Robust Saliency-Driven Quality Adaptation for Mobile 360-Degree Video Streaming

Shibo Wang, Shusen Yang, Hairong Su, Cong Zhao, Chenren Xu, Feng Qian, Nanbin Wang, and Zongben Xu

Abstract—Mobile 360-degree video streaming has grown significantly in popularity but the quality of experience (QoE) suffers from insufficient and variable wireless network bandwidth. Recently, saliency-driven 360-degree streaming overcomes the buffer size limitation of head movement trajectory (HMT)-driven solutions and thus strikes a better balance between video quality and rebuffering. However, inaccurate network estimations and intrinsic saliency bias still challenge saliency-based streaming approaches, limiting further QoE improvement. To address these challenges, we design a robust saliency-driven quality adaptation algorithm for 360-degree video streaming, RoSal360. Specifically, we present a practical, tile-size-aware deep neural network (DNN) model with a decoupled self-attention architecture to accurately and efficiently predict the transmission time of video tiles. Moreover, we design a reinforcement learning (RL)-driven online correction algorithm to robustly compensate the improper quality allocations due to saliency bias. Through extensive prototype evaluations over real wireless network environments including commodity WiFi, 4G/LTE, and 5G links in the wild, RoSal360 significantly enhances the video quality and reduces the rebuffering ratio, thereby improving the viewer QoE, compared to the state-of-the-art algorithms.

Index Terms—360-degree video streaming, Quality adaptation, Saliency, Network estimation

1 Introduction

The virtual reality (VR) market was valued at USD 15.81 billion in 2020 and is expected to grow at an annual growth rate of 18.0% from 2021 to 2028 [1]. As one of the most potential VR applications, mobile 360-degree video streaming, i.e., viewing 360-degree video streams on untethered mobile VR headsets, has experienced a considerable increase in popularity [2], [3], [4], [5]. However, insufficient and fluctuant wireless network bandwidth limits the quality of experience (QoE). First, the delivery of high-definition 360-degree videos can not be fully supported by the current wireless network capacity. A 4K-resolution 360-degree video demands at least 25 Mbps bandwidth [6], while the LTE speeds are only 5-12 Mbps in many regions [7]. Second, mobile 360-degree video streaming would suffer from unexpected rebuffering due to highly variable wireless network bandwidth [8]. Either low-definition VR display or rebuffering would severely degrade the QoE for mobile VR viewers, such as disorientation and nausea [9]. Thus, enhancing the QoE of mobile 360-degree video streaming under limited, dynamic wireless network bandwidth is critical and urgent.

The majority of previous research adopts *head movement trajectory (HMT)-driven* optimization methods [10], [11], [12], [13], [14], [15]. A basic assumption underlying their approaches is that the HMT of a viewer has a strong temporal correlation, which means that the future HMT can be accurately predicted by the historical HMT alone during the playback. However, this temporal correlation would not always be strong enough to achieve accurate HMT predictions, which would severely degrade the QoE in the HMT-driven solutions [10]. Moreover, since the prediction horizon of HMT has to exceed the buffer occupancy on the client player, the HMT-driven approaches are obliged to set a small buffer size (i.e., maximum buffer length) to reduce prediction horizons due to less correlation with the longer interval. The scarce buffer would lead to a high risk of rebuffering, especially under highly dynamic wireless network environments.

Recently, the introduction of saliency information [16], [17], [18] provides a new favorable 360-degree video streaming framework to address the limitations caused by temporal-correlation-dependent HMT predictions. Saliency-based methods acquire the saliency maps from historical viewing data (e.g., gaze data) or video content offline, and tend to allocate high quality levels to high-salient regions during playback. Saliency provides content-aware prior information for long-term viewer attention estimations, free from online HMT data, and thus achieve a better trade-off between video quality and rebuffering. However, saliency-driven quality adaptation still suffers from inaccurate network estimations and saliency bias, limiting further QoE improvement for mobile 360-degree video streaming.

Most existing 360-degree video quality adaptation schemes [10], [11], [17] adopt plausible but under-designed bandwidth estimation methods (e.g., the harmonic mean of past five samples). These methods fail to achieve the satisfactory estimation accuracy in the wild especially in wireless network environments, which degrades the QoE of mobile 360-degree video streaming. Recently, emerging deep neural network (DNN)-driven transmission time prediction (TTP) algorithms [19] consider the effects of *chunk sizes* on throughputs that the user actually experiences,

- *Shibo Wang, Shusen Yang, Hairong Su, Cong Zhao, and Zongben Xu are with Xi'an Jiaotong University.*
- *Chenren Xu is with Peking University.*
- *Feng Qian is with University of Minnesota - Twin Cities.*
- *Nanbin Wang is with Huawei Technology.*
- *This is an extended and enhanced version of a paper that appeared in ACM MobiCom 2022.*

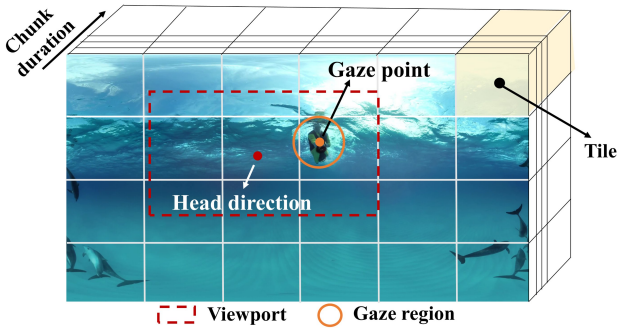


Fig. 1: Viewport, gaze region, chunk, and tiles. Gaze region is a small percentage region of viewport with high visual acuity (more details in [17]). A video chunk is segmented into $N_{row} \times N_{col}$ tiles (4×6 in this example). A tile has the same duration and frame number as the corresponding chunk, but occupies only a small part of spatial region.

and make revolutionary progress in term of TTP accuracy and corresponding QoE. However, these algorithms are designed for conventional non-tiled video streaming and are hardly practical for 360-degree video streaming due to tile-based delivery.

Moreover, the saliency maps are determined by collective preferences using the offline collected historical viewing data, without considering the individual preference of new viewers. Inevitable saliency bias would influence the QoE of outlier viewers. For instance, a viewer that prefers to gaze at collectively-determined non-salient regions would perceive weak QoE due to the quality adaptation using saliency alone (the less salient the lower allocated quality). The prior work [17] tries to compensate the adverse effects induced by saliency bias using a heuristic online correction mechanism, that is, redownloading *improper* in-buffer video tiles. However, in the prior work, *whether a tile is proper* is judged based on the HMT-driven viewport prediction, the methods of which *per se* are not always credible enough to support an accurate proper-or-not judgment. In addition, existing correction schemes suffer from the risk of overdue tile redownloading especially considering the still imperfect TTP outcomes, which would waste network resources and even cause inferior QoE performances compared to a streaming system without the correction mechanism.

To address above challenges and enhance the QoE, we present a robust saliency-driven 360-degree video quality adaptation algorithm, RoSal360. We present a practical, tile-size-aware TTP algorithm for tile-based video transmission. Specifically, we design a self-attention-based DNN model to effectively and efficiently predict the transmission time of each tile by decoupling the network tendency estimation and the effects of tile sizes. Moreover, we present a reinforcement learning (RL)-driven online correction algorithm combining the online estimated credibility of both viewport predictions and network predictions, to reduce the disadvantages caused by saliency bias.

We constructed a gaze-annotated 360-degree video dataset using untethered VR headsets with the built-in eye-tracking device, based on 30 volunteers and 23 various long 360-degree videos. We implemented a prototype of RoSal360 and conducted large-scale evaluations over real-world wireless network environments including commodity WiFi, 4G/LTE, and 5G links in the wild. The evaluation on real viewing datasets, consisting of more than two hundred hours of playback, shows that RoSal360 achieves an up to 4.57 dB improvement in gaze-driven Peak-Signal-to-Noise-Ratio (PSNR) and reduces the rebuffering ratio by up to 4.11×, thereby significantly enhancing the QoE of viewers, compared to the state-of-the-art approaches. All user studies in our work were IRB approved without raising any ethical issues.

In summary, our key contributions are as follows:

- We present RoSal360, a robust, integrated saliency-driven quality adaptation algorithm designed for mobile 360-degree video streaming.
- We design a practical, efficient, and tile-size-aware transmission time prediction algorithm based on an attention-based DNN model with a novel decoupled architecture.
- We design a RL-driven online correction algorithm combining the prediction credibility of both viewer behaviors and network conditions, to reduce the adverse effects of saliency bias.
- By extensive prototype experiments using gaze-annotated 360-degree video datasets on various real wireless networks, we demonstrate that RoSal360 achieves significant improvements on video quality, rebuffering, and viewer QoE over current schemes.

Additional novelties. The previous work, SalientVR [17] (i.e., the conference version of this manuscript), mainly focuses three key challenges in the design of saliency-driven 360-degree video streaming system, i.e., saliency groundtruth judgment, saliency map acquirement, and the saliency-aware quality adaptation. In contrast, this manuscript (i.e., RoSal360) further addresses the new challenges of saliency-based solutions, i.e., tile-level transmission time prediction and online correction, for the robustness to network variation and sailency bias, which are not considered or handled with minimal efforts only by SalientVR.

The remainder of this paper is organized as follows. Section 2 summarizes the background and motivations. Section 3 formulates the tile-level 360-degree video quality adaptation problem with an explicit consideration about the tile's transmission time. Section 4 presents the overview of the RoSal360 system. Section 5 proposes the core design of RoSal360, the robust saliency-driven quality adaptation algorothm. Section 6 introduces the prototype implementation. Section 7 evaluates the performances of RoSal360 over real wireless networks. Section 8 states the limitation and discussion. Section 9 describes the related work. Finally, we conclude in Section 10.

2 Background and Motivation

In this section, we first introduce the background of mobile 360-degree video streaming and the limitations of existing optimization methods. Then, we expound the advantages and remaining defects of saliency-driven quality adaptation for 360-degree videos.

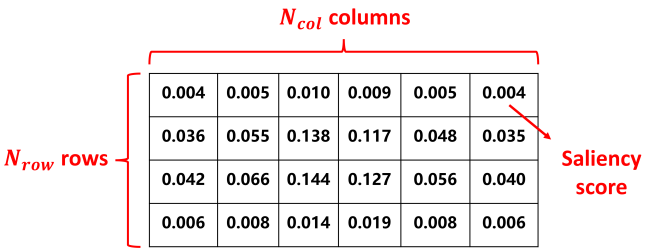


Fig. 2: A tile-level saliency map is a numerical matrix with the same dimension as the tile segmentation. The matrix entries are saliency scores of different tiles in a video chunk, representing the saliency degrees of different tile regions.

2.1 Mobile 360-Degree Video Streaming

Untethered mobile VR headsets enable viewers to enjoy mobile 360-degree video streaming without wired hindrance, but suffer from insufficient wireless network bandwidth. Inspired by the fact that human view is restricted, past works mainly adopt the HMT-driven approaches [10], [12], [13], [14]. They split each video chunk spatially into multiple tiles (e.g., 4×6 grids), as illustrated in Fig. 1, and encode each tile with different quality levels, such as quantization parameters (QP). During the playback, they constantly predict the future HMT in the next 1-3 seconds according to the recently historical HMT using linear regression (LR)-based or DNN-based methods [10], [11]. Based on the predicted HMT, they assess the *importance* of each tile of the next chunk and assign the higher qualities to the more important tiles.

However, these HMT-driven solutions overly rely on the temporal correlation of HMT, leading to two shortcomings. First, the future HMT is scarcely correlated with the historical HMT in some cases even with a short prediction horizon (a.k.a, prediction window or pw) [17]. For mobile VR viewers, the untethered freedom further reduces this temporal correlation. Second, the HMT-driven approaches are highly sensitive to the size of pw due to rapid correlation decrease as the time interval increases. Even three seconds of pw degrades the HMT prediction accuracy into 35.2% [10]. Due to the prefetching feature of on-demand video streaming, pw has to exceed the buffer occupancy during the playback. Therefore, these approaches set a very small buffer size to shorten pw in practice, but suffer from frequent rebuffering especially over variable wireless network conditions.

