SEAWARE: Semantic Aware View Prediction System for 360-degree Video Streaming

Jounsup Park[1], Mingyuan Wu[2], Eric Lee[2], Bo Chen[2], Klara Nahrstedt[2], Michael Zink[3], Ramesh Sitaraman[4]
[1]Department of Electrical Engineering, University of Texas at Tyler, Tyler, TX, 75799
[2]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801
[3]Department of Electrical and Computer Engineering, University of Massachusetts Amherst, Amherst, MA 01003
[4]College of Information and Computer Sciences, University of Massachusetts Amherst, Amherst, MA 01003
Email: jpark@uttyler.edu, {mw34, kylee5, boc2, klara}@illinois.edu, zink@ecs.umass.edu, ramesh@cs.umass.edu

***Abstract*—Future view prediction for a 360-degree video streaming system is important to save the network bandwidth and improve the Quality of Experience (QoE). Historical view data of a single viewer and multiple viewers have been used for future view prediction. Video semantic information is also useful to predict the viewer's future behavior. However, extracting video semantic information requires powerful computing hardware and large memory space to perform deep learning-based video analysis. It is not a desirable condition for most of client devices, such as small mobile devices or Head Mounted Display (HMD). Therefore, we develop an approach where video semantic analysis is executed on the media server, and the analysis results are shared with clients via the Semantic Flow Descriptor (SFD) and View-Object State Machine (VOSM). SFD and VOSM become new descriptive additions of the Media Presentation Description (MPD) and Spatial Relation Description (SRD) to support 360-degree video streaming via the DASH framework. Using the semantic-based approach, we design the Semantic-Aware View Prediction System (SEAWARE) to improve the overall view prediction performance. The evaluation results of 360-degree videos and real HMD view traces show that the SEAWARE system improves the view prediction performance and streams high-quality video with limited network bandwidth.**

I. Introduction

360-degree videos provide a richer multimedia experience than conventional videos by allowing viewers to watch any angle in the 360 content and have access within a fraction of a second to multiple views of a scene instead of a single view only. Like Video-on-Demand (VoD) streaming services, 360-degree videos can be streamed over the Internet. However, to deliver high-quality content, 360-degree videos require more bandwidth and lower latency than conventional videos. It was shown that 360-degree video viewers see 20% of the viewing area at one time because of the limited viewing angle, and the server needs to transmit the 360-degree videos with higher resolution than conventional videos to provide similar Quality of Experience (QoE). However, the bandwidth can be saved by transmitting the pixels of the desired view if the server can receive the client's view information in real-time. Nevertheless, due to the network latency between the server and the client, it is difficult for the server to receive the clients' desired view information in real-time. Therefore, future view prediction is necessary for the 360-degree video streaming systems to overcome the latency problem and improve QoE.

Historical viewing patterns of 360-degree video content, gathered either from a single viewer [1] or from multiple viewers [2], are important information for the prediction of future views. Linear Regression [3][4] using the single viewer's past viewing patterns is useful to predict the near-term future view since most viewers tend to move their head continuously. In addition, a view prediction method using multiple viewers' view history information is introduced in [2]. Since viewers are attracted to the interesting parts of videos, multiple viewers may have similar viewing patterns. In [5], the Navigation Graph (NG) describes both the temporal relationship among 360-degree video segments and the spatial relationship among 360-degree video tiles using both a single viewer's and multiple viewers' viewing patterns. NG achieves better prediction performance than other history-based 360-degree video view prediction algorithms. However, these view prediction algorithms rely only on viewing history data and cannot effectively utilize the correlation between video semantic information and viewing patterns.

Video semantic information describes objects or events within a 360-degree video, and it can improve future view prediction since viewers are attracted by objects or events in the video. For example, in a monster movie [6], the monster may attract viewers' interest and guide their head movement. A deep-learning-based view tracking algorithm [7] is introduced to analyze video content and to predict a user's movement. However, object detection and view tracking require a powerful computing machine to execute deep learning algorithms. Moreover, the view tracking approaches in [7][8][9][10] provide possible future trajectories, but they are more suitable for auto-pilot applications. They require additional processing to be combined with a tiled-video rate adaptation algorithm [11][12][13], while other existing view prediction systems [1][2][3][4][14], designed for streaming systems, can be efficiently combined with the existing video streaming platforms.

Dynamic Adaptive Streaming over the HTTP (DASH) [15] has been successfully used for many existing video streaming platforms because of its client-centric structure. DASH clients can control video quality based on the network condition and their device's features, such as display size, playback buffer status, and battery condition. The media server is only

required to provide Media Presentation Description (MPD) and the client's request bitrates for video segments described in the MPD. As 360-degree videos are becoming more popular, Spatial Relationship Descriptor (SRD) [16] is introduced to support tile-based videos, such as multiview videos or omni-directional videos [17][18]. The high-resolution video frames are cut into smaller tiles. and the tiles are encoded independently using legacy video encoders, such as HEVC [19]. SRD includes the tile configurations to allow the client to render a view of the received tiles. However, there is no descriptor to describe the semantic information of the videos, so the client devices are solely responsible for video analysis to predict future views.

