# Towards Low Latency Multi-viewpoint 360° Interactive Video: A Multimodal Deep Reinforcement Learning Approach

Haitian Pang*,†, Cong Zhang§,†, Fangxin Wang†, Jiangchuan Liu†, Lifeng Sun*
*Department of Computer Science and Technology, Tsinghua University, China
§School of Computer Science and Technology, University of Science and Technology of China, China
†School of Computing Science, Simon Fraser University, Canada

***Abstract*—Recently, the fusion of 360° video and multi-viewpoint video, called multi-viewpoint (MVP) 360° interactive video, has emerged and created much more immersive and interactive user experience, but calls for a low latency solution to request the high-definition contents. Such viewing-related features as head movement have been recently studied, but several key issues still need to be addressed. On the viewer side, it is not clear how to effectively integrate different types of viewing-related features. At the session level, questions such as how to optimize the video quality under dynamic networking conditions and how to build an end-to-end mapping between these features and the quality selection remain to be answered. The solutions to these questions are further complicated given the many practical challenges, e.g., incomplete feature extraction and inaccurate prediction.**

**This paper presents an architecture, called iView, to address the aforementioned issues in an MVP 360° interactive video scenario. To fully understand the viewing-related features and provide a one-step solution, we advocate multimodal learning and deep reinforcement learning in the design. iView intelligently determines video quality and reduces the latency without pre-programmed models or assumptions. We have evaluated iView with multiple real-world video and network datasets. The results showed that our solution effectively utilizes the features of video frames, networking throughput, head movements, and viewpoint selections, achieving at least 27.2%, 15.4%, and 2.8% improvements on the three video datasets, respectively, compared with several state-of-the-art methods.**

## I. Introduction

Thanks to the latest development in network communication (e.g., 4G/5G) and hardware design (e.g., iPhone XS), both 360° video [1] and multi-viewpoint video [2] have enabled an immersive experience for a large number of viewers who only need to own a VR (Virtual Reality) Head-Mounted-Display (HMD) headset, e.g., Oculus Rift, or a VR compatible mobile device with a simple headset, e.g., Google Pixel and Google Daydream. Any viewer can conveniently select a preferred perspective, enjoying the real-time high-definition videos by changing the viewpoints or touching the controllers. A recent report from IDC[1] showed that the shipments of VR headset are expected to grow almost five times in the next five years, from 8.1 million in 2018 to 39.2 million in 2022. Nowadays, such mainstream video service providers as YouTube also support 360° videos up to 8192x4096 resolution. Besides, tile-based HTTP Live Streaming (HLS) provides an opportunity to achieve a high Quality-of-Experience (QoE) under a limited throughput in such scenarios [3]. In the tile-based HLS standard, a video content (i.e., a segment) with a short duration, e.g., one second, can be split into several tiles in the spatial dimension, which allows a client to request only part of needed tiles and differentiate the quality of each tile considering the viewer's Field-of-View (FoV) [4].

Recently, the fusion of 360° video and multi-viewpoint video, called multi-viewpoint (MVP) 360° interactive video, has emerged as a novel application and attracted a great deal of attention from industry and academia [5]. It enables much more immersive and interactive virtual experience for the viewers, but still faces several new challenges: (1) it contains six Degrees-of-Freedom (DoF) interaction that consists of head rotation (3DoF) and viewpoint selection (3DoF), involving in a high-dimensional interactive space; (2) the fidelity of the scene reconstruction mainly depends on high-definition video contents, which needs a great deal of throughput and computation resources on the viewer side to mitigate the download latency and rendering latency; (3) the switch among various viewpoints may occur at the same FoV, which introduces extra switch latency and rebuffering latency.

Therefore, how to decrease these latencies with respect to the viewing experiences when viewers interact with 6DoF videos remains a challenging task. On the one hand, most of the existing works, mainly investigating 360° video [1], [6] and multi-viewpoint video [7], either examine the viewers' rotational movements only or cannot quickly respond to the change of FoVs. On the other hand, recent study [5] in MVP 360° interactive video, assumes that several well-designed prediction methods have already been implemented to provide necessary information to the optimization step, which fails to intelligently adjust prediction based on the optimization results.

In this paper, we for the first time present a low latency MVP 360° interactive video system, called **iView**, which integrates the prediction of viewing-related features and the optimization of tile quality selection. The basic design of iView consists of two modules: a Viewing-Related Prediction (VRP) module and a Tile Selection Optimization (TSO)

The first two authors contributed equally to this work.
[1]https://www.idc.com/getdoc.jsp?containerId=prUS43639318

module. The former utilizes three sub-modules to separately predict the future throughput, head rotation, and viewpoint. The latter receives these prediction results as well as the buffer status on the viewer side to address the tile selection and optimization by solving a formulated Multiple Choice Knapsack (MCK) problem. This basic design is effective, but does have limitations: (1) it cannot take the visual features of video contents into account, due to the TSO module cannot receive such information; (2) the VRP module is independent of the TSO module, so the prediction of the three sub-modules cannot request any feedback from the TSO module, failing to adjust the prediction methods to improve the optimization result; (3) the decoupled scheme may encounters on the two-step delay, that is, the TSO module only can be executed after receiving the prediction results from the VRP module, which cannot be used in a practical system.

We therefore enhance the design of iView to meet the following three requirements: (1) coupling the VRP module and the TSO module to provide an end-to-end solution between the viewing-related features and the optimization objective; (2) accommodating the different types of viewing-related features, including video features, time-series, and the client status on the viewer side; and (3) understanding incomplete viewing features when frequently switching viewpoint or changing FoV. To fulfill these requirements, particularly, the high-dimensional and incomplete feature space, we advocate multimodal learning [8] and deep reinforcement learning [9] to make iView more intelligent.