2.2 Saliency-Driven Quality Adaptation

Saliency [18], [20] is a term in computer vision literature, representing the property of grabbing attention for an object or a region. Recently, inspired by the fact that viewing behaviors of humans are remarkably correlated with the saliency of pixels [18], [20], [21], several works explore the saliency-driven 360-degree video streaming solutions for viewer QoE improvement [16], [17]. These approaches utilize the collected viewing data (e.g., gaze data) to acquire the saliency map (illustrated in Fig 2) of each chunk based on cross-user viewing behavior similarity or DNN-driven video content analysis. Then, they give priority to high-salient video tiles and keep a longish buffer size. The saliency maps acquired offline provide human-centric prior information for accurate long-term attention estimations without depending on the online HMT data. Therefore, the saliency-based framework gets rid of the limitations of temporal correlation and achieves a better trade-off between video quality and rebuffering. Despite great progress of saliency-based solutions, there are several drawbacks in existing saliency-driven quality adaptation approaches, hampering the further QoE enhancement.

First, mobile 360-degree video streaming still suffers from the inaccurate TTP (i.e., transmission time prediction), which plays a fundamentally important role in quality adaptation. An underestimated network condition would incur a low bandwidth utilization and on the other hand, an overestimated bandwidth would increase the risk of buffer exhaustion. Most of previous efforts adopt the intuitive but under-designed bandwidth estimation algorithms (e.g., the harmonic mean of past five samples), which are not accurate and robust enough to support a decent quality allocation. For conventional DASH-based video streaming, several works [19] present the chunk-size-aware, one-stage, and DNN-assisted TTP algorithms, which show promising potential for the accurate TTP. In contrast to *two-stage* TTP frameworks, i.e., predicting the bandwidth and then using the ratio of sizes to estimated bandwidth as TTP results, *one-stage* TTP algorithms consider the effects of chunk sizes on actual throughputs, and predict the transmission time straightforward. However, these TTP methods are inefficient even impractical for tile-based video streaming, where all tiles are independently transmitted. Due to the introduction of tile sizes as model input, each tile has to be separately inferred to output the TTP result. For example, $5 \times 24 = 120$ rounds of forward propagation in neural networks are required for five optional quality levels and 4×6 tile segmentation every chunk, which is inefficient in practice. Therefore, we decouple the network tendency prediction and the effects of tile sizes on throughputs, and put forward to an efficient, practical TTP algorithm based on a novel attention-based DNN architecture.

Second, the saliency maps are prepared offline and are determined by collective preferences without considering individual users. Inevitable saliency bias may result in poor experiences for few individual users with distinctive attention preferences, i.e., outlier viewers. For instance, a viewer that prefers to gaze at collectively-determined non-salient regions would perceive weak QoE due to the quality adaptation using saliency alone (the less salient the lower allocated quality). Therefore, an online correction mechanism is crucial in saliency-driven quality adaptation for robustness to saliency bias. Prior work [17] proposes a heuristic correction scheme to redownload the improper buffer-in video tiles. However, whether an already downloaded tile possesses the proper quality level is judged by HMT-driven viewport predictions, which *per se* are not always reliable. Moreover, the redownloaded tiles will be useless for playback and wasteful of network resources if they fail to arrive before the time they ought to be displayed, which would even cause inferior QoE performances over the streaming system without correction mechanisms. The inaccurate TTP further increases the risk of the overdue tile redownloading. Therefore, to further reduce the negative influence induced

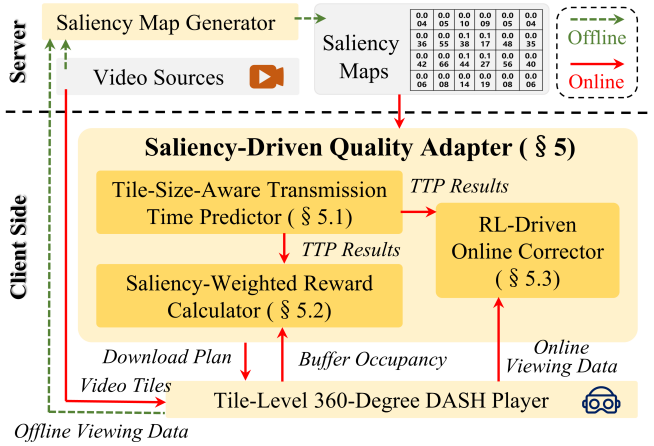


Fig. 3: Overview of the RoSal360 system.

by saliency bias, we propose a RL-driven online correction algorithm combining the credibility of viewing behavior predictions, buffer occupancy, and estimated network variations.

3 Problem Statement

Tile-level 360-degree video quality adaptation, i.e., deciding which quality level to be allocated to each tile, can be formulated as a QoE maximization problem:

$$\max_{l_i(j)} \quad \frac{1}{N} QoE_1^N, \qquad l_i(j) \in \mathbf{\Omega}, \tag{1}$$

where QoE_i^N is the QoE of video chunk 1 through N, $l_i(j)$ is the quality level (e.g., QP value) of tile j of chunk i, and $\mathbf{\Omega}$ is the optional set of quality levels. The QoE is constituted by four components, i.e., average quality, temporal quality variation, spatial quality variation, and rebuffering time, defined as follows:

$$\begin{aligned} QoE_1^N = &\sum_{i=1}^{N} q(\boldsymbol{l}_i, \boldsymbol{v}_i) - \mu_1 \times \sum_{i=1}^{N-1} |q(\boldsymbol{l}_{i+1}, \boldsymbol{v}_{i+1}) - q(\boldsymbol{l}_i, \boldsymbol{v}_i)| \\ &- \mu_2 \times \sum_{i=1}^{N} StdDev(\boldsymbol{l}_i, \boldsymbol{v}_i) - \mu_3 \times Rebuf\_Time \end{aligned} \tag{2}$$

Here, $q(\boldsymbol{l}_i, \boldsymbol{v}_i)$ is the quality (e.g., PSNR) of chunk i, $\boldsymbol{v}_i$ is the viewing behavior including head directions and gaze points in chunk i, and μ_s are non-negative weighting parameters. Different from conventional video, the assessment indexes related to quality should be built upon the viewing behaviors of users for 360-degree video streaming. For example, the region except the viewport is unseen for users, thus the quality in which makes no difference on viewer QoE. The specific and detailed video quality assessment (VQA) method for mobile 360-degree video streaming can refer to the prior work [17]. Moreover, the spatial quality variation, i.e., the tile-across quality variation in one chunk, is a unique evaluation metric for tile-based video delivery. Actually, the unevenly spatial quality distribution would also result in chunk-in temporal quality variation due to the changes of viewer behaviors during one chunk, which is not considered here for simplicity.

Let $B_t \in [0, B_{max}]$ be the buffer occupancy (in seconds) at time t, i.e., the length of the video left in the buffer, and $d_{i(t)}(j)$ be the transmission time of tile j of the to-be-downloaded chunk i at time t. Further, let δ_t be the time cost of solving a tile-level quality allocation problem for the to-be-downloaded chunk at time t. Then the player's buffer dynamics can be formulated as:

$$B_{t_{k+1}} = \left(B_{t_k} - \delta_{t_k} - \sum_j d_{i(t_k)}(j)\right)_+ + L, \tag{3}$$

where L is the duration of a chunk, and the notation $(x)_+ = \max\{x, 0\}$ ensures that the buffer occupancy is non-negative. Compared to chunk-level adaptation, the combinatorial space of the tile-level quality allocation problem is far larger, e.g., 5^{24} combos in one chunk for 4×6 tiling and five quality levels. Hence, δ_t is non-negligible for crude solving solutions (e.g., exhaustive search) in tile-level adaptation especially considering the limited computing resources on mobile devices. Note that due to the limitation of the buffer size B_{max}, when the buffer is full, the fetcher in the player waits for the buffer occupancy to be consumed until a new chunk can be appended. The waiting time Δ_{t_k} can be expressd as follows:

$$\Delta_{t_k} = \left(\left(B_{t_k} - \delta_{t_k} - \sum_j d_{i(t_k)}(j)\right)_+ + L - B_{max}\right)_+, \tag{4}$$

4 Overview of RoSal360

Figure 3 illustrates a overview of RoSal360 system that includes the offline saliency map preparation phase and the online saliency-aware video streaming phase.

Saliency map preparation. When a 360-degree video is uploaded, the media server transcodes and dashifies the video into multiple-bitrate chunks, and then splits each chunk into $N_{row} \times N_{col}$ equal-sized tiles, on top of the DASH standard. Next, the *saliency map* (i.e., the $N_{row} \times N_{col}$ numerical matrix) of each chunk is generated offline on the server based on video content and collected viewing data (refers to [17] for more details about the saliency map acquirement). Finally, the saliency maps are packaged in *JSON* format for ease of transmission and deployment.

Saliency-aware video streaming. At the beginning of video streaming, the corresponding saliency maps are downloaded to the client firstly. During the playback, the *tile-size-aware transmission time predictor* (§5.1) estimates the transmission time of video tiles with diverse quality levels in the next chunk. Then, referring to saliency maps, the *saliency-weighted reward calculator* (§5.2) computes the QoE rewards of all quality combos combining the TTP results and the buffer occupancy to decide the quality allocation (i.e., download plan) of video streaming. Simultaneously, the *RL-driven online corrector* (§5.3) utilizes the online viewing data to correct the unfrequent but improper quality allocations caused by saliency bias. According to the download plan, the videos tiles with the corresponding quality levels are fetched, decoded, merged, projected, and displayed by the client player.

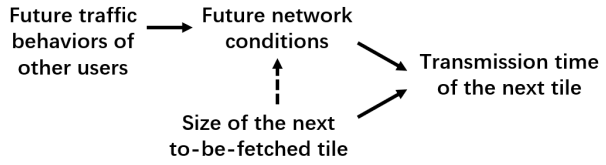


Fig. 4: A causal diagram regarding the tiles' transmission times. The arrows represent the directional causal relationships between two connected variables.

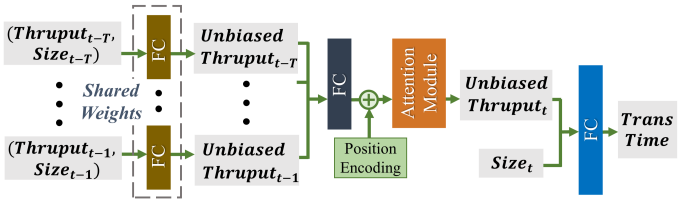


Fig. 5: The basic DNN model architecture of our TTP algorithm. It decouples the prediction of unbiased network tendency and the effects of tile sizes on observed throughputs.

5 Saliency-Driven Quality Adaptation

We propose RoSal360, a robust saliency-driven quality adaptation framework for mobile 360-degree video streaming. We first design a tile-size-aware, attention-based DNN architecture to predict the transmission time of video tiles. Based on the saliency map and TTP results, we model and solve the saliency-weighted reward maximization problem with a transmission time constraint. Further, we design a RL-based online correction algorithm to correct improper quality allocations caused by saliency bias.

5.1 Attention-Based Tile-Size-Aware TTP

Based on the self-attention mechanism in the machine learning domain [22], we design a practical tile-size-aware TTP neural network model that decouples the network condition prediction and the effects of tile sizes on transmission times. Figure 4 shows a causal diagram concerning the related network variables to aid in visualizing how these variables are causally interrelated. The future network conditions and the size of the next to-be-fetched tile jointly determines the transmission time of this tile. Further, the network conditions are influenced by the traffic behaviors of all senders (including ours) that share this identical network link. Given the network resources, all senders compete for bandwidth. If other senders occupy more bandwidth, the actual rate experienced by our sender will be lower and the network condition seems to get worse. Certainly, the behaviors of different senders are interactional especially for elastic flows with fairness-enhanced rate control. It means that our sending rate would affect the others' behaviors and further affect the network condition. However, the influence of our sender on network conditions tends to be insignificant compared to lots of other senders although other users' behaviors are unknowable to us. Therefore, we overlook the effects of tile sizes on dominating network tendency (denoted by the dashed arrow in Fig. 4), and present a novel and practical DNN design by decoupling the effects of network conditions and tile sizes on transmission times.

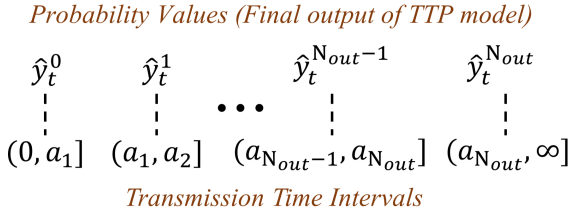


Fig. 6: Final output of our TTP model: an estimated probability distribution of transmission times.