In this paper, we propose the advanced media presentation description method and a semantic-aware view prediction algorithm. We show that the client devices can perform view prediction more efficiently and accurately if the media server provides the semantic information, because the viewers will "see where" interesting objects are located. The proposed Semantic-Aware View Prediction System (SEAWARE) performs the video semantic analysis and records the semantic information on the server. The server provides a part of the semantic information to the clients when the clients request the information. Client devices perform future view prediction using the information provided by the server.

We can summarize the contributions of this work as follows. (1) We propose the advanced media presentation description method that encapsulates new semantic information of 360-degree video content via the Semantic Flow Description (SFD) and View-Object State Machine (VOSM). (2) SEAWARE works on top of the DASH standard and utilizes MPD, SRD, SFD and VOSM information in a coordinated manner. (3) The SEAWARE system utilizes more computing power on the server-side to get more accurate video analysis data. (4) The experimental results show that the SEAWARE system has more efficient and accurate view prediction performance than previous approaches.

This paper is organized as follows. Section II details the semantic-aware 360-degree video streaming systems. Section III presents our view prediction algorithm using advanced meta data. Section IV presents experimental results, and Section V concludes the paper.

II. Semantic-Aware View Prediction System

A. Semantic Information of 360-degree Videos

There are many methods to analyze semantic information of videos, but this information is rarely utilized for view prediction. Some works [7], [20] use a deep learning algorithm to analyze the video and utilize this information to predict viewers' motion, but most of mobile devices are not powerful enough to process deep learning algorithms in real-time. Moreover, the server has whole video segments with the original video quality, while the clients rarely have high quality video segments. The server can provide much accurate semantic information for future segments. Figure 1 shows the relationship of view patterns and objects in the videos.

No	Content	Length	Category
1	Conan	2'44"	Performance
2	Ski	3'21"	Sport
3	Help	4'53"	Film
4	Conan	2'52"	Performance
5	Tahiti Surf	3'25"	Sport
6	The fight for Falluja	10'55"	Documentary
7	Cooking Battle	7'31"	Performance
8	LOSC Football	2'44"	Sport
9	The Last of the Rhinos	4'53"	Documentary

Table I: Video Dataset [21]

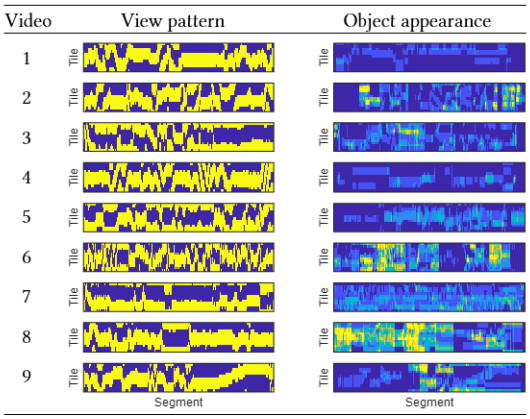


Figure 1: Viewing Patterns and Semantic Information

The videos are encoded with 12 column tiles in this specific example. We cut the videos in a vertical direction only to see how objects are moving in a horizontal direction. The first column of Figure 1 shows the real viewing patterns while a viewer is watching the videos [21] (listed on Table. I), where the yellow tiles represent the visible tiles and dark blue tiles represent non-visible tiles. The second column of Figure 1 shows the object appearances in the videos. Tiles including objects are color-coded to show where the objects appear. The different colors of tiles represent different objects and tiles in dark blue have no object.

Video-1 and 4 are Conan's talk show videos, therefore, the most objects (persons) are shown on stage during the whole video. Video-2 is a Skiing video recorded by one of the skiers and many other skiers appear and disappear frequently in the video. Video-6 is a Football video, therefore, there are many objects in the video and their movements are also very dynamic. Video-3 is a monster movie that has a monster chasing a woman and a police man. We can roughly see how major characters are moving in the video in the second column of the Figure 1. Video-9 is a documentary film and a Rhino is the major object in the video. We can observe that the viewers of video-1 and video-4 usually watch the tiles located in the center of the videos. The viewer of video-1 and video-4 tends to stay in the center of the video because there are objects on stage to watch. The viewer of video-2 also stays in the center of video most of the time, but it is difficult to determine whether the viewer is interested in watching objects

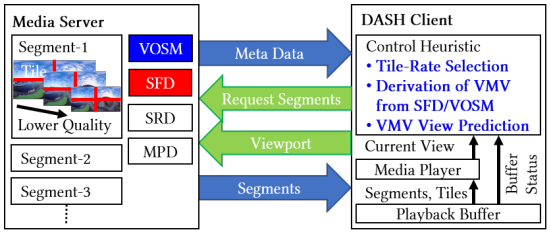


Figure 2: SEAWARE System

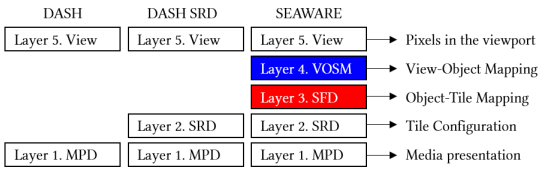


Figure 3: Server-Side Media Descriptive Layers

or more interested in watching other parts of the video, such as mountain or sky, because there are too many objects in the video. The viewer of video-3 has a dynamic viewing pattern, but we can find that the viewer's movement is correlated with objects' movements in the video. The viewing pattern and the object appearance are very similar to video-9, which means that the Rhino object in the video attracts the viewer the most.