Multimodal learning enables the system to fully explore the different types of data to understand an event. A typical application is to detect events using the visual feature, audio feature, and text feature in a multimodal video dataset [10]. Besides, a deep reinforcement learning (DRL) based technique only needs partial information to usher the training step. Compared with previous model-based methods [1], [5], multimodal deep reinforcement learning (MDRL) based solution enables Deep Neural Networks (DNNs) to explore the impact of the high-dimensional features in input data on the objective of an optimization problem, integrating a prediction stage and a optimization stage well, directly building the correlation between input features and an optimization objective. Therefore, MDRL naturally provides an opportunity to address the aforementioned challenges and requirements in designing a low latency MVP 360° interactive video system.

Through this integrated design, iView bridges the gap between different types of complex viewing-related features and viewing QoE, i.e., video quality and latency. The contribution in this work can be summarized as follows:

- We for the first time employ multimodal learning to integrate five types of features related to the viewer's QoE, i.e., video frame, head movement, viewpoint selection, networking throughput and buffer status. The combination of these features creates a high-dimensional space, which is hard to be handled by traditional pre-programmed models.
- We design an actor-critic-based [11] DRL module to generate an end-to-end solution, which integrates all features and learns how to usher the tile selection accordingly.
- We evaluate the performance of iView using three datasets with the viewer's movements and one dataset with networking traces. We also compare the performance with two state-of-the-art solutions to show the superiority of iViewer in the improvement of video quality and the decrease of rebuffering latency and download latency.

The remainder of this paper is organized as follows. Section II briefly introduces the background and related work in this paper. We propose the basic design and enhancement design of iView in Section III and IV . We evaluate iView via trace-driven simulations in Section V and conclude the paper with further discussion in Section VI .

## II. Background and Related Work

### A. Interactive Video

Most of the existing works investigated 360° interactive video streaming or multi-viewpoint video streaming separately. The videos in the former are recorded by a single omnidirectional camera, e.g., Samsung Gear 360°, providing rotational movement around the center point of this camera [12]. The viewers have three degrees of freedom, i.e., yaw, roll, and pitch, as shown in the upper half of Figure 1. The videos in the latter are recorded by a set of cameras around the event, enabling viewers to change positions (i.e., translational movement) among different locations [7]. The viewers have three degrees of freedom, i.e., heave, surge, and sway, as shown in the lower half of Figure 1. An MVP 360° interactive video records an event using a set of omnidirectional cameras, providing the aforementioned 6DoF (i.e., rotational and translational movements) and allowing the viewers to change the viewpoints among these cameras and view the event from every direction at the same time. This novel scenario creates a more immersive and realistic streaming experience, attracting much attention in industry and academia [5].

### B. Tile-based Streaming

Recently, tile-based streaming was proposed as a part of MPEG Dynamic Adaptive Streaming over HTTP (DASH) standard, enabling an opportunity of spatial access to a video [3], [13], [14]. As shown in Figure 3, splitting a video frame to several tiles enables the media player on the viewer side to choose the quality of every tile when pulling it from media servers. It therefore perfectly matches the FoV-related video streaming applications, such as 360° videos, multi-viewpoint videos, and MVP 360° interactive videos. Taking the first part in Figure 3 as an example, it contains a one-second video with three quality versions, i.e., Q1, Q2, and Q3. Each quality version is further split into nine tiles, from T1 to T9. If a media player already has already predicted that a viewer's FoV only contains the area at T5 and T6, it only needs to first download these two tiles, and then requests others, to reduce the delivery latency. Existing works in this

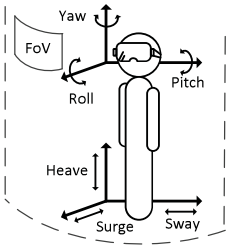



Fig. 1: Viewer's movement


Fig. 2: Viewpoint selection

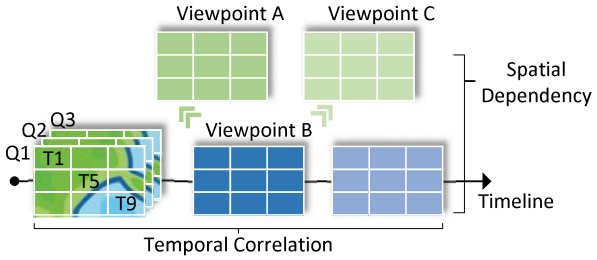



Fig. 3: Tile-based MVP 360° interactive video

topic discussed energy-aware content delivery [15], FoV-based adaptation [12], and tile-based content reconstruction [3].

### C. Viewer's Movement Prediction

Liu *et al.* [6] introduced a FoV-guided strategy to fetch the portions of a scene that viewer will see. To retrieve the FoV information, their method contains a big-data-assisted head movement prediction module, which jointly examines the viewing statistics of the same video across various viewers, the viewing behaviors over multiple videos of a viewer, and some contextual information, e.g., viewers' pose. Fan *et al.* [12] also studied the FoV prediction problem when viewers' watching 360° videos. Their fixation prediction jointly utilized the sensor-related features and video-related features as the inputs of a deep learning network. By analyzing the prediction results, they further conducted a trace-driven simulation to prove the effectiveness in 360° video streaming scenarios. Almquist *et al.* [4] discussed the prefetching aggressiveness tradeoff problem that means how to select the proper prefetching time points when watching a 360° video.

### D. Content Delivery and Rate Adaptation

Hamza *et al.* [2] proposed a quality-aware two-step rate adaptation method from multi-viewpoint videos using a visual view distortion model, which explore the viewer's interaction with the scene and the depth information of videos. Their approach enables the streaming client to find the best set of representations to request from the server based on the estimated viewpoint position. Their another work [7] further investigated the multi-viewpoint video streaming services over mobile networks, considering the viewers' QoE fairness and minimizing the quality fluctuation issues. They formulated a multi-objective QoE-fairness problem and proposed a heuristic algorithm to solve it efficiently. Chakareski *et al.* [16] proposed a viewpoint-driven rate-distortion optimized 360° video streaming, which integrates the viewer's navigation operation and the spatiotemporal rate-distortion characteristics of the 360° video content to maximize the delivered viewing QoE under different available network/system resources.