Basic model architecture. Figure 5 depicts the DNN model architecture of our TTP algorithm. We take the T recently historical $(tile\ throughput,\ tile\ size)$ pairs as model input. The tile throughputs observed by users are actually influenced by the sizes of downloaded tiles, that is, these throughput values are *size-biased*. Thus, we firstly uses a fully-connected (FC) neural network layer to counteract the throughput biases due to the different tile sizes of data pairs, and output the unbiased throughputs. The unbiased throughputs can be regarded as the throughputs observed by hypothetically transmitting the same-size tiles. Then, we use a FC layer followed by the attention-based neural network module (detailed later) to predict the future network tendency variation and output the estimation of the unbiased throughput at time t. Finally, we combine the predicted unbiased throughput and the size of the next to-be-fetched tile to predict the size-biased transmission time of the next tile.

Specific algorithms. The first FC layer in our model undertakes the bias correction, where the weights of all FC models are shared although the input pairs are separated to be processed by the corresponding independent FC model. Actually, every FC model in the first layer generates a vector representing the unbiased throughput instead of a single value. Similar as [19], our TTP model finally outputs a probability distribution $\hat{\mathbf{Y}}$ on possible transmission times rather than a single predicted value, illuminated in Fig. 6 Here, $\hat{\mathbf{Y}}_t = [\hat{y}_t^0, \hat{y}_t^1, \ldots, \hat{y}_t^{N_{out}}]$ is the distribution estimation of transmission times at time t, given the tile size. $\hat{y}_t^i$ is the probability value of the transmission time belonging to the corresponding ith interval, which satisfies $\sum_{i=0}^{N_{out}} \hat{y}_t^i = 1$. In practice, we take the median value of an interval as the transmission time this interval stands for, and compute the expectation value of the output distribution as the estimated transmission time:

$$\hat{d}_t = \sum_{i=0}^{N_{out}} \hat{y}_t^i \times \frac{a_i + a_{i+1}}{2}, \tag{5}$$

where $\hat{d}_t$ is the estimation of transmission times, and $a_0 = 0$.

The *attention module* in our model is similar as the encoder stack of Transformer [22], which includes the N_{atten} blocks of multi-head self-attention layers and position-wise FC layers, as shown in Fig. 7. The Transformer is a well-known attention-based DNN architecture in the artificial intelligence (AI) domain due to its revolutionary progresses in various application areas. Given the time series of position-encoded input $\mathbf{X} = [x_1, x_2, \ldots, x_T]$, the queries, keys, and

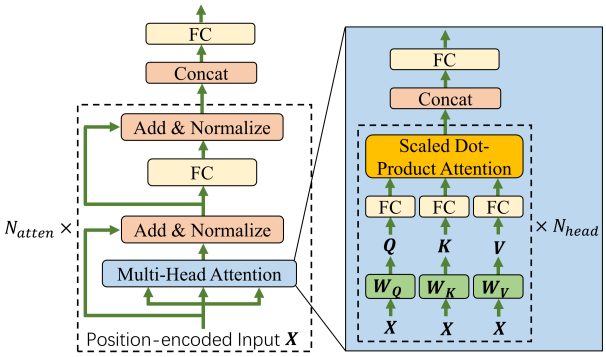


Fig. 7: The attention module of our TTP model.

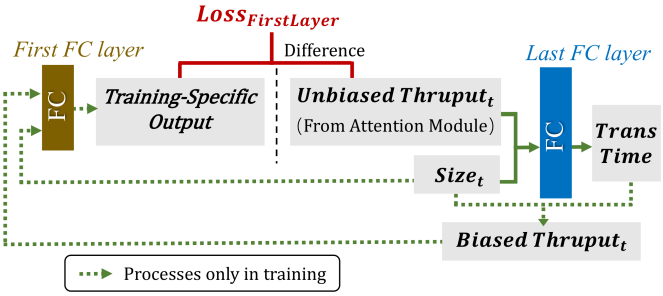


Fig. 8: The second part of the loss function we set in the TTP model training (i.e., $Loss_{FirstLayer}$). The output of the first FC layer is a training-specific ouput for self-supervised bias transformation learning.

values are packed together into matrices $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, respectively, computed as follows:

$$\mathbf{Q} = \mathbf{W_Q}\mathbf{X}^T,\ \mathbf{K} = \mathbf{W_K}\mathbf{X}^T,\ \mathbf{V} = \mathbf{W_V}\mathbf{X}^T, \tag{6}$$

where $\mathbf{W_Q}, \mathbf{W_K}, \mathbf{W_V}$ are learnable weight matrices and $\mathbf{X}^T$ means the transpose of $\mathbf{X}$ (not the time T defined before). Afterwards, the output of the multi-head self-attention layer is computed as follows: (refers to [22] for more details)

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_{key}}}\right)\mathbf{V}, \tag{7}$$

where d_{key} is the dimension of keys (also the dimension of queries). Then, the position-wise FC layer takes the concatenation of outputs from all heads as input and outputs the estimation of the future unbiased throughput.

Model training. We extract two hundred million pairs of real-world network data from Puffer [19] to pre-train our TTP model. Puffer is a free, publicly accessible video streaming service platform that streams commercial television channels through multiple types of network. During the streaming, Puffer constantly records the transmission times of chunks and the corresponding chunk sizes. Based on the pre-trained model weights, we fine-tune our TTP model on a new tile-wise transmission time dataset (over a million pairs of data) that we collected in real network environments. We implemented and deployed the tile-level 360-degree video streaming platforms on five different cloud servers (will be introduced in §6). Multiple clients constantly downloaded the video tiles day and night with a Random ABR algorithm, i.e., the quality level of each tile is randomly selected form the five ascending levels. Different from Puffer in which the collected data are ABR-related due to its evaluation purpose, we adopt a randomly-allocated quality adaptation scheme during the data collection for a larger exploration space and more dissimilar size trajectories of downloaded tiles.

The groundtruth of downloading time is represented as a one-hot vector in which the item corresponding to the time interval the actual downloading time belongs to is one and the others are zero. The loss function in training is constituted by two parts, expressed as follows:

$$Loss_{train} = Loss_{CE} + Loss_{FirstLayer} \tag{8}$$

The first part is the cross-entropy (CE) loss between the estimated transmission time distribution (i.e., the final output of our TTP model) and the discretized actual downloading time (i.e., the groundtruth represented by a one-hot vector). The second part is the loss set for the supervision of bias transformation. After a forward propagation of our TTP model, naturally, we can obtain a model-dependent triad formed by the unbiased throughput, the tile size, and the size-biased throughput at time t, where the biased throughput is the size divided by the corresponding transmission time prediction (generated by the last FC layer) rather than the groundtruth value. We expect the first FC layer in our model to learn to handle the size bias and yield the unbiased throughput, i.e., making biased-to-unbiased transformation, which can be interpreted as the inverse transformation of the last FC layer. Ideally, the transformation (by the first layer) of the transformation (by the last layer) of $UT(t)$ should be $UT(t)$s itself, where $UT(t)$ denotes the unbiased throughput at time t, i.e., the output of the attention module. Therefore, the aforementioned triad provides a group of self-supervised training data for improving the bias transformation of both the first and last FC layer. To this end, as shown in Fig. 8, the second part of the loss function is the difference of $UT(t)$, and the training-specific output from the first FC layer by inputting the predicted biased throughput and the size at time t.

Practical designs. The decoupled design enables the TTP model to efficiently infer the transmission times for a number of tiles at some point in time, with the forward propagation of the neural network only once except the last FC layer. Only the last FC layer repeats network inferences, ingesting the identical unbiased throughput prediction value and the diverse tile sizes to output these tiles' TTP outcomes. It significantly reduces the time cost of quality adaptation per chunk. To further lower the overhead of quality adaptation, we also perform the clustering analysis for the size values of all tiles with different quality levels in each chunk offline, and only infer the transmission time of cluster centers during streaming. Let $\hat{d}_i(j), \hat{d}(cc_i(j))$ be the estimated transmission times of tile j of chunk i and the size center of the cluster that tile j belongs to, respectively. Then, we compute the TTP result of tile j as follows:

$$\hat{d}_i(j) = \hat{d}(cc_i(j)) \times Size_i(j) \,/\, Size(cc_i(j)), \tag{9}$$

which is due to the nearly equal effects of size bias on throughputs with the similar sizes. Moreover, in practice, we

regard the dispersion of the TTP distribution as a measure of the uncertainty degree for future network states. If the dispersion is high, we multiply the final TTP result by a corresponding discounting coefficient (means more conservative TTP) for robustness to network variations.

5.2 Saliency-Weighted Reward Maximization

Based on saliency maps and TTP results, we convert the quality adaptation problem to a constrained, saliency-weighted reward maximization problem. Then we design a fast solution scheme using the simulated annealing optimization method to compute the optimal quality allocation decisions.

Modeling the reward maximization problem. The saliency map implies the probability of gazing at each tile for a new user. Thus, the tiles with larger saliency scores tend to be delivered in higher quality. We also set a minimum buffer threshold, γ, to reduce the risk of rebuffering. Specifically, we convert the quality adaptation problem for the to-be-downloaded chunk i as a constrained, saliency-weighted reward maximization problem as follows:

$$\begin{aligned}\max_{l_i(j)} \quad & reward_i = \sum_j Sal_i(j) \times F\big(l_i(j)\big) \\ & - \lambda_1 \sum_j Sal_i(j) \times Sal_{i-1}(j) \times \big|F\big(l_i(j)\big) - F\big(l_{i-1}(j)\big)\big| \\ & - \lambda_2 \sum_j \sum_{r \in nei(j)} \frac{\big|F\big(l_i(j)\big) - F\big(l_i(r)\big)\big| \times Sal_i(j)}{|nei(j)|}, \\ s.t. \quad & B_{t(i)} - \sum_j d_i(j) > \gamma,\end{aligned} \tag{10}$$

where $Sal_i(j)$ is the saliency score of tile j in the saliency map of chunk i, $nei(j)$ is the indexes of the neighboring tiles of tile j, and λ_s are non-negative weighting parameters.

Here, the first item of the reward we set is the saliency-weighted average video quality of chunk i. $F(\cdot)$ is a function mapping the quality level (e.g., the QP value or zero to five) to an another indicator that is positively related to the actual video quality, e.g., an inversely correlated function of QP values. We used $F(QP) = 60 - 0.8 \times QP$ in RoSal360, which refers to [10], [23], [24]. We used $F(\cdot)$ rather than using the actual video quality $q(\cdot)$ straightforward due to the computing complexity of $q(\cdot)$. The second and third items of the reward represent the chunk i 's saliency-weighted average quality differences between two consecutive chunks, and between neighboring tiles in an identical chunk, respectively. To facilitate computation, we only compute the quality difference of *same-position* tiles of two successive chunks, although the quality difference of *diverse-position* tiles of two successive chunks may influence the QoE due to the cross-tile gaze movement in the junction of two chunks. In practice, computing the diverse-position differences gains little but increases the computation complexity by tens of times. Overall, this setting rewards the increase in saliency-weighted video quality, and penalizes the increase in saliency-weighted quality variance both temporally and spatially.

Saliency-driven quality adaptation design dramatically reduces the reliance on HMT predictions by viewport-free reward computation. Therefore, RoSal360 enables the client player to keep a longish buffer to absorb network variations without worrying about the diminution in HMT's temporal correlation. Inspired by [8], we set a minimum threshold of the buffer occupancy, γ, which means that the buffer occupancy should always exceed γ during video downloading. Specifically, a transmission time constraint is added in Eq.10. If there is no quality combo satisfying the transmission time constraint, RoSal360 will adopt the most conservative quality allocation scheme, i.e., fetching all tiles in the lowest (highest if using QP) quality levels, to expand buffer as best it can.