B. SEAWARE System Architecture

We introduce the SEAWARE system to perform the view prediction using semantic information. SEAWARE is DASH-based 360-degree video streaming system using advanced meta data descriptors. Figure 2 shows a server and a client of SEAWARE. The server of SEAWARE analyzes videos to generate the Semantic Flow Descriptor (SFD), which records semantic information of 360-videos. The View-Object State Machine (VOSM) is added at the server to build a statistical model of viewers' movements, led by the objects. VOSM models the viewers' behaviors, captured by SFD, to understand which objects attract more viewers.

Figure 3 shows the new media presentation layers of the 360-degree video streaming system. Most of the existing video streaming systems including DASH have MPD files to provide video segments encoding (bitrate) information to the clients. MPD allows clients to request subset of the media to produce a "full 360 View" with video segments at the clients' desired bitrates. As 360-degree video has become more popular, SRD was introduced to stream tiled-media. By using SRD, clients can request a subset of tiles in the video segments to render a "visible tiles" only and save bandwidth. However, SRD does not include any information about viewing patterns of viewers (e.g., head movement positions) or semantic information of the videos (e.g., objects or scenes). Therefore, clients perform their view predictions by using their network conditions and their past tile-based view history.

Clients of SEAWARE are the same as DASH clients, but the control plane is much more advanced. SEAWARE clients receive MPD, SRD and SFD with VOSM information from the server to perform individual functions of the control plane as follows: Since clients can extract the semantic description regarding objects and scenes from SFD and VOSM, they do not need to analyze the video content themselves. Using VOSM per video segment, clients predict objects/scenes in the future view. Using SRD, clients map the future objects/scenes to tiles, gaining tile-based representations of future views, and pass the future tile-based view information to their rate selection modules. The clients' rate selection modules use MPD to request the appropriate quality of tiles in future segments from the server.

C. Semantic Flow Descriptor (SFD)

As shown in Figure 1, the view transition is correlated with object trajectories in the videos since the viewers tend to follow interesting objects/scenes of the video. An experiment in [2] shows that some tiles are more popular than other tiles because they encompass interesting points in the videos, which could be interesting objects, a salient part, and/or vanishing points. To find these points, machine learning technologies are mostly used that require a very powerful hardware performance, which is usually not available at mobile devices. The videos are stored at the servers which have better computing power than the mobile devices. Storing the whole video from the start to the end is another server's advantage when executing video analysis prior to any content viewing, while clients have only partial video segments that viewers watch during their content viewing. Therefore, it is a realistic assumption that the server performs video analysis when it receives any new 360-video content and prior to any content viewing by clients. The server can then encode the important semantic paths, and provide the semantic information to the clients right after the clients request the video segments. This semantic information aids clients to predict future views more efficiently without performing complicated video analysis at clients' sides.

We encode the information as a set of tiles in each segment. Therefore, when the clients receive the semantic information, clients directly identify which tiles have certain semantic information. It is different from using bounding boxes to indicate the object or event information. All tiles having overlap with bounding boxes should be received by clients to recover the objects. Therefore, encoding the information as a set of tiles helps reducing the amount of information to transmit and reduce the processing time to find overlaps between tiles and bounding boxes at the client-side. It is also compatible with DASH segments and tiles. Therefore, we define the Semantic Flow Descriptor (SFD) as a set of tiles in a segment that include objects. SFD is an advanced meta data that has an additional semantic information, **the objects description**. Therefore, the clients can get semantic information of the video by receiving SFD from the server. SFD indicates which objects exist in the segment and which tiles are required to display those objects. Figure 4 shows an example of SFD generation for three objects over three

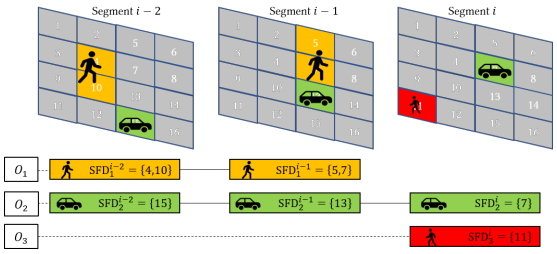


Figure 4: SFD

consecutive video segments. Every object is labeled with a set of tiles including this object. As the clients receive the SFD, clients identify which tiles have the objects in the segment. Moreover, clients can find out whether the current view includes tiles which include objects that the viewer is most likely interested in. Two objects are encoded as SFDs in the segment $i-2$. To locate the objects, SFD_j^{i-2} stores a set of tiles including the object j. These objects move to a new location in the following segment $i-1$, therefore, $SFD_j^{i-1}, \forall j$ have the same object index but a different set of tiles representing different locations. A new object can appear and an existing object can disappear in the future segments. In this example, the person disappears, the car remains, and a new person appears in the segment i.