## III. iView: Basic Design

In this section, we propose the basic design of iView with problem formulation and solution.

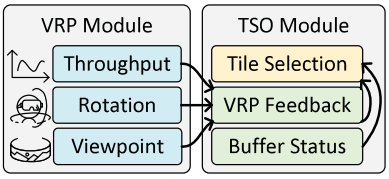



Fig. 4: Basic design of iView

### A. Problem Formulation

The iView architecture consists of two parts: a prediction module and an optimization module. As shown in Figure 4, the former predicts the available bandwidth, head orientation rotations, and viewpoint decisions in the future. This module therefore consists of three sub-parts to fulfill the three types of predictions, using prediction techniques, such as time-series analysis and neural network. The latter uses the feedbacks from the prediction module to optimize the future tile selection and improve the viewer's experience. Here, we consider a general MVP 360° interactive video system, in which a viewer can change head orientation rotations and select viewpoints without any restriction. When the viewer starts a sequence of interactions, the objective of the design is to request the tiles in the FoV as soon as possible under bandwidth constraints, quality constraints, and buffer constraints, etc. We therefore define $b_{u,i}$ and $d_{u,i}$ as the bitrate and size of tile $i$ at quality version $u$ for a given time slot $t$ , where $u \in U_i, i \in I_t, t \in T$. We also define $v(i)$ as a function to map the tile $i$ to its viewpoint $v$, $v \in V$. The objective is to improve a viewer's QoE when s/he watches an MVP 360° interactive video. So we combine the general QoE metrics used by [17] and [18] to define the QoE as follows:

$$Q(u,i) = \frac{F(b_{u,i})}{l^d_{u,i}(1-x_i) + l^r_i x_i}$$

where $F(b_{u,i})$ is a function of bitrate $b_{u,i}$, representing the scene quality perceived by a viewer. We consider two options, i.e., two concave non-decreasing linear and logarithmic functions, in the evaluation. $l^d_{u,i}$ represents the download time of a tile and $l^r_i$ represents the rebuffering time when a viewer switches a new viewpoint; therefore, $x_i$ is an indicator, $x_i = 1$ if current viewer selects a new viewpoint, $x_i = 0$ otherwise. $l^d_{u,i}$ and $l^r_i$ are defined as follows:

$$l_{u,i}^{d} = \frac{d_{u,i}}{B_p}, l_i^r = \sum_{i \in I_t} \sum_{u \in U_i} l_{u,i}^{d}$$

where $B_p$ is the future throughput acquired by the prediction module.

According to the formulation of the objective and above definitions, we define the problem as follows:

$$\max \sum_i \sum_u Q(u,i) y_{u,i} \tag{1}$$

subject to:

$$x_i = \begin{cases} 1 & \text{if } v(i) \neq v(i^{'}), i^{'} \in I_{t-1}, i \in I_t \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$y_{u,i} = \begin{cases} 1 & \text{if file } (u,i) \text{ is requested} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$\sum_u y_{u,i} = 1, \forall i \tag{4}$$

$$\sum_i \sum_u d_{u,i} y_{u,i} \leq D \cdot B_p \cdot D_{buffer} \tag{5}$$

$$F(b_{u,i}) \geq F(b_{\bar{u},\bar{i}}) \tag{6}$$

$$u \in U_i, \bar{u} \in U_{\bar{i}}, i \in I_t^p, \bar{i} \in \bar{I}_t^p \tag{7}$$

$$I_t^p \cap \bar{I}_t^p = \emptyset, I_t^p \cup \bar{I}_t^p = I_t \tag{8}$$

where $y_{i,t}$ be an indicator, $y_{i,t} = 1$ if the tile $(i,t)$ is requested in the future, $x_{i,t} = 0$ otherwise. $D$ is the duration of a tile varies from a few seconds to several minutes (e.g., two seconds), $D_{buffer}$ is the duration of the buffered video content on the viewer-side, $B_p, I_p$, and $V_p$ are the future throughput, the future tile set in the FoV and the viewpoint acquired by the prediction module. The rationale of constraints (3)-(5) is as follow: (1) the system only needs to download one quality version for every tile $i$ in segment $s$, which avoids extra bandwidth consumption; and (2) all tiles in a segment should be downloaded completely before the cache becomes empty, which guarantees a smooth playback in the MVP 360° interactive video scenario. The rationale of constraint (6) is as follows: (1) the system provides the same quality for the best effort; (2) the tiles in predicted FoV can have a higher quality than others, e.g., high bitrate or low distortion, improving the viewing experience. The rationale of constraints (7)-(8) is as follows: (1) The prediction module provides the future tile set and the future viewpoint in FoV; (2) To differentiate the qualities conveniently, there is no overlap between the tiles and viewpoint in FoV and other regions.

### B. Solution

If we only consider the constraints (3) to (5), this optimization can be transformed into the following Multiple Choice Knapsack (MCK) problem, which is NP-hard.

*Theorem 1:* The problem of tile selection and optimization such that the QoE can be maximally achieved, as formulated in (1), is NP-hard.