Fast problem solving using simulated annealing. Huge search space of tile-level adaptation would incur the considerable calculation delay for exhaustive search, which would severely impair adaptation performances. Thus, we adopt two techniques to reduce the search space and speed up

Algorithm 1 Saliency-driven quality allocation with simulated annealing for chunk i

Input: $\boldsymbol{Sal}$, $B_{t(i)}$, $\boldsymbol{d}_i$, $\boldsymbol{l}_{i-1}$, quaLevelOptions, F, λ_1, λ_2, γ
Output: $\boldsymbol{l}_i$

Initialize $\boldsymbol{l}_i$ = the combo with the lowest quality levels, n = 0, maxRew = -10000, stride = 1, strideAdd = 2, strideLim = 2, countAdd = 1, countLim = 1, batch = 100;
while $n <$ len(quaLevelOptions) **do**
  $\boldsymbol{ql}$ = quaLevelOptions[n];
  if $B_{t(i)} - \sum_j d_i(j) \leq \gamma$ **then**
    if countLim % batch == 0 **then**
      strideLim = strideLim $\times$ 2;
    end if
    countLim = countLim + 1;
    stride = strideLim;
  else
    rew = computeReward($\boldsymbol{Sal}$, $\boldsymbol{ql}$, $\boldsymbol{l}_{i-1}$, F, λ_1, λ_2);
    if rew > maxRew **then**
      maxRew = rew; *# update the maximal reward*
      $\boldsymbol{l}_i$ = $\boldsymbol{ql}$; *# update the optimal allocation*
      stride = 1;
    else
      if countAdd % batch == 0 **then**
        strideAdd = strideAdd $\times$ 2;
      end if
      countAdd = countAdd + 1;
      stride = strideAdd;
    end if
  end if
  n = n + stride;
end while
return $\boldsymbol{l}_i$;

	Time cost	Optimal reward
Exhaustive search	3.48s	0.045
Our speed-up	0.04s	0.044

TABLE 1: Our techniques achieve an 87$\times$ computation speed-up to find the near-optimal solution of quality adaptation, compared to the exhaustive search.

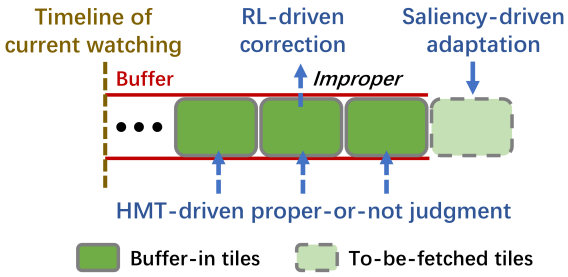


Fig. 9: RL-driven online correction mechanism for robustness to saliency bias.

computation. First, a monotonicity constraint is added that the quality level of each tile never increases (reduces if using QP) as the saliency score drops off. Second, the simulated annealing optimization algorithm [25], [26] is delicately applied to speed up searching for the optimal or suboptimal quality level combo. The simulated annealing method is a probabilistic technique for effectively approximating the global optimum of a given function, which is typically used in discrete but very large configuration spaces, and in the presence of a number of local optima.

The specific algorithm is given in Algorithm 1. Here, *quaLevelOptions* is $N_{row} \cdot N_{col}$-dimension (same as the number of tiles in a chunk) vector space (implemented with multi-dimensional array), enumerating all the optional quality level combos satisfying the monotonicity constraint. The gist of our algorithm is the adaptive stride adjustment for combo searching. After using our searching acceleration techniques, RoSal360 only needs to compute hundreds of paces to find the near-optimal solution (i.e., the quality allocation with the almost maximal reward). Table 1 lists the average time consumption and optimal reward of the exhaustive search scheme and our speed-up methods.

5.3 RL-Driven Online Correction

To improve the QoE of outlier viewers with distinctive preferences, we design a RL-driven online correcting algorithm combining offline saliency information and online viewing data to correct the unfrequent but improper quality allocations caused by saliency bias.

Basic idea. Parallel from saliency-weighted quality allocation (expounded in §5.2) for the next to-be-downloaded chunk, RoSal360 performs the HMT-driven quality allocation algorithm (refer to [10]) for buffer-in chunks (i.e., the chunks that are already downloaded but are not consumed by the viewer yet). If the expected quality level for some buffer-in tile according to the HMT-driven quality allocation decision is far higher than the actual quality level of this tile, this tile would be judged to be with an improper quality allocation. Then the RL agent is responsible for the decision if this buffer-in, improper tile will be finally corrected, i.e., being redownloaded and replaced in high quality, as depicted in Fig. 9. Only the improper tiles will trigger the correction-or-not inference by the RL agent and the proper tiles (account for the larger proportion of tiles, in fact) will not.

An improper quality allocation judged by HMT-driven quality adaptation algorithms will not be necessarily corrected in RoSal360. On one hand, the HMT-driven quality adaptation is not always credible enough despite relatively small prediction horizons (i.e., the size of pw) for buffer-in tiles. On the other hand, redownloading would influence the normal buffer dynamics and consequently increase the risk of rebuffering. Moreover, the redownloaded tiles will be effectless if they fail to arrive before the time they should be displayed. Therefore, outrunning the forecasted transmission time for redownloaded tiles would lead to a overdue-redownloading loss, i.e., bandwidth overhead without any quality gain.

The saliency-driven and HMT-driven quality adaptation modules are running parallelly. However, the downloading processes of the two modules are performed serially rather than parallelly for unambiguous network feedbacks. That is, if the system redownloads some tile for correction, the regular downloading of the next chunk would be delayed until the correction is finished.

The mission of RL agent is to decide whether an improper tile (judged by the HMT-driven quality allocation before RL agent inference) should be finally corrected given the allocation accuracies, current Player states, and network estimations. If a tile is decided to be corrected by the RL agent, RoSal360 will revisit the HMT-driven quality decision and replace this tile using the quality level of the HMT-driven decision (with the same tile position).

Inputs of the RL agent. As shown in Fig. 10, the state inputs of the corrector agent (i.e., a DNN model) include the accuracy of saliency-based behavior estimations, the accuracy of HMT-driven viewport predictions, the buffer occupancy, the time left before playing for the tile, the tile's TTP result, and the TTP confidence. During the playback, RoSal360 constantly tracks the head directions of viewers on-the-fly, and computes the overlapping rate of the actual viewport and the top-four tiles (ranked in order of saliency scores; for 4×6 tiling), which is regarded as the accuracy of saliency-based behavior estimations. The accuracy of HMT-driven viewport predictions is sensitive to the size of pw. Hence, for a tile with an improper quality allocation judged by HMT-driven quality adaptation, RoSal360 samples five recently historical accuracies of viewport predictions with the same pw as this tile, and regards the mean value as the final accuracy estimation of the HMT-driven viewport prediction for this tile. Moreover, the online correction is always executed after transmission time prediction (§5.1) for more robust correction decisions. After computing the TTP result (expressed in Eq. (5)), we revisit the raw probability distribution generated by our TTP model, and regard the corresponding probability value of the transmission time interval that the TTP result belongs to as the TTP confidence. The TTP confidence measures the credibility of TTP results.

Specific algorithms and training. We use the policy-based RL architecture, in which the RL agent makes decisions based on a policy network $\pi(\boldsymbol{s}_t, a_t)$. π maps the state vector $\boldsymbol{s}_t$ and the action a_t to the probability that a_t is taken at state $\boldsymbol{s}_t$. The specific model we use is a multi-layer perception (MLP) network (the most popular DNN architecture), with two hidden layers of 64 and 32 neurons, respectively, and ReLU activation functions. After the final softmax function, the model outputs a two-dimension vector, representing the probabilities of whether a buffer-in tile

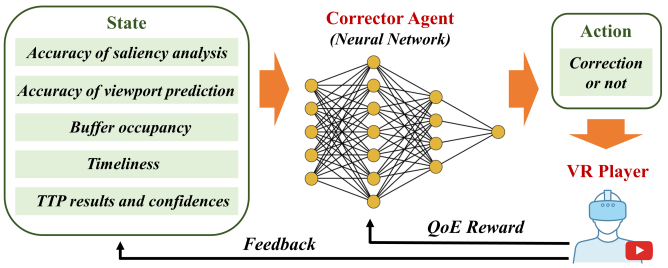


Fig. 10: RL architecture of the tile corrector.

Fig. 11: A snapshot of RoSal360 implementation.

will be redownloaded. We use the policy gradient method [27] to train the above DNN model due to its simplicity and easy implementation.

Let $\boldsymbol{\theta}$ denote the policy parameters and $J(\pi_{\boldsymbol{\theta}})$ denote the expected return of the policy. The gradient of $J(\pi_{\boldsymbol{\theta}})$ can be expressed:

$$\nabla_{\boldsymbol{\theta}} J(\pi_{\boldsymbol{\theta}}) = \mathop{\mathrm{E}}_{\tau \sim \pi_{\boldsymbol{\theta}}} \left[\sum_t \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}} \left(\boldsymbol{s}_t, a_t \right) \left(QoE(\tau) - b \right) \right], \tag{11}$$

where τ is a trajectory of correction decisions over $\pi_{\boldsymbol{\theta}}$, QoE is the training reward similar as the metric defined in §3, and b is a baseline parameter. Then the policy parameters can be updated via stochastic gradient ascent on the policy performance:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \nabla_{\boldsymbol{\theta}} J \left(\pi_{\boldsymbol{\theta}_k} \right), \tag{12}$$

where α is the learning rate that decays from an initial value as the training epoch increases.

The quality allocation model we use can be regarded as a configuration parameter of the RL environment. For example, the buffer occupancy, as a part of RL agent input, is influenced by the underlying quality allocation model. Therefore, we keep the parameters of both saliency-driven and HMT-driven quality adaptation models fixed during the training of the RL agent for a specific training. In other words, the RL agent in RoSal360 learns to make correction decisions based on the dynamic states of user behaviors and network environments, given the specific quality control model.

6 Implementation

We implemented a prototype of RoSal360, client running on a Mini PC, based on 11,300 additional lines of JavaScript and Python code, on top of the HEVC Tiles Merger [28]. The Mini PC, Dell OptiPlex 3080MFF, is equipped with a mobile-class Intel UHD Graphics 610 GPU, an Intel Pentium G6405T 1.8 GHz dual-core processor, 4 GB of RAM, and a 128 GB SSD. It is equivalent to a common commodity smartphone in computing resources. For instance, the Samsung Galaxy S20 phone [29] is equipped with a Qualcomm Adreno 650 GPU, a Qualcomm Snapdragon 865 1.8-2.84 GHz multi-core processor, 8 GB of RAM, and a 128 GB SSD, which is similar to our Mini PC settings. Thus, we elaborately chose the Mini PC to emulate the computing resources of a mobile VR device.

We used FFmpeg [30] and Kvazaar [31] to encode videos in High Efficiency Video Coding (HEVC) formats due to native support for tiled coding [32], [33], and used MP4Box [34] to package and dashify HEVC bitstreams. The video server was a web server over HTTP built in Ubuntu 20.04. We constructed an HTML5-based, DASH-compatible VR Player to stream and display HEVC-tiled 360-degree videos on the Microsoft Edge browser on top of the HEVC Tiles Merger [28]. The Player fetches and merges split tiles with different qualities according to a controllable *quality matrix*. To facilitate online correction, we set a JavaScript *Array* before the buffer queue in the Player. The downloaded video files are placed in the Array firstly and are deferred to enter the buffer queue. The videos in the Array maintain the independence of tiles without merging or decoding, which means that any tile replacement in the Array does not impact other tiles. For user studies on untethered mobile VR headsets, we used *A-Frame* [35], a web framework for building VR experiences, to render and project 360-degree video streams from the PC Player into the mobile VR headset (i.e., Oculus Quest 2 [2]) through the native casting software. Simultaneously, we acquired the online viewing data by *VRFrameData.pose* in *A-Frame* and sent the viewing data to the online corrector. The DNN structures in this work were constructed using the Pytorch repository.

Figure 11 shows a snapshot of our prototype implementation. We adopted the PC-based implementation approach for our client player, largely due to abundant support of the well-developed, compatible software stack and open-source tools on the Windows platform, e.g., browser support for Media Source Extensions (MSE) API and hardware HEVC decoding. A similar prototype implementation scheme was adopted by the previous work [36]. That being said, lack of an implementation totally built on commodity mobile devices adds a complication to the practical deployment and application in industry. We further discuss this limitation in Section 8.

7 Evaluation

In this section, we evaluate the performance of RoSal360 with real prototype implementation over commodity WiFi, 4G/LTE, and 5G wireless networks in the wild. Large-scale evaluations through both gaze-annotated 360-degree video datasets and a survey-based user study demonstrate that RoSal360 achieves the higher video quality, the lower rebuffering ratio, the lower quality variation, and the higher QoE score, compared to the state-of-the-art approaches. We also illuminate the reasons of several key algorithm designs in RoSal360 by a component-wise experimental analysis.

