# SAVG360: Saliency-aware Viewport-guidance -enabled 360-video Streaming System

Yinjie Zhang, Mingyuan Wu, Beitong Tian, Jiaxi Li, Bo Chen, Qian Zhou and Klara Nahrstedt
Department of Computer Science, University of Illinois Urbana-Champaign Urbana, IL
Email: {yinjie2, mw34, beitong2, jiaxili, boc2, qianz, klara}@illinois.edu

***Abstract*—The emergence of 360-video streaming systems has brought about new possibilities for immersive video experiences while requiring significantly higher bandwidth than traditional 2D video streaming. Viewport prediction is used to address this problem, but interesting storylines outside the viewport are ignored. To address this limitation, we present SAVG360, a novel viewport guidance system that utilizes global content information available on the server side to enhance streaming with the best saliency-captured storyline of 360-videos. The saliency analysis is performed offline on the media server with powerful GPU, and the saliency-aware guidance information is encoded and shared with clients through the Saliency-aware Guidance Descriptor. This enables the system to proactively guide users to switch between storylines of the video and allow users to follow or break guided storylines through a novel user interface. Additionally, we present a viewing mode prediction algorithms to enhance video delivery in SAVG360. Evaluation of user viewport traces in 360-videos demonstrate that SAVG360 outperforms existing tiled streaming solutions in terms of overall viewport prediction accuracy and the ability to stream high-quality 360 videos under bandwidth constraints. Furthermore, a user study highlights the advantages of our proactive guidance approach over predicting and streaming of where users look.**

***Index Terms*—360-video, Viewing mode prediction, Viewport guidance, Tile-based adaptive streaming**

## I. Introduction

In recent years, 360-videos have become increasingly popular due to their ability to provide viewers with an immersive multimedia experience that allows them to change viewing directions within a 360-degree sphere space. This trend has been driven by the rapid development of 360-degree capturing devices and Head Mounted Displays (HMDs), as well as the availability of 360-video streaming services on popular online and on-demand platforms such as Twitch and YouTube. Consequently, there has been a growing demand for high-quality 360-video content, prompting research into how to deliver such content efficiently in streaming systems, including viewport guidance systems.

Users can consume 360-degree content through a 2D display or a Virtual Reality HMD. However, in both cases, viewers can only see a limited portion of the 360-degree content within their viewport, which is approximately 20% of the total content. This means that, to display a similar number of pixels in the limited-sized viewports, the entire 360-video requires a much higher resolution than conventional 2D videos, resulting in high bandwidth requirements for streaming 360-videos.

To reduce bandwidth demands without compromising video quality in the user's potential viewport, researchers have proposed viewport prediction-based streaming systems. These systems, such as [1–5], stream only tiles within or nearby the predicted viewport with high-quality. However, these systems only utilize limited client-side perceived information, which includes only the user's historical viewports, resulting in users missing interesting content or storylines within the entire video. To address this issue, viewport guidance algorithms [6–10] have been proposed, which leverage video content information in the 360-degree sphere space across the entire video to guide users' viewport to interesting content. However, these algorithms either prohibit user interaction to choose their own preferred content and storylines or need to recompute future viewport paths online when users deviate from the guidance, making them less feasible for end-to-end streaming. This presents two major challenges: how to conduct viewport guidance in 360-video streaming and how to enhance video delivery with global guidance information.

To address these challenges, we present SAVG360, a Saliency-aware Viewport-guidance-enabled 360-video Streaming System for 2D displays. SAVG360 leverages an advanced media description called Saliency-aware Guidance Descriptor, which encodes pre-computed saliency information in video's meta data. The descriptor is requested by the clients, guiding users to storylines capturing the most salient regions in the video timeline. Additionally, SAVG360 offers a hybrid watching mode that allows users to switch between guidance mode and free mode. In guidance mode, SAVG360 actively guides the user's viewport, while in free mode, users can manually switch their viewport. Additionally, the client employs a viewing-mode-based viewport prediction mechanism to enhance streaming with viewport guidance, which accurately predicts the user's future viewing mode, thereby improving prediction precision and tile-bitrate selection decisions.

Our contributions in this work are: (1) providing a novel user interface that allows users to switch between guidance and free mode to choose video storylines, (2) proposing an advanced media presentation description method that encodes guidance information in the Saliency-aware Guidance Descriptor, (3) proposing a novel Saliency-aware Viewport Guidance algorithm for 360-videos and a streaming system based on the guidance, and (4) proposing a viewing mode prediction mechanism that enhances the accuracy of viewport prediction and the performance of tile-bitrate selection.

This paper is organized as follows: Section II and III covers related work and backgrounds; Section IV details

the SAVG360 system architecture; Section V presents the saliency-aware viewport guidance algorithm; Section VI presents the viewing-mode-based viewport prediction and tile-bitrate selection mechanism; Section VII presents experimental results, and Section VIII concludes the paper.

## II. Related Work

### A. Saliency Detection in 360-video

Saliency Detection refers to the task of identifying visually prominent regions in images or videos. In [11], saliency detection methods are classified into two main categories: Salient Object Detection (SOD) and Eye Fixation Prediction (FP). The goal of SOD [12] is to detect uniformly highlighted salient objects and clear object boundaries that attract people attention. On the other hand, FP [13, 14] methods aim to predict sparse eye fixation locations that capture regions where people actually look in images or videos. Our work focuses on FP Saliency Detection, specifically in the context of 360-videos. Several efforts have been made to study 360-video saliency detection. [15] proposes a novel U-net-based spherical convolutional neural network for 360-videos, where the convolutional kernel is defined on a spherical crown. And in [16], a spatial-temporal network in CubePadding is proposed for 360-videos, using a video dataset with saliency heatmap annotations. These models can be directly incorporated into SAVG360 to compute saliency features in the videos.