*Proof:* We reduce a Multiple Choice Knapsack (MCK) Problem, which is NP-hard [19], to this problem. Given $m$ mutually disjoint classes $N_1, \cdots, N_m$ of items to be placed into a knapsack with capacity $c$. Each item $j \in N_i$ has a value $v_{i,j}$ and a weight $w_{i,j}$, and the problem is to choose exactly one item from each class such that the total value is maximized without exceeding the capacity $c$ in the corresponding total weight. By introducing binary variables $z_{i,j}$ which equal to 1 if and only if item $j$ is chosen in class $N_i$, we can define the MCK problem as follows:

$$\max \sum_{i=1}^{m} \sum_{j \in N_i} v_{i,j} z_{i,j}$$

subject to:

$$\sum_{i=1}^{m} \sum_{j \in N_i} w_{i,j} z_{i,j} \leq c$$

$$\sum_{j \in N_i} z_{i,j} = 1, i = 1, \cdots, m$$

$$z_{i,j} \in \{0,1\}, i = 1, \cdots, m, j \in N_i$$

If we only consider the constraints (3) to (5), the optimization problem (1) can be transformed into the following Multiple Choice Knapsack (MCK) problem: there are $N$ mutually disjoint classes $C_1, \cdots, C_N$ of items (i.e., tiles) to be packed into a knapsack of capacity $B_p \cdot D_{buffer}$. Each item $j \in C_i$ has a value $Q(u,i)$ and a weight $d_{i,j}$, and the problem is to choose exactly one item from each class such that the value sum is maximized without exceeding the capacity of the corresponding weight sum. Through introducing binary variables $x_{i,j}$ which take on 1 if and only if item $j$ is chosen in class $C_i$, we can transform this problem into a corresponding MCK problem. □

Although we can transform the "light" version of this problem into a corresponding MCK problem that has efficient solutions available in practice [20], we still need to consider other constraints. Constraint (2) is effective when calculating the QoE and can be relaxed easily. For constraints (6) to (8), we re-design the function $F(b_{u,i})$ to $F(b_{u,i}, i)$, which checks whether a tile $i$ exists in the FoV or not. We denote $\phi$ as a tunable parameter, which determines how to decrease this part of quality when a tile is not in the FoV. $F(b_{u,i}, i)$ is defined it as follows. We present the definition of $f(b_{u,i}, i)$ in Secion V.

$$F(b_{u,i}, i) = \begin{cases} f(b_{u,i}, i) & \text{if tile } i \text{ is in the FoV} \\ \phi f(b_{u,i}, i) & \text{otherwise} \end{cases} \tag{9}$$

Now, the optimization problem can be solved in practice. Yet this basic design has several limitations: (1) the TSO module cannot take the visual feature of the MVP 360° interactive video into account due to the limitation of the model-based solution; (2) the target of the VRP module is to improve the prediction accuracy, which is not related to the optimization objective directly; (3) current decoupled design has a two-step delay, that is, the optimization step can only be started after receiving the prediction results from the prediction module. To overcome these limitations, we further design an enhanced solution, which inherits the core of the basic design, but combines the prediction and optimization using more intelligent techniques.

## IV. iView with Learning Enhancement

In this section, we present the enhancement design of iView, which generates DRL-based tile selection algorithm and achieves low latency MVP 360° interaction videos. We first introduce the preliminaries of learning techniques. Then, we propose the training methodology and algorithm in iView.

### A. Preliminaries

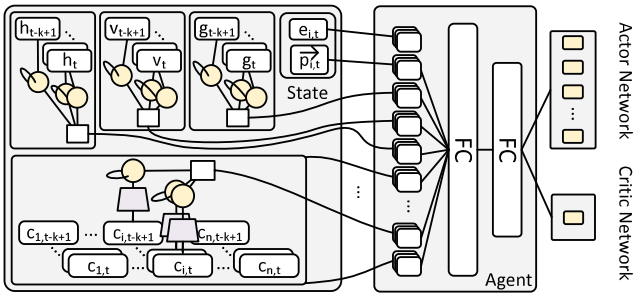



Fig. 5: Enhancement design of iView

**Multimodal Learning** A modality is defined as a way to understand an event. For example, both vision and sound can be used to recognize the meaning of a speech in a video. Therefore, a multimodal research contains multiple such modalities in its problem or dataset. Multimodal learning enables fully exploration for the different types of data in a multimodal dataset. A typical application is event detection using the visual feature, audio feature, and text feature in multimodal video dataset [10]. In multimodal learning, how to integrate the information (i.e., fusion) from different types of modalities is one of the hot topics. Early fusion integrates the features of multiple modalities immediately after they are extracted, while late fusion conducts the integration after pre-processing them or making a decision (e.g., classification). Besides, hybrid fusion and slow fusion were also presented to address the specific challenges. Because different types of information exist in the iView design, simply integrating them at an early stage does not work. Taking the fusion of video frame and head movement as an example, a video frame can be considered as a three-dimensional matrix and a sequence of head movements is a one-dimensional time-series, simply reshaping the video frame to a one-dimensional data overlooks the feature inside this video frame. We therefore select late fusion as a way to integrate all information after pre-processing the original features.

**Deep Reinforcement Learning** DRL-based solutions have been introduced to solve lots of problems from an intelligent perspective, such as, resource allocation [21], adaptive streaming [22], and human-level game control [9], showing its superiority in terms of efficiency, effectiveness, and cost. A general reinforcement learning setup contains an agent interacting with an environment in discrete epoch. At each epoch $t$, the agent receives a state $s_t$ and performs an action $a_t$ according to its policy $\pi(s_t)$, which maps states to actions or probability distributions over actions. The goal of the agent is to receive a reward $r_t$, observe the next state $s_{t+1}$, and calculate the return, i.e., the sum of discounted future reward $R_t = \sum_{i=t}^{\infty} \gamma^{i-t} r(s_i, a_i)$, where $r(s_i, a_i)$ is defined as the reward function and $\gamma \in [0, 1]$ is a discounting factor. The objective of reinforcement learning is how to choose the policy to maximize the expected return, which is defined as $\mathbb{E}[R_t]$ formally.