	Bandwidth	Round-trip time (RTT)
Campus WiFi	1-4 Mbps	15-50 ms
Commodity 4G/LTE	0.5-3 Mbps	45-75 ms
Commodity 5G	0.5-6 Mbps	40-60 ms

TABLE 2: End-to-end bandwidth and RTT of different wireless networks the system actually undergoes from our measurements, in our evaluations.

7.1 Gaze Data Collection

Gaze data have been becoming indispensable annotations for precise 360-degree video quality assessment (VQA) [17], [37]. Hence, we built a gaze-annotated long 360-degree video dataset [1] due to the scarcity of gaze datasets with long-duration videos. We downloaded 23 long 360-degree videos (2-11 minutes, 11 genres, 4K resolution, 30-60 fps, and ∼2 hours in total) from YouTube according to the popularity ranking. The genres include aerial, animal, cartoon, dance, diving, game, mixed, racing, roller coaster, scenery, and outer space. Then, 30 participants viewed these videos using the HTC VIVE PRO EYE VR headset with built-in Tobii's gaze tracking [38]. The participants, aging from 20 to 51, are students, staff, and faculty members from two universities and a company. 47% of them are female, 57% of them wear glasses, and 63% of them are first-time viewers for 360-degree videos. The gaze data of each participant were recorded by the Tobii Pro Lab software [39]. VIVE Wireless Adapter [40] enabled viewers to *explore freely without wired hindrance*. Any participant that felt dizzy or other discomfort discontinued watching and the corresponding data were discarded. The participants experienced the uniformly high video quality to eliminate the effect of spatially-uneven quality distribution on viewing behaviors.

7.2 Experimental Setup

We state the experiment settings in our evaluation.

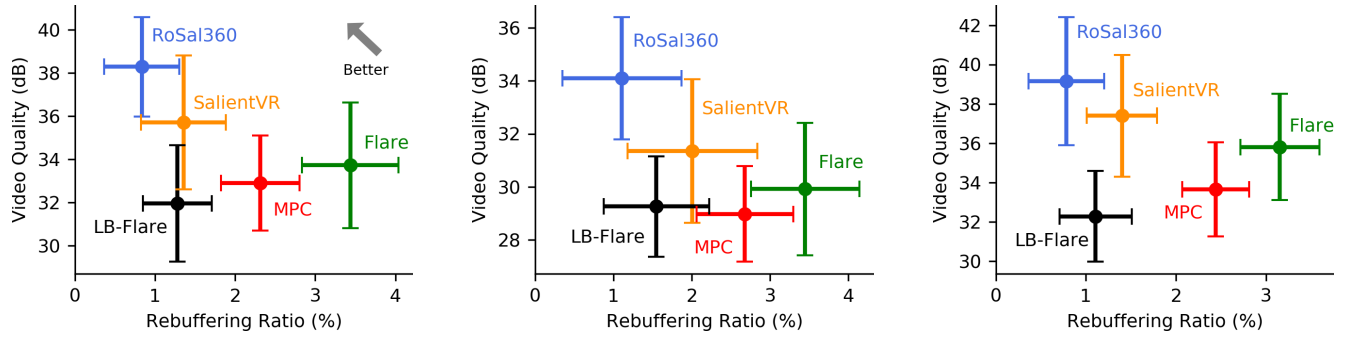
Videos. We used totally four-hour 360-degree videos annotated with head directions and gaze points in our evaluations, including 80 short videos (20-60 seconds) in the Gaze18 dataset [41] and 23 long videos (2-11 minutes) in the dataset we built (§7.1). The collected viewing data are serially ingested by the client player to simulate the viewing process. The videos exhibit a large diversity in terms of genres, and were viewed on commodity VR headsets with free, untethered, and immersive viewer experiences. There are both videos shot from fixed cameras and videos captured with a moving camera, which probably introduces different variances in head/gaze trajectories of viewers. Further, some videos possess relatively more complex content scenes (e.g., numerous foreground objects in a frame) while the scenes in some videos are simple, which would influence the difficulty of saliency analysis. Each video was encoded into 2.13-second chunks with 4×6 tiling and five quality levels (QP=22,27,32,37,42).

Network conditions. We used five remote cloud servers in different cities as video servers through various wireless network links to evaluate RoSal360 in the wild. The real wireless network environments include a campus WiFi link, a commodity 4G/LTE link, and a commodity 5G link. To facilitate large-scale network experiments, we used a cellular network CPE (Customer Premise Equipment) to convert 4G/LTE and 5G signals into WiFi signals. We ran the video streaming system during different time periods, e.g., peak hours and non-peak hours. Table 2 lists the end-to-end bandwidth and round-trip time (RTT) of wireless networks the system actually undergoes from our measurements. We simultaneously ran multiple client players with RoSal360 and baselines (elaborated later), and downloaded videos from the same video server to keep the network condition as equivalent as possible for different algorithms. Note that the popular evaluation scheme of trace-driven network emulations fails to fully emulate the effects of tile sizes on actual transmission times in the wild, which thus was not adopted by us.

Algorithms for comparison. We compared RoSal360 with SalientVR [17], Flare [10], MPC [42] and long-buffered Flare (LB-Flare). SalientVR is a saliency-driven 360-degree video streaming system with a crude TTP scheme and a heuristic online correction mechanism. We compared our work with SalientVR to validate the positive effects of the DNN-based TTP algorithm (§5.1) and the RL-based correction algorithm (§5.3) in RoSal360. Flare is a typical HMT-driven 360-degree video delivery method. LB-Flare, a variant of Flare, expands the buffer size from three seconds to five seconds and tends to keep a longer buffer than Flare. By comparison to Flare and LB-Flare, we illustrate that our saliency-driven quality allocation algorithm (§5.2) achieves a better trade-off between video quality and rebuffering, over the pure HMT-driven approaches. MPC is a control-theoretic adaptation approach designed for non-tiled video streaming. In essence, 360-degree video streaming is a kind of special video streaming. Despite tile-based delivery, allocating the same bitrate to all tiles is technically feasible. Although MPC is designed for general videos without specifically considering 360-degree videos, MPC has potential to be applied to 360-degree video adaptation due to its acknowledged advantages in the trade-off between video quality and rebuffering. Therefore, we aslo compared RoSal360 with MPC for a solid evaluation.

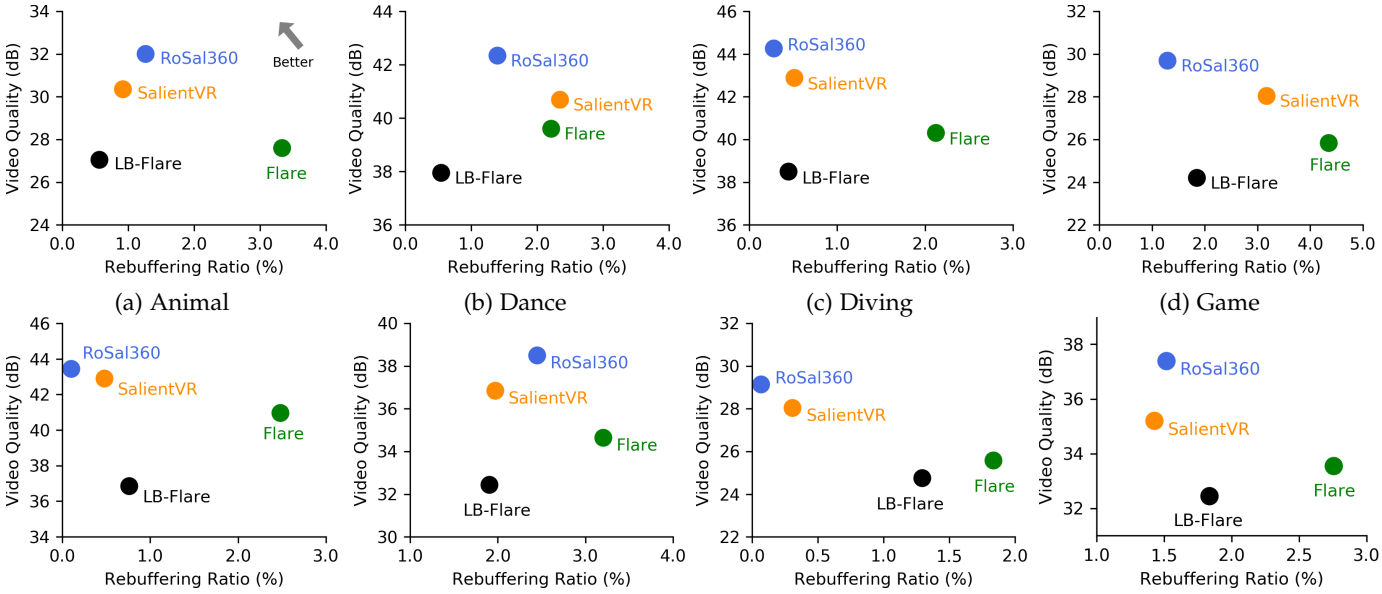
Metrics and settings. We mainly used the gaze-driven PSNR [17] and the rebuffering ratio (i.e., the ratio between the total rebuffering duration and total watching time) to assess the algorithm performances. The gaze-driven PSNR is defined as $\epsilon \cdot PSNR(GazeRegion) + (1-\epsilon) \cdot PSNR(Viewport)$, where the gaze region is the $25°$-radius gaze-centric circular region, and $\epsilon \in (0,1)$ is the weight coefficient. We set $\epsilon = 0.7$ in this work. The survey-based rating by a user study, video quality variations, and viewport-driven PSNR were also discussed. In RoSal360, we adopted the 4×6 tile segmentation for video delivery, i.e., $N_{row} = 4$ and $N_{col} = 6$; we adopted the model configuration with $N_{atten} = 1$, $N_{head} = 3$, $N_{out} = 20$, and $a_{N_{out}} = 2s$; we empirically set $\lambda_1 = 0.1$, $\lambda_2 = 0.3$, and $\gamma = 2.5$ in the quality adaptation module. For other algorithms, we used the same parameter settings as their papers.

1. The raw 360-degree videos, gaze dataset, and the corresponding saliency maps are publicly available at https://github.com/salientVR/gazedata.



(a) Campus WiFi link. (b) Commodity 4G/LTE link. (c) Commodity 5G link.

Fig. 12: Overall evaluation results in video quality and rebuffering ratio for different mobile 360-degree video streaming systems over multiple real wireless network links in the wild. (Error bars show 90% confidence intervals.)



(a) Animal (b) Dance (c) Diving (d) Game

(e) Mixed (f) Racing (g) Roller coaster (h) Scenery

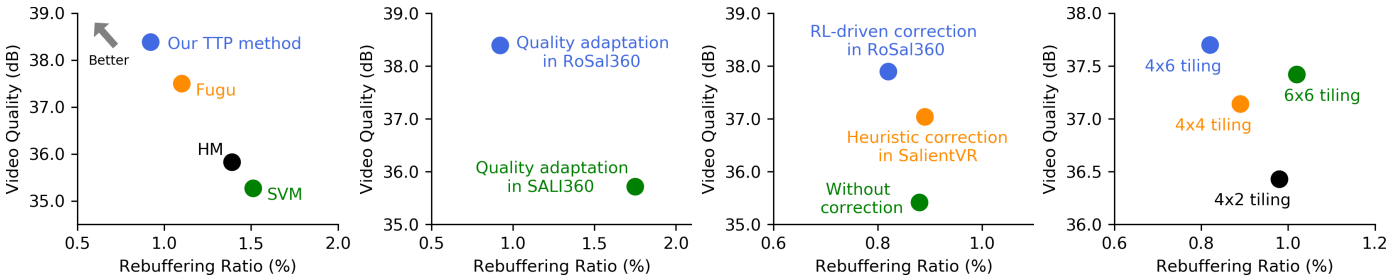
Fig. 13: Separate evaluation results in video quality and rebuffering ratio across eight typical video genres.

7.3 Improvement over Real Wireless Network Links

Figure 12a, Figure 12b, and Figure 12c shows the overall evaluation results of RoSal360 and other alternative algorithms over real-world WiFi, 4G/LTE, and 5G links in the wild, respectively. For WiFi results, RoSal360 achieves a 2.59-6.33 dB video quality improvement and reduces the rebuffering ratio by 1.53-4.12×, compared to existing approaches. For commodity LTE results, RoSal360 achieves a 2.75-5.12 dB video quality improvement and reduces the rebuffering ratio by 1.4-3.11×. For commodity 5G results, compared to alternatives, RoSal360 achieves a 1.76-6.88 dB video quality improvement and reduces the rebuffering ratio by 1.41-4.02×.

