timespan*. This is easy to understand since the user can freely navigate the 3D scene, so the attractive objects are more likely to enter the user's viewport. Therefore, we, for the first time, propose a client-side cache-assisted strategy that aims to buffer the popular VV tiles in the viewport in the near future so as to reduce the redundant VV content transmission. Achieving this goal is not straightforward, facing three major challenges: (1) How to extract and integrate the implicit features in VV for accurate viewport prediction. Compared with 2D video, the spatial complexity and the 6 DoF interactivity in 3D video scenes make accurate prediction extremely difficult. (2) How to achieve an accurate and long-term viewing pattern estimation of each tile to maximize the cache hit ratio. Different from most prediction tasks that only require the next moment prediction, a Long-short term Sequential Prediction is crucial in the future caching decision. (3) Given the limited client-side buffer size, how to find the popular VV tiles and optimally determine the caching policy to reduce the redundant VV transmission and

*Corresponding author: (e-mail:wangfangxin@cuhk.edu.cn)

2642-5254/23/$31.00 ©2023 IEEE
DOI 10.1109/VR55154.2023.00033

achieve a high QoE?

We propose CaV3, an integrated VV streaming framework that tackles all the above challenges. CaV3 employs a Long-short term Sequential Prediction model (LSTSP) that achieves hybrid prediction by integrating the viewer's behavior inertia, current attention, and subjective intention. Besides, CaV3 contains a contextual MAB-based caching adaptation algorithm (CCA) which can fully utilize the viewing pattern and solve the optimal caching problem with a proven upper bound regret.The contributions can be summarized as:

- We propose a novel cache-assisted viewport adaptive VV streaming framework CaV3, which, to our best knowledge, is the first work that leverages content repeatability and client-side intelligent caching to reduce redundant video content transmission and the first work to integrate accurate sequential prediction and client-cache to achieve efficient streaming. Our framework is compatible with existing bitrate adaptation solutions in VV streaming.
- We propose a Long-short term Sequential Prediction (LSTSP) model in VV that expands existing 3 DoF viewport prediction into an integrated temporal 6 DoF prediction. The method allows us to accomplish tile predictions for a long timespan rather than a single moment. Through a three-stage multimodal fusion of the viewer's gaze trajectory, attention perception, and intention inference, it not only outperforms the SOTA single-moment prediction for better performance but also achieves accurate long-term prediction adapting to VV content dynamics and personal preference.
- We propose a contextual MAB-based caching adaptation algorithm (CCA) which effectively assimilates the short-term, mid-term, and long-term predicted viewport from LSTSP as prior information to find the popular VV tiles. It trains a variant of the multi-armed bandit (MAB) model with delayed feedback, which learns the optimal mix of three-stage policies using a regret minimization strategy to make an adaptive client-cache policy under complex interactions. We prove its vanishing regret over time, which shows that its cache accumulation is optimal for a long time.
- Different from existing works only considering VV with single or co-located objects, we collect a new dataset of VV with more practical unbounded 360° scenes. Extensive evaluations on our dataset show that our prediction model outperforms baselines by 15.6%-91.8% and the caching-assisted algorithm outperforms baselines by 13%-40%.

2 Background

Volumetric Video Format: Existing works mainly focused on two main data formats for VV: 3D mesh [57] or 3D point cloud (PtCl) [42]. The mesh-based VV uses dynamic polygon meshes with texture images [4] to store the video component. The mesh topology builds objects out of a set of polygons with vertices, edges, and faces, which makes it difficult and costly to initial. VV based on the PtCl [43] consists of a series of spatial points. Each point contains space coordinates and RGB attributes. Compared to the 3D mesh-based algorithm, the 3D PtCl algorithm does not need complex pre-processing and is more effortless to encode and decode. In this paper, we mainly focus on PtCl-based VV due to its simplicity, flexibility and potential for better spatial resolution.

Volumetric Video Streaming System: The enormous VV size brings a great burden to the network bandwidth for viewer-friendly streaming. Fortunately, the flexibility in PtCl density change enables an easy bitrate adjustment, which further offers an opportunity for bitrate adaptive VV streaming while ensuring user QoE [1]. Existing works like DASH-PC and PCC-DASH [47] extend dynamic adaptive streaming over HTTP (DASH) to VVs and provide bitrate adaptation support. Pietrangelo et al. [36] propose a streaming framework that dynamically fetches objects with their level-of-details (LODs). Park et al. leverage 3D tiling for adaptive volumetric streaming. Furthermore, several existing works [12, 60] extend viewport prediction and tile-based streaming to the VV scenario, which divides the video PtCl into smaller tiles and only transmits the tiles inside the user's FoV (i.e., viewport) with adjusted density or bitrate [35, 47]. For example, ViVo [13] proposes three visibility-aware optimizations for video tiles in order to better save bandwidth consumption. Li et al. [21] consider the high computation complexity of point cloud video encoding during transmission optimization. Furion [18] uses cloud-assisted VR streaming with separated foreground and background. Yuzu [58] utilizes cache for coloring rather than PtCl data transmission. Previous works try to improve the VV streaming system from codec, compression, and mostly the tile-based bitrate adaptive streaming. Different from them, we comprehensively investigate the content repeatability and the long-short term sequential VV viewing patterns, and for the first time propose a caching-assisted framework to improve both the viewport prediction and the streaming efficiency.

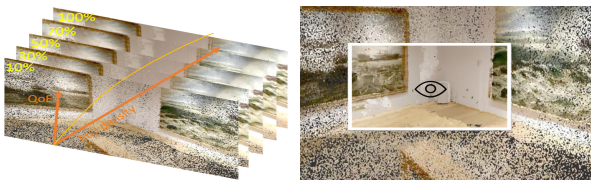
3 Measurement, Motivation and Challenges

Measurement. Unlike existing synthetic datasets with a single dynamic human, we construct a more comprehensive and practical VV dataset, which to our best knowledge is the first VV dataset that covers 360° unbounded scenes. More dataset details can be found in Sec. 7.1. To explore user behaviors under this scenarios, we have recruited 20 volunteers including 13 males and 7 females with ages ranging from 19 to 26, and asked them to watch the VVs from our dataset.Through an analysis of the viewing behaviors together with the VV contents, we have an interesting finding: the popularity of different objects/tiles are quite diverse and some attractive objects/tiles will appear in the viewport back and forth only with small angle shift, even across a long timespan. In other words, *some attractive VV content objects/tiles demonstrate strong temporal repeatability appearing in the viewport.*

Motivation. The key observation on temporal repeatability motivates us to further improve the tile-based VV streaming with PtCl, i.e., instead of streaming the tiles in the viewport every time, we cache those tiles with strong temporal repeatability so as to reduce the later data transmission for the same tiles if cache hits. Note that we mostly focus on the caching of static objects/tiles (accounting for more than 78.3% in a VV scene according to our dataset), of which the tiles generally keep stable only with viewing angle adjustment. Dynamic objects/tiles are also compatible if the incremental transmission is supported, which is our future work.

Challenges. Achieving cache-assisted VV streaming is not intuitive with three key challenges. First, an appropriate VV caching strategy highly relies on the viewport prediction, which is quite challenging, especially in the 360° unbounded 3D scenes. On one hand, the viewport can be quite flexible and dynamic due to the 6 DoF interactivity. On the other hand, people's viewport can be affected by multiple impact factors, such as moving inertia, object saliency, people's subjective intention, etc. How to extract effective features from complex information is a key problem. Second, temporal repeatability characteristics in VV call for more accurate and general sequential viewport prediction even across a long timespan, rendering the traditional next-moment viewport prediction no longer enough. The longer term we can predict, the higher caching efficiency we expect to achieve, while such prediction can become much more difficult. It is also challenging to address this contradiction. Last but not least, given the limited cache capacity and the dynamic network resources, how to better allocate the caching resource and design a caching strategy adaptive to the uncertain user behaviors and network dynamics, remains a challenging problem.

We further propose CaV3 to address the challenges above, which include a Long-short term Sequential Prediction model (Sec. 5) and a contextual MAB-based caching adaptation algorithm (Sec. 6). We will describe it in later sections.