Deep Reinforcement Learning techniques were introduced by Google DeepMind in 2015. The proposed Deep Q-network (DQN) learns successful policies via a deep convolutional neural network, which can handle high-dimensional sensory inputs and employ experience replay to achieve end-to-end reinforcement learning. The objective of DQN is to approximate the optimal action-value function, which is defined as the maximum expected return after observing sate $s_t$, taking an action $a_t$ under state $s_t$ and thereafter following policy $\pi$ per epoch $t$.

$$Q^*(s_t, a_t) = \max_{\pi} \mathbb{E}[R_t | s_t, a_t, \pi]$$

The key capability of DQN is to look ahead to the next epoch. Therefore, the optimal action-value is calculated based on the following rule: if an action $a_{t+1}$ from all possible actions achieves an optimal value $Q^*(s_{t+1}, a_{t+1})$ under a state $s_{t+1}$ at the next epoch $t+1$, it will be selected to maximize the expected value of $r_t + \gamma Q^*(s_{t+1}, a_{t+1})$.

$$Q^*(s_t, a_t) = \mathbb{E}_{s_{t+1}}[r_t + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) | s_t, a_t]$$

As a value-based method, DQN outputs the discrete value for each action, then chooses an action with the maximum value as the action in the next step. Such a mechanism cannot work well in the iView design, because a large number of tiles and their corresponding actions create a high dimension output space. To overcome this issue, we employ a state-of-the-art actor-critic-based asynchronous advantage actor-critic (A3C) algorithm as the fundamental algorithm in iView. An actor-critic-based method has accelerated lots of state-of-the-art RL algorithms. Taking AlphaGo[2] as an example, the core of beating human world champions is to take self-learning in the game, depending on an actor-critic-based method. The basic

[2]https://www.bbc.com/news/technology-35785875

idea of A3C is to enable value-based and policy-based training together. It therefore consists of action network and critic network. The former outputs the probability of each action, the latter provides feedback about this probability to action part. These two networks have the same DNN structure, but represent different outputs. The value predicted by the critic network can be used to train the actor network.

### B. Architecture

iView aims to find a tile selection algorithm; therefore, there exist three challenges when employing DRL techniques: (1) how to map a tile selection problem to a DRL-based action/state learning algorithm; (2) how to determine the space of action/state in the iView design; and (3) how to build a training environment to simulate the MVP 360° interactive video system. To overcome these challenges, we present the enhancement design of iView in Figure 5. Current design of iView consists of two parts: (1) input feature processing, in which multimodal data, i.e., video frame, head movement, viewpoint, network status, and client status, are processed based on their characteristics; (2) DRL-based training algorithm, in which the states, actions, policy, and DNN are designed to accommodate the input features and directly output the tile selection. The rationales of this design are (1) multimodal features are closely related to viewers' experiences, including download latency and video quality; (2) The nature of DRL can create an opportunity for the end-to-end mapping between these multimodal features and viewers experiences, integrating the FoV prediction and the corresponding optimization in MVP 360° interaction video scenarios.

*1) Input feature processing:* Previous works [12], [22] split time-series into a sequence of time slots to fit their DNNs. For example, Lo *et al.* use a sliding window with a fixed length (i.e., 30 frames) to divide the video frames and head movements and integrate them. In this work, because the viewer's movement contains two types: head movement and viewpoint selection, using the sliding window with a fixed length to directly read these data and integrate them will break the temporal correlation when the viewer takes rotational and translational movements. To accommodate MVP 360° interactive videos, we therefore employ a late fusion method in this step. That is, we design two types of DNNs to pre-process video frames and time series traces, i.e., head movements, viewpoint selections, and network throughput, as shown in Figures 5. For the video frames, the first CNN layer [23] (i.e., grey trapezoid) is used to reduce the dimension of each frame. The second LSTM layer [24] (i.e., yellow circle) is used to capture temporal information throughout the network such that higher layers get access to progressively more global information in both spatial and temporal dimensions. For the time series, we use an LSTM layer to collect the temporal correlation among the data among neighboring time slots. In this step, the size of the sliding window is 6 frames, i.e., about 200ms. We consider the data in each short-term slot (i.e., 30 frames) as one batch of the training step in the MDRL module.

*2) DRL-based Training Algorithm:* To train a DRL-based tile selection algorithm, the best way is to deploy a test-bed to generate the traced used in training. Yet such a step is too slow, because the training algorithm must collect all needed information from all viewers during every time slot. We therefore simulate an environment to train the model. Besides the video frames and the viewer's movement traces, the simulator in this environment has to maintain the following information on the viewer side: tile-based representation description, buffer size, viewpoint selection, current throughput. Based on the basic rules in a DRL-based method, our environment takes all of these viewer-related traces and the tile-based selection as states and actions, respectively.

As shown in Figure 5, the training algorithm in iView's agent uses an actor-critic-based asynchronous advantage actor-critic (A3C) algorithm, consists of an actor network (i.e., policy network) and a critic network (i.e., value network). The actor network is trained to predict the tile selection given the current state, where the sequence of actions corresponds to select quality versions for the tiles. The outputs of the actor network are the probability distribution over every tile selection (i.e., action). The critic predicts the value of each state (i.e., six types of viewer-related features and sequence of actions so far), which we define as the expected reward (i.e., viewing QoE metric defined in Section V).

**State:** The agent in iView receives state $s_t = (\vec{c_t}, \vec{h_t}, \vec{v_t}, \vec{g_t}, e_t, \vec{p_t})$ as the inputs of its DNNs. $\vec{c_t}$ is the visual vector, $\vec{c_t} = (\vec{c_{1,t}}, \vec{c_{2,t}}, \cdots, \vec{c_{n,t}}), n = |I|$;[3] $\vec{h_t}$ and $\vec{v_t}$ is the head movement feature and viewpoint feature; $\vec{g_t}$ is the network throughput measured by the past $k$ durations; $e_t$ is current buffer size; $\vec{p_t}$ is the size vector of all tiles in current time slot.

**Policy:** In current scenario, both the inputs and outputs are high-dimensional. We therefore use a three layers neural network to represent the shared structure of actor network and critic network, as shown in Figure 5. Policy $\pi$ is parameterized by $\theta$, which is the vector of the weights in the network. $\pi_\theta$ receives a state $s_t$ and generates the probability distribution over all actions, i.e., $a_{t+1}$ $\pi_\theta(s_t, a_t)$ and $\pi_\theta(s_t, a_t) \rightarrow [0, max(V)]$.