### B. Viewport Prediction in 360-video

Viewport prediction involves predicting a user's future viewing direction based on viewport history of the user or other users' viewing trajectory. This enables the player to download only the content appearing in the viewport or reduce the video resolution outside of the viewport instead of fetching the whole 360-video. Some methods rely solely on viewport data from single viewer, such as [17] that uses linear regression and [18] that uses recent samples that monotonically increase or decrease. Other methods correlate the viewport trajectory of other users, such as [2], which uses multiple-object tracking algorithm to predict that people's viewports might follow object trajectories, and [1, 19], which simply relies on the historical view traces of users. While viewport prediction is a useful technique, one of its disadvantages is that wrong predictions can significantly affect the quality of experience of viewers, as the system may send the wrong content with high resolution and the correct content that users are interested in with low resolutions. In the SAVG360, we also employ viewport prediction, but combine it with viewing mode prediction to enhance the accuracy of traditional prediction methods.

### C. Viewport Guidance for 360-video

To improve user experience, viewport guidance aims to guide the user towards interesting content within the 360-degree sphere space. Unlike viewport prediction, which predicts the user's future viewing direction, viewport guidance automatically switches the viewport based on saliency information. Several methods have been proposed to achieve this goal. Pano2Vid [6] trains a capture worthiness model to extract a guided path from the 360-degree sphere space which generates a normal field-of-view (NFoV) video that resemble human-captured videos. The 360Pilot [7] uses object detection to generate a list of candidate objects and then selects the main object to adjust the NFoV using RNN. [9, 10] generate an optimal camera path based on saliency to navigate the user's viewport and enhance the watching experience. While these works perform viewport guidance locally on the client side, our work aims to integrate viewport guidance into an end-to-end 360-video streaming system.

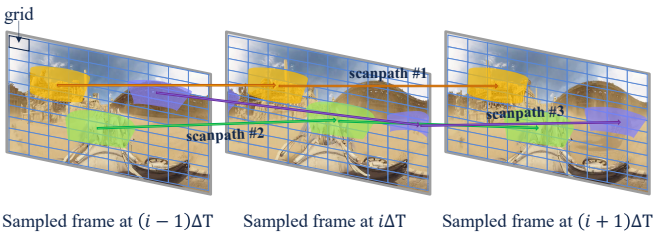



Fig. 1: Scanpaths, sampled frames and grids.

### D. 360-video Streaming

360-video streaming has become popular due to its immersive and interactive viewing experience. Researchers have explored various ways to enhance the streaming quality and user experience of 360-video. Some studies [20–24] have proposed novel video encoding techniques that prioritize salient regions of the 360-video to improve the overall quality of experience. Other studies [1, 2, 4, 17, 25, 26] have developed algorithms to optimize the streaming bitrate allocation and tile-based delivery of 360-videos. SAVG360 differs from these existing systems in that it delivers video for a hybrid watching mode that integrates viewport guidance, as opposed to free mode where users manually choose the viewport.

## III. Backgrounds

### A. 360-video

Traditionally, 360-videos are segmented temporally into **chunks**, each of which has the same time duration. Each chunk is further spatially divided into **tiles**, which share the same duration as the corresponding chunk, but only occupy a small spatial portion. Each tile is encoded into various quality levels, and can be downloaded and decoded independently. During video playback, the portion of the 360-degree spherical video content that is visible to the user is called the **viewport**. And the center of the viewport is called the **viewpoint**.

### B. Viewport Guidance

To facilitate viewport guidance, we need a data structure which stores the corresponding recommended next viewpoint for the current timestamp and viewpoint. However, continuous values of time and viewpoint location cannot be accommodated using finite space offer by any data structure. Therefore, temporal and spatial information is represented discretely. Specifically, as shown in Figure 1, we use **sampled frames**

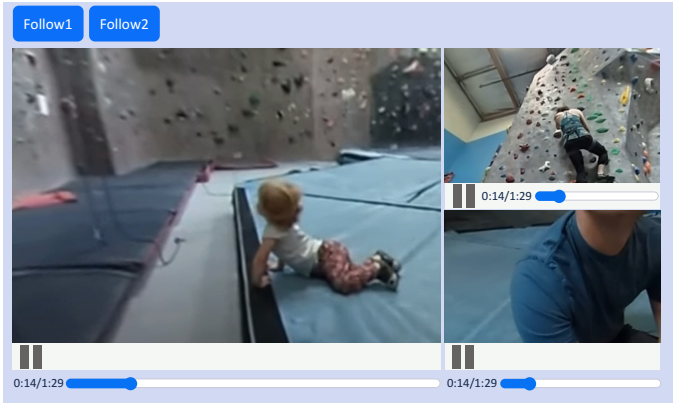



Fig. 2: User interface of SAVG360.