For Flare, the LR-based HMT predictions are not accurate enough to support the video delivery with high perceived quality even keeping a short *pw*. Further, the small buffer size in the client player for the HMT-driven solutions brings more rebuffering under the fluctuant wireless network bandwidth. In contrast, SalientVR outperforms the HMT-driven approaches by virtue of more accurate attention estimations and a larger buffer size to absorb network variations. However, SalientVR is adversely affected by the non-robust saliency bias correction and inaccurate network estimations, and consequently fails to tap the full potential of saliency, resulting in inferior performances over RoSal360. LB-Flare attempts to reduce rebuffering by simply enlarging the buffer size, but suffers from the distinct quality degradation due to the ineffective HMT prediction.

Figure 13 depicts the separate evaluation results of different approaches across eight typical video genres to explain the generalizability of RoSal360. We see that RoSal360 generally performs better in terms of the trade-off between video quality and rebuffering than the state-of-the-art algorithms, by the more robust saliency correction and the more accurate transmission time prediction. That being said, the gains of RoSal360 vary across videos with different genres. For the videos with the relatively *focused* region of interest (ROI) such as the genre of *mixed*, RoSal360 gains less in quality enhancement than the videos with the *dispersed* ROI such as the *dance*, due to more unfrequent saliency bias or HMT prediction errors.



(a) Transmi. time prediction. (b) Quality adaptation. (c) Online correction. (d) Tile segmentation.

Fig. 14: Microbenchmarks: component-wise algorithm analysis for end-to-end system improvement in video quality and rebuffering ratio.

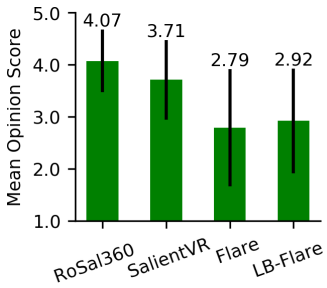


Fig. 15: Survey-based rating (i.e., MOS) by a user study.

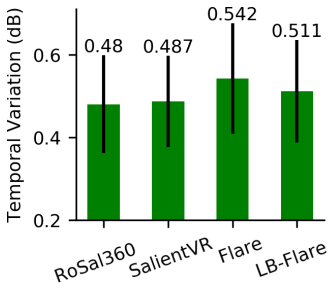


Fig. 16: Temporal video quality variation.

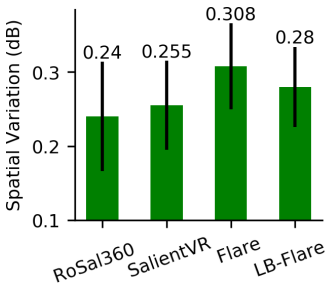


Fig. 17: Spatial video quality variation.

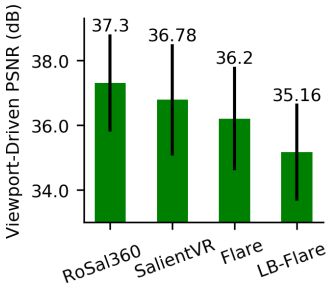


Fig. 18: User Viewport-driven PSNR values.

	MAE ↓	RMSE ↓	MAPE ↓	C-E loss ↓
HM [42]	0.315	0.876	35.7%	12.387
SVM [43]	0.436	0.436	49.2%	7.258
Fugu [19]	0.239	0.358	29.6%	0.556
Ours	**0.172**	**0.242**	**16.8%**	**0.283**

TABLE 3: A deep comparison of transmission time estimation with the state-of-the-art algorithms on multiple error-based evaluation metrics. ↓ means the smaller value is better.

7.4 Microbenchmarks

In this study, we altered some key components in RoSal360 one-by-one to specify the reasons of these settings and better understand their contributions to end-to-end QoE improvements for mobile 360-degree video streaming.

Effects of transmission time estimation. We tried different TTP (i.e., transmission time prediction) algorithms, including HM, SVM, Fugu, and ours (introduced in §5.1), in the mobile 360-degree video streaming system to verify the effects of the TTP module. The harmonic mean scheme (HM) uses the harmonic mean value of past several throughput samples to predict the future bandwidth, and takes the ratio of chunk sizes to the estimated bandwidth as the TTP results, which is widely adopted by previous adaptation methods [10], [17], [42]. Raca et al. [43] utilizes the support vector machine (SVM) to predict the network bandwidth, with the same way as HM to compute TTP results. Fugu [19] designs a simple fully-connected DNN scheme for conventional non-360-degree videos to predict the transmission time of chunks directly. As shown in Fig 14a, our attention-based tile-size-aware TTP algorithm with a decoupled DNN model design significantly boosts the reduction of playback rebuffering by the more robust and accurate TTP over alternatives. Table 3 also lists multiple common error-based quantitative indexes to evaluate the performances of different TTP methods more deeply. We computed the mean absolute error (MAE), the root mean square error (RMSE), the mean absolute percentage error (MAPE), and the cross entropy loss (C-E loss) for different approaches. We see that our TTP approach achieves the lowest prediction errors on different metrics compared to the state-of-the-art schemes.

Effects of our quality adaptation algorithm. Figure 14b compares the formulation-driven saliency-aware quality adaptation algorithm (mentioned in §5.2) designed in RoSal360 with another existing saliency-based quality adaptation method used in SALI360 [16]. SALI360 also notices the potential of saliency in 360-degree video streaming; however, it mainly addresses the issues about cube map encoding and only uses a crude quality allocation scheme. For fairness, we used the identical saliency maps, and only changed the pure quality adaptation methods, i.e., how to utilize the given saliency maps to control video fetching. We observe that the quality allocation strategy in SALI360 fails to fully leverage saliency information to balance the trade-off between video quality and rebuffering, leading to the inferior results over RoSal360. By contrast, the quality adaptation algorithm in RoSal360 performs better by the sophisticated formulation, the robust buffer management, and the simulated-annealing solving acceleration, which demonstrates the superiority of our quality allocation approach *per se* on developing the advantage of saliency.

Effects of online correction. To evaluate the effects of online correction, we chose 15 real viewing trajectories of viewers that preferred to gaze at low-salient regions classified by

collectively-determined saliency inference. Figure 14c compares the RL-driven online correction mechanism (§5.3) with the heuristic correction method proposed by SalientVR [17]. We see that the online correction mechanism effectively enhances the video quality by replacing the improper quality allocations (i.e., low-quality tiles) caused by saliency bias for outlier viewers. Moreover, RoSal360 achieves the more robust correction than SalientVR by the joint awareness of both behavior prediction accuracies and network estimation confidences.

We also found that the correction mechanism increases the possibility of early awareness of network congestion, leading to rebuffering reduction compared to the system without correction. In RoSal360, the updating of estimated network conditions is triggered after the downloading of all tiles in the next chunk or the downloading of all to-be-corrected tiles. The number of to-be-corrected tiles tends to be largely lower than the number of tiles in a chunk (i.e., 4×6=24). Therefore, the correction mechanism increases the frequency of network condition estimating, which lowers the adverse impact of unpredictable network degradation.

Different tile segmentation methods. Figure 14d compares different tile segmentation methods (i.e., 2×4, 4×4, 4×6, and 6×6) in RoSal360. The 4×6 tiling method gains the highest video quality and the lowest rebuffering ratio. A more fine-grained 6×6 tiling method increases the flexibility of quality adaptation but limits the network utilization. The transmission time of video tiles does not scale linearly with tile sizes. Smaller tile sizes lead to lower throughput rates due to the slow-start-restart behavior in the TCP protocol [19], [44]. More fine-grained tile splitting also increases the time consumption of quality adaptation due to the larger search space. Compared to the 4×6 tiling method, the 6×6 tiling method requires a $4\times$ more solving time of the optimization problem (Eq.10). Overall, the 4×6 tiling method makes a better tradeoff among the flexibility of quality adaptation, network utilization, and the time consumption of solving quality allocation problems.

7.5 Improvement in Other Evaluation Metrics

Besides the gaze-driven PSNR and the rebuffering ratio demonstrated above, there are some other QoE assessment metrics that are worthy of measurement and discussion, and therefore, we further quantify the QoE performances on these metrics with different algorithms.

Survey-based rating by a user study. 30 participants viewed 10 videos by mobile VR headsets [2] to evaluate the subjective QoE performances of different algorithms. The users' demographics are similar as illustrated in §7.1 while the participants are diverse for research validity. After watching a video, the viewer was required to rate the experience between 1∼5, i.e., the mean opinion score (MOS), which is a user rating measure used in the domain of QoE [11]. We randomly picked real gaze trajectories from the viewing dataset we collected (§7.1) and the Gaze18 dataset [41], and marked the gaze point on each video frame. The viewers were asked to gaze at the markers to ensure the same viewing trajectory for the same video across different algorithms and viewers. This method of user study was adopted by previous works [11], [17].

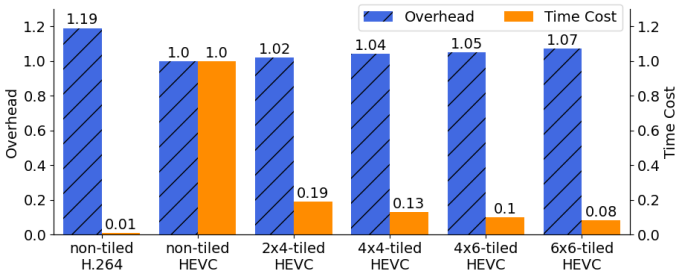


Fig. 19: Compression overhead and time cost of different encoding methods. x264 [46], Kvazaar [31] encoders are used for H.264 and HEVC encoding, respectively.

As shown in Figure 15, RoSal360 gains a significantly higher user rating, with a 31.45% MOS improvement on average, compared to alternatives. The user feedbacks indicate that the rebuffering is more annoying than the quality degradation for the majority of participants when experiencing mobile 360-degree video streaming. For RoSal360, the remarkable decrease of rebuffering without quality degradation significantly boosts the viewer QoE, compared to the state-of-the-art methods.

Quality variations. Figure 16 and Figure 17 compare the temporal and spatial video quality variations (the lower the better) of different algorithms, respectively, which are specifically defined in Section 3. The temporal quality variation quantifies the quality changes between neighboring chunks during the video playback. The spatial quality variation quantifies the quality difference across tiles inside the user viewport, which is a unique metric for spatially quality-uneven video delivery. RoSal360 achieves lower quality variations (both temporal and spatial) than other tile-based algorithms. Moreover, RoSal360 can further reduce the quality variations by tuning the penalty coefficients λ_1 and λ_2 in Eq. (10), although some researchers claim that the quality variations would incur limited QoE degradation [10], [45].

Viewport-driven PSNR. We also evaluated different mobile 360-degree video streaming algorithms using the conventional viewport-driven PSNR (i.e., the PSNR value of the entire user viewport), although it is less precise than the gaze-driven PSNR. As demonstrated in Figure 18, RoSal360 likewise achieves positive gains for the viewport-driven PSNR, compared to other quality adaptation schemes.

7.6 System Overhead

We present a part of system overhead in RoSal360.

Tiled encoding. We used a Ubuntu Server with NVIDIA TITAN X GPU to encode the 360-degree videos. Figure 19 depicts the average compression overhead and time cost for encoding a 2.13-second video chunk with different tiling methods. We measure the compression overhead using the chunk size in units of the size of baseline (i.e., a 2.13-second, non-tiled, HEVC-encoded chunk). The time cost is also expressed in units of the baseline consumption for a clearer comparison. Compared to the non-tiled HEVC encoding, the HEVC-encoded 4×6 tiling increases the compression overhead by 5% only, but reduces the encoding time by $10\times$ by tile-level parallel encoding.

Initial downloading of saliency maps. We present and deliver chunk-level video saliency maps in JSON formats. Although the exact size of the saliency-map profile file for a video depends on the number of chunks in this video and the method of tile segmentation, the compressed saliency map file for initialization only occupies less than 0.5 KB for a chunk with 4×6 tile splitting. Therefore, the traffic cost and time assumption of downloading saliency maps at the initial phase are negligible compared to downloading a video chunk.