**Policy gradient training:** An A3C-based algorithm trains the actor network using a policy gradient method [25]. The objective of the policy gradient method is to maximize the expected cumulated reward by repeatedly estimating the gradient, i.e., $\nabla_\theta \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t]$, where the reward $r_t$ is calculated by the environment, i.e., the simulator, according to the status of the selected actions. Typically, we train the expression of the policy gradient as follows:

$$\nabla_\theta \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t] = \mathbb{E}[\sum_{t=0}^{\infty} A^\pi(s_t, a_{t+1}) \nabla_\theta log \pi_\theta(a_{t+1}|s_t)]$$

where $A^\pi(s_t, a_{t+1})$ is advantage function, which measures whether or not a specific action is better or worse than the default action in policy $\pi$. If and only if $A^\pi(s_t, a_{t+1}) > 0$ , the

[3]Here, we still use the definitions in Section III.

training algorithm changes the policy parameter $\theta$ to increase $\pi_\theta$ in the direction of $\nabla_\theta log \pi_\theta(a_{t+1}|s_t)$.

*3) Critic Network:* Advantage function is unknown, but can be estimated using a critic network, which shares the same structure with the actor network and has different output value. $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$, where $V^\pi(s)$ represents the expected future return as a function of the observed state $s_t$, given the policy $\pi$, actions, and reward function. We use temporal-difference (TD) learning to train the critic network parameters for a one-step iteration.

## V. Performance Evaluation

### A. Dataset

To evaluate the performance of the proposed architecture iView, we employed the following four public datasets, which contain two 360° video datasets with head movements and one MVP 360° interactive video dataset with head movements and viewpoints. We briefly introduce the four datasets and present the reconstruction methods respectively.

**Network traces**: To simulate a practical networking condition, we generate network traces based a broadband dataset provided by the FCC. The original FCC dataset covers the fixed broadband measurement results of major ISPs (Internet Service Providers), which collectively account for over 80% of U.S. residential broadband Internet connections. Each trace records download speed every five seconds. Considering the bandwidth requirement in our scenario, we extract 200 traces, whose minimum throughput is above 5Mbps and average throughput is about 20Mbps.

**Video traces**: To evaluate the performance of iView and other comparisons, we use the following three video datasets:

- *THU dataset* [26]: This dataset contains eighteen 360° videos from YouTube in five categories: Performance, Sport, Film, Documentary, and Talkshow. The HMD sensor traces are collected from 48 subjects, using an HTC Vive headset. The trace of each subject can be divided into two parts, each with nine videos. The first part is free to look around, while the second part asks the subject answers several questions related to the video content.
- *NTHU dataset* [27]: This dataset consists of the video clips (1 minute) of ten 360° videos from YouTube and the HMD sensor trace collected from 50 subjects. Every subject watches all the ten video clips, using an Oculus Rift DK2 headset. The videos used in NTHU dataset are different from those in THU dataset.
- *IRISA dataset* [5]: This dataset includes a 4K video at 30 frames per second, recorded by three Orah Live Spherical VR Camera 4i 360° cameras. Four subjects freely watch an MVP 360° video recorded in an indoor environment and switch the viewpoints using a Google Daydream controller. The sensor trace is collected by an Android-based phone and a Google Daydream. Each trace records the viewer's viewpoints orientation, the current displaced viewpoint, and the switch decision times.

TABLE II: Performance comparison on the three datasets

| Dataset | Reward | BD | FBB | FRB | iView | ↑ |
|---|---|---|---|---|---|---|
| THU | Linear | 33.8 | 24.6 | 29.2 | 43.1 | 29.4% |
| | Log | 52.3 | 32.3 | 44.6 | 67.6 | 27.2% |
| NTHU | Linear | 88.6 | 63.6 | 75.2 | 102.2 | 26.7% |
| | Log | 102.2 | 76.1 | 90.9 | 129.5 | 15.4% |
| IRISA | Linear | 3.6 | 3.4 | 3.3 | 3.8 | 2.8% |
| | Log | 4.1 | 3.9 | 4.0 | 4.2 | 9.4% |

Note that the first two datasets do not have any viewpoint information, therefore, we add viewpoint switching decision for the traces in the two datasets according to the characteristics of viewpoint switching decision in IRISA dataset. We list all related information about the three datasets in Table I. Because the maximum duration in the three datasets is 10m52s, every trace in the network traces extracted from the FCC has a duration of 700 seconds. We use *ffmpeg*, *MP4Box*, and *x264 encoder* to split the video traces into encoded one-second tiles, following the settings in the three datasets.

### B. Methodology

We implement iView using a deep learning framework Tensorflow[4] and an open-source framework Pensieve[5], the node setting of each DNN layer is selected empirically. We follow the objective and formulation of basic design in Section III and propose the following reward design to accommodate the updated architecture of iView. As shown in the following definition, the reward design consists of two parts: quality and latency considering different quality versions of all tiles. The quality part not only uses the bitrate, but also checks whether a tile belongs to the FoV. The latency part is considered as a penalty, which uses the download time of a tile and the rebuffering time when switching a new viewpoint.

$$\sum_i \underbrace{F(b_{u,i}, i)}_{\text{Quality}} - \underbrace{[l^d_{u,i}(1-x_i) + l^r_i x_i]}_{\text{Latency}} \tag{10}$$

where $F(b_{u,i}, i)$ is a function of bitrate $b_{u,i}$ and tile $i$, providing a part of reward for the quality of a tile when it in the FoV or not. In the evaluation, we still follow the definitions in (11) and (12), we set two tunable parameters $\phi_{linear}$ and $\phi_{log}$ to 0.8. We consider six types of bitrate settings: (1) 2160p (40Mbps); (2) 1440p (16Mbps); (3) 1080p (8Mbps); (4) 720p (5Mbps); (5) 480p (2.5Mbps); (6) 360p (1Mbps) due to the recommendation of encoding bitrate from YouTube[6], .

Linear Style:

$$f(b_{u,i}, i) = \frac{b_{u,i}}{\max\{b_{u,i} | \forall u \in U_i\}} \tag{11}$$

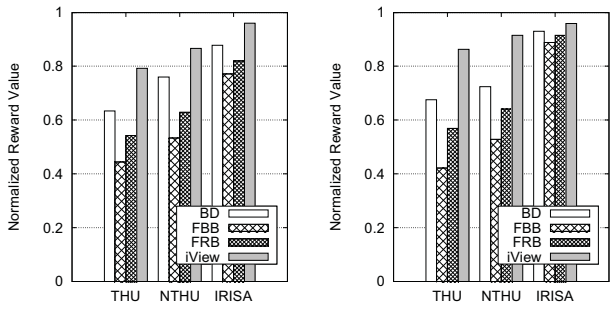
Logarithm Style:

$$f(b_{u,i}, i) = log(\frac{b_{u,i}}{\min\{b_{u,i} | \forall u \in U_i\}}) \tag{12}$$

[4]www.tensorflow.com
[5]https://github.com/hongzimao/pensieve
[6]https://goo.gl/ZoFcL3

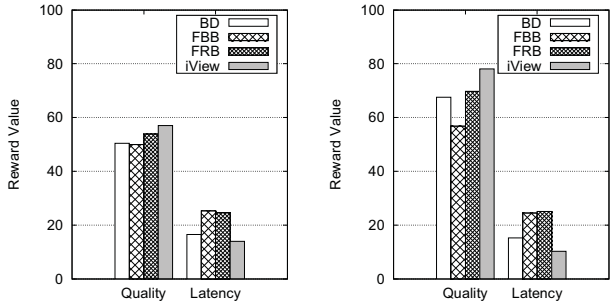
TABLE I: Dataset Description

| Dataset | # of subjects | # of videos | Duration (min/max/avg.) | Resolution(min/max) | MVP | 360° | tile setting |
|---|---|---|---|---|---|---|---|
| THU dataset | 48 | 18 | 10m52s/2m44s/4m52s | 1920x960/2880*1440 | No | Yes | 10x10 |
| NTHU dataset | 50 | 10 | 1m/1m/1m | 3840x1920/3840x2160 | No | Yes | 20x10 |
| IRISA dataset | 4 | 1 | 32s | 4096x2048 | Yes | Yes | 3x2 |




(a) Linear reward (b) Logarithm reward

Fig. 6: Performance comparison on different methods




(a) Linear reward (b) Logarithm reward

Fig. 7: Performance comparison on two reward components

For comparison, we have also implemented the following methods to achieve the tile selection in the MVP 360° interactive video scenario. These methods are derived from several state-of-the-art adaptation algorithms.

- Basic Design (BD): We use two-layer LSTM networks based approach to implement the prediction in the three sub-modules. The problem in the optimization part can be solved by a heuristic algorithm [28].
- FoV-Buffer-based (FBB): This method derives from the buffer-based algorithm proposed by Huang *et al* [29]. FBB method selects the quality of each tile based on the buffer status and calculates the FoV using the sensor data provided by HMD. When the buffer on the viewer side is filling up an extra reservoir $r$, the client requests the tiles in the FoV at a certain level. Here, we set $r$ is equal to 8 seconds [29], which means that if the extra buffering time is more than 8 seconds, the client will select a higher quality version.
- FoV-Rate-based (FRB): Similar to the FBB method, FRB first calculates the FoV, and then estimates throughput using smoothed HTTP/TCP throughput measurement in ten seconds [30]. In this method, the rate adaptation depends on the estimated throughput and the encoded bitrate for all tiles.

### C. Impacts of Different Datasets

We first conduct trace-driven simulations on the three datasets. Table II shows the total reward per time slot of the four methods with different types of reward functions. FBB has the lowest performance on all datasets, because this method only cares about the buffer status on the viewer side, which is not flexible in the MVP 360° interactive video systems. FRB achieves a better reward than FBB, due to the prediction of network throughput. BD utilizes the prediction feedbacks of future throughput, head movements, and viewpoint selection, increasing on average 17.5% improvement. Compared with the BD method, iView has much higher rewards with the two reward functions, achieving at least 27.2%, 15.4%, and 2.8% improvements on the three datasets. Another observation is that the reward of iView on IRISA dataset is the lowest than those on another two datasets. We think that the reason includes two folds: (1) IRISA dataset only has one video trace with a short duration; (2) the tile setting is 3x2, which reduces the benefit from tile selection.

### D. Impacts of Tile Settings

As shown in Table I, three datasets have different tile settings, i.e., 10x10, 20x10, and 3x2 for NTHU, THU, and IRISA datasets, respectively. To better understand the impacts of different tile settings, we also implement an offline optimal solution to compare the performance. Note that the offline optimal solution cannot be used in a practical system, due to the requirement with full knowledge of the whole network and video traces. We plot the average reward of each tile on the three datasets and normalize the value using the results of the optimal solution, as shown in Figure 6. For ease of comparison, the results are normalized by the corresponding maximum values. We observe that the average rewards on the three datasets are similar, but different tile settings show significant impacts on the performance of each method. Note that there exists a significant margin on the performance of different methods in THU and NTHU datasets, but a small improvement of our method is achieved for the IRISA dataset. Because IRISA dataset has a smaller tile setting than another two datasets, generating a smaller tile selection space for every method and a similar performance on the average reward.

### E. Impacts of Different Reward Components

We also investigate the performance of each method on two reward components, i.e., Quality and Latency, in the reward definition (equation (10)). Figure 7 illustrates the values from the two reward components under different quality styles in THU dataset. In the linear design, iView reduces 18%-48% of

the latency part and increases 14%-15% of the quality part. In the logarithm design, it achieves 21% improvement on average compared with another three methods. iView can improve the quality of the tiles in the FoV and prefetch the tiles in a new viewpoint, which systematically focuses on the benefit from the quality of each tile and the penalty from rebuffering and download latency.

## VI. Conclusion

In this paper, we studied multi-viewpoint (MVP) 360° interactive video systems, which combine the features from 360° video and multi-viewpoint video systems and provide a better immersive experience for the viewers. To address the challenges in such a system, we for the first time presented iView to provide low latency MVP 360° interactive videos. The basic design of iView optimizes the tile selection problem by a prediction module and an optimization module. Although this optimization problem can be solved heuristically, it still faces several limitations in practice. We therefore enhanced it with the assistance of multimodal learning and deep reinforcement learning. To this end, iView integrates the prediction and optimization, generating an end-to-end solution between the viewing-related features and optimization objective, i.e., high quality and low latency. We compared the performance of iView with the revised versions of several state-of-the-art algorithms, considering the viewing characteristics of MVP 360° interactive videos. Three video datasets and one network dataset are used to conduct the trace-driven simulations. The results showed that iView effectively utilizes the features of video frames, networking throughput, head movements, and viewpoint selections, achieving at least 27.2%, 15.4%, and 2.8% improvements on the three video datasets, respectively. We are currently implementing iView to further evaluate the performance and investigate the issues in practice.

## Acknowledgment

The work of H. Pang and L. Sun is support by the NSFC under Grant No. 61521002, Key Research and Development Project under Grant No. 2018YFB1003703, Beijing Key Laboratory of Networked Multimedia and Tsinghua-Alibaba Cooperation Project (No.20172001689). The work of C. Zhang and J. Liu is supported by the NPRP grant # [8-519-1-108] from the Qatar National Research Fund (a member of Qatar Foundation) and a Canada NSERC Discovery Grant.

## References

[1] 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 *ACM NOSSDAV, 2017*.

[2] A. Hamza and M. Hefeeda, "Adaptive streaming of interactive free viewpoint videos to heterogeneous clients," in *ACM MMSys, 2016*.

[3] J. Le Feuvre and C. Concolato, "Tiled-based adaptive streaming using MPEG-DASH," in *ACM MMSys, 2016*.

[4] M. Almquist, V. Almquist, V. Krishnamoorthi, N. Carlsson, and D. Eager, "The prefetch aggressiveness tradeoff in 360 video streaming," in *ACM MMSys, 2018*.

[5] X. Corbillon, F. De Simone, G. Simon, and P. Frossard, "Dynamic adaptive streaming for multi-viewpoint omnidirectional videos," in *ACM MMSys, 2018*.

[6] X. Liu, Q. Xiao, V. Gopalakrishnan, B. Han, F. Qian, and M. Varvello, "360° innovations for panoramic video streaming," in *ACM HotNets, 2017*.

[7] A. Hamza, H. Ahmadi, S. Almowuena, and M. Hefeeda, "QoE-fair adaptive streaming of free-viewpoint videos over LTE networks," in *Thematic Workshops of ACM Multimedia, 2017*.

[8] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[10] Z.-z. Lan, L. Bao, S.-I. Yu, W. Liu, and A. G. Hauptmann, "Double fusion for multimedia event detection," in *International Conference on Multimedia Modeling*. Springer, 2012, pp. 173–185.

[11] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in neural information processing systems*, 2000, pp. 1008–1014.

[12] 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 *ACM NOSSDAV, 2017*.

[13] C. Zhang, Q. He, J. Liu, and Z. Wang, "Exploring viewer gazing patterns for touch-based mobile gamecasting," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2333–2344, Oct 2017.

[14] D. Wang, Y. Peng, X. Ma, W. Ding, H. Jiang, F. Chen, and J. Liu, "Adaptive wireless video streaming based on edge computing: Opportunities and approaches," *IEEE Transactions on Services Computing*, pp. 1–1, 2018.

[15] N. Jiang, V. Swaminathan, and S. Wei, "Power evaluation of 360 vr video streaming on head mounted display devices," in *ACM NOSSDAV, 2017*.

[16] J. Chakareski, R. Aksu, X. Corbillon, G. Simon, and V. Swaminathan, "Viewport-driven rate-distortion optimized 360° video streaming," *arXiv preprint arXiv:1803.08177*, 2018.

[17] I. Ketykó, K. De Moor, T. De Pessemier, A. J. Verdejo, K. Vanhecke, W. Joseph, L. Martens, and L. De Marez, "QoE measurement of mobile youtube video streaming," in *ACM MoViD, 2010*.

[18] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over HTTP," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 325–338, Aug. 2015.

[19] H. Kellerer, U. Pferschy, and D. Pisinger, *Introduction to NP-Completeness of Knapsack problems*. Springer, 2004.

[20] K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos, and L. Tassiulas, "Caching and operator cooperation policies for layered video content delivery," in *IEEE INFOCOM 2016*.

[21] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *ACM HotNets, 2016*.

[22] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *ACM SIGCOMM, 2017*.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[24] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[25] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[26] C. Wu, Z. Tan, Z. Wang, and S. Yang, "A dataset for exploring user behaviors in vr spherical video streaming," in *ACM MMSys, 2017*.

[27] W.-C. Lo, C.-L. Fan, J. Lee, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "360° video viewing dataset in head-mounted virtual reality," in *ACM MMSys, 2017*.

[28] D. Pisinger, "A minimal algorithm for the multiple-choice knapsack problem," *European Journal of Operational Research*, vol. 83, no. 2, pp. 394–410, 1995.

[29] 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," *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 187–198, Aug. 2014.

[30] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming," in *ACM MMSys, 2011*.