D. View-Object State Machine (VOSM)

We use a state machine to build a view transition model incorporating objects' descriptions and states. A goal of VOSM is to build a viewing behavior model using semantic information of the video. In other words, VOSM is built to understand and record which objects the viewers' are interested in within a certain segment, and how viewers change their interests in the following segments. The states in the VOSM represent the set of objects the viewers are interested in, and transition probabilities between states represent the probabilities that the viewers change the state in the following segment.

Figure 5 shows an example of VOSM for three consecutive segments. Every segment has a different number of possible View-Object states with three objects, O_1, O_2 and O_3, which appear and disappear in the three segments. The View-Object states are indicating the objects in viewers' views, where the NULL state means that there is no objects in the view. There can be a maximum of eight states when there are three objects, but states are only generated when at least one viewer had visited the state. We assume that the viewers are interested in the object j when the view has an overlapping region with SFD_j. For example, at segment $i-2$, let us assume that 10 viewers watch the segment $i-2$. 2 viewers' views include none of the objects, 3 viewers' views include an object O_2, and 5 viewers' views include two objects O_1 and O_2. Therefore, three states exist in the segment $i-2$, which are $s_0^{i-2} = \{\Phi\}$, $s_1^{i-2} = \{O_2\}$, and $s_2^{i-2} = \{O_1, O_2\}$. In the following segment $i-1$, none of the viewers' views include the object O_2, and the object O_3 attracts 7 viewers while 2 of the viewers are still watching O_1 and 1 viewer is not watching any object.

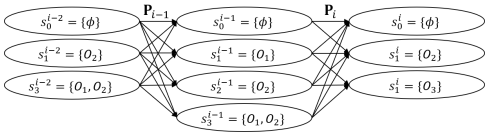


Figure 5: VOSM Generation

Therefore, there are four states generated for the segment $i-1$. Since there is only one object in the segment i, there are two possible states, which are $s_0^i = \{\Phi\}$ and $s_1^i = \{O_3\}$.

When a server receives a new 360-degree video, it can process the video to generate MPD, SRD, and SFD information. The server starts to generate the VOSM when viewers start to watch the 360-degree video. As more users watch the video, the VOSM can gather more data. Every video will have their distinctive VOSM to record viewers' viewing patterns regarding the objects in the videos.

Between states, the transition probability is defined as a probability that viewers change their states to other states in the following segment. P_i is a transition matrix from segment $i-1$ to i and we show P_i next as an example if Figure 5.

$$\mathbf{P}_i = \begin{Bmatrix} p(\mathrm{s}_0^i|\mathrm{s}_0^{i-1}) & p(\mathrm{s}_0^i|\mathrm{s}_1^{i-1}) & p(\mathrm{s}_0^i|\mathrm{s}_2^{i-1}) & p(\mathrm{s}_0^i|\mathrm{s}_3^{i-1}) \\ p(\mathrm{s}_1^i|\mathrm{s}_0^{i-1}) & p(\mathrm{s}_1^i|\mathrm{s}_1^{i-1}) & p(\mathrm{s}_1^i|\mathrm{s}_2^{i-1}) & p(\mathrm{s}_1^i|\mathrm{s}_3^{i-1}) \end{Bmatrix} \tag{1}$$

where the transition probabilities are defined as

$$p(\mathrm{s}_m^i|\mathrm{s}_c^{i-1}) = \frac{\text{number of clients change their state from } \mathbf{s}_c^{i-1} \text{ to } \mathbf{s}_m^i}{\text{number of clients visiting } \mathbf{s}_c^{i-1}}. \tag{2}$$

VOSM is stored in the media server and updated every time the server receives the view information from the clients. Multiple viewers' views information is required to build the transition probability matrix. The transition matrix P_i is provided to the clients as part of VOSM when the clients request segment i to inform clients how other viewers changed their state when they watched segment $i-1$ and segment i.

III. View Prediction Algorithm

Clients can move their view from segment i, which is the current segment that viewer is watching, to the view in the future segment $i+1$ by referring to SFD and VOSM received from the media server. VOSM models the behaviors of viewers in terms of the objects they have watched. Therefore, VOSM can also provide future information, i.e., it can assist in determining which objects the viewer will probably watch under the condition that the viewer is watching certain objects currently. To predict the tiles to be required for rendering future view, SFD helps to identify tiles including the objects the viewer is going to watch. However, it is not sufficient to predict the visible tiles in the future, because SFD only provides the index of the tiles that will include the objects, but actual view of the viewer could include background or surrounding tiles that represent the real view. Therefore, we introduce the View Motion Vector (VMV) that captures motion

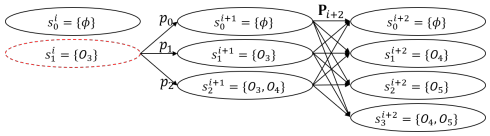


Figure 6: Future State Prediction using VOSM

directions of objects and therefore can move the current view and generate the future views.

A. Future State Prediction using VOSM

Figure 6 shows an example of predicting a state in segment $i+1$ given the current state s_1^i at the segment i using VOSM. The VOSM provides the probability that the viewer will watch a certain object in the future segment.

