Machine Learning based Content-Agnostic Viewport Prediction for 360-Degree Video

SAM VAN DAMME, MARIA TORRES VEGA, and FILIP DE TURCK, IDLab - imec, Department of Information Technology, Ghent University

Accurate and fast estimations or predictions of the (near) future location of the users of Head-Mounted Devices (HMDs) within the virtual omnidirectional environment open a plethora of opportunities in application domains such as interactive immersive gaming or tele-surgery. Therefore, the last years have seen a growing attention to models for viewport prediction in 360° environments. Among the approaches, content-agnostic, trajectory-based methods have the potential to provide very fast solutions, as they do not require complex analysis of the videos to provide a prediction. However, accurate trajectory-based viewport prediction is rather difficult due to the intrinsic variability in user behaviour. Furthermore, even when making use of Machine Learning (ML), current approaches tend to be brute-force and heavily tailored to specific datasets with little comparison to existing benchmarks or publicly available studies. This paper presents a generic, content-agnostic viewport prediction method consisting of a window-based approach combined with a preprocessing system to classify behavioural patterns in terms of user clustering and trajectory correlation. Moreover, as the state-of-the-art does not provide a comparative analysis of different approaches, this paper contributes to this. Based on the obtained results, a combined prediction model is proposed and evaluated. Our method shows a 36.8% to 53.9% improvement when compared to the static prediction baseline for a Prediction Horizon (PH) of 8s. In addition, a 11.5% to 24.0% improvement to a brute-force ML prediction approach is obtained. As such, this work contributes towards the creation of more generic and structured solutions for content-agnostic viewport prediction in terms of data representation, preprocessing and modelling.

Additional Key Words and Phrases: Virtual Reality (VR), 360-degree video, viewport prediction, content-agnostic, Machine Learning

ACM Reference Format:
Sam Van Damme, Maria Torres Vega, and Filip De Turck. 2021. Machine Learning based Content-Agnostic Viewport Prediction for 360-Degree Video. *ACM Trans. Multimedia Comput. Commun. Appl.* , , Article (2021), 23 pages. https://doi.org/10.1145/3474833

1 INTRODUCTION

The constantly increasing demand for better and more immersive user experience in (interactive) multimedia applications, has initiated the rise in popularity of Virtual Reality (VR) and 360° video [14]. To this end, the user wears a Head-Mounted Device (HMD) to freely explore the provided omnidirectional environment during playback. In order to provide good end-user quality, 360° video resolutions of 6K or higher and frame rates of at least 60 fps are required [24]. However, the limited bandwidth of most networks becomes an unavoidable bottleneck. As a result, simply streaming the 360° video as it were traditional video content leads to a waste of resources as the user only watches a limited portion of the delivered content at each instance of time, referred to

Authors' address: Sam Van Damme; Maria Torres Vega; Filip De Turck, {firstname}.{lastname}@ugent.be,IDLab - imec, Department of Information Technology, Ghent University.

as the *viewport*. Therefore, an accurate, fast, and generic viewport prediction algorithm allows the service providers to tailor the scene to the user's viewport, thus optimizing the bandwidth usage and reducing the streaming latency. This would open possibilities to applications, such as interactive, immersive gaming and tele-surgery [3, 8, 13, 22].

To tackle this challenge, a number of viewport prediction algorithms have been developed in recent years. These are mainly classified in two categories [3, 13]: *content-based* and *motion-based/trajectory-based/content-agnostic* approaches. *Content-based* approaches provide predictions based on the characteristics of the video under scrutiny in terms of visual saliency maps, including detection of motion (which tends to capture the user's attention) and specific Regions-of-Interest (ROIs) [3]. They have already proven their benefit for certain types of content in terms of accuracy [9, 16], but typically suffer from high computational requirements as they often include rather complex image processing and deep learning algorithms [2, 5, 9, 10, 23]. Moreover, within the current state-of-the-art there is no strict consensus on the relation between saliency information and the corresponding user behaviour over time [3]. Finally, different users tend to show interest in different ROIs, obstructing the straightforward prediction of a user's gaze given the particular content [3].

In *motion-based* approaches, on the other hand, the user's future fixation point, *i.e.* the center of the viewport, is predicted based on historical motion information from this and/or other users. Often followed approaches include weighted combinations of other user's trajectories[3], statistical extrapolation methods [19] and other heuristics such as dynamic programming approaches. These models are often barely improving on rather straightforward benchmarks such as static prediction, i.e. the user's orientation is assumed not to change between two consecutive samples, especially with respect to long term prediction when user behaviour is complex. As such, classical content-agnostic approaches benefit from reduced computational complexity in comparison with content-aware models, be it often at the cost of reduced accuracy. Machine Learning (ML) models have already shown promising results to tackle this issue. These are often based on Artificial Neural Networks (ANNs), such as Long Short-Term Memorys (LSTMs), although other models are applied as well.

However, the existing content-agnostic ML approaches in literature are often rather brute-force, without a decent user pre-analysis. In addition, the proof of their accuracy is often limited to a tailor-made dataset with a cherry-picked set of users. Finally, there is a tendency to select the most complex ML models to solve the issues without a proper justification. Therefore, the state of the art is missing fast, accurate, and generic viewport prediction models.

This paper presents an end-to-end solution for fast, accurate trajectory-based viewport prediction. We use a window based approach where a set of past samples is used to predict the future location by means of Supervised Learning (SL). In addition, our system introduces a pre-processing scheme which pre-classifies the users and videos to enhance the prediction. To prove the generality of the method, three well-known 360° trajectory datasets have been evaluated [4, 6, 22]. Furthermore, a set of 7 different SL models have been analysed. Therefore, a side contribution of our work is a comparative performance evaluation of different ML models to solve the viewport prediction challenge. The obtained results led to the development of a combined model, which can adapt the complexity of the ML algorithm depending on the user and content type. This work contributes towards the creation of a more generic and structured solution for content-agnostic viewport prediction in terms of data representation, preprocessing and modelling.

The remainder of this paper is organized as follows. In Section 2, the methodology is discussed. Section 3 describes the experimental setup in terms of the used datasets and the training and optimization of ML models. In Section 4 the most important results are analyzed. Section 5 gives an

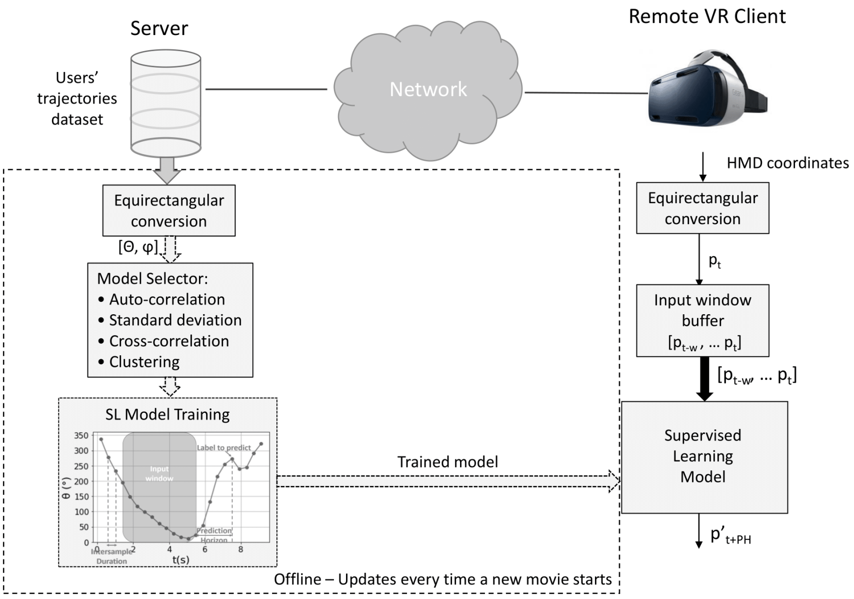


Fig. 1. General overview of the presented end-to-end approach

overview of existing (ML-based) prediction approaches in literature. Section 6, at last, summarizes the most important findings of this work .

2 APPROACH

Figure 1 presents an overview of the proposed, content-agnostic, end-to-end solution. On the server side, a set of historical user trajectories for a particular video is stored. Depending on the nature of the HMD by which the trajectories were gathered, an *equirectangular conversion* of the temporal set of viewport coordinates will be needed to convert them to polar coordinates θ and φ (Section 2.1). Afterwards, a preprocessing analysis of the trajectories is performed in the *Model Selector*, in terms of autocorrelation of user trajectories, cross-correlation between θ and φ, and clustering of users with similar behaviour (Section 2.2). Based on this analysis, the actual training and prediction strategy of the ML model can be decided, *i.e.*, whether ML prediction is needed in comparison with static prediction, simultaneous vs. independent prediction of θ and φ and whether or not to use a pre-clustering algorithm before executing the actual, offline training. This training includes a supervised, window-based ML approach (Section 2.3). This process is performed offline, so that the most recent model can be sent to the client every time they start a new film. As content-agnostic models work on rather low-dimensional, straightforward input signals (such as two-dimensional equirectangular projections), their complexity can be kept to a minimum. As such, sending the trained model to the user can be realized with limited overhead in terms of bandwidth. Detailed results to this end are provided in Section 4.4. On the client side, the same equirectangular conversion is performed to the most current coordinate, after which it is stored in a buffer keeping the last W samples of the user gaze. These are fed to the viewport model, which outputs the predicted viewport p'_{t+PH} at time instance $t + PH$ in the future, with PH being the so-called Prediction Horizon (PH). A more in-depth discussion of these building blocks is provided in the following Sections.