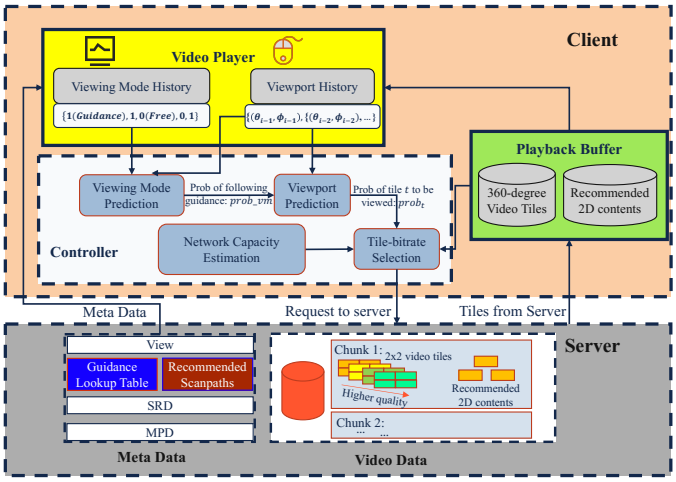



Fig. 3: The system architecture of SAVG360.

$F^{sampled} = \{f_1, ..., f_n\}$, which are sampled with time interval $\Delta T$, in order to divide time into several intervals. Additionally, we use **grids** to spatially divide the equirectangular frames into $W$ by $H$ cells. And a **viewpoint sequence** (or **scanpath**) represents a sequence of grids $P_{l,r} = \{p_l, p_{l+1}, ..., p_r\}$, which are the viewpoints corresponding to the sampled frames spanning from $f_l$ to $f_r$.

## IV. System Overview

### A. User Interface

Figure 2 shows the user interface of SAVG360. The interface is comprised of a saliency-aware viewport-guidance-enabled 360-video player which includes two main components: the **main window** and **side windows**.

The major viewing area is the **main window**, which displays the viewport of users. In the main window, *guidance mode* is deployed, where the system automatically guides the viewport based on a pre-computed saliency path descriptor. However, SAVG360 is designed to also provide users with freedom to choose their own viewing path instead of forcing them to follow one single guided path. To this end, we implement a *hybrid interactive viewing mode*: The users can switch from *guidance mode* to *free mode* by simply dragging a mouse to show different views. If there is no mouse activity for a certain duration, the system will switch back to the *guidance mode*.

In addition, SAVG360 has two **side windows** that compensate for the limited viewport span in the main window. The side windows display interesting video contents that are not visible in the current viewport. By clicking the "Follow 1 or 2" button, users can jump their viewport to the contents displayed in the first or second side window, respectively. This action automatically switches the users to *free mode*.

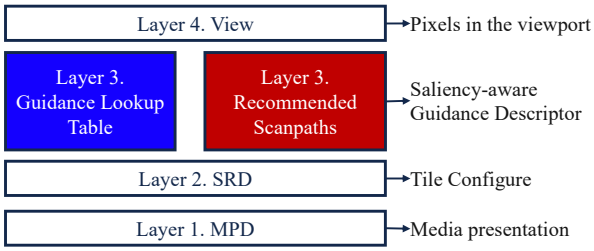
### B. System Architecture

Figure 3 shows a server and a client of SAVG360, which incorporates DASH with advanced meta data descriptors, a modified video storage mechanism, and a controller to deliver a high-quality user experience.

The server stores meta data and video data for each video. For *Meta data*, existing DASH-based video streaming systems typically provide MPD files with video chunk encoding information to support tile-based streaming and save bandwidth. However, as the popularity of 360-videos increased, the need for advanced meta data descriptors has emerged. To address this need, Spatial Relationship Descriptors (SRDs) have been introduced, allowing users to request only "visible" tiles. However, SRDs do not include information on guiding the user's viewport to interesting content in the 360-degree sphere space. To enable a saliency-aware viewport guidance mechanism in the player, we introduce a Saliency-aware Guidance Descriptor (SAGD) with two components: the **Guidance Lookup Table** and the **Recommended Scanpaths**. The server generates the SAGD, and SAGD components support automatic viewport switching on the main window and content selection on the side windows, respectively. Figure 4 illustrates the media descriptive layers of SAVG360, which incorporate advanced meta data descriptors to enhance user's viewing experience.

As discussed in Section IV-A, users are able to switch between two modes, namely the guidance mode and the free mode. In the free mode, users are allowed to freely drag the viewport to any location in the equirectangular coordinate. When the user switches back to guidance mode, the system recommends the next viewpoint based on pre-computed information stored in the **Guidance Lookup Table**. The lookup table is regularly queried at the beginning of each time interval and can also be queried when the system switches back from free mode to guidance mode. Based on the query results, the system guides the viewpoint at a uniform speed until it reaches the next recommended viewpoint at the beginning of the subsequent time interval. Additionally, the **Recommended Scanpaths** provide several (up to $m = 4$) viewpoint sequences for each video chunk. These sequences cover interesting and diverse content in the viewport, centered around the viewpoints during the corresponding chunk.

For *video data*, we have made modifications to the video storage of tile-based 360-video streaming systems to enable the implementation of side windows. In addition to conventional elements like chunks and tiles, recommended 2D con-




(a) Server-Side Media Descriptive Layers of SAVG360.

| Timestamp | Current Viewpoint | Next Viewpoint |
|---|---|---|
| $[0, \Delta T)$ | $(x_1, y_1)$ | $(x_1', y_1')$ |
| $[\Delta T, 2\Delta T)$ | $(x_2, y_2)$ | $(x_2', y_2')$ |
| … | … | … |

(b) Schema of Guidance Lookup Table.

| Chunk ID | Scanpath ID | Viewpoint Seq |
|---|---|---|
| Chunk #1 | Scanpath #1 | $\{(x_{111}, y_{111}), (x_{112}, y_{112}), \dots\}$ |
| Chunk #1 | Scanpath #2 | $\{(x_{121}, y_{121}), (x_{122}, y_{122}), \dots\}$ |
| … | … | … |

(c) Schema of table for Recommended Scanpaths.