(a) Different resolutions (b) Adaptive Bitrate Allocation

Figure 1: Different resolutions and user's viewport

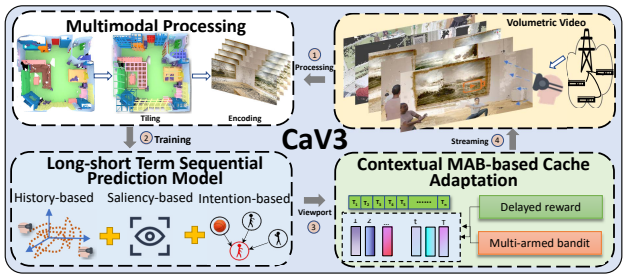


Figure 2: The Framework of CaV3

4 Framework Design of CaV3

In this section, we formulate the VV streaming problem with QoE and bandwidth consumption modeling. We then introduce the overall framework of CaV3.

4.1 Problem Formulation

We first perform fine-grained tiling of all frames in the dataset, dividing the PtCl into small 3D blocks with a size of L × L × L and dividing into T groups of frames (GoFs) with 10 frames each group. Since the video content does not change much during a single GoF time. We assume the number of tiles is the same in a GoF. VVs are encoded into different resolution versions (Shown in Fig. 1(a)) according to the user's network condition.

For the user, his/her QoE depends on the quality of the video with the viewport. We define FoV as δ_t. We define the following indicator matrix $\boldsymbol{A}_{t,n}$ for each GoF t:

$$\boldsymbol{A}_{t,k} = \begin{cases} 1, & \text{if tile k is within the viewport} \\ 0, & \text{if tile k is out of the viewport} \end{cases} \tag{1}$$

where $\boldsymbol{A}_{t,k}$ means the high-quality tile k in FoV in GoF t. We use peak signal-to-noise ratio (PSNR) for PtCl [37] to denote the average video quality, So we can use $a_{t,k} = \sum_t \sum_k \left(PSNR_{t,k} \times \boldsymbol{A}_{t,k}\right)$, and $b_{t,k} = \sum_t \sum_k \left(PSNR_{t,k} \times (1 - A_{t,k}\right)$ to denote QoE of high and low quality tiles. Thus, we can define the QoE at GoF t as

$$U_t = \frac{a_t + \beta b_t}{\sum_k \boldsymbol{A}_{t,k}} \tag{2}$$

The QoE has a proportional relationship between how many tiles are in FoV and their quality. The parameter β is the discount factor as a penalty for changing the resolution.

The bandwidth consumption between the client and content server comes from two quality video tiles. The bandwidth caused by high quality is from requesting video tiles. The bandwidth caused by low quality is all low-quality tiles in the video. Thus, the bandwidth consumption can be defined as:

$$B_t = H_1(S_{t+1} - S_t) + H_2(L_{t+1}) \tag{3}$$

where $H(.)$ is the function to calculate the bandwidth consumption caused by the inputted tiles ID. S_{t+1} is the set of tile ID within the predicted FoV $\hat{\delta_t + 1}$. , C_t is the tile ID that has been cached in GoF t and triggers cache hit in GoF $t+1$, L_{t+1} is the set of tile ID without the predicted FoV $\delta_t + 1$.

The objective of the cache manager is to optimize the system performance,i.e., maximize the QoE and minimize the bandwidth consumption, by finding the best cache policy S, and best on. Thus, the utility of this problem can be formulated as:

$$utility: max \sum_{t=0}^{T} (U_t - \beta B_t) \tag{4}$$

Sec. 5 shows that the prediction of the the future trajectory using LSTSP can improve the QoE of the system. Sec. 6 takes the obtained prediction sequences and uses the CCA algorithm to complete the cache adaptation, which ensures that we reduce the bandwidth consumption at a fixed QoE.

4.2 The Framework of CaV3

Our key idea for optimizing framework performance is to have better prediction accuracy and leverage client cache to reduce bandwidth consumption. Therefore, based on the VV streaming system in Background, we propose CaV3 (Shown in Fig. 2) which consists of two main modules: a Long-short term Sequential Prediction model LSTSPthat obtains an accurate 6 DoF prediction in a complex environment and an efficient cache adaptationalgorithm CCA that can assess the priority of each tile and heavily reduce the bandwidth consumption. We evaluate elaborate investigations on potential factors for viewport prediction. Important sensory input from the fovea's ability to see minute details can guide the human agent's future behaviors when it is controlled by foveal fixation, or alternatively, eye gazing [46]. The viewport movement is also highly correlated with video content, including the scene feature and salient agents. Motivated by these, as shown in Fig. 3, we propose a Long-short term Sequential prediction model (LSTSP) including a three-stage prediction pipeline, which fuses user viewport trajectory, agents trajectories, 3D gaze (intention), and scene feature. Then we adopt an online learning algorithm called Passive-Aggressive (PA) to adapt to personalized preferences.

The procedures of the framework can be summarized as Fig. 2:

1. We segment each frame into tiles with the same granularity and encode them into different resolution versions (As in Fig. 1(a)).
2. Based on results from the Long-short term Sequential Prediction model. CCA calculates the cache priority of each tile and sets the cache policy. The client will request high-quality tiles within the predicted viewport and low-quality tiles out of the viewport (Shown in Fig. 1(b)).
3. If the requested tiles are in the client cache, return them. If not, delayed feedback will be sent to CCA (Discuss in Sec. 6).
4. Prefetched tiles in GoF are stored in the local buffer. Then play and render them.

5 LSTSP: Long-Short Term Sequential Prediction Model

Our main content in this section is to propose a Long-short term Sequential Prediction model LSTSP, which consists of history-based, attention-based, and intention-based predictions, and to explain each of these three prediction methods in detail, and finally fuse, integrate the three-stage predictions to generate the predicted short, medium and long term viewport and adapt using an online learning approach.

It not only extracts the prediction sequences to provide cached content for cache adaptation, but also effectively maximize QoE U_t.

5.1 History-based Prediction

During watching volumetric videos, the user's movement is relatively smooth most of the time. Next motion is essentially a temporal function of user movement, which is quite correlated with the user's

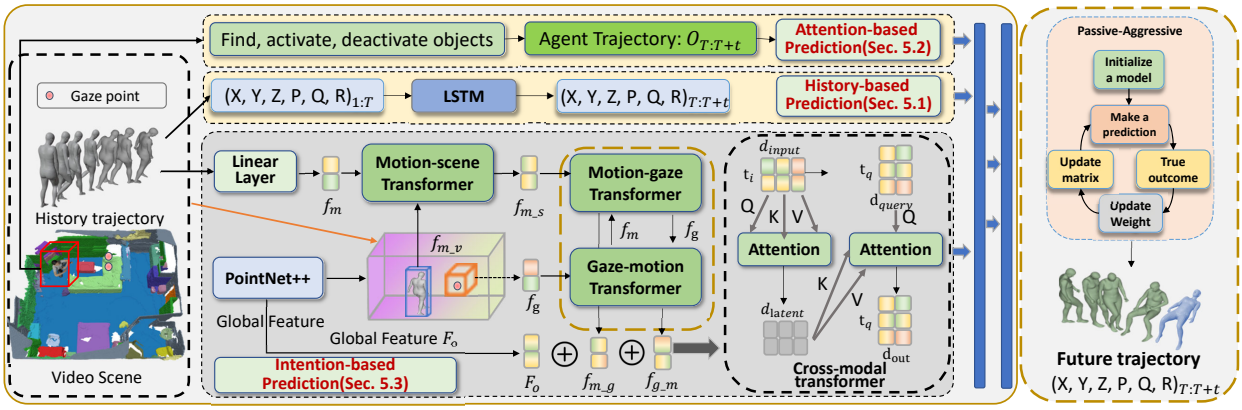


Figure 3: The Structure of LSTSP

historical trajectory under the effect of inertia. Time-series model is effective for short-period prediction within 0.5s [30]. Therefore, we leverage LSTM, an efficient and lightweight model, to predict a series of user's viewport (x, y, z, P, Q, R).

$$H_{T:T+t} = \text{LSTM}\,(H_{T-1}, H_{T-2}, \ldots, H_{T-n}) \tag{5}$$

where H_{T-1} represents user's viewport (x, y, z, P, Q, R) at time $T-1$, given user's previous 100 time steps (totally 2000ms) viewport and the model will outputs $(x_{t+1}, y_{t+1}, z_{t+1}, P_{t+1}, Q_{t+1}, R_{t+1})$, the user's viewport at the next predicted time step, then a series of future viewport $H_{T:T+t}$ as sequential prediction from T to $T+t$. However, the accuracy decreases as time increases. The viewport trajectory can be unstable. The viewer may move and rotate drastically under the impact of saliency-informed agents and the generation and vanishment of intention [40]. For the sake of better prediction accuracy, we design methods introduced in Sec. 5.2 and 5.3.