Downloading of model weights. The weights sizes of TTP model and correction model are 73 KB and 2.2 KB, respectively. Thus, the overhead in term of weights file downloading is negligible. Moreover, the model weights do not need to be downloaded just before a video is watched in most cases. The weights only should be downloaded or updated when the client application is downloaded or updated, or the video list on the streaming website is being loaded parallelly. Therefore, the weights downloading would not increase the start-off delay for video watching in most cases.

Client-side overhead. We used a weak Dell Mini PC (§6) as the client. The time consumption per inference of the TTP neural network model in RoSal360 is about 30 ms with the deployment on weak CPUs, by virtue of a decoupled model architecture design (§5.1). The RL model for online correction (§5.3) is built with two lightweight layers of 64 and 32 neurons, which spends about 5 ms per inference. The quality adapter averagely costs 110 ms per quality allocation of one chunk, which is negligible compared to the transmission time of one chunk. RoSal360 applies the monotonicity constraint and the simulated-annealing optimization algorithm to remarkably reduce the computation complexity and time consumption of quality adaptation. The HTML5-based Player averagely consumes 16% CPU usage, 210 MB memory, and 19.4 ms duration per decoding and tile merging of one chunk.

We also tested the core part of our adaptation algorithms on a commodity Android phone with Qualcomm Snapdragon 860 SoC and 6GB of RAM. The quality adapter costs nearly 180 ms per quality allocation of one chunk on the Android phone, which is acceptable in video streaming applications. Note that the research issues considered in this manuscript is orthogonal to the underlying operating system. Moreover, the majority of the online parts of our algorithms could run on the resource-rich cloud server instead of the end device, with a low communication overhead. It means that the overhead of our algorithms running on mobile devices is not intractable in real-world applications. Therefore, the device-specific challenges are not our main concern. We further discuss this issue in Section 8.

8 Limitation and Discussion

In this section, we present the limitations of RoSal360 as well as a related discussion.

Effect of quality distribution on viewer attention. We observed the effect of spatially uneven video quality distribution on viewer attention. For example, more viewers gaze at the high-quality region than the low-quality region though both regions are deemed high-salient. That is, the quality allocation scheme would unintentionally lead viewer's attention to tiles with allocated high quality. In our future work, we will study this effect which is a common issue faced by quality-uneven video streaming. Note that this effect does not impair our algorithm evaluations because the viewing behavior data used in our experiments were collected with the uniformly high-quality viewing.

Different gains across videos. We found that RoSal360 gains less for saliency-uniform 360-degree videos compared to saliency-uneven videos. For saliency-uniform videos, the saliency scores in different regions are almost same. Therefore, the saliency map would fail to provide meaningful prior information for estimating the new viewers' attention distribution. Fortunately, in light of our observations and experiments, most of 360-degree videos are saliency-uneven to some extent and significantly benefit from RoSal360.

Advanced wireless network. More advanced wireless network (e.g., 5G) would relieve the bandwidth pressure. However, the insufficient and variable bandwidth in real-world wireless network environments remains a challenge for mobile 360-degree video streaming. First, the advanced communication technology such as 5G still suffers from the high power consumption, small coverage range, high susceptibility to blockage, or cybersecurity issue [47]. As far as we know, 4G/LTE and even cellular networks beneath 4G/LTE have not been replaced completely so far. So, the limited bandwidth still challenges the high-quality 360-degree video delivery in some cases. Second, the transmission throughput the user actually experiences is limited by the bottleneck bandwidth in the end-to-end link and the number of the users share this identical link. So, when the link bottleneck is not the last hop or the user number is large, the transmission throughput is probably insufficient. Third, even if the high throughput is technically supported in the future, the internet service provider (ISP) is likely to not allow high download bandwidth for everyone due to the high bandwidth cost. It is obvious that the video transmission in high quality for every direction of spherical views is inefficient and wasteful for network resources. Finally, the inherent volatility trouble of wireless networks has not been fully solved. Consistent (not only average) high throughput and low latency are expected but have been not fulfilled up to now. In sum, the inadequate and unstable wireless bandwidth still challenges the 360-degree video streaming in many cases.

Lack of system implementation on mobile devices. We acknowledge that operating systems (OS) are different between the Mini PC we used and some of all-in-one VR headsets such as Oculus Quest 2. However, we believe that the algorithm evaluations in our work are reasonable and valid enough. First, a large number of previous works adopt the similar evaluation method as ours with the Windows-based implementation, e.g., [17], [36], [48], [49].

Second, the research issues considered in this manuscript is orthogonal to the underlying OS choice. Our work mainly focuses on the tile-level transmission time prediction and online saliency bias correction. The cornerstone of our algorithms in terms of practicality, e.g., the feasibility of tile-based 360-degree video delivery, decoding, merging, and rendering, has been fully constructed and verified on community mobile devices such as Android-based smartphones

by prior works [10]. Thus, the device-specific challenges of tile-level 360-degree video streaming are not the main concern in this manuscript. While the algorithm performances may vary across different OS, our algorithm would be superior to the state-of-the-art approaches on most end devices due to the algorithm design upon the assumption that the computing resource is limited. Therefore, our implementation based on the Mini PC is valid enough to prove the advantage of our algorithms. Note that our algorithms can also be applied to Windows-based VR devices such as Value Index, which are able to stream videos through wireless networks.

Third, in fact, the majority of the online parts of our algorithms such as the DNN-based network estimation could run on the resource-rich cloud server instead of the end device, with only a low overhead concerning the command communication (less than 1 KB per chunk), e.g., using the remote procedure call (RPC). It means that the overhead of our algorithms running on mobile devices is not intractable in real-world applications. Therefore, the device-specific challenges for RoSal360 are not urgent.

That being said, we also tested the core part of our adaptation algorithms on a commodity Android phone with Qualcomm Snapdragon 860 SoC and 6GB of RAM. The corresponding overhead (§7.6 is still acceptable for video streaming applications.

9 Related Work

Most of past works are built upon LR-based [10], [11], [12], [13], [14] or DNN-based [41], [49], [50] HMT predictions, limited by the temporal correlation assumption. Some studies also explore the opportunities of cross-user similarity [51], [52], [53], scalable video coding [54], [55], [56], content-aware saliency analysis [17], [57], personalized field-of-views (FoVs) [58], super resolution [59], and hybrid schemes [48], [52], [60], [61]. By contrast, RoSal360 combines the advantages of saliency analysis and HMT predictions, and introduces the new studies and designs for mobile 360-degree video streaming. The main differences between RoSal360 and other saliency-based 360-degree video streaming approaches are described below.

SALI360 [16] proposes a cubemap-based 360-degree video compression method over saliency; it does not really consider the dynamic, DASH-based quality adaptation problem over wireless networks. Xu *et al.* [52] and Li *et al.* [60] propose multiple viewport prediction approaches using the HMT, cross-user similarity, heatmaps, and saliency, but do not discuss how to use their prediction results for quality adaptation of 360-degree video streaming. By contrast, RoSal360 focuses on the DASH-based, tile-level quality adaptation problem, and designs an integrated saliency-driven adaptation algorithm including the tile-size-aware network estimation, saliency-aware quality allocation, and HMT-assisted error correction.

Nguyen *et al* [62] propose a head movement prediction method for 360-degree videos merging the saliency information and head orientation data with an LSTM-based DNN architecture. However, they do not explore the dynamic quality allocation problem given the HMT prediction results for tile-based video streaming. By contrast, the focus of RoSal360 is the streaming quality adaptation problem, i.e., the trade-off between video quality and rebuffering. Thus, we aim at the accurate network prediction and the practical tile-level bitrate allocation, which are not considered by [62]. In addition, the practicality of [62] in 360-degree video streaming would suffer from the short prediction window (set to be 0.5 seconds) due to the temporal correlation limitation, which leads to a shallow playback buffer and then the high risk of rebuffering. By contrast, RoSal360 fully leverages the resilience of saliency data to prediction window sizes, with the better practicality and robustness for streaming in wireless networks. Unlike [62], RoSal360 decouples the using of offline saliency information and online HMT data by an online correction method, which overcomes the short buffer limitation while improves the robustness to saliency bias.

Lee *et al.* [57] experiment the saliency-driven rate adaptation using a motion-constrained tile set technique. Shen *et al.* [63] integrate saliency information into the quality adaptation scheme based on the Lyapunov optimization. SalientVR [17] formulates the saliency-aware quality adaptation problem and solves the problem using the simulated annealing method. By contrast, RoSal360 utilizes a novel tile-size-aware TTP algorithm to significantly improves the accuracy and robustness of network capacity estimations, which is the cornerstone of quality adapter. SalientVR [17] and Jiang *et al.* [64] both correct the improper tiles by HMT-driven updating judgment. They ignore the inaccuracy of HMT predictions *per se* as well as the risk of overdue tile redownloading. By contrast, RoSal360 considers the reliability of both HMT predictions and network estimations, and achieves the more robust saliency bias compensation with a RL-based correction judgment.

10 Conclusion

We design and implement RoSal360, a robust and sophisticated saliency-driven quality adaptation framework for mobile 360-degree video streaming, to improve the quality of experience (QoE) under the limited and variable wireless network bandwidth. RoSal360 integrates a practical tile-size-aware transmission time prediction (TTP) neural network model, a saliency-aware quality allocation problem formulation with a fast solution method, and a reinforcement learning (RL)-driven online correction mechanism. Through extensive prototype experiments over real wireless networks, RoSal360 significantly improves the video quality, rebuffering reduction, and viewer QoE over existing approaches. Moreover, we constructed a public gaze-annotated long 360-degree video dataset, which are not only used in our evaluations but would also facilitate precise quality assessment and attention behavior analysis for the 360-degree video streaming research.

References

[1] G. V. Research, "Virtual reality market share and trends report, 2021-2028.." https://www.grandviewresearch.com/industry-analysis/virtual-reality-vr-market.
[2] Oculus, "Oculus quest 2 - untether your expectations." https://www.oculus.com/quest-2/.
[3] Samsung, "Gear vr.." https://www.samsung.com/global/galaxy/gear-vr/.

[4] YouTube, "Virtual reality channel - youtube." https://www.youtube.com/channel/UCzuqhhs6NWbgTzMuM09WKDQ.

[5] VRGear, "Netflix vr." https://vrgear.com/news/netflix-vr/.

[6] "Internet connection speed recommendations." https://help.netflix.com/en/node/306.

[7] Verizon, "4g lte speeds vs. your home network.." https://www.verizon.com/articles/4g-lte-speeds-vs-your-home-network/.

[8] T.-Y. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," in *SIGCOMM*, pp. 187–198, 2014.

[9] T. Driscoll, S. Farhoud, S. Nowling, *et al.*, "Enabling mobile augmented and virtual reality with 5g networks," tech. rep., 2017.

[10] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices," in *MobiCom*, pp. 99–114, 2018.

[11] Y. Guan, C. Zheng, X. Zhang, Z. Guo, and J. Jiang, "Pano: Optimizing 360 video streaming with a better understanding of quality perception," in *SIGCOMM*, pp. 394–407, 2019.

[12] 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 *MM*, pp. 315–323, 2017.

[13] L. Sun, F. Duanmu, Y. Liu, Y. Wang, Y. Ye, H. Shi, and D. Dai, "Multi-path multi-tier 360-degree video streaming in 5g networks," in *MMSys*, pp. 162–173, 2018.

[14] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *AllThingsCellular*, pp. 1–6, 2016.

[15] C. Zhou, M. Xiao, and Y. Liu, "Clustile: Toward minimizing bandwidth in 360-degree video streaming," in *INFOCOM*, pp. 962–970, 2018.

[16] D. Baek, H. Kang, and J. Ryoo, "Sali360: design and implementation of saliency based video compression for 360-degree video streaming," in *MMSys*, pp. 141–152, 2020.

[17] S. Wang, S. Yang, H. Li, X. Zhang, C. Zhou, C. Xu, F. Qian, N. Wang, and Z. Xu, "Salientvr: Saliency-driven mobile 360-degree video streaming with gaze information," in *MobiCom*, 2022.

[18] V. Sitzmann, A. Serrano, A. Pavel, M. Agrawala, D. Gutierrez, B. Masia, and G. Wetzstein, "Saliency in vr: How do people explore virtual environments?," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 4, pp. 1633–1642, 2018.