Fig. 4: Meta data representation of SAVG360.

tent, i.e., the projected 2D video of each viewpoint sequence in the Recommended Scanpaths, is encoded at the lowest available quality level and added to the video data.

The client in SAVG360 is similar to typical DASH clients, but with modifications to the playback buffer and an advanced controller. In the *playback buffer*, recommended 2D contents are stored alongside downloaded tiles and queued for playback on the side windows. When a video chunk starts playing, the player queries the Recommended Scanpaths meta data to locate recommended 2D contents that are not visible within the user's current viewport, and displays them in the side windows. In addition, an *advanced viewing-mode-driven controller* is implemented in DASH clients of SAVG360. When the client receives meta data from the server, it performs a series of functions within the control plane. Firstly, it predicts user's future viewing mode based on the user's viewing mode and viewport history. Then, it uses both the Guidance Lookup Table and traditional viewport prediction mechanisms to predict user's future viewport. Next, the client maps the predicted future viewport to tiles using SRD, and passes this tile-based future viewport representation to the tile-bitrate selection module. Finally, the tile-bitrate selection module uses MPD to request the appropriate quality of tiles and recommended 2D contents for future video chunks from the server.

## V. Saliency-Aware Viewport Guidance Algorithm

In this section, we present the saliency-aware viewport guidance algorithm that provides information stored in meta data for the hybrid viewing mode on the client-side. The guidance algorithm is structured into two key components. The first component involves a main-window guidance algorithm, which determines the optimal recommended viewpoint considering user's current viewport and timestamp. The second component encompasses a side-window scanpath recommendation algorithm aimed at generating recommended scanpaths for individual chunks.

### A. Main-window Guidance Algorithm

Utilizing saliency scores for each sampled frame, we have devised a comprehensive main-window guidance algorithm. This algorithm determines the subsequent recommended viewpoint for any given current viewpoint during each sampled frame, stored in the **Guidance Lookup Table**.

The objective of viewport guidance is to enhance the user's perception of saliency during video playback. To achieve this goal, we introduce the concept of the **accumulated perceived saliency score**. This score is calculated through the following steps: Given a continuous segment of sampled frames represented as $F_{l,r}^{sampled} = \{f_l, f_{l+1}, ..., f_r\}$, a corresponding sequence of viewpoints is denoted as $P_{l,r} = \{p_l, p_{l+1}, ..., p_r\}$. The accumulated perceived saliency score $SeqSal_{l,r}(P_{l,r})$ for this viewpoint sequence $P_{l,r}$ is computed as:

$$SeqSal_{l,r}(P_{l,r}) = \sum_{i=l}^{r} PSal_i(p_i) - \alpha \cdot \sum_{i=l}^{r-1} \frac{dist(p_{i+1}, p_i)}{\Delta T}. \quad (1)$$

In particular, to calculate the accumulated perceived saliency score for a given viewpoint sequence, we aggregate $PSal_i(p_i)$, which represents the weighted average saliency score of all grids within the viewport centered around each $p_i$. The weight diminishes as the distance from the grid to the viewport center increases. Furthermore, we introduce a penalty term to address the velocity at which the viewpoint shifts, thus averting abrupt transitions between different salient regions. This penalty term is modulated by parameter $\alpha$, controlling the degree of smoothness's impact on the overall score.

The problem of main-window guidance can be defined as identifying a sequence of viewpoints $P_{i,n} = \{p_i, p_{i+1}, ..., p_n\}$ in the sampled frames $F_{i,n}^{sampled} = \{f_i, f_{i+1}, ..., f_n\}$ that maximizes the accumulated perceived saliency score $SeqSal_{i,n}(P_{i,n})$, given a current timestamp within the time interval $[(i-1)\Delta T, i\Delta T)$ and a user viewpoint situated at the equirectangular grid $p_i$. This problem can be effectively solved through dynamic programming, which calculates the optimal sequence $P_{i,n}^* = \{p_i, p_{i+1}^*, ..., p_n^*\}$ with the highest accumulated score. The dynamic programming process operates in reverse, from the last to the initial sampled frame. Critical information from the process, including the time interval $[(i-1)\Delta T, i\Delta T)$, the current viewpoint $p_i$, and the subsequent viewpoint in the optimal sequence $p_{i+1}^*$, is stored in the Guidance Lookup Table, which is shown in Fig 4b.

### B. Side-window Scanpath Recommendation

The proposed main-window guidance algorithm is designed to direct users toward interesting content within 360-videos, considering constraints related to viewport movement. However, these constraints could potentially result in the algorithm overlooking interesting events that may occur in the near future but are situated far from the current viewport in the main

window. To address this limitation, SAVG360 incorporates side windows, which are employed to display interesting content that currently lies beyond the viewport's scope. As elaborated in Section IV-B, the creation of side windows is facilitated by the utilization of the **Recommended Scanpaths** stored in the video meta data. Notably, each chunk contains several viewpoint sequences in the Recommended Scanpaths. These sequences are sorted in order of their accumulated perceived saliency scores. Consequently, when a chunk is about to play, the relevant video content from the first two viewpoint sequences, not currently visible in the viewport, are retrieved and presented within the two side windows.

The problem of formulating recommended viewpoint sequences for each chunk involves identifying several sequences with maximum accumulated perceived saliency scores and limited overlapping regions. The intuition behind is to make video contents consumed by users capture more salient and diverse regions in all candidate viewing paths. Dynamic programming is adopted to compute and store, from the last frame to the initial frame, the optimal recommended path with the highest accumulated score. In the system implementation, the generated viewpoint sequences are stored within the Recommended Scanpaths, as shown in Fig 4c.