The probability that a viewer will change the state from the current state s_c^i to another state s_m^{i+1} for $0 \leq m \leq M-1$ is the column vector of the $\mathbf{P}_{i+1}$, which is defined as a vector $\mathbf{d} = \mathbb{R}^{1\times M}$ whose elements are $p(s_m^{i+1}|s_c^i)$, for $0 \leq m \leq M-1$, where M is the number of states in the segment $i+1$. We are also interested in the future states for longer prediction horizon. To predict a state in the segment $i+2$, we can perform a matrix multiplication $\mathbf{P}_{i+2} \times \mathbf{d}_1$ to get $\mathbf{d}_2$. In general, we can define a k^{th} future transition probability as

$$\mathbf{d}_k = \mathbf{P}_{i+k} \times ... \times \mathbf{P}_{i+2} \times \mathbf{d_1}. \tag{3}$$

Since the state s_m consists of multiple objects, the probability p_j of watching the object O_j in future segment $i+1$ is given as

$$p_j = \sum_{\forall m, O_j \in s_m} d^m \tag{4}$$

where j is the object index and d^m indicates the m^{th} element of vector $\mathbf{d}$. For example (see Figure 6), the probability of watching the object O_3 is $p_1 + p_2 = p(s_1^{i+1}|s_1^i) + p(s_2^{i+1}|s_1^i)$ because both s_1^{i+1} and s_2^{i+1} include the object O_3. The probability of watching the object O_4 is $p_2 = p(s_2^{i+1}|s_1^i)$, and the probability the viewer will not watch any object is $p_0 = p(s_0^{i+1}|s_1^i)$.

B. Future View Generation using VMV

By using VOSM, we can predict which objects the viewer will probably watch in future segments. However, it does not mean that we can exactly identify which tiles will be shown in the future view. Therefore, we utilize the current view, which is a set of visible tiles in the segment i, and move it properly to include all objects possibly to be shown in the future view $i+1$. If the predicted state in the VOSM is NULL state, then we assume that the viewers have no reason to turn their heads to watch other parts of the video. However, if the predicted state includes certain objects, then we assume that the objects will attract viewers and they will turn their heads to look at the objects. We can generate the possible future views in segment $i+1$ as $currentview + v_1$ and $currentview + v_2$, where the vectors v_1 and v_2 are the motion directions that can properly move all the tiles in $currentview$ to the other set of tiles that include the object O_1 and O_2 respectively, when the predicted states include O_1 and O_2.

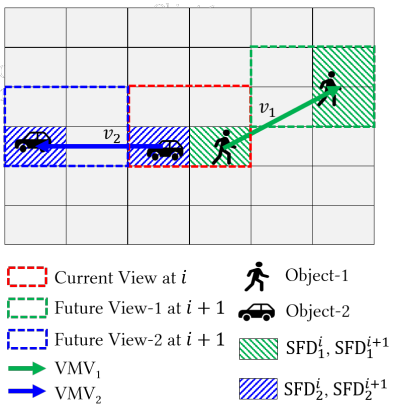


Figure 7: View Motion Vector (VMV) for the Future View Prediction

We define the *View-Motion Vector (VMV)* as a vector that describe a motion of the view from current segment to the future segment to follow the movements of objects. Since the motions of objects will lead viewers' movements, we utilize SFD information to derive and compute VMV. To compute a VMV for the movement of a view from segment i to segment $i+1$, the center point of SFD_j^{i+1} is compared to the center point of SFD_j^i to calculate VMV v_j for object j.

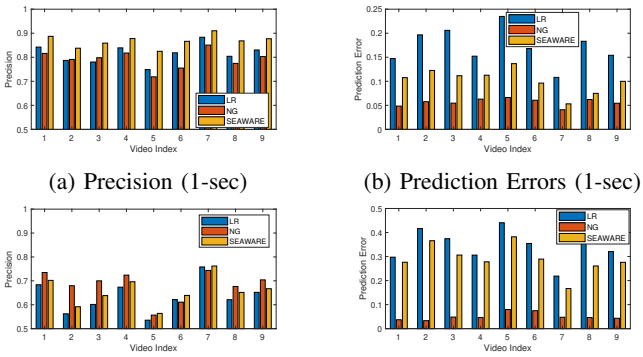
$$v_j = center(SFD_j^{i+1}) - center(SFD_j^i) \tag{5}$$

Figure 7 shows an example of VMVs. The current view consists of four tiles and it includes two objects. SFD will inform clients of which tiles will include the objects in the future segments. We want to find the VMV that can move the current view to the future view including tiles that will have desired objects. In Figure 7 example, a person and a car move in different directions and stay in different sets of tiles. To follow the person, we need to move the current view using a vector v_1. To follow the car, we need to move the current view using vector v_2.

Since the VOSM provides the probability of including the objects in the future view, we can decide the probability w_t that the tile t will be shown in the future segment $i+1$ using p_j from Equation 4 and $futureview_j = currentview + v_j$, where j is the object index. The probability of watching the tiles that belong to $futureview_j$ in the segment $i+1$ is p_j. If the tiles belong to multiple future view in $i+1$, then we have to sum up all p_j for all $futureview_j$ including the tile.

$$w_t = \sum_{\forall j, t \in futureview_j} p_j \tag{6}$$

Since the SFD and VOSM information come from the server, the clients only require to compute VMV to predict the required tiles in the future. This information will be used for rate selection under limited bandwidth.