5.2 Objective Attention-based Prediction

For VV, a kind of 3D dynamic scene, the actionable spatial representations are be divided into **objects** (time-invariant physical) and **agents** (time-variant physical) [39]. The content of the video influence the viewport prediction. Inspired by related works [6, 46, 55] in autonomous driving, we found that the content grabs the user's saliency and attention. For example, depending upon his/her personal choice, the viewer may want to follow his/her favorite player or might be interested in the whole soccer field range. The viewport of the viewer follows the dancer closely as the dancer moves around.

In conclusion, the attention-based method has the potential to perform high prediction accuracy. We claim that the viewport of the viewer is likely to follow or with great potential to follow the movement trajectory of the prime objects in the video, depending upon the user's personal choice. This is a prediction based on objective scenarios. For a VV, we adopt object detection, activation, deactivation and tracking in preprocessing procedures to get the agents' trajectories.

Tracking agents essentially involves tagging them and assigning the same ID to agents in successive frames. We apply GF3D [27], the most advanced object detection network for PtCl, to detect the position and shape of saliency objects. For each agent O_i, its position trajectory from T to $T+t$ can be described as a series of position $(x_T, y_T, z_T), (x_{T+1}, y_{T+1}, z_{T+1}), \cdots (x_{T+t}, y_{T+t}, z_{T+t})$. For the sake of simplicity, we represent the middle point of 3D agents as its position. We get the object's trajectory following the procedures:
Find the active agent O_j^{active}, such that:

$$\begin{aligned} \text{diff}(O_i^{\text{current}}, O_j^{\text{active}}) &<= \text{diff}(O_i^{\text{current}}, O) \forall O \in O^{\text{active}} \\ \text{diff}(O_i^{\text{current}}, O_j^{\text{active}}) &<= \text{diff}(O', O_j^{\text{active}}) \forall O' \in O^{\text{current}} \end{aligned} \tag{6}$$

where O^{current} is the sequence of agents detected in the current frame, O^{active} is the sequence of active agents up to the previous frame. Assign the O_i^{current} with the same ID as the active agent.

Activate New agent. If an agent O_i^{current} in the current frame has not been assigned any existing active object, it shows that it appears for the first time in this video. We assign the agent a new ID and set $O^{\text{active}} = O^{\text{active}} \cup \{O_i^{\text{current}}\}$

Deactivate Old Objects. If an active agent O_k^{active} is not assigned to any new agent in the current frame, it means that either the agent has disappeared or it went undetected. If the number of consecutive frames for which the object is not assigned any new agent crosses a heuristic threshold of 30 , we declare the agent to have disappeared, and we set $O^{\text{active}} = O^{\text{active}} - \{O_k^{\text{active}}\}$.

Object Trajectory. For 3 DoF Rotational Movement Prediction, the rotation caused by object i is varying to the viewer's current position (x_t, y_t, z_t), object i's current position $(x_{t,o_i}, y_{t,o_i}, z_{t,o_i})$. The unit quaternion (qx, qy, qz, qw) represents the rotations of objects in 3D space. With the quaternion of the viewer known, we calculate the unit quaternion from the user's perspective to the object's perspective and convert into (P, Q, R). Hence we get the trajectories of each agent: $\sum_{i=1}^{N_{obj}} O_{i;T:T+t}$, where $O_{i;T:T+t} = ((x_i, y_i, z_i, P_i, Q_i, R_i)_{T:T+t}$

5.3 Subjective Intention-based Prediction

5.3.1 Prediction Problem Formulate

Scene understanding is one of the most essential aspects of intention prediction that has not been explored by prior VV systems.The global and local features of the video scene influence the viewer's tendency to act intentionally. To fully utilize the geometry information from the 3D scene and intention indications from past motions and 3D gaze for viewport prediction, we present an architecture with a bidirectional fusion model that facilitates communication between different modalities. This is a paradigm for making predictions based on subjective intentions. Followed by a variety of cross-modal transformers to transcend information from multi-modality embeddings. We employ PointNet ++ [37] as the encoding backbone to extract per-point scene features.

We represent viewport trajectory sample as a parametric sequence $V_{i:j} = \{v_i, v_{i+1}, \cdots, v_j\}$ where $v_k = (x_k, y_k, z_k, P_k, Q_k, R_k)$ is a viewport sequence at time k. The 3D scene is a point cloud $S \in R^{n\times 3}$, and the 3D gaze point $g \in R^3$ is defined as the intersection locations of the gaze direction and the 3D scene. Given the a series of history viewport trajectories $V_{1:t}$, the corresponding 3D gaze information $G_{T-n:T-1} = \{g_{T-n}, g_{T-n+1}, \cdots, g_{T-1}\}$ and the 3D scene S, we aim to predict the future viewport trajectory from T to $T+t$: $V_{T:T+t} = \Phi(V_{T-n:T-1}, G_{T-n:T-1}, S \mid \theta)$ where θ represents the network parameters, Φ represents the cross-modal transformer.

Cross-modal transformer. The cross-modal transformer [15] is used to capture the interplay of several elements and to establish communications among the multi-modal information. It is largely based on attention mechanisms [48]. The attention function [15] corresponds to a query and key-value pairs to output as:

$$\begin{aligned} \text{Attention}(Q,K,V) &= \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V, \\ Q = qW_q, K &= kW_k, V = vW_v \end{aligned} \tag{7}$$

where d denotes the dimension of the input vector, l is the sequence length, $q \in R^{l_q \times d_q}, k \in R^{l_{kv} \times d_k}, v \in R^{l_{kv} \times d_v}$ are input query, key and value vectors, and three types of weights matrix $W_q \in R^{d_q \times d_K}, W_k \in R^{d_k \times d_K}, W_q \in R^{d_v \times d_V}$ are used to project the omput modality sequence into a different representation subspace. The cross-modal attention block consists of eight multi-head cross-modal attention layers, as depicted in Fig. 3. Each input sequence of t_i length into a t_q length output by querying a t_q length feature. We denote cross-modal transformer as $\Re$. It is also efficient in fusing other multi-modal signals like audio, which has the potential to predict users' future viewport.

5.3.2 Feature Extraction

By inputting sequences of multimodality features, various modalities are processed by the Modal Fusion Cores, resulting in a collection of fused modalities. Instead of extracting the multi-modal features independently [19], we propose a pipeline to overall integrate the history viewport feature, 3D gaze feature, and scene features, which enhances the communication between motion and gaze features, so their future uncertainties can be mutually decreased, resulting in more utilization of the gaze information.

Scene feature extraction. To learn the constraints information (e.g. the surface and topology of furniture) from the 3D scene and retrain the network to pay attention to local geometric structures, we apply PointNet ++, one of the most effective PtCl backbones, to extract both global and local scene features. Specially, we derive the per-point feature map and a global descriptor of the scene as follows:

$$F_P, F_o = \Phi_{\text{scene}}\ (S \mid \theta_s) \tag{8}$$

where $S \in R^{n \times 3}$ is the input point cloud, $F_P \in R^{n \times d_p}$ is the per-point d_p dimensional feature map, and $F_o \in R^{d_o}$ is the global descriptor of the scene. Given the per-point feature F_P, the feature of an arbitrary point e can be computed through the inversed distance weighted interpolation [37]:

$$F_{P|e} = \frac{\sum_{i=1}^{n_e} w_i F_{P|p_i}}{\sum_{i=1}^{n_e} w_i}, w_i = \frac{1}{\|p_i - e\|_2} \tag{9}$$

where $\{p_1, p_2, \cdots, p_{n_e}\}$ are the nearest neighbors of e in the scene.

Gaze feature extraction. The gaze point feature f_g is then retrieved from the per-point scene feature map F_P using Eq. 9, i.e., $f_g = F_{P|g}$. Consequently, the interpolated gaze feature with corresponding scene information provides indications to infer the intention.

Viewport feature extraction. A linear layer is used to extract the viewport feature embedding f_m from multidimensional viewport information input. The viewport is well-aligned with the video scene. To endow the feature awareness of the 3D scene, we further query the scene features with the viewport features using Eq. 9. These viewport-related scene features are then supplied to PointNet++ to get the contextual scene context feature f_{m_v} of the current viewport:

$$f_{m\_v} = \text{PointNet++}(F_{P|v}) \tag{10}$$

5.3.3 Multimodal Feature Fusion

In lieu of directly concatenating the features, which would bring modalities features redundancy and impair the prediction accuracy [23], we propose a model by deploying a cross-modal transformer [29] to fuse the gaze, viewport, and scene features (Fig. 3).