## VI. Viewport Prediction and Rate Adaptation with Viewing Mode Prediction

In this section, we present a viewport prediction and rate adaptation mechanism in the controller (see details in Fig 3) of SAVG360. First, the controller utilizes the user's viewing mode and viewport history to predict the user's future viewport. Subsequently, the predicted viewport is passed to the tile-bitrate selection module, which decides appropriate quality of tiles for a future video chunk.

### A. Viewport Prediction

Based on viewport guidance, the client is capable of enabling a hybrid viewing mode for 360-videos. In this mode, users have the option to watch the video in guidance mode, which switches their viewport based on the Guidance Lookup Table. If the users continue to follow the guidance mode in the next several video chunks, i.e., they do not switch to free mode by dragging a mouse or change the viewport to side windows by clicking the "follow" button, the system can accurately predict the upcoming viewport by repetitively querying the Guidance Lookup Table for the subsequent viewport center.

To harness the advantages of the guidance mode, we present a two-stage algorithm towards viewport prediction in SAVG360. The first stage is a Deep-Learning-based approach that predicts the user's future viewing mode. Based on this viewing mode prediction result, the second stage predicts user's future viewport.

In the first stage, the algorithm takes information related to the user's current viewing mode and her viewport history as inputs. Specifically, the inputs include the user's viewing states while watching the last $hw$ (history window) chunks, and the states for each chunk include: (1) Viewing-mode switching flag: whether the user remains in the guidance mode from the beginning to the end of the chunk. (2) Ending viewport: the location of user's viewport at the end of the chunk. (3) User viewing time within individual recommended scanpath: duration for which the user's viewpoint in the main window follows each of the recommended scanpaths, and deviates from any recommended scanpaths. Then the 1D-CNN model forecasts the likelihood $prob\_vm$, that the user will continue to follow the guidance in several subsequent video chunks.

In the second stage, the user will either i) follow the guidance with probability $prob\_vm$, where a tile will be viewed if it is covered by the guidance or ii) switch into free mode with probability $1 - prob\_vm$, where the viewport will be predicted by traditional 360-degree viewport prediction methods. Therefore, we can predict the viewing probability of each tile $t$, denoted as $prob_t$, as shown in Equation 2:

$$prob_t = prob\_vm \cdot \mathbf{In}(t) + (1 - prob\_vm) \cdot prob\_vp_t \quad (2)$$

where $\mathbf{In}(t)$ is 1 if tile $t$ is visible to the user when the user follows the guidance and 0 otherwise. $prob\_vp_t$ represents the viewing probability of tile $t$ predicted by a traditional method, e.g., Linear Regression [17] and Navigation Graph [1].

### B. Tile-bitrate Selection

In this section, we formulate the tile-bitrate selection as an optimization problem, which is supplied with Viewport Prediction results, estimated throughput, and current buffer status. Let us consider a video chunk containing $M$ tiles, each of which has $Q$ quality levels with corresponding bitrates $r_{t,j}$ and perceived quality $u_{t,j}$ for the $t$-th tile at the $j$-th quality level. The goal is to determine the optimal quality level $l_t$ for each tile, which maximizes the sum of perceived quality while adhering to the constraints of throughput and buffer. This is defined as:

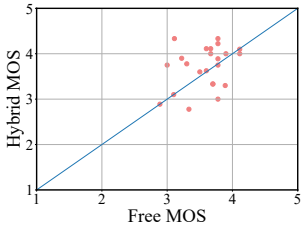
$$\begin{aligned} \max_{l_t \in [1,Q], \forall t} \quad & \Sigma_{t=1}^{M} prob_t \cdot u_{t,j} \\ \text{s.t.} \quad & \Sigma_{t=1}^{M} r_{t,j} \leq BW \end{aligned} \quad (3)$$

Here, $prob_t$ represents the viewing probability of each tile, which is estimated using the proposed viewing-mode-based viewport prediction mechanism. To ensure seamless playback, the constraint $BW$ is set to be the product of the estimated throughput and the current buffer size, where the throughput is estimated using the throughput predictor of RobustMPC [27]. The problem can be solved as a Multiple-choice Knapsack problem [28], which can be efficiently solved by dynamic programming [19]. The computational complexity of the dynamic programming solution is $O(BW \cdot M \cdot Q)$, and $BW$ can be discretized to reduce the decision space.

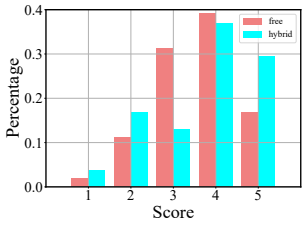
## VII. Performance Evaluation

### A. Methodology

**Dataset:** the video dataset used in our study consists of 23 360-videos sourced from two datasets: [6] and [29]. To generate meta data for these videos, we apply pre-computed viewport guidance with a time interval length of $\Delta T$ = 2 seconds and a grid division of $W = 45$ and $H = 80$. The videos




(a) MOS on each video for free mode and hybrid mode.




(b) Rating distribution for both free mode and hybrid mode.

Fig. 5: Survey result for free mode and hybrid mode.

are then divided into chunks lasting 2 seconds each, which are further divided into tiles of various configurations, including 2x4, 4x6, and 6x8. We encode each tile independently into five different qualities using ffmpeg [30]. And we use PSNR as the perceived quality $u_{t,j}$ in the tile-bitrate selection module.