2.1 Equirectangular conversion

The format in which the trajectory data is provided depends exclusively on the HMD used. For instance, the HTC Vive provides its location data in quaternions[1] [22], while the Oculus does so in 3D (yaw x, pitch y, roll z). In order to be able to provide the same evaluation for different datasets, there is a need for a general representation. In this approach, we selected the equirectangular representation as it has been broadly used in literature. Therefore, the viewport data are first equirectangularly projected, obtaining the θ and φ coordinate vectors. This process might vary slightly depending on the format of the original data, but in general, if the original location data is provided in quaternions (qx, qy, qz, qw), it will be first transformed to 3D (yaw, pitch, roll). Next, data in 3D format (either originally or as the output of the quaternion transformation) are then projected to the equirectangular plane [22].

Moreover, to minimize the influence of the spherical nature of the data on the results, the equirectangularly projected trajectories are transformed to exclude numerically large jumps between values over time (*e.g.,* 0° and 359° only differ 1° in practice, but are numerically far away). To this end, each of the consecutive samples is checked to make sure the difference between them covers the smallest angular difference possible, *i.e.* within the [0°, 180°] interval. If this is not the case, a full 360° turn is added or subtracted accordingly. Afterwards, the trajectory is shifted such that 180° becomes the mean value and all samples lay within the [0°, 360°] interval again, i.e. the new trajectory $T'(x)$ is given by $T'(x) = T(x) + 180° - M$, with $T(x)$ the original trajectory and M its average.

2.2 Model selector

As the variability of both the 360° video content and user behaviour are intrinsically high, a one-size-fits-all approach is difficult to establish. As such, a mechanism is needed to select the modelling approach that is best suited to the situation at hand. Therefore, four fundamental aspects have to be analyzed when deciding upon the best suited prediction method.

2.2.1 Auto-correlation. First, it should be decided whether an ML approach is needed in comparison with the more straightforward, static prediction, as ML approaches tend to show little improvement for smaller values of PH. The autocorrelation of the viewport trajectory, i.e., the correlation of the trajectory with a shifted version of itself over a distance PH, can provide insight in this matter.

The resulting value is a decimal number between -1 and 1, where -1 and 1 indicate a strong negative or positive correlation respectively, and 0 no correlation at all. If a strong auto-correlation exists, this indicates that straightforward static prediction as such might be sufficient in terms of accuracy. However, auto-correlation typically tends to drop heavily with increasing values of *PH* such that ML approaches might provide better performance for more long-term prediction.

2.2.2 Standard deviation. Second, the standard deviation of a user trajectory over time is also worth exploring. For user trajectories with larger deviations, the probability of static prediction making larger errors is higher while more complex (ML-based) prediction approaches can better grasp patterns and thus limit prediction errors. For trajectories with lower deviation, on the other hand, ML might provide unnecessary overhead in terms of time and memory such that straightforward static prediction might be better suited.

2.2.3 (Cross-)correlation. Third, it should be determined whether θ and φ need to be predicted simultaneously, thus combining $[\theta_{t-W}, ..., \theta_t]$ and $[\varphi_{t-W}, ..., \varphi_t]$ as input features, or independent of each other (using two independently trained ML models). This depends on the strength of the correlation between the two variables. If a strong correlation between θ and φ exists, a combined

[1] https://en.wikipedia.org/wiki/Quaternion

Table 1. Overview of the 360° videos included in the datasets [4, 6, 22]. The prepositions B, C and W indicate the respective datasets Bao, Corbillon and Wu, respectively.

ID	Content	Category	users	Dur.	ID	Content	Category	users	Dur.
B0	Angels	Action	65	40s	C0	Diving	Nature	58	412s
B1	Barreled	Nature/Sport	57	40s	C1	Paris	Culture	58	244s
B2	Basketball	Sport	46	40s	C2	Rollercoaster	Action	58	206s
B3	BasketballFlying	Sport	49	40s	C3	Timelapse	Culture	58	91s
B4	Boomerang	Action	63	40s	C4	Venice	Culture	58	175s
B5	Boxing	Sport	51	40s					
B6	DancingGirl	Sport	46	40s	W0	Conan360°-Sandwich	Performance	48	164s
B7	Dolphin	Nature	84	40s	W1	Freestyle Skiing	Sport	48	201s
B8	Flying	Sport	62	40s	W2	Google Spotlight-HELP	Film	48	293s
B9	footballTeam	Sport	48	40s	W3	Conan360°-Weird Al	Performance	48	172s
B10	GiantDinosaur	Nature	64	40s	W4	GoPro VR-Tahiti Surf	Sport	48	205s
B11	GrandCanyon	Nature/Sport	73	40s	W5	The Fight for Falluja	Documentary	48	655s
B12	Hollywood	Action	79	40s	W6	360° Cooking Battle	Performance	48	451s
B13	Ski	Sport	66	40s	W7	LOSC Football	Sport	48	164s
B14	Soccer	Sport	59	40s	W8	The Last of the Rhinos	Documentary	48	292s
B15	Survivorman	Nature	57	40s					

prediction model is preferred such that their mutual relationship remains unbroken. Otherwise, it is best to build two different prediction blocks. The Pearson Linear Correlation Coefficient (PLCC) between both provides an indication, and can be calculated similarly to an auto-correlation, but between the two different signals and without time shift.

2.2.4 Clustering. Fourth, depending on the behaviour of the users and on the structure of the videos themselves, it might be beneficial to divide the multiple users in different clusters with similar viewing pattern. In such case, one model per cluster is to be trained and used as a viewport predictor for unseen trajectories (after assigning them to the appropriate cluster as well). To classify patterns, we propose to use the spectral clustering based algorithm proposed by Petrangeli et al. [17]. It clusters the individual user trajectories based on an affinity score $K(\mathbf{P}, \mathbf{Q})$ that expresses the resemblance of two trajectories based on the user behaviour. This is done by means of the average distance between both curves over time [17].

Calculating these scores on all the users' trajectories results in an affinity matrix, based on which the users' trajectories are divided in clusters [1]. The optimal number of clusters is derived from the *eigengap* heuristic proposed by Zelnik-Manor et al. [27]. Equivalent to the definition of Petrangeli et al.[17], clusters with more than 3 trajectories are considered as *main clusters*, while trajectories included in the remaining clusters are considered *outliers*, and therefore harder to predict.

2.3 Sliding window approach

The ML approach followed in this paper consists of a sliding window based setup as shown in Figure 1. For every future position $p_{t+PH} = (\theta_{t+PH}, \varphi_{t+PH})$ the ML model makes a prediction based on a set of current and historical values $[p_{t-W}, ..., p_t]$, which are called the *input window*. The number of samples within this input window is defined by the length of the window W and the *Inter-Sample Duration (ISD)*, which is the inverse of the *sample frequency*. The time between the last sample of the input window and the future position to predict is called the *Prediction Horizon (PH)*. The values of the PH and the window W can be configured depending on the characteristics of the videos and users. In the evaluation, different values of both parameters at put to test. Finally, it is worth noting that the window based approach is independent from the type of ML method used as long as it is SL-based.

3 EXPERIMENTAL SETUP

This Section describes the experimental testing methodology followed in this paper. First, a thorough description of the three datasets under examination is provided (Section 3.1). Next, the different ML models with which our approach is evaluated are given in (Section 3.2). This section provides technical details of in terms of optimization, training and evaluation.

3.1 Datasets

This Section provides a brief description of the datasets [4, 6, 22] used for analysis and prediction. For all three of them, a general overview of the dataset characteristics is given (Table 1). The dataset provided by Bao et al. [4] consists of viewport trajectories of 16 40s videos in 4K resolution, covering varying content roughly divided in three categories: *Action*, *Nature* and *Sport*. Viewport trajectories are collected over a total of 153 subjects. 35 subjects watched all 16 videos, while the others watched 3 to 5 randomly selected clips. A total of 969 views were gathered. For each of the views, the pitch, yaw and roll coordinates were collected at a sampling rate of 10 Hz. The pitch and yaw can be directly mapped to φ and θ respectively [22]. The roll is not further analyzed within this paper.