(a) Precision (1-sec) (b) Prediction Errors (1-sec)

(c) Precision (5-sec) (d) Prediction Errors (5-sec)

Figure 8: View Prediction Performance

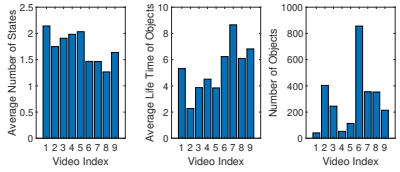


Figure 9: Video Semantic Information

IV. Performance Evaluation

Nine videos (Table I) and 48 users' view traces are used for the performance evaluation. The data set [21] provides the 360-degree videos along with view traces. The videos are divided into 6 by 12 tiles and tiles are encoded independently using ffmpeg [22] to four different qualities. A segment duration is 1-second. Objects are found in the video using Yolov3 [23] based object detection in every frame and the Deepsort multiple object tracking algorithm[24] is used to track the objects. This information is encoded as a SFD by mapping the location of the objects into the set of tiles for each video and stored at the media server along with MPD and SRD. 43 viewers' view traces are used to build a VOSM at the server, and remaining 5 viewers' data is used to test the performance of the view prediction algorithm.

To evaluate the performance of the proposed view prediction algorithm in the end-to-end streaming system, we have generated a networking model that transmits the video segment data. A real HSDPA network trace [25] is used to evaluate the performance of the 360-degree video streaming system with the network variation.

A. View Prediction

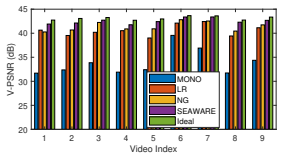
We first measure the performance of view prediction algorithms without considering the bandwidth variation. The precision (prediction accuracy) is measured as

$$\sum_{t=1}^{T} \min(w_t, g_t) \tag{7}$$

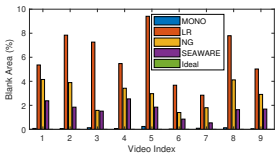
where g_t is a normalized ground truth that has 0 for non-visible tiles and $1/$(number of visible tiles) for visible tiles, and w_t is a predicted probability that tile t will be shown in the future segment. Figure 8a shows the average precision for 1-sec prediction horizon of the proposed SEAWARE system, Linear Regression (LR) [3], and Navigation Graph (NG) [5] based view prediction method. We could find that the proposed SEAWARE system performs better than the LR and NG for all videos. We have analyzed the videos in terms of objects shown in the videos to find the reason why the prediction algorithm performs differently in each video. Figure 9 shows the average number of states per object in VOSM, the average life time of the objects, and the total number of objects shown in nine videos. It shows that the video-1 has two states per object on average, which means that there are two possible choices for the next segment on average when the viewer is watching a certain object. Other videos have less than two states per object on average and especially the video-8 has the smallest average number of states per object. Therefore, VOSM can make more accurate prediction of the view in the next segment for video-8. In other words, if the number of states per object is 1, it means that all viewers make the same choice for the next view. Therefore, the performance gain we can get by using the SEAWARE is larger with smaller number of states per object. The average life time of the objects counts the average number of segments that the objects appear in the video. Video-6, 7, 8 and 9 have longer life time of the objects than videos-1, 2, 3, 4 and 5, which helps predicting the next view relying on the objects' trajectories. We also count the number of objects shown in the videos. Video-6 has a large number of objects found in the video and it helps build a viewing behavior model in terms of objects.

The average prediction errors for 1-sec prediction horizon (Figure 8b) are also measured to find how often the view prediction algorithms fail to predict visible tiles in the future segment. The Prediction Error is measured by counting required tiles that have $p_t = 0$ over total required tiles to render the view. Since the clients will not request the tiles having $p_t = 0$, there is a risk to show blank area to the viewer when the Prediction Error is high. Figure 10b shows that LR has the highest prediction error compared to NG and SEAWARE. The Precision of LR is better than the Precision of NG, but NG has the smallest prediction error, therefore, we cannot directly decide which prediction method is better. Since NG refers all other viewers' view history to produce the prediction matrix, NG usually requests every tile that other viewers watched in the past. It makes the Prediction Error small, but there is some loss in Precision performance. SEAWARE has the best precision performance compared to NG and LR, and has better prediction error performance than LR.