**Survey-based evaluation:** we have implemented SAVG360 using the code from existing 360-video player [31] and made SAVG360 available for participants in the evaluation through a web interface. Each participant watched each of the collected videos and contributed two viewport trajectories, one under free mode and the other under hybrid mode, and assigned grades to them. The user IDs, trajectories, and scores of ten participants were stored in our database.

**Trace-driven simulation:** To evaluate the two-stage viewport prediction algorithm, we simulate the network that transmits the video data using real network traces from the 3G/HSDPA [32] and Oboe [33] datasets. In the simulation, we compare the two-stage algorithm with baselines. The experiment also serves as an ablation study to show the importance of the viewing mode prediction (the first stage).

**Hyper-parameter settings:** During the training process, we use 80% of the viewing trajectories from all users in the user study to train the CNN model for viewing mode prediction, as well as the model related to viewport prediction (Linear Regression (LR) and Navigation Graph (NG)), while the remaining 20% are utilized for evaluating the performance of the proposed viewing-mode-based viewport prediction. It's worth noting that our proposed viewing mode prediction aims to optimize viewport prediction. In contrast, systems like Pano [20] and SalientVR [21] focus on enhancing video delivery post viewport prediction. Thus, we compare our viewing mode prediction with viewport-based methods, and our viewing mode prediction can be seamlessly integrated with Pano and SalientVR. We train the viewing mode prediction model to look ahead $lookahead = 1, 2, 3$ chunks using the mode-based states and viewport history of the most recent $hw = 5$ chunks. For the viewport prediction method LR, we use viewport history in the past 10 seconds to predict the future viewport in 0.4 seconds, with the prediction result being used iteratively to predict the viewport in the distant future. Additionally, we set the client-side buffer limit to be 6 seconds, which causes the controller to pause downloading to drain the buffer when the buffer size exceeds the limit, which means the controller makes downloading requests of at most 3 chunks after the current chunk.

| | TP | FP | TN | FN |
|---|---|---|---|---|
| Lookahead=1 | 58.2% | 5.8% | 32.1% | 3.9% |
| Lookahead=2 | 56.3% | 7.7% | 33.0% | 3.0% |
| Lookahead=3 | 53.5% | 9.0% | 33.8% | 3.7% |

TABLE I: Performance of Viewing Mode Prediction: TP(True Positive), FP(False Positive), TN(True Negative), FN(False Negative).

### B. Survey-based Evaluation

Figure 5a displays the Mean Opinion Score (MOS) ratings provided by users for each video. It can be observed that in 16 out of 23 videos, the hybrid viewing mode receives higher MOS than the free mode, while in the remaining 7 videos, the free mode obtains higher MOS. This discrepancy may be attributed to the imperfection of saliency and its potential to provide inaccurate guidance.

Figure 5b illustrates the distribution of user ratings for free mode and hybrid mode. The proposed hybrid mode receives 100% more full scores (5) than free mode, but it also obtains more low scores. Thus, it can be concluded that viewport guidance has the potential to enhance the viewing experience, but it may also have a negative impact at times.

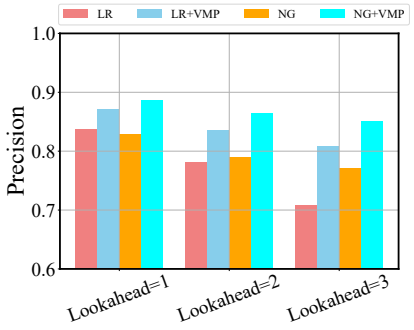
### C. Viewport Prediction

Before considering network variation, we evaluate the performance of viewport prediction methods (NG, LR) and assess how they can benefit from being combined with Viewing Mode Prediction (VMP). In this regard, the performance of viewing mode (free or guidance) prediction is measured to determine the extent to which it can improve the accuracy of viewport prediction by predicting when users are following the guidance mode. Table I presents the performance of VMP on the test set with varying lookahead values (1, 2, and 3 chunks), where a positive prediction is made when the probability that the user will continue following guidance ($prob\_vm$) is greater than 0.5. The results show that the performance of VMP is influenced by the lookahead value, as higher values result in increased FP rates and decreased TP rates. Nonetheless, the model still achieves a high recall rate of 94% even with a lookahead value of 3 chunks, demonstrating its ability to accurately identify when users are following the guidance mode. This implies that the combination of VMP with viewport prediction methods can lead to improved accuracy when users adhere to guidance mode.

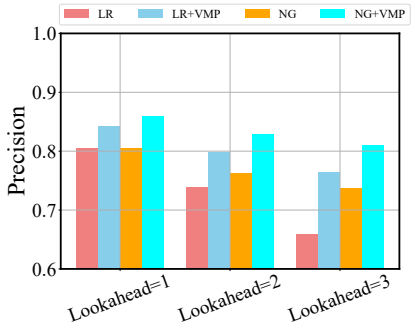
We evaluate viewport prediction algorithms by:

$$precision = \frac{\Sigma_{t=1}^{M} prob_t \cdot g_t}{\Sigma_{t=1}^{M} g_t}, \tag{4}$$

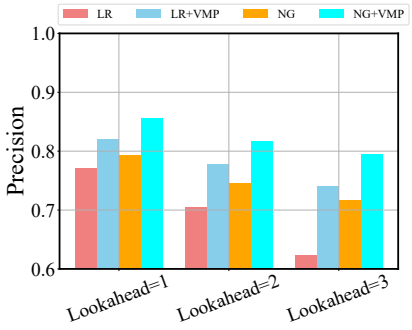
where $g_t$ is the ground truth that has 0 for non-visible tiles and 1 for visible tiles, and $prob_t$ is a predicted probability that tile $t$ will be shown in the future chunk. Figure 6 illustrates the mean precision of viewport prediction methods for different lookahead values (1, 2, and 3 chunks) and tile configurations.