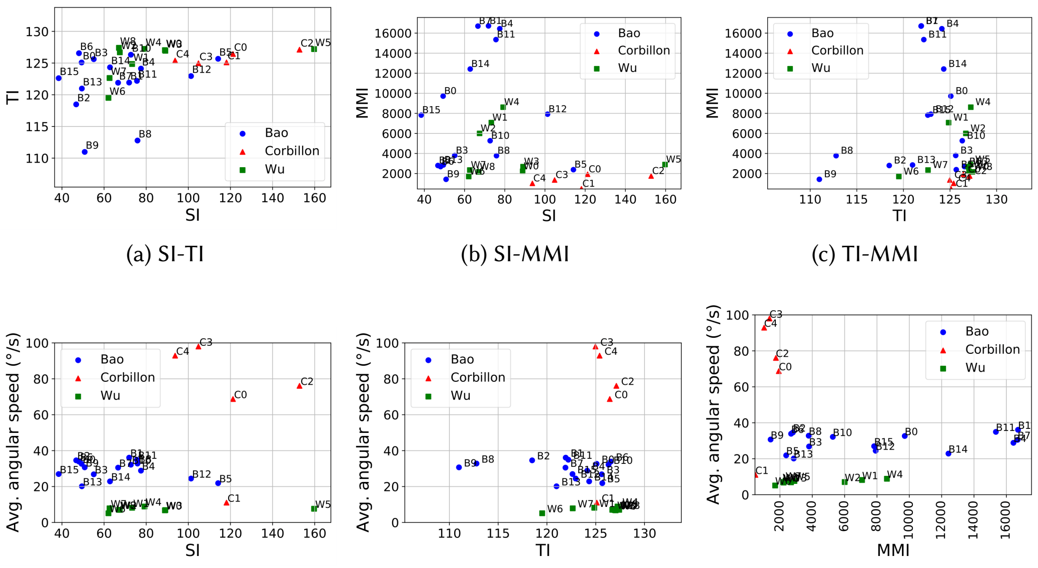
The dataset of Corbillon et al. [6] includes five 360° 4K YouTube videos. Their content is split into three categories: *Nature*, *Action* and *Culture*, as shown in Table 1. Each of the videos was watched by a total of 58 users. As such, a total of 290 trajectories is gathered, where samples are represented as quaternions and collected at a sampling rate of 45 Hz. For further analysis, the quaternion representation is transformed to spherical coordinates [22].

The dataset of Wu et al. [22] consists of two experiments, covering the same 48 participants each. In the first experiment, the users are not provided with any sort of particular instructions, therefore being allowed to look around freely in the virtual scene. In the second experiment, users are told they will have to take a test on the content of the video afterwards, therefore forcing them to focus on the main content of the video (*e.g.* the number of points a team scores in a basketball match). To mimic the real-life user behaviour of omnidirectional video streaming as closely as possible, only the dataset resulting from the first experiment is used for analysis. The data consists of 9 different 360° videos covering a multitude of content within the fields of *Performance*, *Sport*, *Film* and *Documentary* and lengths varying between 164s and 655s. Each of the videos was watched by all 48 participants, thus resulting in a total of 432 user trajectories collected. For each video and each participant, the trajectory of the viewport within the hemisphere of the VR-device (HTC Vive) was sampled over time at 25 Hz and stored in unit quaternion representation. These user trajectories were rephrased in terms of the horizontal (θ) and vertical (φ) angular displacement to a fixed coordinate system as measured from the origin.

For consistency between the datasets, the Corbillon and Wu dataset have been further subsampled such that they equal the 10 Hz sampling rate of the Bao dataset. Furthermore, a spatiotemporal analysis is performed to gain further insight into the characteristics of the videos that might relate to user behaviour. Figure 2 shows the spatiotemporal characteristics of the videos in terms of Spatial Information (SI), Temporal Information (TI) and Mean Motion Intensity (MMI).

Here, f_k is the k-th frame in the video sequence and K, M and N are the number of frames, height and width of the video, respectively.

Based on this spatiotemporal classification, a subset of videos is selected for more detailed analysis. As shown in Figures 2d, 2e and 2f, the datasets span three different classes in terms of average user speed. The Corbillon videos show the highest average user speed (except for C1), while the Wu videos show the slightest movement. The Bao videos hold the middle between both. As such, two videos per dataset are selected to obtain a representative distribution of user speeds.



(a) SI-TI (b) SI-MMI (c) TI-MMI

(d) SI-Angular speed (e) TI-Angular speed (f) MMI-Angular speed

Fig. 2. Spatiotemporal characteristics of the videos in terms of SI, TI, MMI and average user speed.

Furthermore, videos are selected based on their characteristics in SI-TI-MMI space. As such, both B9 and C2 are chosen as extreme cases. B4 is also selected due to its high MMI. C4, W2 and W4 are added to the selection due to their rather average characteristics. A summary of the selected videos is provided in Table 2, while Figure 3 shows a representative screenshot for each of them.

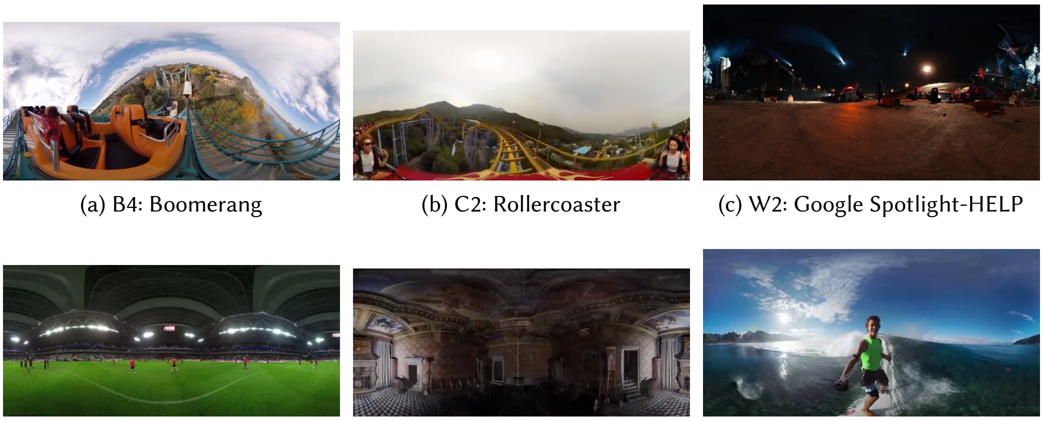
3.2 ML optimization, training and evaluation

A set of seven ML models is put forward of which the applicability towards sliding window based viewport prediction will be investigated. Four of them fall within the category of ANNs, i.e. Feedforward Neural Network (FNN), Convolutional Neural Network (CNN), LSTM and Gated Recurrent Unit (GRU), which are often applied in literature. The three other models (Linear Regression (LR), Support Vector Regressor (SVR) and Random Forest (RF)) are chosen because of their wide application in ML problems in general.

3.2.1 Optimization. These models are optimized by splitting the users in the available data in a training and test set in a 90%-10% ratio. The training set is used for hyperparameter tuning using a 3-fold Cross-Validation (CV) approach. For each run (and thus each validation fold) a grid-search over the parameters is performed. Afterwards, the three resulting validation errors are averaged per parameter configuration to obtain the optimal solution. Here, the squared angular difference $E(\hat{y}, y)$ between prediction $\hat{y}$ and ground truth y is used as the error function (Equation 1). This is done for a default window size and PH of 2s each. The samples in the input window are further normalized between -1 and 1 using a *MinMaxScaler* before training.

$$E(\hat{y}, y) = (-180^\circ + (\hat{y} - y + 180^\circ) \mod 360^\circ)^2 \tag{1}$$

No specific hyperparameters are tuned for the LR model. For the SVR, the kernel is fixed at a Radial Basis Function (RBF) while for both the *regularization parameter* λ and the *tube width* ϵ values


(a) B4: Boomerang (b) C2: Rollercoaster (c) W2: Google Spotlight-HELP

(d) B9: FootballTeam (e) C4: Venice (f) W4: GoPro VR-Tahiti Surf

Fig. 3. Representative screenshot of each of the selected videos.

within $\{10^n | n = -4..4\}$ are investigated. With respect to the RF, the Mean Squared Error (MSE) is chosen as the *optimization criterion* and the *minimal number of samples* per split and per leaf is fixed at 2 and 1, respectively. The *number of estimators*, *maximum tree depth* and *minimum impurity decrease* are varied over the respective sets $\{1, 5, 10, 50, 100, 500, 1000\}$, $\{5, 10, 25, 50, 100, 500, 1000\}$ and $\{0.0, 0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1.0\}$. Furthermore, the *activation function*, *optimizer* and *loss function* are chosen as a
acrelu,
acadam and
acmse, respectively. The other parameters (*number of layers*, *number of neurons/layer*, *learning rate*, *regularization parameter* λ, *momentum* μ, *decay* δ, *number of epochs*, *batch size* and *dropout rate*) are allowed to move freely in order to obtain the best possible convergence of the validation curves. The same accounts for the
accnn, with the additional *convolutional kernel width* fixed to 3. For the
aclstm and
acgru, the same optimizer and loss function are used while $\tanh(x)$ and $\text{sigmoid}(x)$ are selected as the *activation function* and *recurrent activation function*, respectively. The other parameters, which are similar to the FNN, are also tuned in function of convergence.

3.2.2 Training. After hyperparameter tuning, the model is retrained on the full training set using the most optimal configuration. Predictions are performed on the test set to evaluate the models' error. This error is expressed as the Root Mean Squared Error (RMSE) of the per-sample *angular difference*. In the evaluations, different scenarios have been investigated, with both input window sizes between and PHs varying from 0.4 to 8 s.

3.2.3 Evaluation. The proposed models are evaluated for each possible combination of the former. Implementation of the ML models is done in *Python*[2] making use of the *Scikit-learn*[3] library for the

[2]https://www.python.org/
[3]https://scikit-learn.org/stable/

LR, SVR and RF models and the *Tensorflow Keras*[4] library for the ANNs in addition to *Pandas*[5] and *NumPy*[6] for data processing.

4 RESULTS

This Section gives an overview of the results obtained from the ML prediction approach presented in Section 2. The results in this Section focus on the prediction errors of θ as the distribution of φ over time has a rather small standard deviation by nature such that prediction errors are small. As such, complex ML algorithms have little benefit in the φ-dimension in comparison with plain interpolation or static prediction. This is shown in Table 3, with the standard deviations σ of both θ and φ averaged over all users, for each video as well as per dataset. Note that the significantly longer videos of the Wu dataset show higher values for $\sigma(\theta)$ on average. A possible explanation lays in the fact that users might become more interested in the surroundings other than the main topic of the video after being exposed to the virtual environment for a certain amount of time. The same assumption can be made for the Corbillon dataset.