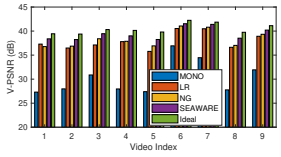
Figure 8c and Figure 8d show the precision and the prediction error performance for 5-sec future view prediction results. The view prediction performance degrades with the longer prediction horizon. In terms of the precision, NG works best for video-1,2,3,4,8,9 and SEAWARE works best for video-6,7. video-6 has the largest number of objects shown in the video, and video-7 has the longest life time of the objects. In other



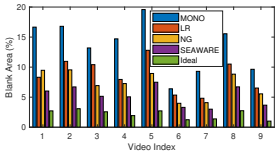
(a) Average V-PSNR with Stable Network



(b) Average Blank Area in the View with Stable Network



(c) Average V-PSNR with Variable Network



(d) Average Blank Area in the View with Variable Network

Figure 10: Average V-PSNR and Average Blank Area (BA)

words, SEAWARE works better when the video has many objects and their life time is long. NG has the lowest prediction errors because NG usually request more tiles that were seen by other viewers. However, we could find that the precision performance of 1-sec prediction horizon affects the most on V-PSNR and Blank Area performance, which are more closely related to viewers' actual experience. We discuss the results in the following section.

B. Viewport-PSNR (V-PSNR) and Blank Area (BA)

To evaluate more realistic performance of the view prediction algorithms in 360-degree video streaming system, we have applied the rate selection algorithm to measure quality of tiles that will be shown to the viewers. The Window-based rate adaptation algorithm (WBA) [12], [11] is used for rate selection of tiles with given probability matrix derived from the view prediction algorithms. WBA allows the rate selection of tiles [13], the client buffer management, and the fast switching [26], [12] to update the existing video segments in the buffer. Two types of network traces are used to test the algorithms. A stable network condition is considered for a wired network, and a variable network condition is considered for a wireless network.

Figure 10 shows the average V-PSNR and the average BA for nine videos (Table. I). V-PSNR represents the quality of the view, which is measured by comparing a received view with the original view. BA indicates the average number of tiles that are missing over total number of tiles required to render the view. Therefore, the viewers will see more blank (or black) area in their view when BA is larger. BA is an important performance evaluation metric for 360-degree videos. In case of conventional video streaming systems, video freezing rate (or re-buffering rate) is important because it indicates how often the client-side buffer depletes and the clients should wait until the video resumes. However, in tiled media, a video may not freeze even if the client does not receive tiles to render the view, but those missing tiles will show blank areas in the view and it will affect the viewer's experience degrading. Therefore, we have to measure how large the blank area is in the view.

MONO represents the system that has no view prediction, assuming every tile has the same probability of view. MONO has poor V-PSNR performance because it streams all tiles in the same quality. Therefore, most of the bandwidth is wasted due to streaming non-visible tiles. However, MONO has very low BA with the stable network condition because it will request every tile with lower video quality. LR, NG, and SEAWARE perform much better than MONO since they perform view prediction using single viewer's view data, multiple viewers' view data, and multiple viewers' view data combined with semantic information respectively. Figure 10a shows that SEAWARE can achieve the best V-PSNR performance. Ideal represents the video streaming system with perfect view prediction. Therefore, we know how much performance loss we had because of view prediction errors. BA (Figure 10b) shows an advantage of using SEAWARE. In most of the videos, SEAWARE can achieve less than 4% of BA, which means that viewers' will rarely see a blank area while other schemes have higher BA (up to 10%). Ideal has almost 0% BA because the view prediction is perfect.

Figure 10c and 10d show the performance with variable network condition [25]. We can observe that the V-PSNR and BA performance are worse compared to V-PSNR and BA performance in the stable network condition. The network bandwidth variation makes it more difficult to receive all required tiles because the network sometimes reaches a too low bandwidth status to receive all required tiles with the lowest quality. However, LR, NG, and SEAWARE have less performance degradation than MONO because they can receive visible tiles utilizing given bandwidth more efficiently. SEAWARE achieves the best V-PSNR and BA performance in the variable network condition since it's prediction accuracy is higher than other prediction schemes and prediction errors are smaller than LR.

We can observe that MONO has very poor BA performance in the variable network condition. MONO cannot stream visible tiles when the network bandwidth is very small, because it wastes most of the bandwidth streaming non-visible tiles. Moreover, even with the Ideal view prediction, there are some blank area in the view in variable network, which means that the network bandwidth is sometimes too small to transmit all visible tiles.

C. Overhead Analysis

SEAWARE requires SFD and VOSM in addition to SRD. The total storage required for the SFD files is up to 100KB per video and the storage required for the VOSM file (including states and transition matrix) is 10KB per video on average. SFD file is generated for each segment and delivered only when clients request corresponding segment. The average size of SFD files is smaller than 1KB per segment which is significantly smaller compared to video segments (up to 5.8MB).

Processing the VOSM based view prediction algorithm takes 2.2ms on average in the client device (Intel Core i7, 8GB memory) not using GPU when the processing time of LR and NG are 1.9ms and 19ms respectively on average in the same client device. We have used YOLOv3 based object detection and tracking algorithms off-line. This process is done on the Ubuntu machine that has CPU (Intel Xeon) and GPU (NVIDIA Quadro P4000) with 32GB memory. The process takes 4-min per video on average.

SEAWARE can achieve better performance than other schemes using minimal overhead.