Feature fusion. As an intermediary element, the viewport features strive to be cognizant of the 3D scene features and the subject's intention inferred from the gaze features. First, we utilize the scene feature f_{m_v} acquired from the 3D environment (Eq. 10) as the query to update the viewport feature f_m in viewport-scene transformer. Then, the output viewport embedding f_{m_s} is expected to be aware of the 3D scenewhich results in the final viewport embedding f_{m_g}. Inspired by [59], we handle the gaze embedding in a bidirectional manner, i.e., the viewport embedding f_m is also utilized as the query to update the gaze features into f_{g_m}. The bidirectionally fused multi-modal features are then assembled into holistic temporal input representations to perform human viewport prediction. As shown in Fig. 3, the updated gaze feature f_{g_m}, viewport feature f_{m_g} and the global scene feature F_O are used to predict the future viewport trajectories from t to T by:

$$V_{T:T+t} = \Re\left(h_{\text{pos}}, \text{concat}\left(f_{g\_m}, f_{m\_g}, F_O\right)_{T-n:T-1}\right) \tag{11}$$

where concat denotes operator of concatenation, and h_{pos} is the latent vector containing temporal positional encodings for the output [37]. Experiments validate the effectiveness of our design in utilizing gaze, 3D scene and previous viewport trajectories.

5.4 Model Fusion: Passive-Aggressive Regression

The efficiency of each step of a three-stage prediction might vary greatly depending on the setting and the user's preferences. The learning job must be rapid, precise, and online, and it should progressively update the model weights in order to quickly adjust to user choices. Therefore, we execute a lightweight Passive-Aggressive Regression model that integrates the results of three-stage and outputs a series of (x, y, z, P, Q, R) future viewport sequences.

The above set of intermediate viewports obtained is fed as an input to the Passive-Aggressive Regression model.

General Model Definition: Passive-Aggressive Regression [7] is an efficient online learning regression algorithm that computes the mapping $f : \mathbb{R}^n \to \mathbb{R}, f(\mathbf{x}; \theta) = \theta^T \mathbf{x}$ where, parameters θ, predictors $\mathbf{x} \in \mathbb{R}^n$. The algorithm uses the Hinge Loss Function, given by:

$$L(\theta, \varepsilon) = \max\left(0, |y - f(\mathbf{x_t}; \theta)| - \varepsilon\right) \tag{12}$$

where y is the actual value of the response variable. The parameter ε determines a tolerance for prediction errors. The weight update rule for PA Regression is:

$$\theta^{t+1} = \theta^t + \alpha \frac{\max\left(0, |y_t - \theta^T \mathbf{x_t}| - \varepsilon\right)}{\|\mathbf{x_t}\|^2 + \frac{1}{2C}} \text{sign}\left(y_t - \theta^T \mathbf{x_t}\right) \mathbf{x_t} \tag{13}$$

We run Passive-Aggressive Regression model that predicts the 6 DoF viewport (X, Y, Z, P, Q, R) for the next set of GoF (see Fig. 3). Along with the history viewport, the model uses the object trajectories that were pre-calculated by object detection, and object tracking and predicted viewport from multimodal fusion. The equations for the predictions in a future chunk of viewports T to $T + t$ are given by:

$$F_{T:T+t} = \theta_0 + \theta_H \cdot H_{T:T+t} + \sum_{i=1}^{N_{obj}} \theta_i \cdot O_{i;T:T+t} + \theta_V \cdot V_{T:T+t} \tag{14}$$

where $F_{T:T+t}$ is the predicted sequential viewports from LSTSP, $H_{T:T+t}$ is the predicted viewport obtained from history-based model for GoF f, $O_{i;T:T+t}$ is the space coordinates and calculated rotation

of the i^{th} object, $V_{T:T+t}$ is the predicted viewport obtained from intention-based model from T to $T+t$, and $\theta_{0;T:T+t}$,θ_H, $\theta_{i;T:T+t}$, θ_V are bias and corresponding parameters of each model. $\in R^{t \cdot fps}$

For Eq. 14, we get the contribution of each stage in the prediction. For example, the contribution of object trajectories at frame f are $\sum_{i=1}^{N_{obj}} \theta_{it} \cdot O_{Tif}$. When the next chunk of point cloud frames is rendered in the video streaming, the current set of user viewports is obtained, and the weights of each part of the Passive Aggressive Regression model are updated using the predicted and actual values of viewports as per the update rule in Eq. 13.

Especially, having obtained the sequential viewport, we extract short, medium and long-term predicted viewport for time T as cache policy for CCA. The calculation process can be is given by:

$$
\begin{aligned}
F_{T+n}^{short} &= \theta_0 + \theta_{H;T+n} \cdot H_{T+n} + \sum_{i=1}^{N_{obj}} \theta_{i;T+n} \cdot O_{i;T+n} + \theta_{V;T+n} \cdot V_{T+n} \\
F_{T+3n}^{med} &= \theta_0 + \theta_{H;T+3n} \cdot H_{T+3n} + \sum_{i=1}^{N_{obj}} \theta_{i;T+3n} \cdot O_{i;T+3n} + \theta_{V;T+3n} \cdot V_{T+3n} \\
F_{T+5n}^{long} &= \theta_0 + \theta_{H;T+5n} \cdot H_{T+5n} + \sum_{i=1}^{N_{obj}} \theta_{i;T+5n} \cdot O_{i;T+5n} + \theta_{VT+5n} \cdot V_{T+5n}
\end{aligned}
\tag{15}
$$

where F_{T+n} is the predicted single-point viewport at time $T+n$, n represents the value of time segmentation. In this paper, we assume n = 1s.

6 CCA: Contextual MAB-based Cache Adaptation

In Sec. 5, we get the sequential predicted viewport and extract the predicted viewport of short, medium and long term, which have one-to-one correspondence caching policy. Based on three time-varying policies, We propose the CCA algorithm, a variant of MAB with delayed reward, whose goal is to minimize the number of cache misses under limited cache capacity and thus minimize the bandwidth consumption B_t. We show its upper bound and prove its regret vanishes over time.

6.1 Background and Motivation of CCA

In our scenario, classical algorithms such as LRU and LFU and existing adaptive algorithms like ARC [32]perform poorly. The most adaptive algorithm assumes that the best strategy at a given time is a probabilistic mix of the two policies (LFU/LRU). Among the recent, the SOTA adaptive cache replacement algorithm LeCaR [51] using Multi-armed bandit(MAB) to learn the optical probabilistic mix outperforms ARC by 18x. Their performances are highly limited because of viewport shifting. So we introduce the CCA, an enhanced adaptive cache adaptation algorithm based on MAB. The enhancement mainly lies in three aspects, including (1) We address viewport shifting using the predicted viewport from LSTSP as a cache strategy. We assume the best cache strategy is a probabilistic mix of short, medium and long-term policy and learn the optimal mix using a regret minimization strategy. (2) If an evicted item is requested immediately after its eviction, the regret is much higher [9], which assumes the cost decreases with increased delay. So we introduce a delayed feedback module (3) One of the main drawbacks of LeCaR is its fixed learning rate. We give a theoretically adaptive optimal learning rate to improve its performance.

6.2 Cache Adaptation Problem Formulation

For cache replacement problems, each expert represents a distinct cache replacement policy, which advises cache and evicts when reaching the cache size. When playing the VV GoF of t, three experts (short, medium, long term prediction model) are consulted. From their three-stage predicted FoV for $t+1$, tiles within the

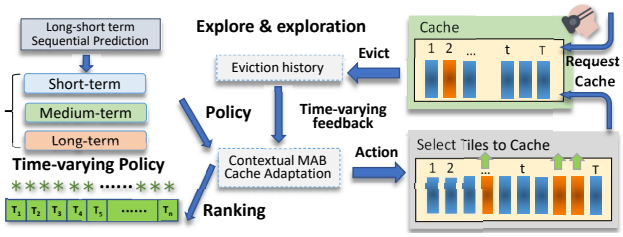


Figure 4: The Structure of CCA

FoV will be cached, and tiles out of the FoV will be evicted. The probability of each tile depends on the three-stage FoV and their weights. Without scene-cut, the number of tiles is stable, and the content of nearly 90% of tiles does not change. The recorded newly activated and deactivated tiles will be empty. Assume there are K tiles in the video scene.

If no prior knowledge is available about the experts, CCA initializes by assigning equal weights. Thus, $w_i(1) = 1, i = 1, \ldots, N$. The probability, $p_j(t)$, of picking cache action for tile $j = \{1, \ldots, K\}$ in GoF t is proportional to the sum of the weights of the experts that recommend action j. If more experts recommend an action, that action will have a higher probability to be chosen by the algorithm. This is interpreted as exploitation the information accumulated by the learning algorithm so far. If the player chooses to explore with a random action, the algorithm will choose an action randomly from the available tiles. The probability can be summarized by Eq. 16:

$$
p_j(t) = (1-\eta) \sum_{i=1}^{N} \frac{w_i(t) \times \xi_i^j(t)}{W_t} + \frac{\eta}{K} \tag{16}
$$

where η is the learning rate, $w_i(1) = 1, i = 1, \ldots, N$, $W_t = \sum_{i=1}^{N} w_i(t)$, and $\xi_i^j(t)$ is the probability of choosing action j by expert i at GoF t. The learning rate η controls the amount of exploration and exploitation at each GoF. If the learning rate is too high, the algorithm will explore more and exploit less. Feedback on the eviction comes in the form of a cache miss(extra), but at an indeterminate time after the action is taken, and the cost of the eviction is set to be inversely proportional to the response time.

We assume that in GoF t, the vector of values of the cost function for each cache action are denoted by $\mathbf{x}(t) = (x_1(t), \ldots, x_K(t))$, where $x_j(t) \in [0,1]$. The main objective of the CCA is to minimize the regret (minimize bandwidth consumption) over a T GoFs. The cumulative cost after T GoFs for the best strategy is given by:

$$
C_{\text{best}}(T) = \min_{1 \le i \le N} \sum_{t=1}^{T} \xi_i(t) \cdot \mathbf{x}(t) \tag{17}
$$

which assumes that the best of the N expert recommendations are followed in each GoF. Also, the cumulative cost incurred by CCA selecting action $a(t)$ in t GoF is given by:

$$
C_{\text{CCA}}(T) = \sum_{t=1}^{T} \xi(t) \cdot x_{i_t}(t) \tag{18}
$$

To measure the performance of adaptive learning algorithms, regret can be defined in terms of the cumulative difference between the costs of the best strategy in each step and the algorithm in consideration. Thus, the regret $R_{\text{CCA}}(T)$ of algorithm CCA after GoF T can be calculated from:

$$
R_{\text{CCA}}(T) = C_{best}(T) - C_{\text{CCA}}(T) \tag{19}
$$

The main objective of any learning algorithm is to minimize the total regret over time while ensuring it vanishes over a long time horizon.

In other words, It aims to minimize the number of cache misses thus minimizing bandwidth consumption.

When feedback is non-zero, the weights of the experts get updated. A feedback is generated at GoF t, triggered by an action, $i_{t'}$, taken back in GoF t'. As the delay in feedback, $t-t'$, increases, the relative cost of the chosen action decays. Since regret is proportional to cost, the weights of experts who chose the action $i_{t'}$ are decreased using the estimated cost value observed at GoF t. The estimated cost is calculated as:

$$\hat{\mathbf{x}}_{i_t}(t) = \frac{x_{i_t}(t)}{d \times p_{i_t}(t)} \tag{20}$$

In summary, the pseudocode of our cache adaptation algorithm is shown in Algorithm 1.

6.3 CCA Algorithm

In conclusion, the CCA algorithm is formulated in Algorithm 1: Fi-

Algorithm 1 The CCA Algorithm

Input: Request sequence, σ; Cost vectors, $\mathbf{x}(t)$; Learning rate, $\eta \in (0,1]$; Cache size, h; Number of tiles, K; set of experts $= \{Short, Medium, Long\}$; Number of experts, N;
for $t = 1,2,...$ to T **do**
  while Existing cache $\leq$ Cache Size h **do**
    Obtain expert advice $\xi_1(t),\ldots,\xi_N(t)$ for request $\sigma(t)$;
    Cache tile $j \in \{1,\ldots,K\}$ with prob,
    $p_j(t) = (1-\eta)\sum_{i=1}^{N} \frac{w_i(t)\times\xi_i^j(t)}{W_t} + \frac{\eta}{K}$
  end while
  for $i = 1,\cdots,N$ **do**
    $\hat{x}_{i_t}(t) = \begin{cases} \frac{x_{i_t}(t)}{d}, & \text{if } i_{t'} = j \text{ and } \sigma(t') \text{ is in history in position } d \\ 0, & \text{otherwise} \end{cases}$
    $w_i(t+1) = w_i(t)\exp\left(\frac{-\eta\hat{\mathbf{x}}(t)\cdot\xi_i(t)}{K}\right)$
  end for
end for

nally, we would like to draw a conclusion on the performance of the generic CCA algorithm. As with other RL algorithms, we formalize the regret bound for CCA. Based on the proof of the previous MAB-based caching method [2, 52, 56], we theoretically guaranteed to have vanishing regret over time.

Our method is not necessarily restricted to short, medium and long-term policy. In Theorem 1, we prove the upper bounded of CCA for any policies and tiles of any positive number.

Upper Bound of the vanishing regret:

Theorem 1. *For any $K, T > 0$, for any learning rate, $\eta \in (0,1]$, for any family of N experts, and for any assignment of arbitrary costs decaying with delay d, the expected regret of the algorithm can be upper bounded by: $R_{CCA}(T) \leq 2\eta T + \frac{K\ln N}{\eta}$.*

The Proof of Theorem 1 is available in the supplementary material.

Selecting the optimal learning rate for vanishing regret:

One of the main drawbacks of MAB-based cache replacement is its fixed learning rate. Theorem 1 shows that cumulative regret is a function of the learning rate. We argue that picking the right learning rate can minimize regret and ensure that it vanishes. We differentiate the above quantity and set it to 0. Thus the equation, $R'_{CCA}(T) = 2T - \frac{K\ln N}{\eta^2} = 0$, gives us the optimal value of η as follows:

$$\eta_{Opt} = \min\left(1, \sqrt{\frac{K\ln N}{2T}}\right) \tag{21}$$

Finally, plugging this back into the upper bound gives us the following regret bound:$R_{CCA}(t) \leq 2\sqrt{2KT\ln N}$. As $T \to \infty$, the regret bound vanishes with the time horizon.

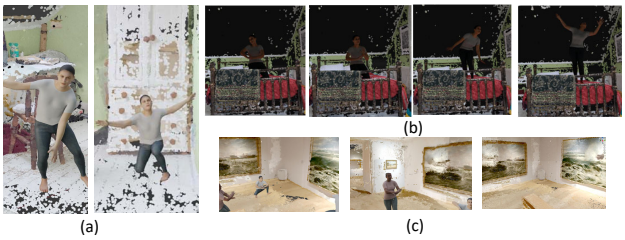


Figure 5: (a) Movement of agent. (b) A series of sequences of woman's movement. (c) are the layout in the room in VV.

7 Evaluation

7.1 Dataset

Dataset overview. Limited by capturing techniques, existing dynamic PtCl datasets [3, 17] use a single or co-located synthetic 3D object/person (e.g. *loot* and *longdress*), which lack authenticity and expression and restricts people's viewport to a limited range. To this end, we construct a more comprehensive and practical VV dataset. It achieves unbounded 360-degree scenarios with five daily scenes , eight sets of dance moves and 1-4 moving agents in each VV. Human interaction in social XR settings are dipicted. As shown in Fig. 5, it provides ten kinds of intention-scene relation. It provides data including gaze, scene, and tracking trajectories. We collect tracking data from twenty people under different settings. In summary, our dataset is more enjoyable, interactive, and realistic.

Dataset setup: We convert existing dynamic 3D models into dynamic PtCl sequences. For example, we collect the joint capture data of several dance sequences, fit them into the SMPL model [28] and convert it into dynamic PtCl, which averagely takes 17MB/s per agent. We then combine them with the existing scene dataset Scannet [8], Sunrgbd [45], animate and render the VV in Unity and Blender to produce scenes containing different interior layouts, different objects, and agents that move with different patterns in the scenes. Having areas ranging from $2.5 \times 2.5m^2$ to $7 \times 7m^2$ with a height of 3.5m. Viewers freely watch and explore these one-minute videos. The 6 DoF information is recorded every 0.01s.

Dataset capture: The gaze and head tracking data are measured by VIVE Pro Eye[1], a VR headset with precision eye tracking. VIVE provided SRanipal software development kit (SDK)[2] for developers to collect the gaze data. We develop a Unity project based on OpenVR[3] and OpenXR[4] to capture and save to a log file gaze data and head tracking data from the headset.

7.2 Setup and Baseline

Model Setup: The implementation of the prediction model and cache management is based on Python and Pytorch. We build our pipeline by incorporating 6 cross-modal transformer layers to extract 256 dimension gaze and viewport features. We adopt L1 loss between the predicted viewport and the ground Truth, Adam optimizer and learning rate $1e^{-5}$. The head number of each multi-head cross-modal attention and self-attention is 5. For cache adaptation, the learning rate is from the proof in Sec. 6.2. We set each video chunk to be 1s and the buffer capacity to be 5 seconds.

Communication and Computational Resource: In the experiments, two typical real wireless network traces, denoted as BW1 and BW2, are selected with average downlink rates of about 40 Mbps and 120 Mbps, respectively. We assume that the decoding process

[1]https://www.vive.com/us/product/vive-pro-eye/overview/
[2]https://developer-express.vive.com/resources/vive-sense/
[3]https://github.com/ValveSoftware/openvr
[4]https://www.khronos.org/openxr/

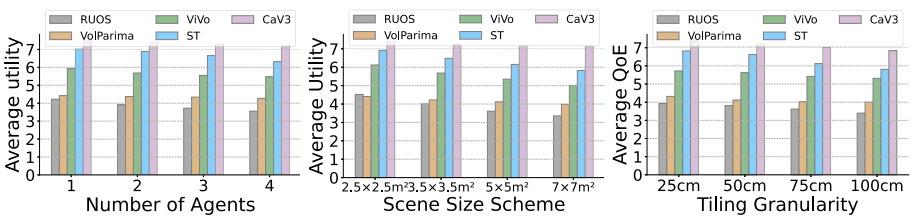


Figure 6: (a) Average utility value over different number of agents, and (b) over different scene size schemes(c) Average QoE over different tiling granularity,

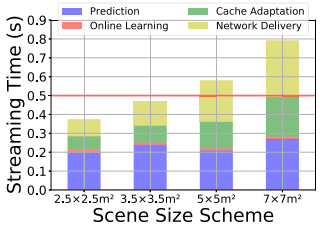


Figure 7: Maximal streaming times over different scene size schemes.

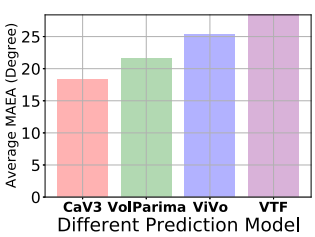


Figure 8: Average MAEA for viewport prediction

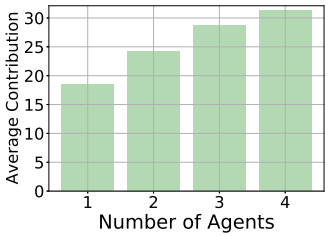


Figure 9: Attention-based prediction over different numbers of agents

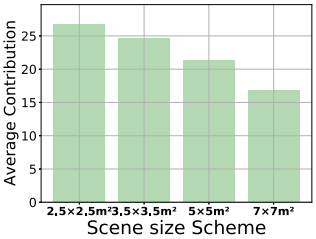


Figure 10: Intention-based prediction over different scene schemes

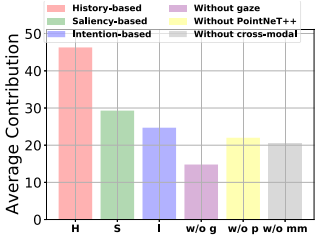


Figure 11: Average contribution of each part in prediction model

can be multithreaded. The whole pipeline is running on a high-performance laptop with GTX 1660 Ti GPU, Intel i7 9750H CPU, and 32GB memory. The number of CPU cores owned by the client is assumed to be 2 (denoted as NC1) and 4 (denoted as NC2), which reflects the different amounts of computational resources available.

Baseline: We compare CaV3 with systems and algorithms below.

- ViVo [13]: an integrated streaming system adopting visibility-aware optimizations. It determines the density of PtCl to fetch based on the viewer's perception. The bitrate selection scheme is assigned homogeneous bitrate to the content within the viewport.
- Saliency tiling (ST) [20]: A saliency-based tiling method that achieves VV streaming with high viewer's QoE. It calculates the saliency value of the cells and derives a tiling optimization for joint computational and communication resources.
- VolParima: We extend the key idea of Parima [5] into VV. It adopts a pyramid prediction algorithm using both LSTM and video content to predict and Passive-Aggressive to adjust.
- Rate Utility Optimized Streaming (RUOS) [35]: It utilizes a rate-utility greedy heuristic optimization to allocate bitrate and adopts a non-learning statistical approximation method to predict.
- LRU/LFU [31, 34]: The least recently/frequently used cache is evicted when reaching the cache's capacity limit.
- ARC [32]: It keeps track of both frequently used and recently used pages plus a recent eviction history for both.
- LeCaR [52]: It models cache replacement as a MAB problem and explores online learning with variants of MAB.

7.3 Evaluation Result

In this part, we explain our experimental results. From the perspective of the dynamic and static states, we analyze the impact of the number of agents and different scene sizes. Our goal is to examine the effectiveness of the whole framework, prediction model, and cache adaptation algorithm and analysis.

7.3.1 Long-Short Term Sequential Prediction Performance

We evaluate LSTSP against ViVo, VolParima and transformer-based Vanilla-TF (VTF) [49]. We also do the ablation study to compare the contribution of each part of our framework.

Prediction Accuracy: We can use the Average Mean Absolute Error Angle (MAEA) of the 3 DoF rotation prediction. As shown in Fig. 8, our CaV3 reduces MAED by 15.63%, 23.56%, and 40.12% compared with VolParima, ViVo and VTF, which demonstrates that our model can fuse our three-stage models, learn user's preference and achieve a higher prediction accuracy.

Ablation Study: We then compare each part contribution of the model: history-based (H), saliency-based (S), intention-based (I), and its corresponding ablation study include: without gaze (w/o g), without PointNet ++ (w/o p) and without cross-modal transformer (w/o mm). It proves that each part has a positive contribution. The gaze and scene understanding is well utilized and helpful to the task. It can be seen from Fig. 11 that CaV3 can effectively fuse the information, adjust the parameter according to preference and achieve better accuracy.

7.3.2 CCA Performance

We evaluate CCA against LRU, LFU, ARC, and LeCaR under different cache sizes to show its performance. Compared with other cache replacement algorithms, our contextual CCA effectively assimilates the predicted viewport as prior information. As shown in Fig. 13 and Fig. 14, our model can heavily resist interference from dynamic agents and complex static scenes, which proves that our model could well fuse the results from three stages and adopt a reinforcement learning algorithm to find the novel cache strategy. To further present the finding, we conduct ablation studies shown in Fig. 12. Compared with the single prediction model result, our model reduces bandwidth consumption from 13.4% to 30.2%. It also shows that the delayed mechanism plays an important role and is beneficial to the task. As Fig. 15 shows, the CCA has a relatively fast increase in hit rate as the cache size increases, which means it can well utilize the limited cache size.

7.3.3 Impact of Number of Agents

Different numbers of agents (dynamic objects in the viewport) can have different impacts on the framework. We evaluate the same scene with 1 to 4 agents inside the video with 25cm tiling granularity. The average utility of different schemes is shown in Fig. 6(a). As the number of agents increases, the performance of the proposed model increases and achieves the highest utility, showing an average improvement of 11.7% over other best frameworks. To further excavate the reason, results in Fig. 9 and Fig. 13 show that the contribution (its corresponding parameter in online learning) of saliency-based prediction increase, and the hit rate of CCA is basically unaffected, which show that our model has the capacity to capture the dynamic scene and get an accurate prediction.

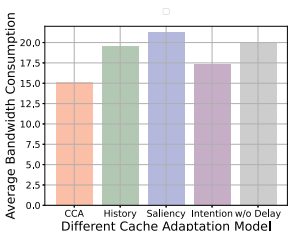


Figure 12: Bandwidth consumption over different prediction methods.

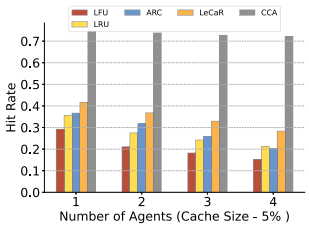


Figure 13: Hit rate over different numbers of agents.

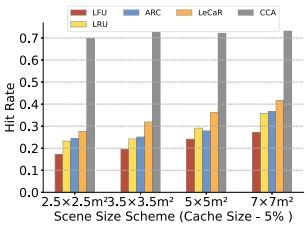


Figure 14: Hit rate over different number of scene schemes.

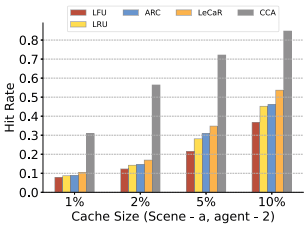


Figure 15: Hit rate of different algorithms over different cache sizes .

7.3.4 Impact of Different Video Scenes

A further finding is about scene understanding. Different types of video scenes can have different impacts on the framework, which is highly correlated with scene understanding and intention prediction. We first fix the scene size to be several schemes, a: $2.5 \times 2.5m^2$, b: $3.5 \times 3.5m^2$, c: $5 \times 5m^2$, d: $7 \times 7m^2$, two agents. The height of all scenes is $3.5m$ and the tiling scheme is $50 \times 50 \times 50cm^3$. According to Fig. 6(b), `CaV3` outperforms the four baselines, showing an average 8.3%-20.3% improvement of utility value over the best working framework. As shown in the figure, when the size of the scene increases, only `CaV3` and `VolParima` maintain a stable utility, while the other three baselines have relatively declining utilities. Since the rest of the three baselines' prediction models only focus on the viewer's pattern. The prediction model will result in lower accuracy when the size of the scene and bandwidth consumption increase. Fig. 9 and 14 show that our intention-based model contributes more and the cache hit rate remains stable compared with another model in the large scene, which demonstrates higher prediction accuracy and lower bandwidth consumption.

7.3.5 Impact of Different Tiling Granularity

When the tiling granularity changes, the performance of each model and streaming time is affected, especially for models with adaptive bitrate allocation like `ViVo, ST, RUOS`. A fine-grained bitrate allocation will increase QoE to varying degrees. We select four tiling schemes, including 25cm, 50cm, 75cm and 100cm. Then we compare `CaV3` against four baselines under scene including one agent. Fig. 6(c) shows that `CaV3` has a higher QoE value than other baselines under all tiling schemes. As the granularity increases, the QoE of these adaptive model decrease, which shows that `CaV3` is also sensitive to granularity because of fine-grained cache adaptation. Our model achieves an average QoE promotion of 14.6% over `ST`, 32.4% over `ViVo`, 78.3% over `VolParima` and 91.8% over `RUOS`. As the PtCl is a series of 3D points, we can directly implement the methods at the coordinate level without extra overhead. Too many tiles will bring extra overhead and computation consumption for processes such as bitrate allocation and cache adaptation.

7.3.6 Impact of Video Chunk Size

We deploy our streaming framework in different scene size schemes. Fig. 6(a) shows a little QoE promotion from 50cm to 25cm. Therefore, we select the 25cm tiling scheme to investigate the overhead caused by tiling. As the scene becomes larger, larger numbers of tiles increase the computation consumption and network consumption. The network time, prediction time, and cache adaptation time are highly corrected with numbers of tiles. The model update is meager and not a bottleneck for any of the videos. As Fig. 7 shows, in schemes $5x5m^2$ and $7x7m^2$, the maximal streaming time exceeds 0.5s. The total streaming time is comfortably under 1 second for a chunk duration of 1s. Hence, we use a chunk size of 1s. The results show that the refine tiling scheme and large numbers of tiles will not break the real-time transmission.

8 Discussion

More application support: Our model provides a universal viewport prediction paradigm for 3D video, including mesh-based VV and potential future 3D video formats like Neural radiance field (NeRF) video [24]. Existing mobile VV streaming based on cloud rendering [25] is heavily dependent on viewport prediction to alleviate the increased latency. Our prediction model can offer great relief to discomforted spinning sensations by Motion-To-Photon latency and higher QoE. For Multi-user XR or Metaverse, our model has excellent compatibility that can be combined with Multi-Range Transformers [54] for interaction-based prediction.

More optimization support: Our framework can concatenate with more streaming optimizations like bitrate selection [38], coarse-grained cell bitrate allocation, [20] and CODEC methods for Point cloud [33]. For simplicity, our model assumes each cell has the same bitrate. This approach can be improved into arbitrary bitrate by adopting a variant of the knapsack problem. The multimodal transformer can fuse more features like spatial audio [50] and caption text [10], which will bring better immersion and prediction accuracy.

Streaming dynamic content: Without the scene-cut, the content of each tile change slightly over time. We conduct differential transmission that extra residual is transferred when cache hits. For tiles with dynamic agents, the content may largely changed because of the movement. We search the target tile, calculate the motion vector [20], and conduct differential transmission. With the scene-cut, the recorded newly activated and deactivated tiles will be empty, the client request the entire contents.

9 Conclusion

In this paper, We propose a VV streaming framework `CaV3`, where accurate prediction and cache adaptation algorithms are first integrated to achieve efficient streaming. The Long-short term Sequential Prediction model `LSTSP` utilizes a multi-modal transformer to achieve accurate 6 DoF viewport prediction and quickly learn to adapt to the viewer's preference and video contents. In addition, with prior information from `LSTSP`, the cache adaptation model `CCA` can make a novel cache policy and minimize the total regret over time. Extensive trace-driven experiments have shown the superiority of our framework in prediction, cache adaptation and achieving better utility under different scenarios.

Acknowledgments

The work was supported in part by the Basic Research Project No. HZQB-KCZYZ-2021067 of Hetao Shenzhen-HK S&T Cooperation Zone, by National Natural Science Foundation of China with Grant No.62102342, by Shenzhen Outstanding Talents Training Fund 202002, by Guangdong Research Projects No. 2017ZT07X152 and No. 2019CX01X104, by the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No. 2022B1212010001), and by The Major Key Project of PCL Department of Broadband Communication.

REFERENCES

[1] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, J. Levine, A. Sharf, and C. Silva. State of the art in surface reconstruction from point clouds. Eurographics 2014-State of the Art Reports, 1(1):161–185, 2014.

[2] A. Beygelzimer, J. Langford, L. Li, L. Reyzin, and R. E. Schapire. Contextual bandit algorithms with supervised learning guarantees. arXiv: Learning, 2010.

[3] F. Bogo, J. Romero, G. Pons-Moll, and M. J. Black. Dynamic faust: Registering human bodies in motion. computer vision and pattern recognition, 2017.

[4] M. Campen and L. Kobbelt. Exact and robust (self-) intersections for polygonal meshes. In Computer Graphics Forum, vol. 29, pp. 397–406. Wiley Online Library, 2010.

[5] L. Chopra, S. Chakraborty, A. Mondal, and S. Chakraborty. Parima: Viewport adaptive 360-degree video streaming. In Proceedings of the Web Conference 2021, pp. 2379–2391, 2021.

[6] V. Clay, P. König, and S. Koenig. Eye tracking in virtual reality. Journal of eye movement research, 12(1), 2019.

[7] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. Journal of Machine Learning Research, 2006.

[8] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5828–5839, 2017.

[9] T. Desautels, A. Krause, and J. W. Burdick. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. Journal of Machine Learning Research, 2014.

[10] S. Y. Feng, K. Lu, Z. Tao, M. Alikhani, T. Mitamura, E. Hovy, and V. Gangal. Retrieve, caption, generate: Visual grounding for enhancing commonsense in text generation models. arXiv: Computation and Language, 2021.

[11] D. B. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai. An overview of ongoing point cloud compression standardization activities: video-based (v-pcc) and geometry-based (g-pcc). APSIPA Transactions on Signal and Information Processing, 2020.

[12] C. Guo, Y. Cui, and Z. Liu. Optimal multicast of tiled 360 vr video in ofdma systems. IEEE Communications Letters, 22(12):2563–2566, 2018.

[13] B. Han, Y. Liu, and F. Qian. Vivo: Visibility-aware mobile volumetric video streaming. In Proceedings of the 26th annual international conference on mobile computing and networking, pp. 1–13, 2020.

[14] M. Hosseini and C. Timmerer. Dynamic adaptive point cloud streaming. In Proceedings of the 23rd Packet Video Workshop. ACM, jun 2018. doi: 10.1145/3210424.3210429

[15] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer, O. J. Hénaff, M. Botvinick, A. Zisserman, O. Vinyals, and J. Carreira. Perceiver io: A general architecture for structured inputs & outputs. arXiv: Learning, 2021.

[16] Y. Jin, J. Liu, and F. Wang. Ebublio: Edge assisted multi-user 360-degree video streaming. In 2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops, VR Workshops, Christchurch, New Zealand, March 12-16, 2022, pp. 600–601. IEEE, 2022. doi: 10.1109/VRW55335.2022.00151

[17] K. Koide, M. Yokozuka, S. Oishi, and A. Banno. Voxelized gicp for fast and accurate 3d point cloud registration. international conference on robotics and automation, 2021.

[18] Z. Lai, Y. C. Hu, Y. Cui, L. Sun, N. Dai, and H.-S. Lee. Furion: Engineering high-quality immersive virtual reality on today's mobile devices. IEEE Transactions on Mobile Computing, 19(7):1586–1602, 2019.

[19] J. Li, Y. Yin, H. Chu, Y. Zhou, T. Wang, S. Fidler, and H. Li. Learning to generate diverse dance motions with transformer. arXiv: Computer Vision and Pattern Recognition, 2020.

[20] J. Li, C. Zhang, Z. Liu, R. Hong, and H. Hu. Optimal volumetric video streaming with hybrid saliency based tiling. IEEE Transactions on Multimedia, pp. 1–1, 2022. doi: 10.1109/TMM.2022.3153208

[21] J. Li, C. Zhang, Z. Liu, W. Sun, and Q. Li. Joint communication and computational resource allocation for qoe-driven point cloud video streaming. IEEE International Conference on Communications (ICC), 2020.

[22] L. Li, Z. Li, S. Liu, and H. Li. Efficient projected frame padding for video-based point cloud compression. IEEE Transactions on Multimedia, 2021.

[23] R. Li, S. Yang, D. A. Ross, and A. Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++. arXiv: Computer Vision and Pattern Recognition, 2021.

[24] T. Li, M. Slavcheva, M. Zollhoefer, S. Green, C. Lassner, C. Kim, T. Schmidt, S. Lovegrove, M. Goesele, and Z. Lv. Neural 3d video synthesis. arXiv: Computer Vision and Pattern Recognition, 2021.

[25] Y. Liu, B. Han, F. Qian, A. Narayanan, and Z.-L. Zhang. Vues: Practical mobile volumetric video streaming through multiview transcoding. 2022.

[26] Z. Liu, Q. Li, X. Chen, C. Wu, S. Ishihara, J. Li, and Y. Ji. Point cloud video streaming: Challenges and solutions. IEEE Network, 2021.

[27] Z. Liu, Z. Zhang, Y. Cao, H. Hu, and X. Tong. Group-free 3d object detection via transformers. arXiv: Computer Vision and Pattern Recognition, 2021.

[28] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. Smpl: a skinned multi-person linear model. international conference on computer graphics and interactive techniques, 2015.

[29] S. Luo, H. Dai, L. Shao, and Y. Ding. C4av: learning cross-modal representations from transformers. In European Conference on Computer Vision, pp. 33–38. Springer, 2020.

[30] O. Makansi, J. von Kügelgen, F. Locatello, P. V. Gehler, D. Janzing, T. Brox, and B. Schölkopf. You mostly walk alone: Analyzing feature attribution in trajectory prediction. arXiv: Learning, 2021.

[31] D. Mátáni, K. Shah, and A. Mitra. An o(1) algorithm for implementing the lfu cache eviction scheme. arXiv: Data Structures and Algorithms, 2021.

[32] N. Megiddo and D. S. Modha. Outperforming lru with an adaptive replacement cache algorithm. IEEE Computer, 2004.

[33] R. Mekuria, K. Blom, and P. Cesar. Design, implementation, and evaluation of a point cloud codec for tele-immersive video. IEEE Transactions on Circuits and Systems for Video Technology, 2017.

[34] E. O'Neil, P. O'Neil, and G. Weikum. The lru-k page replacement algorithm for database disk buffering. international conference on management of data, 1993.

[35] J. Park, P. A. Chou, and J.-N. Hwang. Rate-utility optimized streaming of volumetric media for augmented reality. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 9(1):149–162, 2019.

[36] S. Petrangeli, G. Simon, H. Wang, and V. Swaminathan. Dynamic adaptive streaming for augmented reality applications. In 2019 IEEE International Symposium on Multimedia (ISM), pp. 56–567. IEEE, 2019.

[37] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, eds., Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pp. 5099–5108, 2017.

[38] F. Qian, B. Han, J. Pair, and V. Gopalakrishnan. Toward practical volumetric video streaming on commodity smartphones. In Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications. ACM, feb 2019. doi: 10.1145/3301293.3302358

[39] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone. 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans. In M. Toussaint, A. Bicchi, and T. Hermans, eds., Robotics: Science and Systems XVI, Virtual Event / Corvalis, Oregon, USA, July 12-16, 2020, 2020. doi: 10.15607/RSS.2020.XVI.079

[40] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras. Human motion trajectory prediction: A survey. The International Journal of Robotics Research, 2019.

[41] O. Schreer, I. Feldmann, S. Renault, M. Zepp, M. Worchel, P. Eisert, and P. Kauff. Capture and 3d video processing of volumetric video. international conference on image processing, 2019.

[42] O. Schreer, I. Feldmann, S. Renault, M. Zepp, M. Worchel, P. Eisert,

and P. Kauff. Capture and 3d video processing of volumetric video. In 2019 IEEE International conference on image processing (ICIP), pp. 4310–4314. IEEE, 2019.

[43] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuća, S. Lasserre, Z. Li, et al. Emerging mpeg standards for point cloud compression. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 9(1):133–148, 2018.

[44] X. Sheng, L. Li, D. Liu, and Z. Xiong. Attribute artifacts removal for geometry-based point cloud compression. arXiv: Computer Vision and Pattern Recognition, 2021.

[45] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 567–576, 2015.

[46] B. W. Tatler, M. Hayhoe, M. F. Land, and D. H. Ballard. Eye guidance in natural vision: reinterpreting salience. Journal of Vision, 2011.

[47] J. van der Hooft, T. Wauters, F. De Turck, C. Timmerer, and H. Hellwagner. Towards 6dof http adaptive streaming through point cloud compression. In Proceedings of the 27th ACM International Conference on Multimedia, pp. 2405–2413, 2019.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. neural information processing systems, 2017.

[49] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[50] P. Verma and J. Berger. Audio transformers:transformer architectures for large scale audio understanding. adieu convolutions. arXiv: Sound, 2021.

[51] G. Vietri, L. V. Rodriguez, W. A. Martinez, S. Lyons, J. Liu, R. Rangaswami, M. Zhao, and G. Narasimhan. Driving cache replacement with ml-based lecar. In A. Goel and N. Talagala, eds., 10th USENIX Workshop on Hot Topics in Storage and File Systems, HotStorage 2018, Boston, MA, USA, July 9-10, 2018. USENIX Association, 2018.

[52] G. Vietri, L. V. Rodriguez, W. A. Martinez, S. Lyons, J. Liu, R. Rangaswami, M. Zhao, and G. Narasimhan. Driving cache replacement with ml-based lecar. usenix annual technical conference, 2018.

[53] P. Visutsak, F. Pensiri, and O. Chaowalit. Smooth voxel surface for medical volumetric rendering. 2019 International Conference on Image and Video Processing, and Artificial Intelligence, 2019.

[54] J. Wang, H. Xu, M. Narasimhan, and X. Wang. Multi-person 3d motion prediction with multi-range transformers. 2022.

[55] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao. Gaze prediction in dynamic 360 immersive videos. In proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5333–5342, 2018.

[56] F. B. Yusuf, V. Stebliankin, G. Vietri, and G. Narasimhan. Cache replacement as a mab with delayed feedback and decaying costs, 2020. doi: 10.48550/ARXIV.2009.11330

[57] E. Zerman, C. Ozcinar, P. Gao, and A. Smolic. Textured mesh vs coloured point cloud: A subjective study for volumetric video compression. In 2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX), pp. 1–6. IEEE, 2020.

[58] A. Zhang, C. Wang, B. Han, and F. Qian. Yuzu: Neural-enhanced volumetric video streaming. In A. Phanishayee and V. Sekar, eds., 19th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2022, Renton, WA, USA, April 4-6, 2022, pp. 137–154. USENIX Association, 2022.

[59] Y. Zheng, Y. Yang, K. Mo, J. Li, T. Yu, Y. Liu, K. Liu, and L. J. Guibas. Gimo: Gaze-informed human motion prediction in context. 2022.

[60] M. Zink, R. Sitaraman, and K. Nahrstedt. Scalable 360° video stream delivery: Challenges, solutions, and opportunities. Proceedings of the IEEE, 2019.