In the following Sections, the potential of the window-based approach will first be shown (Section 4.1). Next, the modifications of the *model selector*, i.e. the auto- (Section 4.1) and cross-correlation analysis (Section 4.2) and the user clustering algorithm of Petrangeli et al. [17] (Section 4.3), will be introduced and their influence on the prediction results will be discussed. Based on these, a hybrid solution is proposed in Section 4.4, accompanied by a short discussion on its complexity and bandwidth requirements.

4.1 Simultaneous prediction: Brute-force window-based model evaluation

The most straightforward approach for viewport prediction includes a simultaneous prediction of θ and φ. To this end, for each video the hyperparameters of each of the models are tuned on 90% of the data using a 3-fold CV. Afterwards, the optimized models are retrained on the full 90% of the data. The remaining 10% is used for evaluation and to obtain the errors discussed in the following Sections. This training and evaluation procedure is repeated for either the window size or the PH varying from 0.4 to 8s while the other variable is fixed at 2s during the process.

Figure 4 shows the angular RMSE of θ between the predicted and actual trajectory as a function of the input window size and PH, respectively. The other variable is always fixed at 2s. The accuracy of the static prediction is included as a benchmark. From Figure 4a, there can be seen that the influence of the window size on the prediction accuracy is rather limited, especially for W2 and W4. For C2, some performance is gained with increasing the window size, except for the LSTM. A possible explanation is that a larger input window allows the models to better "memorize" the rollercoaster, which is closely related to the viewport trajectories. On the other hand, the performance of the

[4]https://www.tensorflow.org/api_docs/python/tf/keras
[5]https://pandas.pydata.org/
[6]https://numpy.org/

Table 2. Summarizing table of the selected videos in terms of SI, TI, MMI and angular speed.

Video	SI	TI	MMI	Angular speed
B4	medium	medium	high	medium
B9	low	low	low	medium
C2	high	high	low	high
C4	medium	medium	low	high
W2	medium	high	medium	low
W4	medium	high	medium	low

Table 3. Overview of the standard deviation values of θ and φ over time, for each video and overall. The lowest and highest deviations per dataset and coordinate are indicated in italic blue and bold red respectively. The subset of selected videos is marked in gray.

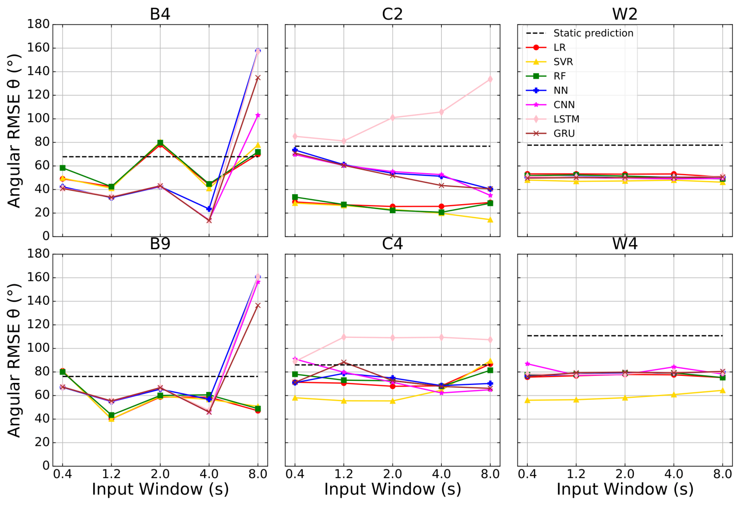
ID	$\sigma(\theta)$	$\sigma(\varphi)$	**ID**	$\sigma(\theta)$	$\sigma(\varphi)$
B0	56.6	**22.5**	C0	105.3	**19.0**
B1	61.3	17.5	C1	100.8	*11.5*
B2	60.8	12.7	C2	*73.0*	15.7
B3	46.9	10.3	C3	100.6	14.4
B4	52.9	17.8	C4	**107.5**	17.4
B5	39.4	11.2	**Overall**	100.0	16.1
B6	66.9	11.5	**ID**	$\sigma(\theta)$	$\sigma(\varphi)$
B7	54.1	15.2	W0	77.4	13.9
B8	**67.8**	21.7	W1	96.7	15.8
B9	46.8	*8.1*	W2	**133.5**	**19.9**
B10	49.1	13.3	W3	*57.7*	14.3
B11	60.2	21.8	W4	98.4	15.5
B12	38.8	21.0	W5	101.7	12.9
B13	*29.8*	11.5	W6	98.4	*7.8*
B14	32.4	15.6	W7	76.3	12.6
B15	53.2	18.4	W8	81.6	16.1
Overall	52.0	16.7	**Overall**	97.5	14.2

SVR in video C4 is decreasing with increasing window size. With respect to the Bao videos, no straightforward relationship can be found between the window size and the models' performance.

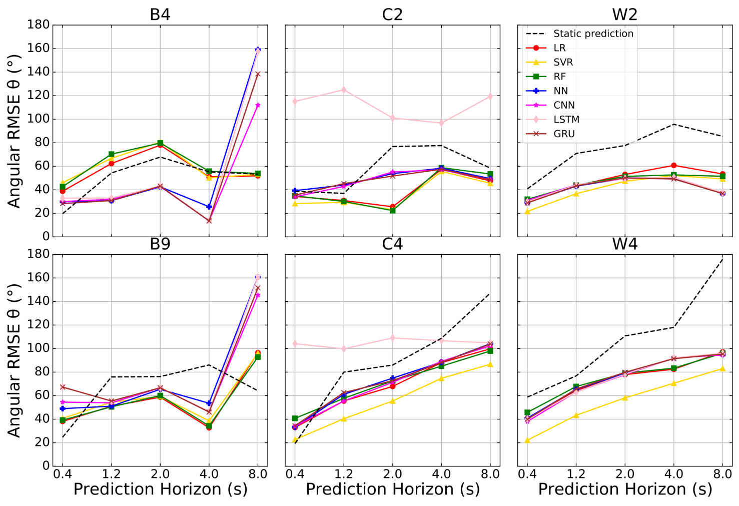
If we have a look at the PH figures (Figure 4b), a first important observation is that the ML-based prediction clearly shows improved accuracy for most of the models when compared to static prediction, especially for higher values of PH. In general, the ANNs (FNN, CNN, LSTM, GRU) show worse performance than LR, SVR and RF. B4 is an important exception to this finding as only the ANNs improve on the static prediction benchmark. Their performance severely worsens, however, for the rather high PH of 8s. This behaviour cannot be noticed for the Corbillon and Wu videos, where the increase in angular error as a function of the PH is rather smooth. Note that similar behaviour can be seen in the autocorrelation curves (on the θ-dimension) as shown in Figure 5. While the Corbillon and Wu videos show the expected, smooth transition from high to low PLCC, the behaviour of B4 and B9 is more irregular. In addition, it can be seen from Figure 4b that the intersection where the performance of the best performing ML model surpasses the static prediction (if this point exists) always lays below 1s, as is the case for B4, B9 and C4. As such, viewport prediction already starts to benefit from ML implementations for PHs as low as 1s. Based on the autocorrelation curves, this indicates that an autocorrelation of 0.6 to 0.7 is needed for the static prediction to level or improve the ML performance. Furthermore, it should be noted that the SVR model is showing good performance overall, especially for videos C4 and W4, with video B4 again as the only exception. Another interesting conclusion is the under-performance of the LSTM model for the Corbillon videos.

4.2 Independent prediction: Enabling the correlation analysis block

In order to show the need for a pre-processing block, this section focuses on the PLCC analysis and how this can improve the prediction of the models. Therefore, in this case the viewport prediction consists of the separate prediction of θ and φ using two different ML models, thus not exploiting their entanglement (if any). This entanglement can be expressed as the PLCC between θ and φ over the course of the trajectory. Table 4 shows the PLCCs between θ and φ, averaged over the



(a) Influence of the input window. The PH is fixed at 2s.



(b) Influence of the PH. The input window size is fixed at 2s.

Fig. 4. Influence of both the input window size and the PH on the accuracy of the simultaneous prediction model, without clustering.

users of each video as well as each dataset as a whole. It can be seen that θ and φ show limited correlation in general, providing an indication that θ and φ can be predicted independently of each other. Furthermore, it can be noticed that some videos tend to show a much stronger correlation relative to the other videos and the dataset average. Video B4, for example, shows a PLCC of -0.274, which is a factor 10 stronger than the average of the dataset. A possible explanation is the fact that this video features a 360° ride in a rollercoaster, in which it is not unlikely that users are focusing on the next few meters of the rollercoaster trajectory. This, in combination with the fact that the rollercoaster is mainly going up and counterclockwise, therefore forcing the user

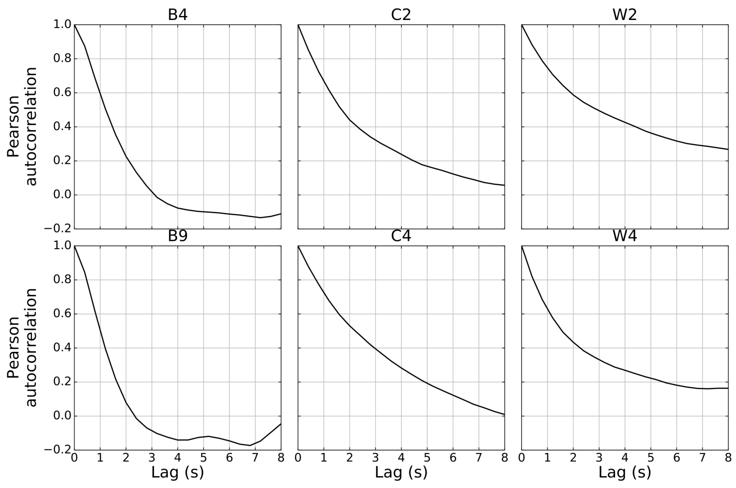


Fig. 5. Autocorrelation curves for the θ dimension of each of the selected videos.

Table 4. Overview of the correlation values between θ and φ for each video and overall. The strongest and weakest correlations per dataset are indicated in italic blue and bold red respectively. The subset of selected videos is marked in gray.

ID	PLCC(θ, φ)	ID	PLCC(θ, φ)
B0	-0.043	C0	0.347
B1	0.069	C1	0.093
B2	-0.012	C2	*0.397*
B3	0.097	C3	**-0.013**
B4	*-0.274*	C4	0.233
B5	-0.009	**Overall**	0.232
B6	-0.104	**ID**	**PLCC(θ, φ)**
B7	0.084	W0	**0.013**
B8	-0.113	W1	-0.050
B9	0.070	W2	*0.221*
B10	0.026	W3	0.015
B11	**0.007**	W4	0.024
B12	**0.007**	W5	-0.016
B13	0.077	W6	-0.114
B14	-0.069	W7	0.019
B15	-0.050	W8	-0.037
Overall	-0.026	**Overall**	0.029

to look to the upper left relative to the center, might provide an explanation to this behaviour. Another example is video W2 with a PLCC of 0.221, in which a big part of the content consists of a scene in which the user is chased by a giant monster (forcing the user to look up). In addition, the monster is coming from the right relative to the starting position, which might explain the positive correlation. A third exception is given by video C2, with a positive PLCC of 0.397. Once again, the video shows a rollercoaster ride which can explain this finding analogous to video B4. As a result, it can be concluded that most videos tend to show an almost non-existing correlation between both coordinates, providing an indication for the fact that both θ and φ can be predicted independently of each other in most cases. For some specific videos, however, the content will play a relevant

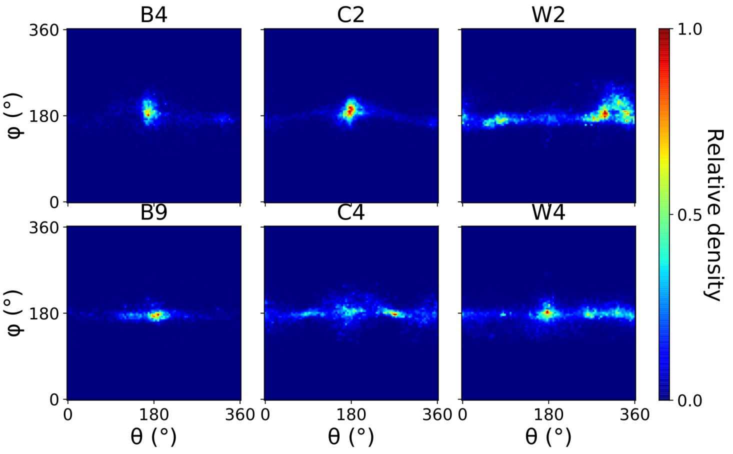


Fig. 6. Heatmaps indicating the probability density of user fixation for each of the selected videos.

role in viewport prediction indicating that such cases might benefit from predicting both angles as one tuple. Instead of requiring complex image or video analysis, this characteristic can be easily detected by the (θ, φ)-PLCC correlation.

The users' trajectory tendencies can be further explored by means of the heatmaps provided in Figure 6. These show the relative fixation point probabilities for each video, as obtained by averaging over the users. The major part of the fixation is centered around (180°,180°). Furthermore, the probability clearly shows a higher spread over the θ-axis as is the case for φ. This is also the case for video B4, although it is clear that an important part of the density is spread out in the vertical direction as well. This might be explained by the fact that the content of the video covers a ride in a theme park attraction in which users naturally tend to look up. This can also be noticed in video C2, although less pronounced. Furthermore, one can clearly see the fixation on the chasing monster in the right part of the heatmap of video W2. Videos B9, C4 and W4, on the other hand, show rather generic user behaviour (with most fixation centered around (180°, 180°) and little spread in the vertical direction) without any particular indication of the content at hand.

From the evaluation of the PLCC, it can be concluded that for most of the videos, predicting the two coordinates independently is bound to provide better results. To verify this hypothesis, we performed the same type of analysis as in the previous section but this time with separated coordinates. Figure 7 shows the results of this analysis by means of the angular RMSE on θ after predicting it independently of φ. With respect to autocorrelation and the window size, almost identical conclusions can be drawn as is the case for the simultaneous prediction. As such, the according discussion is left out and the according Figure is not displayed.

For videos B4, C2 and W2, it can be seen that some of the models are negatively influenced by the independent nature of the prediction when compared to the simultaneous prediction results for PHs up to 4s. This is the case for the ANNs (FNN, CNN, LSTM and GRU) for video B4; LR, SVR and RF for video C2 and the SVR for video W2. For video B9, on the other hand, independent prediction increases the performance of all models for most of the PHs. This is also the case for the LSTM in videos C2 and C4, while the other models in C4 don't seem to experience any fundamental change in accuracy due to the independent instead of simultaneous prediction. A similar conclusion can be made for W4, although a small improvement can be noticed for the SVR model. Note that B4, C2 and W2 are showing rather strong correlations between θ and φ while for B9 and W4 this

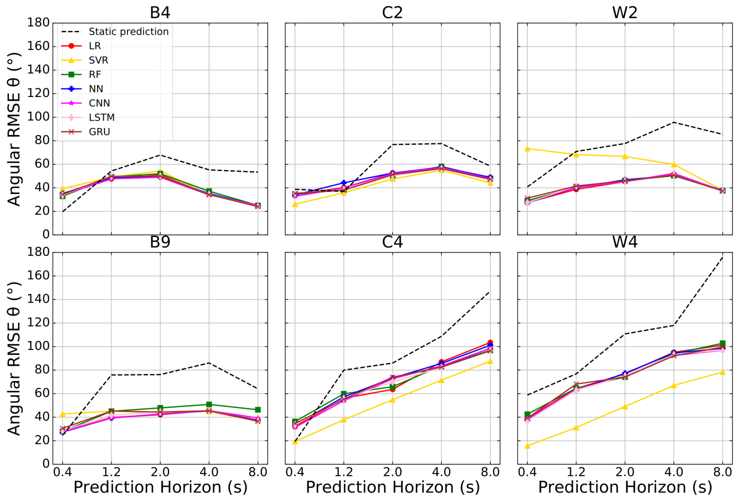


Fig. 7. Influence of the PH on the accuracy of the independent prediction model, without clustering. The input window size is fixed at 2s.

Table 5. Results of the Petrangeli [17] clustering algorithm on both datasets. The total number of clusters as well as the amount of main clusters (>3 trajectories) are shown. Between parentheses, the fraction of the video trajectories contained within each main cluster is given.

ID	Clusters	Main clusters	ID	Clusters	Main clusters
B0	28	3 (30.3%, 18.2%, 6.1%)	C0	12	1 (63.8%)
B1	24	2 (32.8%, 22.4%)	C1	1	1 (100%)
B2	14	3 (27.7%, 12.8%, 10.6%)	C2	14	1 (75.9%)
B3	18	4 (28%, 18%, 12%, 8%)	C3	24	1 (56.9%)
B4	24	5 (21.9%, 18.8%, 9.4%, 9.4%, 6.3%)	C4	16	1 (65.5%)
B5	15	2 (63.5%, 7.7%)	**Avg.**	13.4	1 (72.4%)
B6	12	3 (25.5%, 25.5%, 8.5%)			
B7	31	4 (29.4%, 12.9%, 9.4%, 8.2%)	W0	1	1 (100%)
B8	22	4 (22.2%, 22.2%, 9.5%, 9.5%)	W1	8	2 (75%, 8.3%)
B9	16	3 (32.7%, 20.4%, 12.2%)	W2	1	1 (100%)
B10	29	3 (35.4%, 13.8%, 7.7%)	W3	1	1 (100%)
B11	31	2 (41.9%, 5.4%)	W4	3	3 (77.1%, 14.6%, 8.3%)
B12	25	3 (43.8%, 10%, 10%)	W5	3	2 (85.4%, 8.3%)
B13	24	3 (37.3%, 14.9%, 13.4%)	W6	8	2 (70.8%, 14.6%)
B14	18	1 (61.7%)	W7	7	4 (58,3%, 12.5%, 10.4%, 8.3%)
B15	19	4 (27.6%, 17.2%, 12.1%, 6.9%)	W8	4	1 (89.6%))
Avg.	21.9	3.1 (60.7%)	**Avg.**	4.0	1.9 (93.5%)

correlation is almost non-existent (Table 4). As such, the findings presented in this Section confirm that correlation plays an important role when deciding upon the best suited prediction strategy. Only C4 does not seem to show specific improvement when applying simultaneous rather than independent prediction, although no decrease in accuracy is noticed either, be it apart from the LSTM.

4.3 Clustering: Pre-classifying users within the video

To complete the pre-processing analysis presented in the methodology, this section explores the effects of user clustering on the accuracy of the evaluations. To achieve this, we took the approach presented by Petrangeli et al. [17], as explained in Section 2.2.4, and applied it to the three datasets. Table 5 shows these results (with α, T_l and σ set to 0.95, 2s and 10 respectively, equal to the original implementation). Both the total number of clusters and the number of main clusters are shown. Between parentheses, the fraction of the user video trajectories contained within each main cluster is given. It can be seen that, on average, a lower amount of clusters is identified for the Wu dataset (4.0) than is the case for Corbillon (13.4) and Bao (21.9). Furthermore, a higher fraction of the user trajectories in the Wu dataset is classified within a main cluster (93.5% on average) than in the Corbillon (72.4% on average) and Bao dataset (60.7% on average). This leads to the conclusion that the Bao dataset contains much more outlying behaviour than the set of Wu, while Corbillon holds the middle between the two. For the latter, it is also important to point out that only one main cluster is found for each video, meaning a clear distinction is made between expected and outlying behaviour. This is seen in more detail in Figure 8.

Figure 8a shows the total number of clusters identified by the Petrangeli algorithm as well as the amount of clusters being identified as main clusters, *i.e.* containing more than three users, for each of the selected videos. The Wu videos in the selection show no outliers whatsoever and only a limited amount of clusters in total, i.e. 1 and 3 for videos W2 and W4, respectively. The Bao videos, on the other hand, show a much higher amount of clusters (24 and 16 respectively) of which only a small amount identifies as main cluster (5 and 3). Corbillon holds the middle between the two in terms of the number of total clusters (14 and 16 for C2 and C4 respectively), but has less main clusters, i.e. one each.

Figure 8b shows the percentual division of the total amount of users into multiple main clusters and outliers, for each of the selected videos. As mentioned, no outliers are detected for the Wu videos. All the users of W2 are incorporated in one overarching cluster, while the users of W4 are divided into one large cluster (77.1% of users) and two smaller ones (14.6% and 8.3%). It can be noticed that the Bao videos show the highest amount of main clusters and a large amount of outliers (5 and 3 main clusters and 34.4% and 34.7% outliers respectively). The Corbillon videos show only one main cluster, such that the clustering algorithm in fact acts as an outlier removal mechanism. 24.1% and 34.5% of the users are marked as outliers, respectively.

Once the clusters were selected, we performed the prediction analysis following the same procedure as in previous sections, this time on the identified clusters. Given the better performance of the independent prediction, Figures 9 and 10 show the accuracy of the models' prediction in function of the PH after clustering of the simultaneous and independent models, respectively. When comparing the simultaneous case with its counterpart without clustering, it should be pointed out that user clustering prior to model training results in a heavy decrease of accuracy for some of the videos. Especially the ANN algorithms are showing a heavy decrease in accuracy as is the case for B4 and B9 and, to a more limited extent, C2 and C4. For the Bao videos, this might be a result of the fact that a rather high amount of main clusters is found, thus resulting in a limited amount of data per cluster to train the neural models upon. The other models seem to deal better with this issue, however, even showing limited improvements when compared to the non-clustering case. User clustering on the C2 and C4 videos, on the other hand, does not result in multiple main clusters. However, 24.1% and 34.5% of the users are labeled as an outlier, respectively, therefore drastically reducing the dataset. As such, this lack of data is canceling out the benefits from outlier removal here as well. Note that, once again, improvement can be noticed for LR, SVR and RF in video C2 compared to the non-clustering case. For the Wu videos, no outliers are being detected by

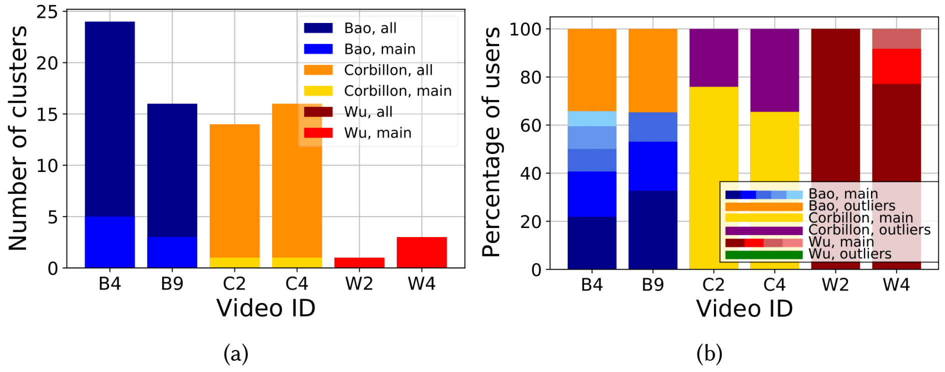


Fig. 8. The total number of clusters as well as the amount of main clusters (a) and the percentual division of the users in main clusters and outliers (b) as identified by the Petrangeli algorithm for each of the selected videos.

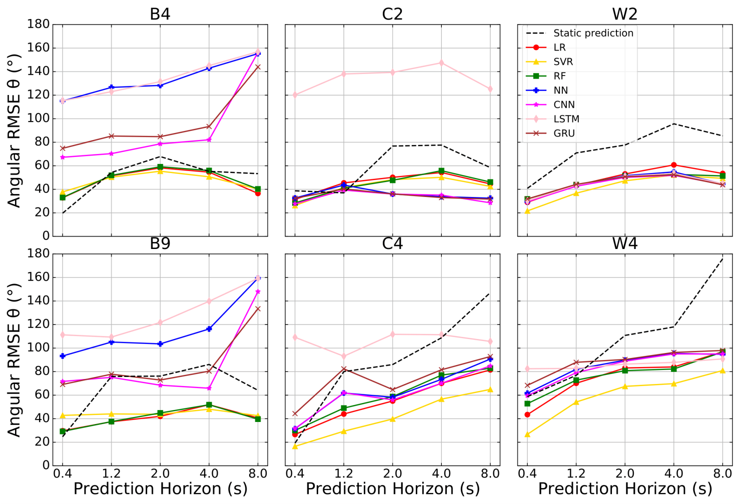


Fig. 9. Influence of the PH on the accuracy of the simultaneous prediction model, after clustering. The input window size is fixed at 2s.

the Petrangeli algorithm. This is also reflected in the results of Figure 9, as only a limited change in behaviour can be noticed in comparison with the non-clustering case. The differences that do occur are most likely a result from a lack of data in the two smaller clusters of W4 and a differing initialization during the optimization phase in case of W2.

When comparing the two independent prediction approaches to each other, it can be noticed that clustering does decrease the angular error for most of the videos and models. Video B4, for example is showing a significant improvement for PHs up to 4s. The LSTM and GRU are showing worse performance for the 8s PH, however. For video C2, improvement can be seen for PHs of 2s and higher, with the FNN at a PH of 8s as its sole exception. Video C4 also shows major improvement, with a decrease in error of up to 20° for PHs of 2s and higher. Once again, the

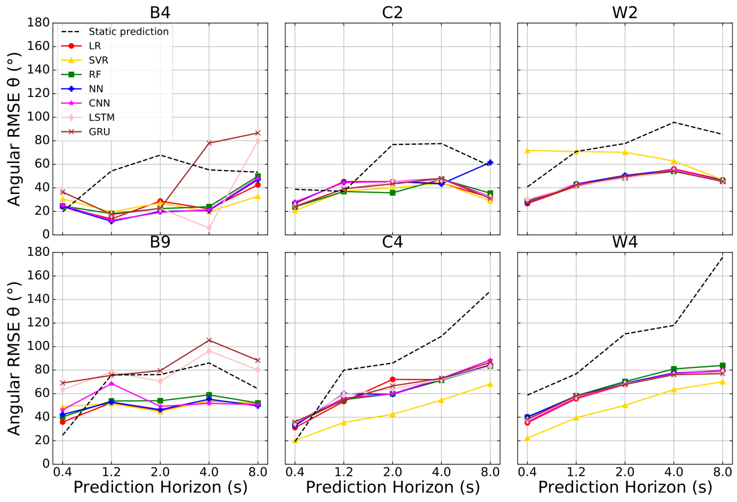


Fig. 10. Influence of the PH on the accuracy of the independent prediction model, after clustering. The input window size is fixed at 2s.

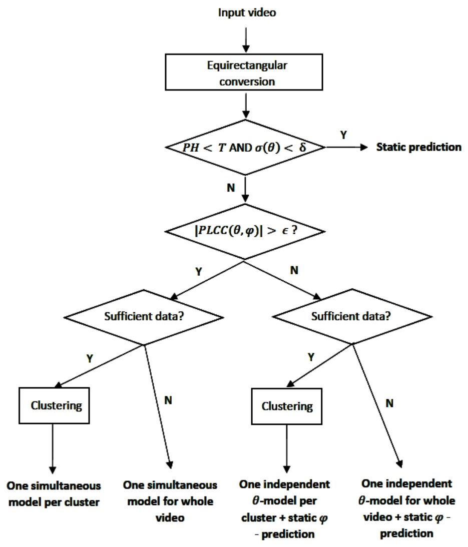
performance for the Wu videos is similar with and without clustering, as the Petrangeli algorithm is not identifying outliers. B9 forms an exception to the conclusions above, as clustering seems to negatively influence performance over all models. Furthermore, the window size again does not show to have a significant influence is therefore not further analyzed. The same conclusion is drawn concerning the autocorrelations.

4.4 Proposed method

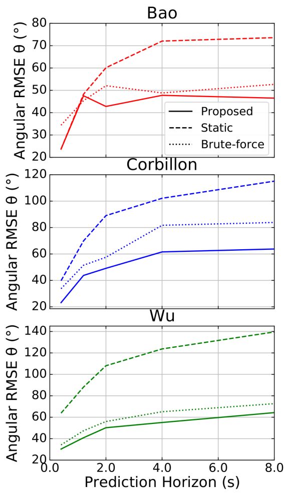
Based on the observations made in the previous Sections, four key factors were identified that influence the choice of prediction approach. These are the PH aimed at, the standard deviation $\sigma(\theta)$ over time, the correlation between θ and φ in a given video, and the amount of available data to train models upon. Based on these results, we understood that there is not a "one-size-fit-all" solution. Therefore, we designed the model selection mechanism schematically illustrated in Figure 11a.

A first question to be answered is which PH the prediction algorithm is aiming at. The results show that for a rather short PH, resulting in high autocorrelation, it is often difficult to improve on the static prediction benchmark. ML models are even showing worse results in such cases. Thus, a ML-based approach, which adds complexity, is unnecessary. Based on our results, the PH threshold T is estimated to be below 1s, which corresponds to autocorrelation values around 0.7. However, it can be noticed that for the Corbillon and Wu videos some of the ML approaches often outperform the static prediction for low PH. This can be explained by the difference in standard deviation of θ (Table 3), as higher standard deviations increase the probability of the static prediction approach to produce higher prediction errors. This results from the fact that the distribution of possible angular values is wider. As such, we propose to add a second threshold to $\sigma(\theta)$, in addition to the PH, to prefer static prediction above ML. Based on our results, the prediction error for PHs below 1s, as averaged over all datasets and videos, is minimized by setting the threshold δ between 73° (video C2) and 76.3° (video W7). δ is therefore fixed to 74.65, as the average between both.

When more long-term predictions are needed, a content-agnostic ML prediction approach is more suited for viewport prediction. A correlation analysis can be applied to decide upon the nature of the prediction algorithm. If the absolute value of the PLCC is higher than a certain value ϵ, a


Input video
Equirectangular conversion
$PH < T$ AND $\sigma(\theta) < \delta$
Y
Static prediction
N
$|PLCC(\theta,\varphi)| > \epsilon$?
Y
N
Sufficient data?
Sufficient data?
Y
N
Y
N
Clustering
Clustering
One simultaneous model per cluster
One simultaneous model for whole video
One independent θ-model per cluster + static φ - prediction
One independent θ-model for whole video + static φ - prediction

(a) Flowchart presenting the proposed viewport prediction model selecting mechanism.



(b) The obtained angular RMSE as a function of the requested PH using the proposed method, and compared to the static and brute-force prediction for each dataset.

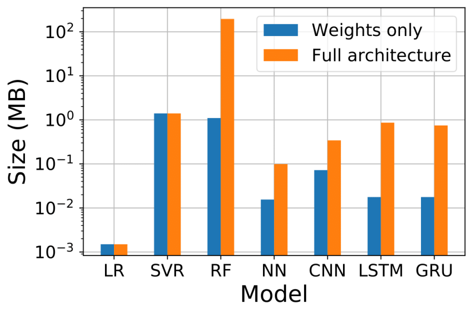
Fig. 11. The proposed method (a) and obtained results (b)

simultaneous prediction of θ and φ has shown to be more beneficial. Otherwise, an independent prediction of both angles is put forward. For most videos, the standard deviation in the φ-dimension is rather limited such that a static prediction is sufficient. However, it is possible to train a second ML-model for the φ-dimension in case more complex user behaviour is observed. Based on the findings in this paper, an ϵ value of about 0.2 is sufficient to choose simultaneous prediction over its independent counterpart.

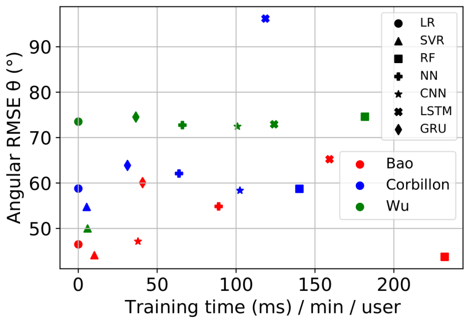
A last, important factor is the amount of data that is available at the server side to train the proposed model upon. As was shown in the results, user clustering shows to decrease the error of the model, but only when a large enough amount of data is available within each cluster. If this is not the case, one overarching model for the whole video tends to show better performance in general.

When it comes to the ML model itself, different videos and prediction scenarios tend to show different models with optimal performance such that selecting the best suited ML algorithm is a video specific task. Generally speaking, including the SVR model into the proposed method shows to be the most promising approach for accurate viewport prediction. This is the case especially for longer PHs with angular RMSE between 20 and 40° for some particular 8s PH cases resulting in an 27 to 55% improvement when compared to the static prediction baseline. When expressed as viewport overlap, this comes down to 81.8% and 63.6% respectively, which is in line with the results from other content-agnostic approaches in literature.

Figure 11b shows the obtained angular RMSE as a function of the requested PH using the proposed method. Results are shown for each of the datasets, averaged over all users and videos. The static prediction and brute-force approach, i.e. simultaneous prediction without clustering, are added by means of comparison. Based on the above discussion, T, δ and ϵ are set to 1s, 74.65° and



(a) Worst-case bandwidth requirements to send the full model architecture and the trained weights only



(b) Average model training times per user trajectory and per minute of video for each dataset

Fig. 12. Overview of training complexity and bandwidth requirements for each of the presented models

0.2 respectively. The SVR is chosen as the prediction model, and is trained and optimized per video sequence as described in Section 3.2. The amount of data is assumed to be sufficient to allow for user clustering when needed. It can be seen that the proposed methodology constantly stays below both the static prediction and the brute-force benchmark. It shows average prediction errors between 23.2° and 64.3 °, depending on PH and dataset, corresponding to viewport overlaps between 78.9% and 41.5%. Especially for larger PH, e.g. 8s, improvements of 11.5% (Wu) to 24.0% (Corbillon) in comparison with the brute force approach can be seen, as well as a 36.8% (Bao) to 53.9% (Wu) improvement of the static prediction.

Apart from accuracy, it is relevant to investigate training times and bandwidth requirements of the models. To this end, each model is stored after training and its memory requirements and training times are logged. Figure 12a shows the worst-case ML model sizes encountered during optimization and training over all videos and datasets. A distinction is made between the memory requirements of the full architecture of the model and its trained weights only. This is because it is possible to send the full model at each request of a new video or, alternatively, pre-install a (set of) model architecture(s) on the HMD and only send the appropriate weights upon video request. As can be seen, the LR model is (logically) showing the smallest overhead with only 1.5 kB in size. The ANNs, on the other hand, are each showing sizes in the order of tens of kBs for the weights only and 1 MB at most for the full architecture. The latter is comparable to both the weight and architecture size of the SVR as well. It is worth pointing out that the RF needs an architecture size of almost 200 MB in some cases to reach optimal accuracy. As modern 5G networks are showing average download speeds in the order of 100+ Mbps, this should not be considered as a bottleneck as only a limited timespan is required at the beginning of each session to transfer the model. Due to its higher memory constraints and lower average accuracy than the SVR, however, the RF remains the least obvious choice.

Figure 12b shows the average training times per user trajectory and per minute of video playthrough for each model and each dataset, as measured on a 1.58 MHz *nVidia GeForce GTX 1080 Ti GPU https://www.nvidia.com/en-sg/geforce/products/10series/geforce-gtx-1080-ti/* with 10.92 GB of memory, in comparison to their average accuracy following the proposed approach in 11a. No other tasks were running at the time of training. Not surprisingly, the LR model is showing the lowest training times around 0.03 ms. The ANNs and the RF model, on the other hand are showing much higher complexity with training times in the order of 100 ms. The SVR holds the middle

Table 6. Summary of the presented related work

Author	Year	Approach	PH	Results
Ban et al. [3]	2018	LR + probability voting	2-6s	Viewport overlap = 60-98%
Heyse et al. [11]	2019	Contextual Bandits (CBs)	1s	Error = +-25°
Jiang et al. [12]	2018	LSTM	3s	Error < 25° in 65% of cases
Li et al. [13]	2019	LSTM	1-10s	Hitrate = 75-95%
Petrangeli et al. [17]	2019	User clustering + trend trajectory	5s	Viewport overlap > 50% in 70% of cases
Qian et al. [18]	2016	Weighted Linear Regression (WLR)	0.5-2s	Accuracy = 71.2-96.6%
Van der Hooft et al. [19]	2019	Spherical walks	2s	Error = 5-60°
Van der Hooft et al. [20]	2020	Spherical walks	2s	Error = 20-35°
Vielhaben et al. [21]	2019	Linear SVM	1s	Precision = 75%
Xie et al. [25]	2018	DBSCAN + probability theory	1-5s	Precision > 80% for 30-50% of users

between both, with training times between 5.3 and 10.2 ms. Prediction times are all in the order of 0.1 ms. As such, real-time viewport prediction is no issue. It is also interesting to see that the SVR is showing the best prediction results on average while keeping its complexity on a lower level than most of the other models. This, in comparison with its average bandwidth requirements, makes it the best suited model choice.

5 RELATED WORK

The last years have seen a growth in the number of approaches for viewport prediction of 360° videos. As mentioned above, these can be roughly divided into two categories: content-aware and content-agnostic methods, both with and without the application of ML. In addition, a lot of hybrid solutions exists that combine aspects of the former two. As this paper focuses on the creation of a content-agnostic prediction method, this Section will be limited to this particular type of approach. Table 6 shows an overview of the most prominent studies.

Ban et al. [3] present a content-agnostic model in which the three angles (yaw, pitch, roll) are predicted independently using a combination of a LR on the past user trajectory and a probability voting mechanism on other user trajectories. The latter is done by selecting the K fixations of other users closest to the LR prediction by means of a K-Nearest-Neighbours (KNN) approach. Afterwards, a weighted combination of the LR prediction and the K neighbours is calculated to obtain the eventual prediction. They evaluated their approach on three videos from the Wu trajectory dataset [22]. Their results show viewport deviations from 2-30% for a prediction of 2s in the future to 15-40% for a 6s prediction. This is an absolute improvement of about 20% in comparison with a standard LR prediction approach.

Heyse et al. [11] present a content-agnostic, CB-based learning approach. A CB bank of four agents is used, where two of the agents predict whether or not movement will occur in the horizontal and vertical dimension, respectively, while the other two predict the actual movement in the same directions. Their approach shows an average error of 25° as averaged over a limited custom-made dataset of 5 users for predictions of 1s in the future.

Jiang et al. [12] developed a content-agnostic prediction approach as part of a 360° video streaming framework called *Plato*. Their predictor consists of an LSTM layer followed by a tanh transformation layer. Evaluation on the publicly available Corbillon dataset [6] shows yaw prediction errors below 25° for approximately 65% of the cases using a 3s PH.

Li et al. [13] propose a content-agnostic, ML-based prediction method based on a viewport representation consisting of center trajectories. The method includes an LSTM based sequence-to-sequence model that combines the user's historical movement with other users' trajectories. Their model is evaluated on the publicly available datasets of Xu et al [26] and Wu et al. [22], respectively.

The trajectory-based method obtains viewport hitrates from 95% to 75% for PHs varying from 1 to 10s.

Petrangeli et al. [17] aim at structuring the user trajectories by clustering them using an adapted version of the spectral clustering algorithm [15] as proposed by Atev et al. [1]. Afterwards, a per-cluster *trend trajectory* is calculated which is further used as the content-agnostic predictor for every user within that specific cluster. The horizontal and vertical angle of the viewport center (relative to a fixed coordinate system) are predicted independently, although no preliminary analysis is performed on the single, publicly available dataset [4] under scrutiny to justify this approach over a combined prediction of both angles. Their results show that at least 50% of the viewport area is predicted correctly for 70% of the users for a prediction time of 5s.

Qian et al. [18] attempt to optimize 360° video delivery over cellular networks by means of a content-agnostic, trajectory-based viewport prediction algorithm. They use a self-created, limited dataset of 4 YouTube 360 videos, whereas fixation points are tracked in terms of yaw, pitch and roll. An average predictor, LR and WLR are investigated. Their results show the WLR to be the most accurate predictor. With prediction considered to be accurate if all three dimensions differ from the ground truth with less than 10° , average prediction accuracies of 96.6%, 92.4% and 71.2% are found for PHs of 0.5, 1 and 2, respectively.

Van der Hooft et al. [19] use a subset of the Wu dataset, with user trajectories expressed as polar coordinates. Both coordinates are predicted simultaneously using a content-agnostic viewport prediction scheme based on spherical walks. Their best performing model shows average prediction errors ranging from 5° to 60°, depending on the average user speed, for a prediction of 2s in the future. In a subsequent study [20], they reuse their viewport prediction scheme as a part of a tile-based adaptive streaming framework for VR video. Evaluation on a subset of three videos from the dataset of Wu et al. [22] shows prediction errors between 20° and 35° for a PH of 2s.

Vielhaben et al. [21] present a content-agnostic prediction method based on a linear Support Vector Machine (SVM). Both prediction from historical user movement and from population statistics (Standard deviation, autocorrelation, Root Mean Squared Distance (RMSD)...) is researched. There approach was evaluated on a self-created dataset consisting of 50 spherical YouTube and Vimeo videos, whereas each video is watched by 10 subjects. The model is build to predict whether the future gaze orientation (after a certain PH) will surpass a given threshold. The results show precision values up to 0.75 for a PH of 1s and the threshold set to 20°.

Xie et al. [25], at last, also use the dataset of Wu et al. [22] for their research. They represent viewport centers in terms of unit vectors in the Cartesian coordinate system. Based on this representation, a *DBSCAN* [7] user clustering algorithm is developed to combine users with similar interests. Per class, the viewing probabilities of the 2D hemisphere tiles are calculated, which act as a content-agnostic model for future viewport prediction. Their method is evaluated on a self-created dataset for PHs of 1, 3 and 5s. Their results show a prediction precision of >80% for 50%, 35% and 30% of the users, respectively.

The above literature review has revealed that the number of studies related to content-agnostic viewport prediction is rather limited. In addition, ML prediction methods are rarely used and the ones that do exist often consist of LSTM-based methods. This is rather remarkable, as our research shows SVMs to be more applicable both in terms of accuracy, with an average 36.8% to 53.9% to the static prediction baseline, and training complexity. This even holds for longer PHs up to 8s, countering the belief that Recurrent Neural Networks (RNNs) such as LSTMs are needed for this task [12, 13]. Only one study was found that also applies an SVM [21], with that difference that it has only be applied to the rather short PH of 1s. More straightforward approaches include the use of LR [3, 18], which is understandable from a complexity point of view as our results confirm. From the point of accuracy however, it seems that LRs tend to show rather unpredictable behaviour with results varying heavily depending on the particular video and/or dataset. This can be seen from Bao et al. [4], with results between 60% and 98% viewport overlap; Qian et al. [18], with an accuracy varying between 71.2% and 96.6% and our own results that show angular RMSE from 46.5° (Bao) to 73.5° (Wu).

Furthermore, it is worth mentioning that most studies use a rather brute force prediction approach without examining specific user behaviour or any other form of preprocessing. Only three studies investigate similarities in user trajectories in their prediction approach: Ban et al. [3] (probability voting), Xie et al. [25] (DBSCAN) and Petrangeli et al. [17] (user clustering), whereas the latter is the clustering algorithm included in the analysis of this work. This is an interesting observation, as our results show that choosing the appropriate prediction approach based on standard deviation, correlation and trajectory similarity can reduce the brute-force ML error with 11.5 to 24 %.

6 CONCLUSIONS

This paper has proposed an end-to-end solution for real-time content-agnostic viewport prediction in 360° videos. The solution consists of an equirectangular conversion to angles θ and φ, followed by an autocorrelation analysis and standard deviation threshold based on which there is decided whether ML prediction could improve upon straightforward static prediction. Next is the preprocessing step, which consists of a correlation analysis, based on which there is decided between simultaneous and independent prediction of the two angles. In addition, an optional user clustering is performed if the amount of available data allows so. Afterwards, per-video or per-cluster models are trained in an offline manner, which are updated on the client every time a new video is started. The method has been evaluated on three well-known datasets. We have shown that the best suited ML-model is video and case dependent, although the SVR has shown to be the most promising in general for inclusion in our hybrid method, with an angular RMSE between 20 and 40° for some particular 8s PH cases and 46.5° (Bao) and 64.3 ° (Wu) on average. In terms of viewport overlap, this translates to values between 41.5% and 57.7% respectively. This comes down to a 36.8% to 53.9% improvement when compared to the static prediction baseline. Furthermore, the proposed method shows RMSEs between 42.8° and 64.3° (i.e. 41.5-61.1% viewport overlap) for PHs above 2s. As a result, this work provides a generic and structured solution for content-agnostic viewport prediction in terms of data representation, preprocessing and modelling. Future research is planned to decide upon the PH, σ and PLCC thresholds for ML prediction and simultaneous vs. independent modelling, respectively, as well as the minimal amount of data needed to allow for user clustering.

ACKNOWLEDGEMENTS

This research is part of a collaborative project between Huawei and Ghent University, funded by Huawei Technologies, China. Maria Torres Vega is funded by the Research Foundation Flanders (FWO), grant number 12W4819N.

REFERENCES

[1] S. Atev, G. Miller, and N. P. Papanikolopoulos. 2010. Clustering of vehicle trajectories. *IEEE Transactions on Intelligent Transportation Systems* 11, 3 (2010), 647–657.

[2] T. Aykut, J. Xu, and E. Steinbach. 2019. Realtime 3D 360-Degree Telepresence With Deep-Learning-Based Head-Motion Prediction. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9, 1 (2019), 231–244.

[3] Y. Ban, L. Xie, Z. Xu, X. Zhang, Z. Guo, and Y. Wang. 2018. CUB360: Exploiting Cross-Users Behaviors for Viewport Prediction in 360 Video Adaptive Streaming. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, San Diego, CA, USA, 1–6.

[4] Y. Bao, H. