Cross-Frame Resource Allocation with Context-Aware QoE Estimation for 360° Video Streaming in Wireless Virtual Reality

Cheng-Yeh Chen and Hung-Yun Hsieh

Abstract

Wireless virtual reality (VR), aiming to provide an untethered immersive experience through 360° videos, could be facilitated by viewport-guided streaming with the help of viewport prediction. Although many recent viewport predictors can output a series of predictions over upcoming frames, most existing work on video streaming does not fully utilize the capability of these predictors. In this paper, we investigate the problem of 360° video streaming by incorporating the complete series of viewport predictions for maximizing the quality of experience (QoE) through cross-frame resource allocation. To address the problem of viewport prediction errors that could result in erroneous estimation of QoE contribution of tiles in upcoming frames, we develop a novel approach based on contextual multi-armed bandit (CMAB) to "learn" *online* the viewing behavior of the user and the capability of the predictor such that resource can be preferentially allocated to tiles with significant QoE contribution. Further, to address the problem of transmission failures during wireless streaming, we formulate a constrained Markov decision process (CMDP) and apply model predictive control (MPC) to account for resource competition among reactive and proactive transmissions as well as retransmissions of tiles. The performance of the proposed streaming system is evaluated using a real-world VR dataset, state-of-the-art viewport predictors, and realistic mmWave channel models. An improvement of 10.2% in QoE and a reduction of 18.7% in resource waste are achieved across various videos, users, and predictors. Simulation results substantiate that the context-aware QoE learned by the proposed CMAB effectively addresses prediction errors for tiles with different temporal and spatial contexts, and the proposed CMDP can achieve the desired performance even under tight bandwidth constraint and severe channel condition.

Index Terms

Viewport-guided streaming, model predictive control, contextual multi-armed bandit.

I. Introduction

Wireless virtual reality (VR) is gathering attention among various research areas due to its potential to provide an untethered immersive experience and bring the “sense of presence” to the customers through 360° video streaming [1]. To avoid sickness caused by the conflicts between virtual sense and actual movement, the delay between the user action and the video playback, also known as the motion-to-photon (MTP) latency, should be bounded to the order of 10 ms [1], [2]. New-generation cellular technology like millimeter wave (mmWave) thus has been envisioned as an enabler to support the required data rate up to 1Gb/s for delivering high-resolution 360° videos [2]. As mmWave is vulnerable to fading and blockage, however, it is challenging to provide a reliable experience within the desired MTP latency for a user wearing a head-mounted display (HMD) with limited communication and computation power.

Viewport-guided 360° video streaming has thus been adopted in the literature for facilitating wireless virtual reality with cost-effectiveness. A 360° video is usually mapped into equirectangular projection (ERP) and split into smaller rectangles called “tiles” [3], [4]. Only the tiles in the viewport of a user are transmitted such that the required data rate could be drastically decreased since a viewport is typically 20% of the whole panoramic video [5]. Further, with the help of viewport prediction, transmission of tiles is allowed to start “proactively” well before the frame playback time based on the *predicted viewports for upcoming frames*. The latency budget for transmission thus could be substantially increased to the order of 100 ms or even higher [4], allowing more room to accommodate channel failure through retransmissions.

Despite the benefit of viewport prediction, prediction errors lead to deviation between the coverage area of transmitted tiles and the actual viewport. Such deviation results in waste of radio resource (for transmitting unviewed tiles) and potentially lowers user experience (for not being able to transmit tiles in the actual viewport). Recent development in 360° video streaming thus aims to leverage the information provided by viewport predictors while minimizing the impact of prediction errors. For example, extended area surrounding the predicted viewport is transmitted [5], [6] while lower bit rates are allocated to tiles outside the predicted viewport [7], [8] to compensate for prediction errors. The concept of viewing probability of each tile has been proposed to facilitate resource allocation based on some error models [9], [10]. Related work has also proposed to refer to other users’ behavior to mitigate the impact of prediction errors through clustering-based viewport prediction [11] or collaborative viewport prediction [12].

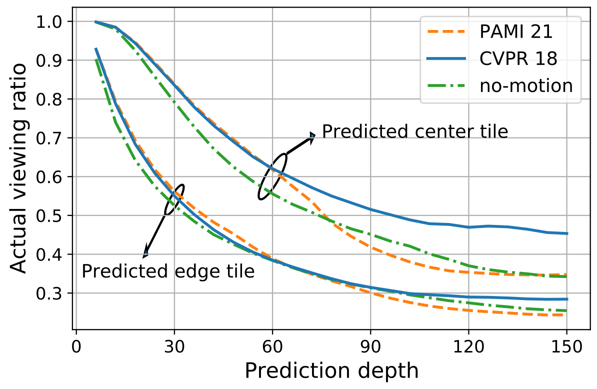


Fig. 1. Actual viewing ratio of predicted center and edge tiles from three viewport predictors [13]–[15] showing that some distant predicted center tiles could have higher viewing ratio than near predicted edge tiles.

A common theme of these research endeavors is to schedule or allocate radio resource to tiles for optimizing streaming performance under viewport prediction error. These streaming systems are typically designed to leverage the *spatial information* of a single viewport (either directly from one predicted viewport or through merging several predicted viewports into one combined viewport [8]) such that *tiles of the same spatial context* (relative position with respect to the predicted viewport) within each frame or group of pictures (GOP) can be prioritized for resource allocation to minimize resource usage. Such an approach, however, fails to fully exploit the benefit of state-of-the-art predictors that can output *a series of viewport predictions* in the "prediction window." A complete series of viewport predictions provides not only *spatial information* but also *temporal information* of tiles in the $360°$ video that are helpful in determining their importance to maximize the viewing experience under channel failure and prediction error.

To motivate how a complete series of viewport predictions could benefit $360°$ video streaming, we use Fig. 1 to show the actual viewing ratio of tiles in the predicted viewport.[1] *Prediction depth* as labeled in the x-axis is the difference of frame indices between the predicted and current frames and it denotes the "futureness" of the predicted viewport. A value of 150, for example, denotes a prediction that is 5s later for a frame rate of 30 fps. Note that a lower viewing ratio of tiles implies a larger prediction error. It can be observed that the actual viewing ratio of a

[1]The actual viewing ratio is measured from state-of-the-art viewport predictors over a real VR dataset [16]. We compute the actual viewing ratio by considering tiles located in the center or edge region of the predicted viewports and counting whether they are actually viewed by users or not. Details on viewport predictors are elaborated in Section VII.

center tile in the predicted viewport is higher than that of an edge tile. In addition, prediction error increases (e.g. view ratio decreases) for frames in the distant future in both center and edge tiles. It is interesting to note that some center tiles in the distant future (say, 2s later or 60 in prediction depth) is more likely to be viewed than some edge tiles in the near future (say, 1s later or 30 in prediction depth).

Such a phenomenon is common to all predictors and it indicates an intriguing yet counter-intuitive insight for proactive scheduling of tile transmissions. Since tiles in the distant future may have higher viewing ratio than tiles in the near future, they should preferentially be allocated resource for performance optimization. Transmission of tiles with low viewing ratio could be delayed until their viewing ratio increases (as they become less "distant" in the future) or until they fall into the actual viewport (reactive scheduling) to avoid transmitting unused tiles. The approach adopted in related work that through *sequential optimization* allocates resource to tiles by considering only their spatial contexts thus fails to address such characteristics of prediction error. To address such a pitfall, therefore, we formulate a *cross-frame optimization* problem for resource allocation[2] that considers both the *spatial and temporal contexts* of individual tiles as provided by state-of-the-art viewport predictors such that the quality of experience (QoE) of a user in viewing the 360° video can be maximized.

Clearly, the QoE for the streamed video is contributed by viewed tiles in each frame and hence resource allocation to tiles should consider the QoE contribution of individual tiles. In estimating the QoE contribution of tiles within the predicted viewports, however, one should take into consideration the impact of prediction error. *Cross-frame optimization is beneficial only if the impact of prediction error on individual tiles is properly accounted for.* Toward this objective, we have observed that QoE contribution of tiles with different spatial contexts (e.g. at the viewport center or edge) as well as temporal contexts (e.g. in the distant or near future) suffers differently from prediction error. Such an observation motivates the proposed design of *context-aware QoE estimation* for cross-frame resource allocation to maximize the QoE of wireless 360° video streaming, and leads to the contribution of this paper as follows:

- To jointly address prediction errors and channel errors, we formulate a constrained Markov decision process (CMDP) for cross-frame resource allocation. We apply *model predictive*

[2]Similar to [17], [18], we assume GOP to be 1 frame to offer higher flexibility for tile selection and satisfy the stringent MTP requirement. Such an assumption is without loss of generality and could be applied to scenarios with GOP longer than 1 frame.

control (MPC) to account for *resource competition* among a series of proactive and reactive transmissions of tiles as well as retransmissions of tiles due to channel failures.

- We propose a QoE model to account for the contribution of individual tiles in a frame of the panoramic video. The proposed QoE model considers the absolute position of a tile in ERP as well as its relative position and viewed area in the viewport. Such a QoE model facilitates the calculation and estimation of the QoE contribution for individual tiles across multiple frames for resource allocation, even under viewport prediction error.
- With the proposed QoE model, we develop a novel contextual multi-armed bandit (CMAB) algorithm to learn the relation between the tile context and the corresponding QoE contribution. We further treat the length of the prediction window as an arm to dynamically adjust the prediction window depending on the prediction quality and channel condition *online* during the 360° video streaming.

To the best of our knowledge, this work is the first to provide a solution for cross-frame resource allocation in 360° video streaming that can leverage full prediction information provided by state-of-the-art viewport predictors *while addressing the artifacts of prediction errors and transmission failures in wireless channel.*

II. Related work

We first explain how viewport prediction is incorporated into the 360° video streaming system in recent literature and why these endeavors differ from the proposed cross-frame optimization. We then explain how related work addresses the problem of prediction error and how they differ from the proposed context-aware error estimation. Finally, we explain the concept of predictive scheduling based on different predictions for different queueing systems and why these research endeavors could not be used directly in the target problem.

A. Incorporating viewport prediction for 360° video streaming

State-of-the-art viewport predictors can provide prediction information in terms of either viewport positions or viewing probabilities of tiles in future frames. Related work has proposed to utilize the predicted viewport position to reduce the bandwidth consumption for video streaming in the single-user [5] and multi-user [6] scenarios. In [7], viewport prediction and bandwidth prediction are both utilized to facilitate rate adaptation, while [9], [11], [19] use the predicted viewing probability of each tile to optimize rate adaptation. In [8], bit rate for each tile is allocated based on the Manhattan distance between the tile and the predicted viewport center. [20] jointly optimizes proactive and reactive streaming based on the current viewport and the predicted viewing probability of each tile. However, these streaming systems have been designed to use only the spatial information from viewport prediction while ignoring the temporal information available from the series of predictions.

To the best of our knowledge, there exists little work that incorporates a series of viewport predictions into tile-based $360°$ streaming. In [21], the authors propose to use the complete series of predictions to rank the transmission priorities of all tiles, where tiles are ranked first by their angles of view to the predicted viewport centers and then by their playback orders. While such a heuristic helps to optimize QoE through rate adaption, it does not consider the impact of prediction error and channel failure on the QoE contribution of individual tiles. We compare in Section VII the performance of [21] against the proposed approach.

B. Addressing viewport prediction error for viewport-guided streaming

Viewport prediction inevitably contains errors and these artifacts should be properly addressed to avoid the impact on achievable QoE. The most common way in literature to tackle viewport prediction error is to model it by certain probability distributions like Gaussian distribution [6], [9] and Laplace distribution [10] and then incorporate the modeled prediction error to derive the variance of the actual viewport position or per-tile viewing probability. Prediction error could also be learned from the past during the streaming. In [21], the number of extra tiles to compensate for prediction error is determined by the moving average of the past prediction error. [5] proposes to estimate prediction error by another predictor using linear regression and neural networks. Furthermore, the uncertainty of prediction could be inferred by other users through the similarity of user behavior, like the clustering-based prediction in [11] or collaborative prediction in [12]. However, these endeavors are not designed to address the difference of prediction uncertainty among various spatial and temporal context aforementioned. A comprehensive consideration on the underlying context information should be incorporated to fully address the series of prediction errors incurred by the series of predictions.

C. Scheduling with a series of predictions for different systems

The concept of predictive scheduling and/or proactive scheduling based on a series of predictions has been investigated in related work that sheds light to the solution proposed in this paper.

The authors in [22] consider a general multi-queue system with a series of arrival predictions and they show how system delay can be reduced under different prediction capabilities in controlled queueing systems. The follow-up work in [23] further investigates how a series of predictions on packet arrivals could improve the system throughput for delay-guaranteed services in general computing and communication systems. In [24], the authors consider a general server providing time-sensitive information that aims to minimize age-of-information given a series of predictions on user's requests for information update. We note that these endeavors are not explicitly designed for video streaming with predictions of user viewports, and hence the formulations and solutions proposed thereof cannot be directly applied in our work. By specifically leveraging the information from state-of-the-art viewport predictors for optimizing the performance of $360°$ video streaming, we are able to identify non-trivial problems and observe interesting phenomena for the target system as we show in the following.

III. System models

We consider a scenario where the head-mounted display (HMD) of a user connects to a base station (BS) through mmWave to stream $360°$ video to the user. The HMD periodically updates its current viewport to the BS, which then forwards the viewport to the viewport predictor at the edge server. As is the case for many state-of-the-art predictors, we assume that the viewport predictor provides a series of predicted viewports in the prediction window according to the latest viewport. These predictions are then sent to the BS to perform resource allocation that can optimize the QoE of the user, while considering prediction errors and transmission failures. In the following, we describe the underlying transmission scheme (III-A) for proactive and reactive streaming and define the channel model (Section III-B) to address channel uncertainty. A new QoE model (Section III-C) is proposed to consider the effect of distortion, position, and completeness for each tile received at the HMD through $360°$ video streaming.

A. Transmission model

The timeline is discrete and labeled by $t \in \{1, 2, \cdots, T\}$ with T being the total number of frames in a video. We assume equirectangular projection and split each frame into $N_x \times N_y$ tiles, where N_x and N_y denote the number of columns and rows respectively such that $\mathcal{N}_x = \{1, 2, \cdots, N_x\}$ and $\mathcal{N}_y = \{1, 2, \cdots, N_y\}$. We index each tile in each frame by a triple (i, j, t) with $i \in \mathcal{N}_x$, $j \in \mathcal{N}_y$, and $t \in \{1, 2, \cdots, T\}$ to denote the (i, j)-th tile in the t-th frame.

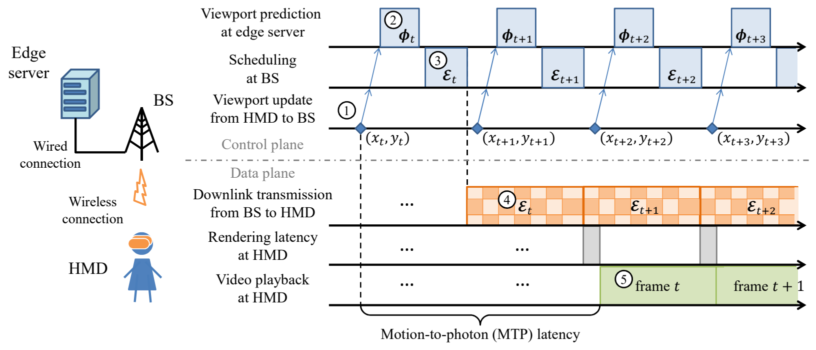


Fig. 2. Transmission scenario and timeline for 360° video streaming to enable wireless VR. 1) The HMD updates the current viewport center (x_t, y_t) to the BS and the BS forwards (x_t, y_t) to the edge server. 2) The edge server provides viewport prediction $\boldsymbol{\phi}_t$. 3) The BS schedules resource allocation $\boldsymbol{\mathcal{E}}_t$ based on $\boldsymbol{\phi}_t$. 4) The BS transmits required tiles according to $\boldsymbol{\mathcal{E}}_t$ in the downlink. 5) The HMD renders the received tiles for video playback.

The underlying transmission timeline is elaborated in Fig. 2. Before the playback of frame t, the HMD first updates the user's current viewport center (x_t, y_t) to the BS. The BS forwards (x_t, y_t) to the edge server to obtain prediction information $\boldsymbol{\phi}_t = \{(\tilde{x}_t(t'), \tilde{y}_t(t')) \,|\, \forall t' \in \{t+1, \cdots, t+W_t\}\}$, where $(\tilde{x}_t(t'), \tilde{y}_t(t'))$ denotes the predicted viewport center for a future frame t' and W_t represents the prediction window for current frame t. According to $\boldsymbol{\phi}_t$, the BS performs resource allocation $\boldsymbol{\mathcal{E}}_t = \{e_t(i,j,t') \,|\, \forall i \in \mathcal{N}_x, j \in \mathcal{N}_y, t' \in \{t, \cdots, t+W_t\}\}$, where $e_t(i,j,t')$ is the amount of radio resource allocated to transmit tile (i,j,t'). The BS could allocate resource reactively for tiles in frame $t' = t$ according to current viewport center (x_t, y_t) or proactively for tiles in frame $t' \in \{t+1, \cdots, t+W_t\}$ according to $\boldsymbol{\phi}_t$. Hence, the resource allocation $\boldsymbol{\mathcal{E}}_t$ may include transmission of tiles in frame $t' \in \{t, \cdots, t+W_t\}$.

B. Channel model

Channel uncertainty is considered in this work since failure exists in transmission, especially for mmWave communication. We model the channel dynamics by a Markov chain with K channel states in a state space $\mathcal{S} = \{s_1, \cdots, s_K\}$ and we use $P(s, s'), \forall s, s' \in \mathcal{S}$ to represent the transition probability from s to s'. We denote S_t as the channel state for the downlink transmission specified in $\boldsymbol{\mathcal{E}}_t$ as Fig. 2 shows. To address channel failure, let $\zeta\big(s, e_t(i,j,t')\big)$ be the probability of successful transmission jointly decided by s and $e_t(i,j,t')$.[3] We further

[3]Formulation of $\zeta(\,\cdot\,)$ depends on the given channel model. Section VII-A shows the case for the mmWave channel model.

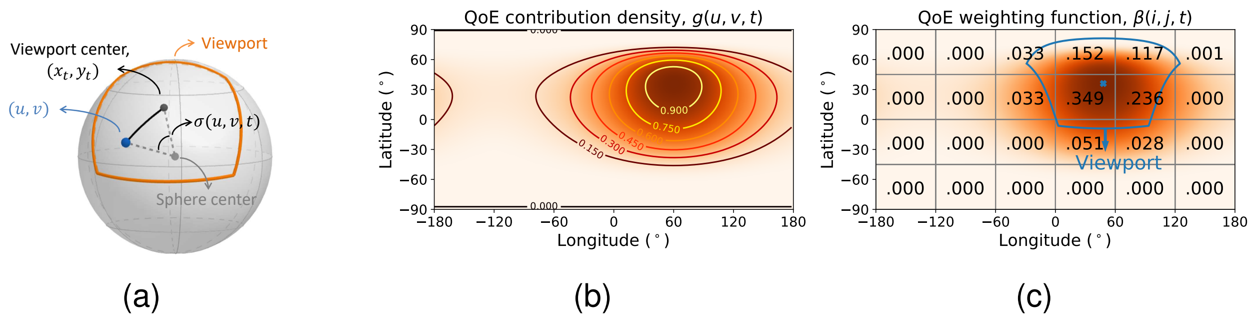


Fig. 3. Visualization of the proposed QoE metrics. (a) $\sigma(u,v,t)$ denotes the angle of view from the viewport center (x_t, y_t) to any point (u,v) on the viewing sphere. (b) QoE contribution $g(u,v,t)$ given the viewpoint $(x_t, y_t) = (48^\circ, 36^\circ)$. (c) Visualization of the value of QoE weighting function $\beta(i,j,t)$ for each tile.

use a Bernoulli random variable $b_t(i,j,t') \in \{0,1\}$ with $\mathbb{E}\left[b_t(i,j,t')\right] = \zeta\big(s, e_t(i,j,t')\big)$ to denote whether the transmission of tile (i,j,t') in $\boldsymbol{\mathcal{E}}_t$ is successful or not. Let tile availability $a_t(i,j,t') \in \{0,1\}$ denote whether tile (i,j,t') is available at the HMD after the transmission of $\boldsymbol{\mathcal{E}}_t$. By initializing $a_0(i,j,t') = 0$, $\forall i \in \mathcal{N}_x, j \in \mathcal{N}_y, t' \in \{1, \cdots, T\}$, we have

$$a_t(i,j,t') = \begin{cases} 1, & \text{if } b_t(i,j,t') = 1, \\ a_{t-1}(i,j,t'), & \text{otherwise.} \end{cases} \tag{1}$$

C. QoE model

Related work has proposed models to quantify the QoE for 360° video by considering factors such as projection distortion (WS-PSNR) [25] and visual attention (VASW-PSNR) [26]. These QoE models, however, have been proposed to measure the impact of rate adaptation that could potentially result in reconstruction errors of the streamed video. To focus on the achievable QoE as the result of different tile scheduling decisions, we propose a QoE weighting model that addresses compound effects unique to tiles in 360° video as follows:

1) Position in the viewport: A missing center tile affects QoE more severely than a missing edge tile does. Therefore, a tile's relative position in the viewport should be considered to account for its QoE contribution. We measure the position of a given point by its *angle of view* from the viewport center. As shown in Fig. 3a, let (u,v) be a point in ERP, where $-\pi \le u \le \pi$ is the longitude and $-\frac{\pi}{2} \le v \le \frac{\pi}{2}$ is the latitude. Let $\sigma(u,v,t)$ be the angle of view from a given point (u,v) to the viewport center (x_t, y_t) for frame t as follows:

$$\sigma(u,v,t) = \cos^{-1}\left(\cos(u - x_t)\cos v \cos y_t + \sin v \sin y_t\right). \tag{2}$$

The QoE contribution of a point (u, v) is scaled by $\sigma(u, v, t)$ such that the tile near the viewport center should have higher QoE contribution than a tile far from the viewport center.

2) Distortion in the projection: Since a tile with higher latitude occupies less area in a 360° video due to ERP, we model such distortion to better measure the actual QoE of a user. Inspired by [25], we define a distortion function $\rho(v) = \cos v$ as a function of latitude v to capture the distortion of the actual area viewed by a user. Note that $\rho(v) \to 1$ for $v \to 0$ (no distortion in low latitude region) while $\rho(v) \to 0$ for $v \to \pm\pi/2$ (severe distortion in high latitude region). Combining $\sigma(u, v, t)$ and $\rho(v)$, we define the QoE contributed by point (u, v) in frame t as

$$g(u, v, t) = \exp\left(-\frac{\sigma^2(u, v, t)}{\rho^2(v)}\right) \tag{3}$$

to jointly address the effect of position and distortion. Eq. (3) resembles a 2D Gaussian function with its highest value centered at the actual viewport center (x_t, y_t) and shaped by $\rho^2(v)$. Notice that we adopt the 2D Gaussian function to account for the QoE contribution due to its commonness in computer vision to shape a given quantity like saliency [27], head movement map [28], or the ground-truth heatmap for object detection [29]. Fig. 3b shows the contour of $g(u, v, t)$, where the QoE contribution of (u, v) decreases when the distance from (x_t, y_t) increases and shrinks even faster for higher absolute value of latitude $|v|$.

3) Completeness in the viewport: To account for the effect that a tile might be viewed partially, let $V(i, j, t)$ be the viewed region of a tile (i, j, t). An example is shown in Fig. 3c, where there are 9 tiles with non-empty $V(i, j, t)$. With $V(\cdot)$, the proposed QoE weighting function $\beta(i, j, t)$ is formulated by integrating $g(\cdot)$ over $V(\cdot)$ to obtain the QoE contribution from tile (i, j, t) and normalizing it by the overall contribution as follows:

$$\beta(i, j, t) = \frac{\iint_{V(i,j,t)} g(u, v, t)\, du\, dv}{\sum_{i' \in \mathcal{N}_x, j' \in \mathcal{N}_y} \iint_{V(i',j',t)} g(u, v, t)\, du\, dv}, \tag{4}$$

which has a value in $[0, 1]$ for each tile as labeled in Fig. 3c.

The overall QoE Q_t for frame t is contributed by all of its received tiles as follows:

$$Q_t = \sum_{i \in \mathcal{N}_x} \sum_{j \in \mathcal{N}_y} \beta(i, j, t) a_t(i, j, t), \tag{5}$$

where $a_t(i, j, t)$ is tile availability defined in (1). Notice that if all tiles in the actual viewport are received at the HMD, Q_t reaches its maximum value 1 under the proposed QoE model. As mentioned before, while we do not consider the PSNR (Peak Signal-to-Noise Ratio) of each tile in (5) to keep the paper focused, it can also be incorporated in our model to account for reconstruction errors if rate adaptation were to be performed during scheduling.

IV. Problem formulation

A. Cross-frame resource allocation

Since the viewport predictor could reveal prediction information in several future frames, we propose to fully optimize the decision of each frame within the prediction window by model predictive control (MPC) [24], [30]. The main idea is to optimize resource allocation for all future frames within the prediction window $\boldsymbol{\mathcal{E}}_{t'}, \forall t' \in \{t, \cdots, t+W_t\}$ but commit only the first allocation $\boldsymbol{\mathcal{E}}_t$ for frame t. Such approach allows resource allocation to take the impact of channel conditions and viewport prediction errors across multiple frames into consideration. In this way, resource needed for tile retransmissions could also be accounted for and resource competition between reactive and proactive transmission can be jointly optimized.

To determine the resource allocation for frame t based on MPC, we formulate the QoE maximization through cross-frame resource allocation for tiles in frame $t' \in \{t, \cdots, t+W_t\}$:

$$\max_{\boldsymbol{\mathcal{E}}_{t'}, \forall t' \in \{t, \cdots, t+W_t\}} \quad \mathbb{E}\left\{ Q_t + \sum_{t'=t+1}^{t+W_t} \tilde{Q}_{t'} \right\} \tag{6}$$

$$\text{s.t.} \quad \mathbb{E}\left\{E_{t'}\right\} = \mathbb{E}\left\{ \sum_{i \in \mathcal{N}_x} \sum_{j \in \mathcal{N}_y} \sum_{t''=t'}^{t+W_t} e_{t'}(i,j,t'') \right\} \leq B, \forall t' \in \{t, \cdots, t+W_t\}, \tag{7}$$

where $E_{t'}$ is the total radio resource consumed for $\boldsymbol{\mathcal{E}}_{t'}$ and B is the total amount of radio resource. In (6), the objective is to maximize the sum of QoEs for all frames within the prediction window $t' \in \{t, \cdots, t+W_t\}$ through the determination of $\boldsymbol{\mathcal{E}}_{t'}$, where Q_t represents the actual QoE for current frame and $\sum_{t'=t+1}^{t+W_t} \tilde{Q}_{t'}$ represents the estimated QoE for future frames in the prediction window. The estimated QoE $\tilde{Q}_{t'}$ for frame $t' \in \{t+1, \cdots, t+W_t\}$ can be expressed as follows:

$$\tilde{Q}_{t'} = \sum_{i \in \mathcal{N}_x} \sum_{j \in \mathcal{N}_y} \tilde{\beta}(i,j,t') a_{t'}(i,j,t'), \tag{8}$$

where $a_{t'}(i,j,t')$ is availability of tile (i,j,t') at its playback frame t' and $\tilde{\beta}(i,j,t')$ is the estimated QoE contribution (weighting function) of tile (i,j,t') based on prediction information $\boldsymbol{\phi}_t$ that would be elaborated in Section IV-B. Note that Q_t indicates the objective of *reactive transmission* and $\sum_{t'=t+1}^{t+W_t} \tilde{Q}_{t'}$ indicates the objective of *proactive transmission*, which are jointly optimized under our problem formulation.

Eqs. (6) and (7) are represented in the expectation form due to the uncertainty in channel condition as modeled by the Markov chain described in Section III-B. Specifically, the BS should determine resource $e_{t'}(i,j,t'')$ allocated for each tile (i,j,t'') within the prediction window. The value of $e_{t'}(i,j,t'')$ influences the probability of successful transmission $\zeta\big(s, e_{t'}(i,j,t'')\big)$ and the dynamics of tile availability $a_{t'}(i,j,t'')$ used in (8). We present how we solve such an MPC-driven cross-frame resource allocation problem by constrained Markov decision process (CMDP) in Section VI.

B. Context-aware QoE estimation

With the prediction information $\boldsymbol{\phi}_t$ at frame t, one can estimate the QoE contribution of tile (i,j,t') by applying (4) to calculate $\tilde{\beta}(i,j,t')$. The problem, however, is that the prediction information such as the predicted viewport and its center is subject to prediction error. Such prediction error could result in wrong QoE estimation and the optimization of cross-frame resource allocation. To cope with such a problem, in this paper we propose to estimate QoE for future frames t' in the prediction window through *context-aware online learning* and use (4) only for the current frame t.

As we have mentioned in Section I, the impact of prediction error on QoE depends on the *context of the tile*, including its position in the predicted viewport (spatial context) and the "futureness" of its frame in the prediction window (temporal context). Specifically, we propose to learn the QoE contribution $\tilde{\beta}(i,j,t')$ of a future tile (i,j,t') in the prediction window based on three contexts, (i) prediction depth $t'-t$, (ii) row index j, and (iii) angle of view $\tilde{\sigma}_t(i,j,t')$ from the tile center (u_i, v_j) to the predicted viewport center $(\tilde{x}_t(t'), \tilde{y}_t(t'))$, as follows:

$$\tilde{\beta}(i,j,t') = \Theta\Big(t'-t, j, \tilde{\sigma}_t(i,j,t')\Big), \tag{9}$$

where Θ is a mapping function to be learned. Note that the prediction depth $t'-t$ indicates whether the frame is in near or distant future and it helps to profile the extent of prediction uncertainty. The row index j represents the latitude of the tile in ERP and it indicates the amount of distortion and the actual area viewed by a user. The angle of view $\tilde{\sigma}_t(i,j,t')$ can be calculated through proper modification of (2) as follows: $\tilde{\sigma}_t(i,j,t') = \cos^{-1}\big(\cos(u_i - \tilde{x}_t(t'))\cos v_j \cos \tilde{y}_t(t') + \sin v_j \sin \tilde{y}_t(t')\big)$. It denotes whether the tile is near or far from the predicted center and it is informative to reveal the impact of prediction error on QoE.

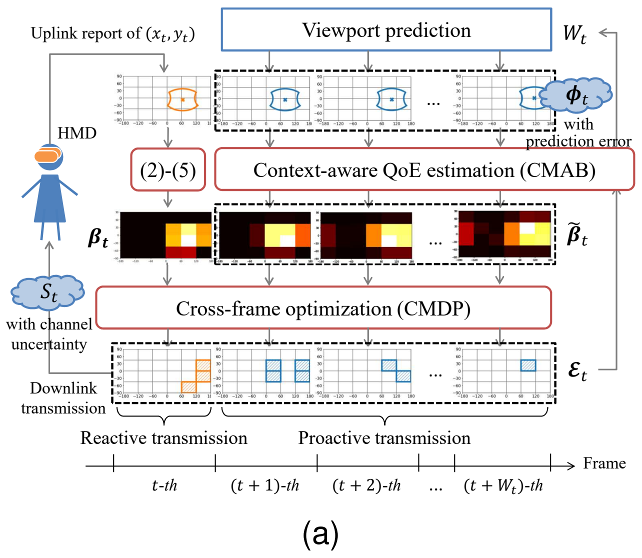
Notice that Θ in (9) depends on the characteristics of different viewport predictors as well as the users and/or videos these predictors operate on. Instead of pre-training or assuming Θ based on some mathematical model, we believe that online learning based on contextual multi-armed bandit (CMAB) is more suitable since it can *automatically adapt to the user and/or viewport* *predictor involved during the streaming of the video.* Another benefit of applying CMAB in the target problem is the determination of the prediction window W_t that can adapt to the channel condition and prediction quality. Improper length of the prediction window could undermine the system performance since longer prediction window may incur more prediction errors, while shorter prediction window may lower tolerance to channel variation. We explain in Section V how we use CMAB to jointly perform QoE estimation and prediction window adjustment.

C. System overview

Fig. 4a provides an overview of the proposed system that involves context-aware QoE estimation and cross-frame optimization for 360° video streaming. For each frame t, the BS first obtains the actual viewport position (x_t, y_t) from the user's HMD and the viewport prediction $\boldsymbol{\phi}_t$ from the viewport predictor at the edge server. The actual QoE $\boldsymbol{\beta}_t = \{\beta(i,j,t) \,|\, \forall i \in \mathcal{N}_x, j \in \mathcal{N}_y\}$ for all tiles in the current frame is computed from (x_t, y_t), while the QoE's $\tilde{\boldsymbol{\beta}}_t = \{\tilde{\beta}(i,j,t') \,|\, \forall i \in \mathcal{N}_x, j \in \mathcal{N}_y, t' \in \{t+1, \cdots, t+W_t\}\}$ for future frames are obtained through the proposed CMAB algorithm. With $\boldsymbol{\beta}_t$ and $\tilde{\boldsymbol{\beta}}_t$, the BS solves the proposed CMDP in (6) and (7) to output resource allocation $\boldsymbol{\mathcal{E}}_t$ that includes resource used for reactive transmission as well as proactive transmission of tiles in the actual and predicted viewports respectively. Based on $\boldsymbol{\mathcal{E}}_t$, a proper length of the prediction window W_{t+1} for the next frame is decided by the proposed CMAB.

V. Context-aware QoE estimation with dynamic window adjustment

As described in Section IV-B, we assume tiles with same spatial and temporal contexts have same QoE contribution. Hence, we could keep records for QoE's in different contexts as computed through (4) and then use the recorded QoE to estimate QoE for future tiles in corresponding contexts during the streaming of the video. To proceed, we construct a hypercube structure to partition the context space and formulate a contextual multi-armed bandit (CMAB) problem. We treat the length of the prediction window as an arm to allow the system to determine a length of the prediction window. Through exploration and exploitation in CMAB, the relation between context information and QoE estimation, e.g. Θ in (9), is learned during the exploration phase, and a proper length of the prediction window for viewport prediction is determined during the exploitation phase. In this way, estimation of QoE for future frames and dynamic adjustment of the prediction window can be jointly determined in the proposed CMAB algorithm.



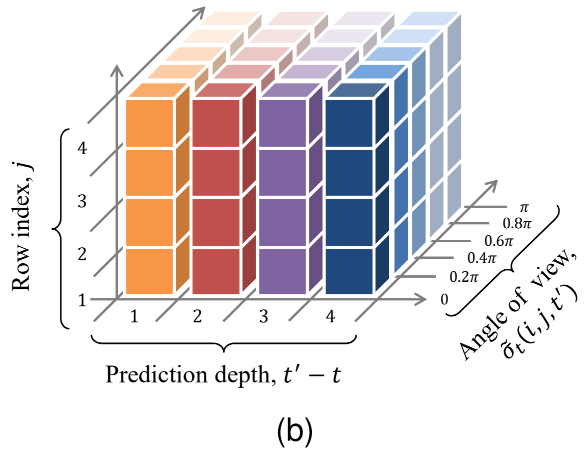

Row index, j
Prediction depth, $t' - t$
Angle of view, $\tilde{\sigma}_t(i,j,t')$
0
0.2π
0.4π
0.6π
0.8π
π
(b)

Fig. 4. (a) System overview of the proposed cross-frame optimization with context-aware QoE. This example shows the procedure in frame t aiming to optimize resource allocation for frame t to $t + W_t$. (b) An example of hypercube containing three dimensions of information with $W_t = 4$.

In Algorithm 1, a hypercube structure is first initialized to partition the context space. For frame t, the BS obtains the actual viewport from the HMD (line 4) and computes the per-tile QoE in the current frame to update QoE estimation for the corresponding contexts and hypercubes (line 5 to 10). The BS then collects the latest prediction information from the viewport predictor at the edge server (line 11) and looks up the QoE estimation for future tiles in the corresponding hypercubes (line 12 to 15). The acquired QoE is used to facilitate the cross-frame optimization elaborated in Section VI (line 16). Finally, the algorithm determines the length of the prediction window for the next frame through exploration or exploitation of CMAB (line 17 to 20).

A. Hypercube structure for context-aware QoE

As mentioned in (9), the proposed "context" for QoE estimation has 3 dimensions, including the prediction depth $t' - t$ of the frame, the row index j of the tile, and the angle of view $\tilde{\sigma}_t(i, j, t')$ from the tile center to the predicted viewport center. Let $\mathcal{X} \in \mathcal{R}^D$ be a context space with D dimensions. We split the d-th dimension of $\mathcal{X}$ into h_d identical partitions such that the context space consists of $\prod_{d=1}^{d=D} h_d$ hypercubes. Let $\mathcal{P}_\mathcal{X}$ be the hypercube structure thus formed, where each hypercube $p \in \mathcal{P}_\mathcal{X}$ corresponds to a particular context in the context space. Fig. 4b illustrates a hypercube structure with 80 hypercubes for the case of $W_t = 4$.

Algorithm 1 CMAB for context-aware QoE optimization

```
1:  Input: $T$, $B$, $W_{\max}$, $\mathcal{X}$, $\mathcal{P}_{\mathcal{X}}$, $K_t$.
2:  Initialization: $\mathcal{A}_p = \emptyset, \forall p \in \mathcal{P}_{\mathcal{X}}$; $W_1 = W_{\max}$.
3:  for $t \in \{1, 2, \cdots, T\}$ do
4:      Obtain the actual viewport $(x_t, y_t)$ from the HMD.
5:      for $i \in \mathcal{N}_x, j \in \mathcal{N}_y$ do                    ▷ Hypercube update
6:          for $w \in \{1, \cdots, W_{\max}\}$ do
7:              if $t - w > 0$ and $W_{t-w} \geq w$ then
8:                  $p \leftarrow f_{\mathcal{X}}\big(w, j, \tilde{\sigma}_{t-w}(i,j,t)\big)$.
9:                  $\tilde{\beta}_p \leftarrow \big(\tilde{\beta}_p|\mathcal{A}_p| + \beta(i,j,t)\big)/\big(|\mathcal{A}_p| + 1\big)$.
10:                 $\mathcal{A}_p \leftarrow \mathcal{A}_p \cup \{(i,j,t)\}$.
11:     Obtain predicted viewport center $(\tilde{x}_t(t'), \tilde{y}_t(t'))$ in prediction window $W_t$ from the
        viewport predictor.
12:     for $i \in \mathcal{N}_x, j \in \mathcal{N}_y$ do                    ▷ QoE lookup
13:         for $t' \in \{t+1, \cdots, t+W_t\}$ do
14:             $p \leftarrow f_{\mathcal{X}}\big(t'-t, j, \tilde{\sigma}_t(i,j,t')\big)$.
15:             $\tilde{\beta}(i,j,t') \leftarrow \tilde{\beta}_p$.
16:     Solve (6) using $\boldsymbol{\beta}_t$, $\tilde{\boldsymbol{\beta}}_t$ and $W_t$ by Algorithm 2 to obtain $\boldsymbol{\mathcal{E}}_t$ and allocate resource
        accordingly.
17:     if $\exists\ p \in \mathcal{P}_{\mathcal{X}}$ s.t. $|\mathcal{A}_p| \leq K_t$ then                    ▷ Exploration
18:         $W_{t+1} \leftarrow W_{t+1}^{\text{explore}}$ defined in (11).
19:     else                    ▷ Exploitation
20:         $W_{t+1} \leftarrow W_{t+1}^{\text{exploit}}$ defined in (12).
```

B. Estimation on context-aware QoE

Our goal is to learn QoE contribution of tiles in each hypercube and utilize the learned value to estimate QoE for future tiles. Two procedures as follows are involved:

1) Hypercube update (QoE learning): After the update of the actual viewport center (x_t, y_t) (line 4 in Algorithm 1), the actual QoE contribution $\beta(i,j,t)$ of every tile (i,j,t) in frame t, $\forall i \in \mathcal{N}_x, j \in \mathcal{N}_y$, can be calculated based on (4). For each such tile (i,j,t), we loop back for the past $w \in \{1, \cdots, W_{\max}\}$ frames to see if there is past prediction covering the current tile (line 7). A function $f_{\mathcal{X}}\big(t'-t, j, \tilde{\sigma}_t(i,j,t')\big)$ is then invoked to find the hypercube p corresponding to the context of the tile (line 8), where its QoE $\tilde{\beta}_p$ is updated with the actual QoE $\beta(i,j,t)$ and its tile set $\mathcal{A}_p$ is updated to include tile (i,j,t) (line 9 to 10). Note that the original QoE $\tilde{\beta}_p$ of hypercube p before the update has been obtained by averaging the actual QoE's of all transmitted tiles in $\mathcal{A}_p$ as follows: $\tilde{\beta}_p = \sum_{(i,j,t)\in\mathcal{A}_p} \beta(i,j,t)/|\mathcal{A}_p|$. It is clear that as more predictions are given and more actual viewports are reported from the user, the size of $\mathcal{A}_p$ would grow for all $p \in \mathcal{P}_{\mathcal{X}}$ and $\tilde{\beta}_p$ would converge as we show in Section V-D.

2) QoE lookup: For all future tiles (i,j,t'), $\forall i \in \mathcal{N}_x, j \in \mathcal{N}_y, t' \in \{t+1,\cdots,t+W_t\}$ in the prediction window W_t, we first calculate their angles of view $\tilde{\sigma}_t(i,j,t')$ against the predicted viewport information (line 11) and then look up the corresponding hypercube p (line 14). The context-aware QoE $\tilde{\beta}_p$ maintained at hypercube p is then used as the estimated QoE contribution for the tile $\tilde{\beta}(i,j,t')$ (line 15) without relying on the calculation of (4) for future tiles.

C. Adaptation of the prediction window length

If the number of tiles $|\mathcal{A}_p|$ maintained at any hypercube p is less than a certain threshold, such a hypercube is called *under-explored* since the QoE estimation of the corresponding context may be inaccurate. To avoid this, the system needs to guarantee a minimum number of QoE updates for each hypercube. Note that in our context space, a longer prediction window would result in more predicted viewports and hence a larger set of hypercubes and tiles. However, longer prediction window would incur higher computation complexity for solving the cross-frame optimization in Section VI. Therefore, the CMAB algorithm needs to determine a proper length of the prediction window through trade-off between exploration and exploitation.

To proceed, define K_t as a control function monotonically increasing in t. A proper choice of K_t is important for the convergence of CMAB as we show in Section V-D. With K_t as a threshold for $|\mathcal{A}_p|$, we update the length of the prediction window as follows:

$$W_{t+1} = \begin{cases} W_{t+1}^{\text{explore}}, & \exists\ p \in \mathcal{P}_{\mathcal{X}} \text{ s.t. } |\mathcal{A}_p| \le K_t, \\ W_{t+1}^{\text{exploit}}, & \text{otherwise.} \end{cases} \tag{10}$$

That is, if there exists an under-explored hypercube p such that $|\mathcal{A}_p| \le K_t$, the algorithm is in the explore phase; otherwise, the algorithm is in the exploit phase. By (10), the length of the prediction window can be automatically adjusted as needed.

1) Exploration: The value of W_{t+1}^{explore} is determined by

$$W_{t+1}^{\text{explore}} = \max_{p \in \mathcal{P}_{\mathcal{X}}, |\mathcal{A}_p| \le K_t} (t'-t)_p, \tag{11}$$

where $(t'-t)_p$ denotes the prediction depth corresponding to hypercube p with $|\mathcal{A}_p| \le K_t$. Eq. (11) guarantees all under-explored hypercubes would be covered in the next prediction task.

2) Exploitation: The value of W_{t+1}^{exploit} is determined by

$$W_{t+1}^{\text{exploit}} = \min\left\{ \max_{(i,j,t') \in \boldsymbol{\mathcal{E}}_t, e_t(i,j,t')>0} (t'-t+1), W_{\max} \right\}. \tag{12}$$

The rationale is to set W_{t+1}^{exploit} to one plus the largest prediction depth of tiles that have been allocated resource in $\boldsymbol{\mathcal{E}}_t$. If recent channel condition is good and/or prediction error is low, the largest prediction depth of tiles allocated in $\boldsymbol{\mathcal{E}}_t$ could grow larger since resource used to transmit lost tiles (due to channel failure) or unviewed tiles (due to prediction error) is low. Otherwise, the largest prediction depth of tiles allocated in $\boldsymbol{\mathcal{E}}_t$ might shrink since resource is wasted due to lost or unviewed tiles, and there might be no sufficient resource left for proactive streaming.

D. Regret analysis with delayed update

An online learning algorithm may incur regret to the solution since the learned estimation may be inaccurate, leading to sub-optimal actions. The *regret of learning* in this paper can be defined as $R_T = \sum_{t=1}^{T} \mathbb{E}\left\{Q_t^* - Q_t\right\}$, where Q_t^* is the optimal QoE obtained by the oracle algorithm and Q_t is the QoE obtained by the proposed CMAB learning algorithm. It is necessary to prove that *the regret bound is sub-linear in time* to validate the convergence of an online algorithm.

Unlike other algorithms based on CMAB, we should consider the effect of *delayed update* in the proposed algorithm. The reason is that although the viewport predictor at frame t could provide prediction up to the next $W_{\max}$ frames, hypercubes with tile context related to $t' - t > 1$ cannot be updated immediately. They have to wait until $t = t'$, indicating that the update for these hypercubes are delayed. Such delay may incur mis-exploitation and impact the convergence of R_T. The sub-linearity of the proposed algorithm with delayed update is stated as follows:

Theorem 1. *(Regret bound with delayed update) Let* $K_t = t^{\frac{2\alpha}{3\alpha+D}} \log(t)$ *and* $h_d \leq \lceil T^{\frac{1}{3\alpha+D}} \rceil$, $\forall d = 1, \cdots, D$. *If there exists* $L > 0$ *and* $\alpha > 0$ *such that* $|\beta(\phi) - \beta(\phi')| \leq L\|\phi - \phi'\|^{\alpha}$ *for any two contexts* ϕ *and* ϕ'*, where* $\beta(\cdot)$ *is a function mapping each context to its QoE, then the complexity of* R_T *belongs to* $O\left(\left((W_{\max} - 1)T^{\frac{2\alpha}{3\alpha+D}} + T^{\frac{2\alpha+D}{3\alpha+D}}\right)\log(T)\right)$.

Proof. Reference [31] has a proof for a similar problem, where the numbers of hypercubes in all dimensions of the context space are the same. In this paper, the numbers of hypercubes in each dimension d is set to h_d in consideration of the heterogeneity of the context involved. Yet with some minor modifications we can prove the same for the proposed algorithm, details of which are omitted here for lack of space. □

Hence, the complexity of the leading order of R_T is sub-linear and we confirm that even with delayed update, the proposed CMAB algorithm is able to converge sub-linearly.

VI. Cross-frame optimization with predictive scheduling

The cross-frame optimization of QoE as defined in (6) with constraint (7) could be treated as a constrained Markov decision process (CMDP) with finite horizon. Related work [23] solved a similar problem, where a series of prediction information is available to facilitate the resource allocation. However, the prediction model in [23] could only accommodate uniform prediction error for each step in the prediction window. In this paper, we generalize the framework to accommodate general prediction error over various prediction depth by the incorporation of QoE estimation. Note that while a closed-form policy can be derived in [23], our generalization complicates the problem formulation that requires a different solution approach. To proceed, we apply the subgradient method on Lagrangian multipliers, solve the backward induction on the defined Bellman equations, and compute the expected resource consumption by forward induction to obtain an MPC-based resource allocation.

A. Tile-level decomposition

We first decompose (6) with constraint (7) into a *per-tile optimization* problem. Define $\boldsymbol{\lambda} = \{\lambda_t, \cdots, \lambda_{t+W_t}\}$ as the Lagrangian multipliers for the cost of resource violation. The Lagrangian of (6) with the constraint (7) is

$$\begin{aligned}\mathcal{L}_{\boldsymbol{\lambda}} =& \mathbb{E}\left\{Q_t + \sum_{t'=t+1}^{t+W_t} \tilde{Q}_{t'}\right\} - \sum_{t'=t}^{t+W_t} \lambda_{t'}\Big(\mathbb{E}\{E_{t'}\} - B\Big) = \sum_{i,j}\Bigg(\beta(i,j,t)\mathbb{E}\left\{a_t(i,j,t)\right\} + \\ & \sum_{t'=t+1}^{t+W_t} \tilde{\beta}(i,j,t')\mathbb{E}\left\{a_{t'}(i,j,t')\right\} - \sum_{t'=t}^{t+W_t} \lambda_{t'}\mathbb{E}\left\{\sum_{t''=t'}^{t+W_t} e_{t'}(i,j,t'')\right\}\Bigg) + \sum_{t'=t}^{t+W_t} \lambda_{t'}B.\end{aligned} \tag{13}$$

By using the equality $\sum_{t'=t}^{t+W_t}\sum_{t''=t'}^{t+W_t} e_{t'}(i,j,t'') = \sum_{t'=t}^{t+W_t}\sum_{t''=t}^{t'} e_{t''}(i,j,t')$ and defining *per-tile Lagrangian* $\mathcal{L}_{\boldsymbol{\lambda}}(i,j,t')$ as

$$\mathcal{L}_{\boldsymbol{\lambda}}(i,j,t') = \begin{cases} \beta(i,j,t')\mathbb{E}\left\{a_{t'}(i,j,t')\right\} - \sum\limits_{t''=t}^{t'} \lambda_{t''}\mathbb{E}\left\{e_{t''}(i,j,t')\right\}, \forall t' = t, \\ \tilde{\beta}(i,j,t')\mathbb{E}\left\{a_{t'}(i,j,t')\right\} - \sum\limits_{t''=t}^{t'} \lambda_{t''}\mathbb{E}\left\{e_{t''}(i,j,t')\right\}, \forall t' \in \{t+1, \cdots, t+W_t\}, \end{cases} \tag{14}$$

we can decompose the maximization of the original Lagrangian $\mathcal{L}_{\boldsymbol{\lambda}}$ into the maximization of per-tile Lagrangian $\mathcal{L}_{\boldsymbol{\lambda}}(i,j,t')$ as follows:

$$\arg\max_{\boldsymbol{\mathcal{E}},\boldsymbol{\lambda}} \mathcal{L}_{\boldsymbol{\lambda}} = \arg\max_{\boldsymbol{\mathcal{E}},\boldsymbol{\lambda}} \sum_{i,j}\sum_{t'=t}^{t+W_t} \mathcal{L}_{\boldsymbol{\lambda}}(i,j,t') + \sum_{t'=t}^{t+W_t}\lambda_{t'}B = \arg\max_{\boldsymbol{\mathcal{E}},\boldsymbol{\lambda}} \sum_{i,j}\sum_{t'=t}^{t+W_t} \mathcal{L}_{\boldsymbol{\lambda}}(i,j,t'). \tag{15}$$

Since resource allocation $\boldsymbol{\mathcal{E}} = \{\boldsymbol{\mathcal{E}}_t, \cdots, \boldsymbol{\mathcal{E}}_{t+W_t}\}$ is irrelevant to $\lambda_{t'}$ and B, maximizing $\mathcal{L}_{\boldsymbol{\lambda}}$ is equivalent to maximizing $\mathcal{L}_{\boldsymbol{\lambda}}(i,j,t')$ tile by tile.

B. Per-tile Bellman equations

To solve the per-tile optimization problem derived in (15), we first define the Bellman equations for each tile (i,j,t') in the prediction window, $t' \in \{t, \cdots, t+W_t\}$, under different system states. We design the system state to encompass (i) channel condition $S_t \in \mathcal{S}$ to take channel uncertainty into consideration for resource allocation $\boldsymbol{\mathcal{E}}_t$ and (ii) allocation time $t'' \in \{t, \cdots, t'\}$ to indicate the time in the prediction window to schedule transmission. Note that tile (i,j,t') should be received at the HMD at or before frame t' for playback, and hence $t' - t''$ can be considered as the *delay budget* left for the tile before it times out. Allocation time t'' is crucial to our problem in dealing with retransmissions as well as scheduling of reactive and proactive transmissions. For simplicity, in the following we use (s, t'') to denote the aforementioned system state.

We design the value function $V(s, t'' \,|\, i,j,t')$ to denote the payoff of resource allocation for tile (i,j,t') under state (s,t'') based on the per-tile Lagrangian in (14). The value function should be able to consider retransmission caused by channel failure and resource competition between reactive and proactive transmission. We first define the value function for reactive transmission ($t'' = t' = t$). Since any failure in reactive transmission results in the outdated tile, the value function could be given by the product of the QoE contribution $\beta(\cdot)$ and the probability of successful transmission $\zeta(\cdot)$ while considering the cost of allocated resource $e_{t''}(\cdot)$:

$$V(s,t'' \,|\, i,j,t') = \max_{e_{t''}(i,j,t')} \Big\{ \zeta(s, e_{t''}(i,j,t'))\beta(i,j,t') - \lambda_{t''} e_{t''}(i,j,t') \Big\}, \forall s, t'' = t' = t. \quad (16)$$

Note that (16) involves the optimization of $e_{t''}(\cdot)$ to cope with channel uncertainty through $\zeta(\cdot)$ and resource competition with other tiles through $\lambda_{t''}$. For proactive transmission ($t \leq t'' \leq t', t < t' \leq t + W_t$), the payoff should consider the possibility of retransmission in the future if the current transmission fails, since the delay budget may still be sufficient for a proactive tile. Hence, the value function depends on a recursive relation for future value functions as follows:

$$\begin{aligned} V(s,t'' \,|\, i,j,t') = \max_{e_{t''}(i,j,t')} \Big\{ & \zeta(s, e_{t''}(i,j,t'))\tilde{\beta}(i,j,t') - \lambda_{t''} e_{t''}(i,j,t') + \\ & \big(1 - \zeta(s, e_{t''}(i,j,t'))\big) \sum_{s'} P(s,s') V(s', t''+1 \,|\, i,j,t') \Big\}, \forall s, t \leq t'' \leq t', t < t' \leq t + W_t. \quad (17) \end{aligned}$$

Note that if transmission is successful, an immediate payoff worth of its QoE contribution $\tilde{\beta}(\cdot)$ is obtained. For failed transmission with probability $1 - \zeta(\cdot)$, the current payoff depends on

Algorithm 2 CMDP for cross-frame optimization

```
1: **Input:** $\boldsymbol{\beta}_t$, $\tilde{\boldsymbol{\beta}}_t$, $W_t$.
2: **Initialization:** $k = 1$, $\boldsymbol{\lambda}^{(1)} = \mathbf{0}$.
3: **repeat**
4:     Compute (16)-(18) by the backward induction to obtain $\boldsymbol{\mathcal{E}}^{(k)}$.
5:     Compute (20) by the forward induction to obtain $\mathbb{E}\{E_{t''}|\boldsymbol{\mathcal{E}}^{(k)}\}$ in (21).
6:     Update $\boldsymbol{\lambda}^{(k+1)}$ using (19).
7:     $k = k + 1$.
8: **until** $\boldsymbol{\lambda}$ converges.
9: Obtain $e_t^*(i, j, t')$, $\forall i, j, t'$ from $\boldsymbol{\mathcal{E}}^* = \{\boldsymbol{\mathcal{E}}_t^*, \cdots, \boldsymbol{\mathcal{E}}_{t+W_t}^*\}$.
10: **Output:** $\{e_t^*(i, j, t'), \forall i, j, t'\}$.
```

further payoffs $V(s', t'' + 1 \,|\, i, j, t')$ recursively. If the tile is retransmitted but fails repeatedly, such a recursive relation ends when the tile becomes outdated ($t' < t''$) and the resulting payoff is defined as 0:

$$V(s, t'' \,|\, i, j, t') = 0, \, \forall s, t' < t'', t < t' \leq t + W_t. \tag{18}$$

Finding the maximum of (16) is rather straightforward but the optimization of (17) involves recursion on each further step until (18) is reached. One could apply backward induction [32] from $t'' = t'$ to $t'' = t$ step by step to obtain a series of optimal resource allocation $e_{t''}^*(i, j, t'), \forall t'' \in \{t, \cdots, t'\}$.

C. Overall scheduling optimization

To solve the whole system of Bellman equations and obtain optimal resource allocation $\boldsymbol{\mathcal{E}}^* = \{\boldsymbol{\mathcal{E}}_t^*, \cdots, \boldsymbol{\mathcal{E}}_{t+W_t}^*\}$, we require a procedure to also determine the set of Lagrangian multipliers $\boldsymbol{\lambda} = \{\lambda_t, \cdots, \lambda_{t+W_t}\}$, which is crucial to balance the resource competition between different tiles based on all "per-tile" Bellman equations.

1) Subgradient method for $\boldsymbol{\lambda}$*:* $\boldsymbol{\lambda}$ could be found by the subgradient method [33] by iteration. Let $\boldsymbol{\lambda}^{(k)}$ be the Lagrangian multipliers updated for the k-th iteration. Given $\boldsymbol{\lambda}^{(k)}$, one could first solve the system of Bellman equations (16)-(18) to obtain $\boldsymbol{\mathcal{E}}^{(k)}$ and then compute the corresponding resource consumption $\mathbb{E}\{E_{t''}|\boldsymbol{\mathcal{E}}^{(k)}\}$ conditioned on the policy obtained in $\boldsymbol{\mathcal{E}}^{(k)}$. We update $\boldsymbol{\lambda}^{(k+1)}$ for the next iteration as follows:

$$\lambda_{t''}^{(k+1)} = \lambda_{t''}^{(k)} + \epsilon^{(k)} \left(\mathbb{E}\{E_{t''}|\boldsymbol{\mathcal{E}}^{(k)}\} - B \right), \forall t'' \in \{t, \cdots, t + W_t\}, \tag{19}$$

where $\epsilon^{(k)}$ is a sequence of step size. With proper design on $\epsilon^{(k)}$ as shown in [33], $\boldsymbol{\lambda}^{(k)}$ would converge to an optimal solution $\boldsymbol{\lambda}^*$ such that $\boldsymbol{\lambda}^* = \arg\max_{\boldsymbol{\lambda}} \mathcal{L}_{\boldsymbol{\lambda}}$.

2) Induction on $\mathbb{E}\{E_{t''}|\boldsymbol{\mathcal{E}}^{(k)}\}$: As $\boldsymbol{\lambda}^{(k)}$ converges to an optimal value $\boldsymbol{\lambda}^*$, the corresponding $\boldsymbol{\mathcal{E}}^{(k)}$ can also converge to $\boldsymbol{\mathcal{E}}^*$. Note that in each iteration, $\boldsymbol{\mathcal{E}}^{(k)}$ could be obtained by solving the backward induction on the defined Bellman equations (16)-(18). However, to facilitate the update of (19), $\mathbb{E}\{E_{t''}|\boldsymbol{\mathcal{E}}^{(k)}\}$ is required. The computation of $\mathbb{E}\{E_{t''}|\boldsymbol{\mathcal{E}}^{(k)}\}$ involves the steady-state probabilities of all states (s, t''). Let $\omega^{(k)}(s, t'' \,|\, i, j, t')$ denote the steady-state probability of state (s, t'') under $\boldsymbol{\mathcal{E}}^{(k)}$. We could exploit forward induction to obtain all $w^{(k)}(s, t'' \,|\, i, j, t')$ as follows. Starting with (s_t, t) as the initial system state at t, we set $w^{(k)}(s_t, t \,|\, i, j, t') = 1$ and $w^{(k)}(s, t \,|\, i, j, t') = 0$, $\forall s \in \mathcal{S} \setminus s_t$. We then calculate all $w^{(k)}(s, t'' \,|\, i, j, t')$ step by step from $t'' = t + 1$ to $t'' = t'$ as follows:

$$w^{(k)}(s, t'' \,|\, i, j, t') = \sum_{s' \in \mathcal{S}} \big(1 - \zeta(s', e^{(k)}_{t''-1}(i, j, t'))\big) P(s', s) w^{(k)}(s', t'' - 1 \,|\, i, j, t'), \tag{20}$$

which is to sum up the probability to remain unreceived at t'' after transmission of $e^{(k)}_{t''-1}(i, j, t')$ at $t'' - 1$. With all $w^{(k)}(s, t'' \,|\, i, j, t')$ in hand, the expected resource consumption is given by

$$\mathbb{E}\{E_{t''}|\boldsymbol{\mathcal{E}}^{(k)}\} = \sum_{\forall s, i, j, t' \in \{t'', \cdots, t+W_t\}} w^{(k)}(s, t'' \,|\, i, j, t') e^{(k)}_{t''}(i, j, t'), \; \forall t'' \in \{t, \cdots, t + W_t\}. \tag{21}$$

With (21), one could update $\boldsymbol{\lambda}^{(k+1)}$ for the next iteration using (19) repeatedly until $\boldsymbol{\lambda}$ converges.

We conclude the whole procedure in Algorithm 2, which takes $\boldsymbol{\beta}_t$, $\tilde{\boldsymbol{\beta}}_t$, and W_t as the input in line 1 and repeatedly updates $\boldsymbol{\lambda}^{(k)}$ and $\boldsymbol{\mathcal{E}}^{(k)}$ until convergence in line 3 to 8. Although all allocations $\boldsymbol{\mathcal{E}}^* = \{\boldsymbol{\mathcal{E}}^*_t, \cdots, \boldsymbol{\mathcal{E}}^*_{t+W_t}\}$ in the prediction window have been obtained, in line 9 we commit only the decision $\boldsymbol{\mathcal{E}}^*_t$ for the current time frame t as per the principle of MPC.

VII. Performance Evaluation

A. Evaluation environment

Similar to the scenario used in [35], we consider a user that could roam freely inside a 5×5 m^2 room while wearing an HMD that connects to a BS through mmWave. The BS has a height of 2.5 meters and is located at (1.5, 3), where the origin (0, 0) is at the lower-left corner of the room and the height of the user (and HMD) is assumed to be 1.6 meters. To model user mobility, we assume random walk as used in [36] with a speed of 3 km/hr for the indoor scenario [37]. To consider realistic head movement, we resort to a real VR dataset [16] and randomly select the head-movement traces of 20 users while viewing 19 videos. The viewport has a size of $110° \times 110°$, and we assume 6×4 tiling with ERP for its popularity in the literature [21]. The video resolution is set as 60 pixels per degree [2] such that the required bit rate for a tile

TABLE I
SIMULATION PARAMETERS

Parameter	**Value**	**Parameter**	**Value**
Room size	5×5 m^2	User mobility	3 km/hr
Field of view	$110^\circ \times 110^\circ$	Tiling	6×4 in ERP
Video resolution	60 pixels/degree	Bandwidth	500 MHz
Carrier frequency	28 GHz	LOS/NLOS path loss model	[34]

$(60^\circ \times 45^\circ)$ in 6×4 tiling is 150 Mbps on average in the typical H.265 HEVC encoding [2]. We apply CVPR 18 [13] as the default viewport predictor. It can output a series of viewports for future frames by taking saliency, motion information, and user's historic gaze path as the input for the deep neural network (DNN) predictor.

We use mmWave as the underlying communication technology due to its capability to afford high data rate. We set the BS transmit power as 15 dBm and the total amount of resource B as 500 MHz. Channel state for mmWave is typically characterized into line-of-sight (LOS) and non-line-of-sight (NLOS), where LOS denotes the state free to blockage while NLOS denotes the state subject to blockage. We assume indoor scenario and refer to [34] to determine $P_{\text{LOS}}^{\text{indoor}}(d_{\text{BS}})$ based on d_{BS}. To find channel conditions during video streaming, we record the traces of random walk in the 5×5 m^2 room up to 500,000 consecutive time steps and calculate the distance d_{BS} between the user and the BS. We thus obtain the following state transition matrix of LOS and NLOS based on the channel models proposed in [34]:

$$\begin{bmatrix} P_{\text{LOS-LOS}} & P_{\text{LOS-NLOS}} \\ P_{\text{NLOS-LOS}} & P_{\text{NLOS-NLOS}} \end{bmatrix} = \begin{bmatrix} 0.81 & 0.19 \\ 0.72 & 0.28 \end{bmatrix}, \tag{22}$$

where $P_{s\text{-}s'}, \forall s, s' \in \{\text{LOS}, \text{NLOS}\}$ denotes the conditional probability from s to s'. Let $s \in \{\text{LOS}, \text{NLOS}\}$ be the channel state and $e_t(i,j,t') \in [0, 500 \text{ MHz}]$ be the resource consumption. The probability of successful transmission $\zeta\big(s, e_t(i,j,t')\big)$ is determined by

$$\zeta\big(s, e_t(i,j,t')\big) = \Pr\left\{e_t(i,j,t') \log_2(1+\gamma_s) \geq C(i,j,t')\right\}, \tag{23}$$

where $C(i,j,t')$ is the required bit rate for tile (i,j,t') and γ_s is the signal-to-noise ratio (SNR) under channel state s. The path loss model for γ_s follows [34].

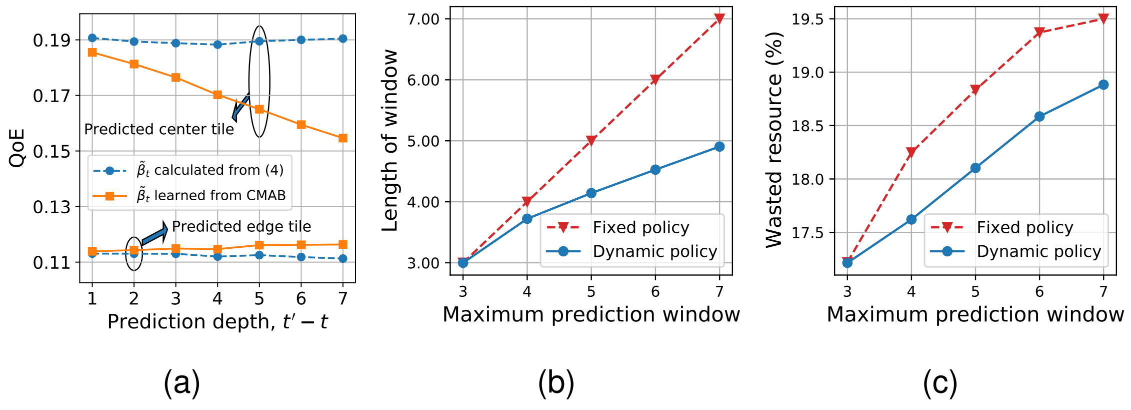


Fig. 5. (a) Comparison between $\tilde{\beta}_t$ learned from CMAB and $\tilde{\beta}_t$ calculated from (4). (b) Average prediction window between fixed and dynamic policy. (c) Percentage of wasted resource between fixed and dynamic policy.

B. *Performance of the proposed CMAB algorithm*

One of our contributions is the design of the CMAB algorithm, which learns the context-aware QoE and determines the required length of prediction window during the streaming. We first demonstrate how the learned QoE could address prediction error among various frames and how the dynamic prediction window could benefit the performance.

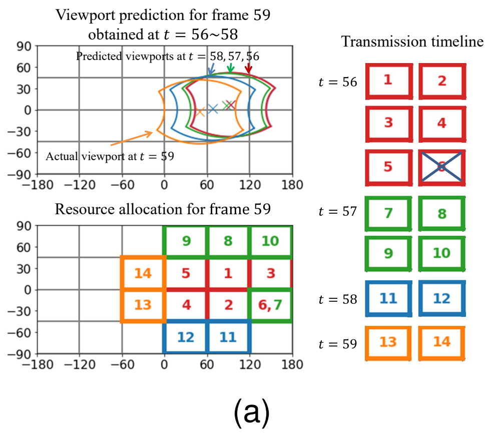
1) Context-aware QoE estimation: We demonstrate how prediction error differs across prediction depths as Fig. 1 has depicted. Fig. 5a shows the estimated QoE $\tilde{\boldsymbol{\beta}}_t$ learned from the proposed CMAB algorithm in contrast to $\tilde{\boldsymbol{\beta}}_t$ calculated directly from (4) using the predicted viewport $\boldsymbol{\phi}_t$. Note that $\tilde{\boldsymbol{\beta}}_t$ calculated from (4) assumes the predicted viewport to be the same as the actual viewport. Thus, it does not consider the impact of prediction error and remains the same for all prediction depths. On the other hand, the learned $\tilde{\boldsymbol{\beta}}_t$ for center tiles decrease substantially (from 0.186 to 0.163) and that for edge tiles increase slightly (from 0.115 to 0.120) as the prediction depth increases. This phenomenon reflects a fact that prediction error could make a predicted center tile deviate from the actual viewport center, thus leading to lower QoE contribution as error increases with the prediction depth. The predicted edge tile, on the other hand, may become closer to or farther away from the actual viewport center due to prediction error, and hence the impact on QoE estimation is not pronounced. Note that the proposed QoE metric involves normalization of all viewed tiles in the viewport such that decrease of QoE contribution for center tiles leads to relative increase of QoE contribution for edge tiles. In conclusion, Fig. 5a demonstrates that by learning from the past observation through the proposed CMAB, the proposed QoE estimation could capture the actual impact that *prediction error increases as prediction depth increases* and such impact varies among spatial context.

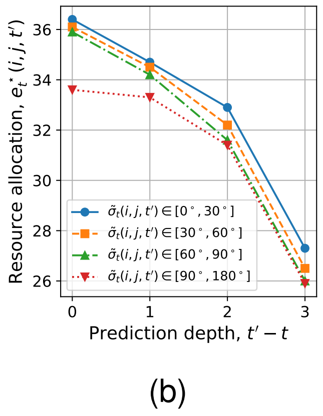
2) Dynamic prediction window adjustment: To see the benefit of dynamic adjustment of the prediction window, we consider a policy that fixes W_t to $W_{\max}$ in comparison to the proposed dynamic policy. We vary the value of $W_{\max}$ in Fig. 5b and keep track of the average length of the prediction window during streaming for all user-video combinations sampled from the dataset. We find that for the fixed policy, W_t increases linearly with $W_{\max}$ as expected. Although W_t for the dynamic policy still grows with $W_{\max}$ as Fig. 5b shows, we could observe that the dynamic policy effectively reduces W_t compared to the fixed policy, especially for higher $W_{\max}$. Note that under the dynamic policy, W_t^{exploit} as defined in (12) converges to 3.81 for $W_{\max} > 4$. That W_t still grows indicates that the proposed window adjustment could adapt to the need of resource allocation as driven by W_t^{explore} (higher W_t^{explore} is needed to update all under-explored hypercubes). In Fig. 5c, we further define *wasted resource* as the percentage of resource used on transmitting unviewed tiles out of total allocated resource. It can be observed that the proposed dynamic W_t could reduce wasted resource for $W_{\max} > 3$ compared to the fixed policy. Note that prediction error is typically larger when W_t is large and hence a smaller W_t could prevent resource allocation to be misled by prediction error in the distant future. In summary, dynamic prediction window could save prediction overhead and wasted resource without sacrificing QoE.

C. Performance of the proposed CMDP algorithm

To demonstrate how cross-frame optimization based on CMDP is beneficial, we first provide microscopic observation on tile selection and transmission order. We then show how resource is allocated among tiles and frames to mitigate channel uncertainty and prediction error.

1) Microscopic observation: Fig. 6a visualizes one instance of resource allocation based on CMDP for tiles to be consumed in frame 59 of the video `Cows` as viewed by user 43 in the dataset. Since proactive transmission is allowed, these tiles (for frame 59) may be transmitted before $t = 59$, as labeled in Fig. 6a in terms of the transmission times (from $t = 56$ to $t = 59$) and the transmission orders (from 1 to 14). The predicted viewports for frame 59 obtained at $t = 56$, 57, and 58 deviate from the actual viewport at $t = 59$ with errors in the positions of the viewport centers for $44.8°$, $38.6°$, and $19.2°$ respectively. Based on these predictions, the BS first delivers transmissions 1 to 6 (red tiles), which cover most of the predicted viewport's center region. Transmission 6 fails due to bad channel. Later, transmissions 7 to 10 at $t = 57$ (green


Viewport prediction for frame 59 obtained at t = 56~58
Predicted viewports at t = 58, 57, 56
Actual viewport at t = 59
Resource allocation for frame 59
Transmission timeline
(a)



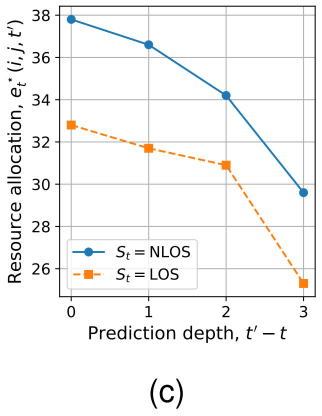


Fig. 6. (a) Visualization of resource allocation and viewport prediction for all tiles in frame 59 for user 43 viewing video `Cows`. (b) Optimal resource allocation under different prediction depths for different angles of view $\tilde{\sigma}_t(i,j,t')$. (c) Optimal resource allocation under different prediction depths for different channel conditions S_t.

tiles) and 11 to 12 at $t = 58$ (blue tiles) cover the predicted viewport's edge region. However, at $t = 59$ it is found that the actual viewport deviates from all previous predictions and hence transmissions 13 and 14 (orange tiles) are allocated reactively to accommodate prediction error.

The scenario demonstrated in Fig. 6a is for the case with relatively higher prediction error. Yet even under such a case, all required tiles are transmitted *in time* due to the design of cross-frame optimization for reactive and proactive transmissions. Note that center tiles are transmitted first, followed by edge tiles as observed in Fig. 6a. Such transmission orders are due to the proposed QoE estimation that can rank tiles for resource allocation based on the position, viewed area of each tile, and prediction depth. Packet loss is also addressed since proactive transmission allows for retransmission opportunity as shown by transmissions 6 and 7 in Fig. 6a.

2) Optimal resource allocation among tiles and frames: In Fig. 6b and Fig. 6c, we show the average value of resource allocation $e_t^*(i,j,t')$ for all user-video combinations under different prediction depths. Since a transmission loss for a tile in the distant future could be retransmitted later (as long as there is still delay budget), both Fig. 6b and Fig. 6c reveal that $e_t^*(i,j,t')$ decreases for tiles of larger prediction depth ($t'-t$). On the other hand, more imminent transmission (smaller $t'-t$), such as reactive transmission, is shown to receive higher amount of resource to avoid transmission loss. Such phenomenon echoes the results observed in [23].

Fig. 6b also compares resource allocation for tiles with different angle of views $\tilde{\sigma}_t(i,j,t')$. It can be observed that the algorithm allocates more resource to tiles near the predicted center area (with angle of view less than $30°$). The reason is because these tiles have higher QoE contribution (due to lower angle of view and more complete viewing area) compared to edge tiles. Additionally, Fig. 6c compares resource allocation for tiles under different channel conditions (transmission states S_t). We observe that tiles in the NLOS state are provided with higher resource allocation compared to the LOS state. This is because when the channel condition is poor (NLOS state), more resource is needed for each tile to ensure that the tile can be received successfully with large probability.

D. Overall performance comparison

We compare our proposed streaming system with two benchmark systems, 360ProbDASH [9] and Flare [21] as described in Section II. 360ProbDASH is based on sequential optimization while Flare is based on cross-frame optimization with heuristic ranking as follows:

- *360ProbDASH* [9]: This work models prediction error by Gaussian distribution to compute the viewing probability for each tile. Resource allocation problem is optimized sequentially according to the derived viewing probability for each tile according to the order of playback.
- *Flare* [21]: In this work, a number of tiles are selected for transmission that can cover an *extended viewport* while accommodating prediction error. To proceed, tiles are first ranked into several classes based on their angles of view to the predicted viewport centers. The scheduler then determines the transmission priorities for all tiles first by their ranking classes and then by their playback orders. Resource allocation across various frames is thereby solved according to such tile ranking.

The comparisons are in terms of the user experience and system efficiency, where user experience is evaluated based on the QoE and system efficiency is measured by the percentage of resource wasted on transmitting unused tiles defined previously in Section VII-B2.

1) Performance under various prediction errors: To show how the proposed system could mitigate the impact of prediction error, we add additional Gaussian noises to the latitude and longitude of the viewport center predicted by CVPR 18 [13]. The noise is with zero mean and different variances to model a predictor with worsening prediction error. Note that the first value on the x-axis ($11.6°$) of Fig. 7a and Fig. 7b is the original average prediction error, while the following three values ($20.3°$, $34.8°$, and $46.9°$) are results due to the added Gaussian noise. As we observe in Fig. 7a and Fig. 7b, the proposed system achieves the highest QoE and the lowest percentage of resource waste. Performance improvement is larger under larger

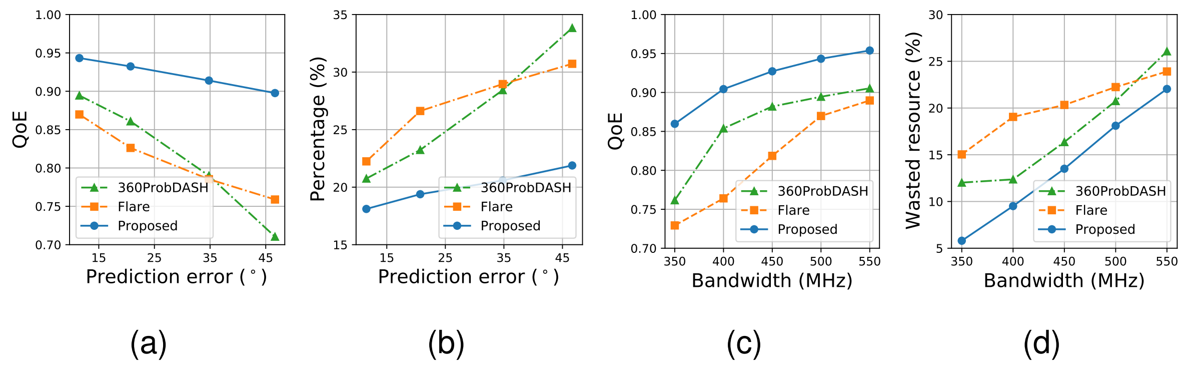


Fig. 7. Performance comparison between the proposed and benchmark systems. (a) QoE under various viewport prediction errors. (b) Percentage of wasted resource under various viewport prediction errors. (c) QoE under various bandwidth constraints. (d) Percentage of wasted resource under various bandwidth constraints.

prediction errors, indicating that the proposed system with context-aware QoE estimation and cross-frame optimization is more resilient to prediction error. On the contrary, 360ProbDASH optimizes resource allocation sequentially based on the predicted viewing probability, which results in sub-optimal performance especially when the prediction error is high. Although Flare considers the whole series of prediction information, larger prediction errors still undermine its performance since its mechanism on extended viewport would invite too many tiles to transmit for each frame, thus leading to more wasted resource.

2) Performance under various bandwidth constraints: Fig. 7c and Fig. 7d compare the performance of these 3 streaming systems under different bandwidth constraints B. Note that 360ProbDASH could only optimize resource allocation sequentially based on spatial information and it does not handle channel uncertainty. Although Flare allows resource allocation across various viewport predictions, the heuristics used in Flare may lead to resource waste since tiles are preferentially ranked (and allocated resource) by their playback orders. Transmitting too many tiles in the distant future is prone to prediction error (refer to Fig. 1) and hence may result in wasted resource under channel uncertainty. On the contrary, the proposed system could address both of channel uncertainty and prediction error by optimizing the retransmission and resource competition between reactive and proactive transmissions facilitated by cross-frame optimization. Hence, better performance could be achieved under various bandwidth constraints.

3) Performance under various videos, users, and predictors: To demonstrate the applicability of the proposed system, in addition to "CVPR 18," we also apply another state-of-the-art viewport

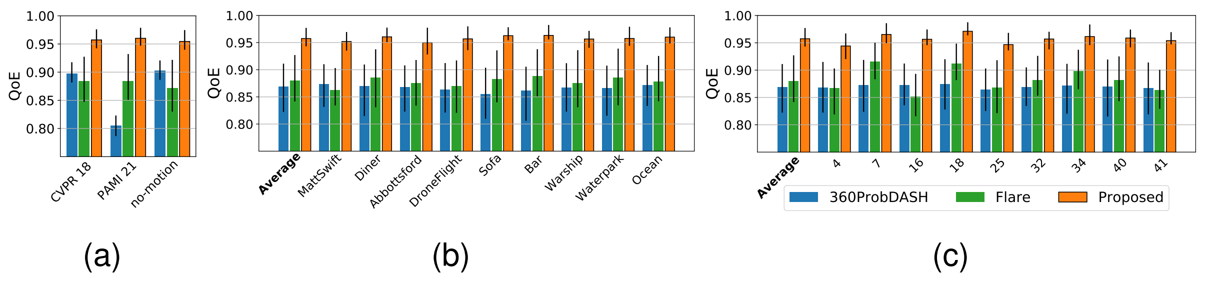


Fig. 8. QoE achieved by the proposed and benchmark systems under various predictors, videos, and users. The vertical black line atop each bar indicates the range from 25-th to 75-th percentiles while the bar indicates the average value. (a) QoE across various predictors. (b) QoE across various videos. (c) QoE across various users.

predictor, “PAMI 21” [14], as well as a baseline predictor, “no-motion” [15], for validation:

- *PAMI 21* [14]: This predictor handles the user’s past position and video content independently by using separate recurrent neural network (RNN) units and fuses them by a shared RNN layer to predict a series of future viewports.
- *no-motion* [15]: This predictor assumes all future viewports stay still (same as the current viewport) during the prediction window, thus providing baseline prediction for validation.

Fig. 8a shows the average QoE across all videos and users in the dataset for all viewport predictors (CVPR 18, PAMI 21, and no-motion). Fig. 8b and Fig. 8c further breakdown the average QoE (of the three predictors altogether) for different videos and users under consideration. For sake of space, we show only the results for 9 videos viewed by all users in Fig. 8b and 9 users viewing all videos in Fig. 8c. It can be observed that the proposed system consistently outperforms 360ProbDASH and Flare, with an overall average improvement on QoE by 10.2% and 8.8% as well as reduction on wasted resource by 18.7% and 17.4% respectively. (The figure on wasted resource is omitted due to space constraint.) As indicated by the lengths of vertical lines (the range from 25-th to 75-th percentiles) in the figure, the proposed system can also achieve lower variance on the performance compared against the two benchmark systems.

VIII. Conclusion

We have formulated a cross-frame QoE optimization problem for wireless 360° video streaming. Instead of optimizing the scheduling decision for a specific future frame, our formulation aims to jointly consider a series of future predictions. By leveraging all prediction information within the prediction window, optimal resource allocation for the upcoming frames within the prediction window could be determined. Evaluation results have shown that the proposed system better accommodates prediction error and channel uncertainty compared to related work.

References

[1] 3GPP, "Virtual Reality (VR) Media Services over 3GPP," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 26.918, 11 2020, version 16.0.0.

[2] M. S. Elbamby, C. Perfecto, M. Bennis, and K. Doppler, "Toward Low-Latency and Ultra-Reliable Virtual Reality," *IEEE Netw.*, vol. 32, no. 2, pp. 78–84, 2018.

[3] 3GPP, "3GPP Virtual Reality Profiles for Streaming Applications," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 26.118, 07 2021, version 16.3.0.

[4] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 Video Delivery over Cellular Networks," in *Proc. ACM Workshop on All Things Cellular: Operations, Applications and Challenges*, 2016, pp. 1–6.

[5] Y. Bao, H. Wu, T. Zhang, A. A. Ramli, and X. Liu, "Shooting a Moving Target: Motion-Prediction-Based Transmission for 360-Degree Videos," in *Proc. IEEE International Conference on Big Data*, 2016, pp. 1161–1170.

[6] J. Chen, X. Qin, G. Zhu, B. Ji, and B. Li, "Motion-Prediction-Based Wireless Scheduling for Multi-User Panoramic Video Streaming," in *Proc. IEEE INFOCOM*, 2021, pp. 1–10.

[7] Y. Zhang, P. Zhao, K. Bian, Y. Liu, L. Song, and X. Li, "DRL360: 360-Degree Video Streaming with Deep Reinforcement Learning," in *Proc. IEEE INFOCOM*, 2019, pp. 1252–1260.

[8] L. Chopra, S. Chakraborty, A. Mondal, and S. Chakraborty, "PARIMA: Viewport Adaptive 360-Degree Video Streaming," in *Proc. ACM WWW*, 2021, pp. 2379–2391.

[9] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360ProbDASH: Improving QoE of 360 Video Streaming Using Tile-Based HTTP Adaptive Streaming," in *Proc. ACM MM*, 2017, pp. 315–323.

[10] J. Zou, C. Li, C. Liu, Q. Yang, H. Xiong, and E. Steinbach, "Probabilistic Tile Visibility-Based Server-Side Rate Adaptation for Adaptive 360-Degree Video Streaming," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 1, pp. 161–176, 2020.

[11] L. Xie, X. Zhang, and Z. Guo, "CLS: A Cross-User Learning Based System for Improving QoE in 360-Degree Video Adaptive Streaming," in *Proc. ACM MM*, 2018, pp. 564–572.

[12] L. Sun, Y. Mao, T. Zong, Y. Liu, and Y. Wang, "Flocking-Based Live Streaming of 360-Degree Video," in *Proc. ACM MMSys*, 2020, pp. 26–37.

[13] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao, "Gaze Prediction in Dynamic 360° Immersive Videos," in *Proc. IEEE CVPR*, 2018, pp. 5333–5342.

[14] M. F. Romero Rondon, L. Sassatelli, R. Aparicio-Pardo, and F. Precioso, "TRACK: A New Method from a Re-Examination of Deep Architectures for Head Motion Prediction in 360-Degree Videos," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–1, 2021.

[15] M. F. R. Rondón, L. Sassatelli, R. Aparicio-Pardo, and F. Precioso, "A Unified Evaluation Framework for Head Motion Prediction Methods in 360° Videos," in *Proc. ACM MMSys*, 2020, pp. 279–284.

[16] E. J. David, J. Gutiérrez, A. Coutrot, M. P. Da Silva, and P. L. Callet, "A Dataset of Head and Eye Movements for 360° Videos," in *Proc. ACM MMSys*, 2018, pp. 432–437.

[17] C. Ozcinar, A. De Abreu, and A. Smolic, "Viewport-Aware Adaptive 360° Video Streaming Using Tiles for Virtual Reality," in *Proc. IEEE ICIP*, 2017, pp. 2174–2178.

[18] X. Zhang, G. Cheung, Y. Zhao, P. Le Callet, C. Lin, and J. Z. G. Tan, "Graph Learning Based Head Movement Prediction for Interactive 360 Video Streaming," *IEEE Trans. Image Process.*, vol. 30, pp. 4622–4636, 2021.

[19] X. Hou, S. Dey, J. Zhang, and M. Budagavi, “Predictive Adaptive Streaming to Enable Mobile 360-Degree and VR Experiences,” *IEEE Trans. Multimedia*, vol. 23, pp. 716–731, 2021.

[20] I.-H. Hou, N. Z. Naghsh, S. Paul, Y. C. Hu, and A. Eryilmaz, “Predictive Scheduling for Virtual Reality,” in *IEEE INFOCOM*, 2020, pp. 1349–1358.

[21] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, “Flare: Practical Viewport-Adaptive 360-Degree Video Streaming for Mobile Devices,” in *Proc. ACM MobiCom*, 2018, pp. 99–114.

[22] L. Huang, S. Zhang, M. Chen, and X. Liu, “When Backpressure Meets Predictive Scheduling,” *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2237–2250, 2016.

[23] K. Chen and L. Huang, “Timely-Throughput Optimal Scheduling with Prediction,” *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2457–2470, 2018.

[24] B. Yin, S. Zhang, Y. Cheng, L. X. Cai, Z. Jiang, S. Zhou, and Z. Niu, “Only Those Requested Count: Proactive Scheduling Policies for Minimizing Effective Age-of-Information,” in *Proc. IEEE INFOCOM*, 2019, pp. 109–117.

[25] Y. Sun, A. Lu, and L. Yu, “Weighted-to-Spherically-Uniform Quality Evaluation for Omnidirectional Video,” *IEEE Signal Process. Lett.*, vol. 24, no. 9, pp. 1408–1412, 2017.

[26] C. Ozcinar, J. Cabrera, and A. Smolic, “Visual Attention-Aware Omnidirectional Video Streaming Using Optimal Tiles for Virtual Reality,” *IEEE Trans. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 217–230, 2019.

[27] N. Bruce and J. Tsotsos, “Saliency Based on Information Maximization,” in *Proc. NeurIPS*, vol. 18. MIT Press, 2005.

[28] M. Xu, Y. Song, J. Wang, M. Qiao, L. Huo, and Z. Wang, “Predicting Head Movement in Panoramic Video: A Deep Reinforcement Learning Approach,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2693–2708, 2019.

[29] H. Law and J. Deng, “CornerNet: Detecting Objects as Paired Keypoints,” in *Proc. ECCV*, September 2018.

[30] E. F. Camacho and C. B. Alba, *Model Predictive Control.* Springer Science & Business Media, 2013.

[31] L. Chen and J. Xu, “Task Replication for Vehicular Cloud: Contextual Combinatorial Bandit with Delayed Feedback,” in *Proc. IEEE INFOCOM*, 2019, pp. 748–756.

[32] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 2014.

[33] D. Bertsekas, A. Nedic, and A. Ozdaglar, *Convex Analysis and Optimization*. Athena Scientific, 2003, vol. 1.

[34] K. Haneda, L. Tian, H. Asplund, J. Li, Y. Wang, D. Steer, C. Li, T. Balercia, S. Lee, Y. Kim, A. Ghosh, T. Thomas, T. Nakamurai, Y. Kakishima, T. Imai, H. Papadopoulas, T. S. Rappaport, G. R. MacCartney, M. K. Samimi, S. Sun, O. Koymen, S. Hur, J. Park, J. Zhang, E. Mellios, A. F. Molisch, S. S. Ghassamzadeh, and A. Ghosh, “Indoor 5G 3GPP-like Channel Models for Office and Shopping Mall Environments,” in *Proc. IEEE ICC Workshops*, 2016, pp. 694–699.

[35] A. Borrego, J. Latorre, M. Alcañiz, and R. Llorens, “Comparison of Oculus Rift and HTC Vive: Feasibility for Virtual Reality-Based Exploration, Navigation, Exergaming, and Rehabilitation,” *Games for Health Journal*, vol. 7, no. 3, pp. 151–156, 2018.

[36] L. Sanguinetti, A. L. Moustakas, E. Björnson, and M. Debbah, “Large System Analysis of the Energy Consumption Distribution in Multi-User MIMO Systems with Mobility,” *IEEE Trans. Wireless Commun.*, vol. 14, no. 3, pp. 1730–1745, 2015.

[37] 3GPP, “Study on Scenarios and Requirements for Next Generation Access Technologies,” 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.913, 07 2020, version 16.0.0.