[19] F. Y. Yan, H. Ayers, C. Zhu, S. Fouladi, J. Hong, K. Zhang, P. Levis, and K. Winstein, "Learning in situ: a randomized experiment in video streaming," in *NSDI*, pp. 495–511, 2020.

[20] A. Borji, D. N. Sihite, and L. Itti, "What stands out in a scene? a study of human explicit saliency judgment," *Vision research*, vol. 91, pp. 62–77, 2013.

[21] Y. Li, X. Hou, C. Koch, J. M. Rehg, and A. L. Yuille, "The secrets of salient object segmentation," in *CVPR*, pp. 280–287, 2014.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, pp. 5998–6008, 2017.

[23] D. Schroeder, A. El Essaili, E. Steinbach, D. Staehle, and M. Shehada, "Low-complexity no-reference psnr estimation for h. 264/avc encoded video," in *2013 20th International packet video workshop*, pp. 1–6, IEEE, 2013.

[24] I. U. Khan, M. Ansari, S. H. Saeed, and K. Khan, "Evaluation and analysis of rate control methods for h. 264/avc and mpeg-4 video codec," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 5, p. 2788, 2018.

[25] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[26] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon, "Optimization by simulated annealing: An experimental evaluation; part i, graph partitioning," *Operations research*, vol. 37, no. 6, pp. 865–892, 1989.

[27] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in neural information processing systems*, vol. 12, 1999.

[28] Chrille89, "Hevc tiles merger." https://github.com/Chrille89/DashPlayer, 2019.

[29] Samsung, "Samsung galaxy s20.." https://www.samsung.com/us/mobile/galaxy-s20-5g.

[30] "Ffmpeg." https://github.com/FFmpeg/FFmpeg.

[31] M. Viitanen, A. Koivula, A. Lemmetti, A. Ylä-Outinen, J. Vanne, and T. D. Hämäläinen, "Kvazaar: Open-source hevc/h.265 encoder," in *MM*, 2016.

[32] M. Gert, "Viewport dependent mpeg-dash streaming of 360 degree natively tiled hevc video in web browser context."

[33] C. Concolato, J. Le Feuvre, F. Denoual, F. Mazé, E. Nassor, N. Ouedraogo, and J. Taquet, "Adaptive streaming of hevc tiled videos using mpeg-dash," *IEEE transactions on circuits and systems for video technology*, vol. 28, no. 8, pp. 1981–1992, 2017.

[34] J. Le Feuvre, C. Concolato, and J.-C. Moissinac, "Gpac: Open source multimedia framework," in *MM*, p. 1009–1012, Association for Computing Machinery, 2007.

[35] A-Frame, "A web framework for building virtual reality experiences.." https://github.com/aframevr/aframe.

[36] K. Boos, D. Chu, and E. Cuervo, "Flashback: Immersive virtual reality on mobile devices via rendering memoization," in *MobiSys*, pp. 291–304, 2016.

[37] J. van der Hooft, M. T. Vega, S. Petrangeli, T. Wauters, and F. De Turck, "Quality assessment for adaptive virtual reality video streaming: A probabilistic approach on the user's gaze," in *ICIN*, pp. 19–24, IEEE, 2019.

[38] "Htc vive pro eye.." https://www.vive.com/uk/product/vive-pro-eye/.

[39] "Tobii pro lab." https://www.tobiipro.com/product-listing/tobii-pro-lab/.

[40] H. VIVE, "Vive wireless adapter - untethered virtual reality is here.." https://www.vive.com/us/accessory/wireless-adapter/.

[41] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao, "Gaze prediction in dynamic 360 immersive videos," in *CVPR*, pp. 5333–5342, 2018.

[42] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," in *SIGCOMM*, pp. 325–338, 2015.

[43] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, and V. Gopalakrishnan, "On leveraging machine and deep learning for throughput prediction in cellular networks: Design, performance, and challenges," *IEEE Comm. Mag.*, vol. 58, no. 3, pp. 11–17, 2020.

[44] "Pensieve project." https://github.com/hongzimao/pensieve.

[45] H. Wang, V.-T. Nguyen, W. T. Ooi, and M. C. Chan, "Mixing tile resolutions in tiled video: A perceptual quality assessment," in *NOSSDAV*, pp. 25–30, 2014.

[46] VideoLAN, "x264, the best h.264/avc encoder.." https://www.videolan.org/developers/x264.html.

[47] V. Chauhan *et al.*, "A review on 5g network system with its limitation and different approaches to build strong 5g network system," in *2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)*, pp. 403–410, IEEE, 2022.

[48] X. Wei, M. Zhou, S. Kwong, H. Yuan, and W. Jia, "A hybrid control scheme for 360-degree dynamic adaptive video streaming over mobile devices," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

[49] Y. Zhang, Y. Guan, K. Bian, Y. Liu, H. Tuo, L. Song, and X. Li, "Epass360: Qoe-aware 360-degree video streaming over mobile devices," *IEEE Transactions on Mobile Computing*, vol. 20, no. 7, pp. 2338–2353, 2021.

[50] Y. Bao, T. Zhang, A. Pande, H. Wu, and X. Liu, "Motion-prediction-based multicast for 360-degree video transmissions," in *SECON*, pp. 1–9, IEEE, 2017.

[51] Y. Ban, L. Xie, Z. Xu, X. Zhang, Z. Guo, and Y. Wang, "Cub360: Exploiting cross-users behaviors for viewport prediction in 360 video adaptive streaming," in *ICME*, pp. 1–6, 2018.

[52] T. Xu, Q. Fan, and B. Han, "Content assisted viewport prediction for panoramic video streaming," in *Workshop on Computer Vision for AR/VR*, 2019.

[53] M. Hu, J. Chen, D. Wu, Y. Zhou, Y. Wang, and H.-N. Dai, "Tvg-streaming: Learning user behaviors for qoe-optimized 360-degree video streaming," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[54] A. T. Nasrabadi, A. Mahzari, J. D. Beshay, and R. Prakash, "Adaptive 360-degree video streaming using scalable video coding," in *MM*, pp. 1689–1697, 2017.

[55] J. He, M. A. Qureshi, L. Qiu, J. Li, F. Li, and L. Han, "Rubiks: Practical 360-degree streaming for smartphones," in *MobiSys*, pp. 482–494, 2018.

[56] X. Zhang, X. Hu, L. Zhong, S. Shirmohammadi, and L. Zhang, "Cooperative tile-based 360 panoramic streaming in heterogeneous networks using scalable video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 1, pp. 217–231, 2018.

[57] S. Lee, D. Jang, J. Jeong, and E.-S. Ryu, "Motion-constrained tile set based 360-degree video streaming using saliency map prediction," in *Proceedings of the 29th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, pp. 20–24, 2019.
[58] M. Tang and V. W. Wong, "Online bitrate selection for viewport adaptive 360-degree video streaming," *IEEE Transactions on Mobile Computing*, 2020.
[59] M. Dasari, A. Bhattacharya, S. Vargas, P. Sahu, A. Balasubramanian, and S. R. Das, "Streaming 360-degree videos using super-resolution," in *INFOCOM*, pp. 1977–1986, IEEE, 2020.
[60] C. Li, W. Zhang, Y. Liu, and Y. Wang, "Very long term field of view prediction for 360-degree video streaming," in *MIPR*, pp. 297–302, 2019.
[61] C.-L. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Fixation prediction for 360 video streaming in head-mounted virtual reality," in *NOSSDAV*, pp. 67–72, 2017.
[62] A. Nguyen, Z. Yan, and K. Nahrstedt, "Your attention is unique: Detecting 360-degree video saliency in head-mounted display for head movement prediction," in *MM*, pp. 1190–1198, 2018.
[63] W. Shen, L. Ding, G. Zhai, Y. Cui, and Z. Gao, "A qoe-oriented saliency-aware approach for 360-degree video transmission," in *2019 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, IEEE, 2019.
[64] Z. Jiang, X. Zhang, W. Huang, H. Chen, Y. Xu, J.-N. Hwang, Z. Ma, and J. Sun, "A hierarchical buffer management approach to rate adaptation for 360-degree video streaming," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2157–2170, 2020.

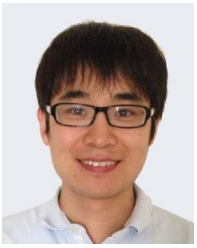
Hairong Su received her B.Sc. degree in Mathematics from Xi'an Jiaotong University (XJTU) in 2021. She is currently working toward a Ph.D degree at XJTU. Her research interests include video streaming, networked systems and deep learning.

Cong Zhao received his Ph.D. degree in Computer Science and Technology from Xi'an Jiaotong University (XJTU) in 2017. He is currently an Associate Professor at XJTU. His research interests include edge intelligence, meta learning, federated learning, and industrial intelligence.

Shibo Wang received his B.Sc. degree from Xi'an Jiaotong University (XJTU) in 2019. He is currently working toward a Ph.D. degree at XJTU. His research interests include networked systems, video streaming, and video analytics.

Chenren Xu (http://ceca.pku.edu.cn/chenren) is a Boya Young Fellow Associate Professor (with early tenure) in the School of Computer Science at Peking University (PKU) where he directs Software-hardware Orchestrated ARchitecture (SOAR) Lab. His research interests span wireless, networking and system, with a current focus on backscatter communication for low power IoT connectivity, future mobile Internet for high mobility data networking, and collaborative edge intelligence system for mobile and IoT computing. He earned his Ph.D. from WINLAB, Rutgers University, and worked as postdoctoral fellow in Carnegie Mellon University and visiting scholars in AT&T Shannon Labs and Microsoft Research. He is the General Secretary of ACM SIGBED China, Executive Committee of ACM SIGMOBILE and ACM SIGBED, Associate Editor of ACM IMWUT and Communications of the CCF. He published papers and has been serving as organization committee and/or TPC in top venues including ACM SIGCOMM, MobiCom, SenSys, UbiComp, and IEEE INFOCOM. He is a recipient of NSFC Excellent Young Scientists Fund (2020), Alibaba DAMO Academy Young Fellow (2018), ACM SIGCOMM China Rising Star (2020), CCF-Intel Young Faculty (2017) and CIE Outstanding Scientific and Technological Worker (2021) awards. His work has been featured in MIT Technology Review.

Shusen Yang received his Ph.D. degree in Computing from Imperial College London in 2014. He is a professor and director of the National Engineering Laboratory for Big Data Analytics, and deputy director of Ministry of Education (MoE) Key Lab for Intelligent Networks and Network Security, both at Xi'an Jiaotong University (XJTU), Xi'an, China. He is also the deputy director of the Industrial Artificial Intelligence Center at Pazhou Laboratory, Guangzhou, China. Before joining XJTU, He worked as a lecturer (assistant professor) at the University of Liverpool from 2015 to 2016, and a research associate at Intel Collaborative Research Institute (ICRI) on sustainable connected cities from 2013 to 2014. Shusen is a DAMO Academy Young Fellow, and an honorary research fellow at Imperial College London. He is a senior member of IEEE and a member of ACM. His research focuses on distributed systems and data sciences, and their applications in industrial scenarios, including data-driven network algorithms, distributed machine learning, Edge-Cloud intelligence, industrial internet and industrial intelligence.



Feng Qian (Member, IEEE/ACM) received the BS degree from Shanghai Jiao Tong University, and the Ph.D. degree from the University of Michigan. He is currently an Associate Professor in the Computer Science and Engineering Department at University of Minnesota - Twin Cities. Prior to joining UMN, he worked at AT&T Labs and Indiana University. His research interests cover mobile systems, AR/VR, mobile networking, wearable computing, real-world system measurements, and system security.

Nanbin Wang received his M.Sc. degree from Nanjing University of Posts and Telecommunications in 1999. He has been with Huawei Technology for more than 20 years and has high professional honors. Currently, he is the Fellow of Global Technology Service, the chief architect of mobile network, and also the chief architect of 5G service and experience, at Huawei Technology.

Zongben Xu received the Ph.D. degree in mathematics from Xi'an Jiaotong University (XJTU) in 1987. He is the Director of the Institute for Information and System Sciences in XJTU. His research interests include intelligent information processing and applied mathematics. Dr. Xu was elected as a member of the Chinese Academy of Science in 2011. He was a recipient of the National Natural Science Award of China in 2007 and the Winner of the CSIAM Su Buchin Applied Mathematics Prize in 2008. He delivered a speech at the International Congress of Mathematicians 2010. He serves as the Chief Scientist for the National Basic Research Program of China (973 Project).