(a) The precision of viewport prediction under tile configuration 2x4.

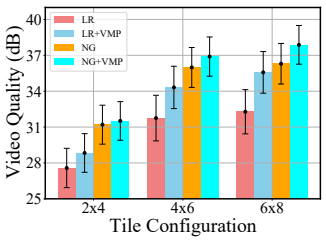



(b) The precision of viewport prediction under tile configuration 4x6.

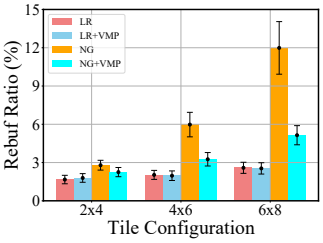



(c) The precision of viewport prediction under tile configuration 6x8.

Fig. 6: The precision of different viewport prediction methods including LR(Linear Regression), LR+VMP(Viewing Mode Prediction), NG(Navigation Graph), NG+VMP, under various tile configurations.

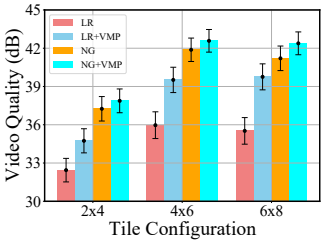



(a) V-PSNR.

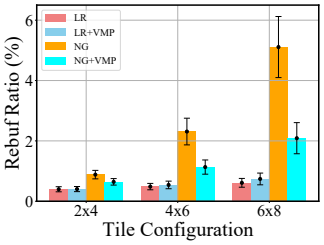



(b) Rebuffer ratio.

Fig. 7: V-PSNR and rebuffer ratio in HSDPA with different tile configurations. (Error bars show 95% confidence intervals.)




(a) V-PSNR.




(b) Rebuffer ratio.

Fig. 8: V-PSNR and rebuffer ratio in Oboe with different tile configurations. (Error bars show 95% confidence intervals.)

The results demonstrate that combining VMP with LR and NG can improve prediction accuracy by 4%-20% due to the high recall rate of VMP, which significantly enhances the accuracy of viewport prediction when users follow the guidance. In particular, LR performs well for 1 chunk future prediction, but its accuracy drops significantly as the lookahead value or number of tiles increases. However, combining LR with VMP enhances prediction accuracy for both near and distant future predictions and for all tile configurations, with the improvement increasing with the increase in lookahead value or number of tiles. For NG, the accuracy is relatively stable across different lookahead values and tile configurations, but combining it with VMP can still lead to an improvement in precision by 4%-11%.

### *D. Quality of Experience (QoE)*

To assess the impact of viewing mode prediction on QoE under various network conditions, we evaluate the performance of different viewport prediction methods using average viewport PSNR (V-PSNR) and rebuffer ratio.

The results of V-PSNR and rebuffer ratio achieved by SAVG360 with different viewport prediction methods in HSDPA traces are presented in Figure 7. The results reveal that using LR to predict users' viewports results in a decrease in average video quality due to its inaccurate prediction. However, the use of VMP along with LR significantly improves the V-PSNR in all tile configurations, achieving an increase of 10% under 6x8 and 8% under 4x6, without adding rebuffer. NG tends to predict and stream more tiles, resulting in high rebuffer ratios when the video is split into many tiles (4x6 or 6x8). However, when combined with VMP, the rebuffer ratio can be significantly reduced by 47%-58%.

Figure 8 shows the metrics of different viewport prediction methods under Oboe traces. We can observe that both LR+VMP and NG+VMP achieve higher V-PSNR and lower rebuffer ratios compared to LR and NG, respectively, for all tile configurations. Specifically, when using a tile configuration of 6x8, the combination of LR with VMP results in an improvement of 4.2 dB in V-PSNR, while only slightly increasing the rebuffer ratio by 1.2%, as compared to using LR alone. Similarly, when using NG with the same tile configuration, the rebuffer ratio is initially high at 5.1%. However, when combined with VMP, the ratio is significantly reduced to only 2%, with a slight increase in V-PSNR of 1.2 dB. These results demonstrate the effectiveness of the proposed viewing mode prediction mechanism in improving the performance of 360-video delivery and user watching experience.

## VIII. Conclusion

Streaming 360-videos requires a much higher data rate compared to non-360-videos. Previous works have attempted to solve this problem by predicting user viewports, but this

approach may result in a suboptimal user experience due to missed interesting storylines. To address this issue, we present SAVG360, a novel Saliency-aware Viewport-guidance-enabled 360-video Streaming System that integrates viewport guidance into the 360-video streaming process. Experimental results demonstrate that SAVG360 yields higher MOS compared to free mode, and viewing mode prediction can significantly improve 360-video delivery and quality of experience.

## ACKNOWLEDGMENT

This work was funded by the National Science Foundation under grant contracts NSF 1835834, NSF 1900875, NSF 2106592. Any results and opinions are our own and do not represent views of National Science Foundation.

## REFERENCES

[1] J. Park and K. Nahrstedt, "Navigation graph for tiled media streaming," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 447–455. [Online]. Available: https://doi.org/10.1145/3343031.3351021

[2] J. Park, M. Wu, K.-Y. Lee, B. Chen, K. Nahrstedt, M. Zink, and R. Sitaraman, "Seaware: Semantic aware view prediction system for 360-degree video streaming," in *2020 IEEE International Symposium on Multimedia (ISM)*, 2020, pp. 57–64.