V. Conclusion

The semantic-aware view prediction (SEAWARE) system represents a novel 360-degree video streaming system. The SEAWARE system utilizes semantic information of 360-degree video content to predict the viewer's behavior. It ensures minimal bandwidth usage with very strong V-PSNR and low BA performance results. Semantic information is encoded as SFD and view information from the clients are gathered to generate VOSM. SFD and VOSM descriptions are stored in the media server and the server provides the information to the clients when the clients request their video segments. The client devices can predict future view using SFD and VOSM information since the viewers tend to follow objects in the video. We introduce VMV, derived from SFD, VOSM, and client's current view, to generate possible future views. The SEAWARE system achieves better precision performance compared to the history-based prediction algorithms. We have shown that video semantic information helps to improve QoE of the 360-degree videos streamed over the Internet. Moreover, SEAWARE is compatible with the DASH platform and clients can perform the semantic-aware view prediction without analyzing the video.

Acknowledgement

This work is supported by the National Science Foundation under Grant No. CNS-1900875 and No. CNS-1901137.

References

[1] Y. Bao, H. Wu, T. Zhang, A. A. Ramli, and X. Liu, "Shooting a moving target: Motion-prediction-based transmission for 360-degree videos," in *2016 IEEE International Conference on Big Data (Big Data)*, Dec 2016, pp. 1161–1170.

[2] X. Lan, Z. Xinggong, and G. Zongming, "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*, ser. MM '18. New York, NY, USA: ACM, 2018, pp. 564–572. [Online]. Available: http://doi.acm.org/10.1145/3240508.3240556

[3] X. Lan, X. Zhimin, B. Yixuan, Z. Xinggong, and G. Zongming, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *ACM International Conference on Multimedia (MM)*, Oct. 2017.

[4] 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, March 2019.

[5] 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

[6] "360 google spotlight stories: Help," 2016. [Online]. Available: https://youtu.be/G-XZhKqQAHU

[7] M. Xu, Y. Song, J. Wang, M. Qiao, L. Huo, and Z. Wang, "Predicting head movement in panoramic video: A deep reinforcement learning approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2693–2708, 2018.

[8] 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.

[9] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

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

[11] J. Park, P. A. Chou, and J. Hwang, "Volumetric media streaming for augmented reality," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018, pp. 1–6.

[12] ——, "Rate-utility optimized streaming of volumetric media for augmented reality," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 149–162, March 2019.

[13] J. Park, J. Hwang, and H. Wei, "Cross-layer optimization for vr video multicast systems," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec 2018, pp. 206–212.

[14] J. Le Feuvre and C. Concolato, "Tiled-based adaptive streaming using mpeg-dash," in *ACM Multimedia Systems Conference (MMSys)*, ser. MMSys '16. New York, NY, USA: ACM, 2016, pp. 41:1–41:3. [Online]. Available: http://doi.acm.org/10.1145/2910017.2910641

[15] I. Standard, "Dynamic adaptive streaming over http (dash)—part 1: media presentation description and segment formats," *ISO/IEC*, pp. 23 009–1, 2014.

[16] M. Hosseini and V. Swaminathan, "Adaptive 360 vr video streaming based on mpeg-dash srd," in *2016 IEEE International Symposium on Multimedia (ISM)*, Dec 2016, pp. 407–408.

[17] V. R. Gaddam, M. Riegler, R. Eg, C. Griwodz, and P. Halvorsen, "Tiling in interactive panoramic video: Approaches and evaluation," *IEEE Transactions on Multimedia*, vol. 18, no. 9, pp. 1819–1831, Sep. 2016.

[18] R. A. Patrice, M. Jean-Franois, and V. Nico, "Interactive omnidirectional video delivery: A bandwidth-effective approach," *Bell Lab. Tech. J.*, vol. 16, no. 4, pp. 135–147, Mar. 2012. [Online]. Available: http://dx.doi.org/10.1002/bltj.20538

[19] K. Misra, A. Segall, M. Horowitz, S. Xu, A. Fuldseth, and M. Zhou, "An overview of tiles in hevc," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 6, pp. 969–977, Dec 2013.

[20] P. Zhao, Y. Zhang, K. Bian, H. Tuo, and L. Song, "Laddernet: Knowledge transfer based viewpoint prediction in 360○ video," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 1657–1661.

[21] W. Chenglei, T. Zhihao, W. Zhi, and Y. Shiqiang, "A dataset for exploring user behaviors in vr spherical video streaming," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, ser. MMSys'17. New York, NY, USA: ACM, 2017, pp. 193–198. [Online]. Available: http://doi.acm.org/10.1145/3083187.3083210

[22] F. Bellard, FFmpeg https://www.ffmpeg.org.

[23] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[24] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," *CoRR*, vol. abs/1703.07402, 2017. [Online]. Available: http://arxiv.org/abs/1703.07402

[25] R.Haakon, E. Tore, V. Paul, G. Carsten, and H. Paul, "Video streaming using a location-based bandwidth-lookup service for bitrate planning," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 8, no. 3, pp. 24:1–24:19, Aug. 2012. [Online]. Available: http://doi.acm.org/10.1145/2240136.2240137

[26] K. Spiteri, R. Sitaraman, and D. Sparacio, "From theory to practice: Improving bitrate adaptation in the dash reference player," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 15, no. 2s, Jul. 2019. [Online]. Available: https://doi.org/10.1145/3336497