[3] A. T. Nasrabadi, A. Samiei, and R. Prakash, "Viewport prediction for 360° videos: A clustering approach," in *Proceedings of the 30th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, ser. NOSSDAV '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 34–39. [Online]. Available: https://doi.org/10.1145/3386290.3396934

[4] F. Qian, B. Han, Q. Xiao, and V. Gopalakrishnan, "Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices," in *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*, ser. MobiCom '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 99–114. [Online]. Available: https://doi.org/10.1145/3241539.3241565

[5] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, 2016, pp. 1–6.

[6] W. Bares, V. Gandhi, Q. Galvane, and R. Ronfard, "Pano2vid: Automatic cinematography for watching 360○ videos," in *Proc. Eurograph. Workshop Intell. Cinematogr. Editing*, 2017, p. 1.

[7] H.-N. Hu, Y.-C. Lin, M.-Y. Liu, H.-T. Cheng, Y.-J. Chang, and M. Sun, "Deep 360 pilot: Learning a deep agent for piloting through 360 sports videos," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 1396–1405.

[8] Y.-C. Su and K. Grauman, "Making 360 video watchable in 2d: Learning videography for click free viewing," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 1368–1376.

[9] K. Kang and S. Cho, "Interactive and automatic navigation for 360 video playback," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–11, 2019.

[10] S. Cha, J. Lee, S. Jeong, Y. Kim, and J. Noh, "Enhanced interactive 360° viewing via automatic guidance," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 5, pp. 1–15, 2020.

[11] W. Wang, Q. Lai, H. Fu, J. Shen, H. Ling, and R. Yang, "Salient object detection in the deep learning era: An in-depth survey," 2019. [Online]. Available: https://arxiv.org/abs/1904.09146

[12] G. Ma, S. Li, C. Chen, A. Hao, and H. Qin, "Stage-wise salient object detection in 360 omnidirectional image via object-level semantical saliency ranking," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 12, pp. 3535–3545, 2020.

[13] 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.

[14] H. Lv, Q. Yang, C. Li, W. Dai, J. Zou, and H. Xiong, "Salgcn: Saliency prediction for 360-degree images based on spherical graph convolutional networks," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 682–690.

[15] Z. Zhang, Y. Xu, J. Yu, and S. Gao, "Saliency detection in 360° videos," in *The European Conference on Computer Vision (ECCV)*, September 2018.

[16] H.-T. Cheng, C.-H. Chao, J.-D. Dong, H.-K. Wen, T.-L. Liu, and M. Sun, "Cube padding for weakly-supervised saliency prediction in 360° videos," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1420–1429.

[17] 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 *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 315–323.

[18] L. Sun, F. Duanmu, Y. Liu, Y. Wang, Y. Ye, H. Shi, and D. Dai, "A two-tier system for on-demand streaming of 360 degree video over dynamic networks," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 43–57, 2019.

[19] L. Xie, X. Zhang, and Z. Guo, "Cls: A cross-user learning based system for improving qoe in 360-degree video adaptive streaming," in *Proceedings of the 26th ACM international conference on Multimedia*, 2018, pp. 564–572.

[20] Y. Guan, C. Zheng, X. Zhang, Z. Guo, and J. Jiang, "Pano: Optimizing 360 video streaming with a better understanding of quality perception," in *Proceedings of the ACM Special Interest Group on Data Communication*, 2019, pp. 394–407.

[21] 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 *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, 2022, pp. 542–555.

[22] D. Baek, H. Kang, and J. Ryoo, "Sali360: design and implementation of saliency based video compression for 360 video streaming," in *Proceedings of the 11th ACM Multimedia Systems Conference*, 2020, pp. 141–152.

[23] M. Xiao, C. Zhou, Y. Liu, and S. Chen, "Optile: Toward optimal tiling in 360-degree video streaming," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 708–716.

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

[25] Y. Zhang, P. Zhao, K. Bian, Y. Liu, L. Song, and X. Li, "Drl360: 360-degree video streaming with deep reinforcement learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 1252–1260.

[26] 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 *Proceedings of the 9th ACM multimedia systems conference*, 2018, pp. 162–173.

[27] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, "A control-theoretic approach for dynamic adaptive video streaming over http," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 325–338.

[28] P. Sinha and A. A. Zoltners, "The multiple-choice knapsack problem," *Operations Research*, vol. 27, no. 3, pp. 503–515, 1979.

[29] A. Nguyen and Z. Yan, "A saliency dataset for 360-degree videos," in *Proceedings of the 10th ACM Multimedia Systems Conference*, 2019, pp. 279–284.

[30] F. Bellard, "Ffmpeg multimedia system," *FFmpeg.[Last accessed: November 2015]. https://www. ffmpeg. org/about. html*, 2005.

[31] "Simple360Player Demo — slawrence.github.io," https://slawrence.github.io/simple-360-player/, [Accessed 29-08-2023].

[32] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Commute path bandwidth traces from 3g networks: analysis and applications," in *Proceedings of the 4th ACM Multimedia Systems Conference*, 2013, pp. 114–118.

[33] Z. Akhtar, Y. S. Nam, R. Govindan, S. Rao, J. Chen, E. Katz-Bassett, B. Ribeiro, J. Zhan, and H. Zhang, "Oboe: Auto-tuning video abr algorithms to network conditions," in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, 2018, pp. 44–58.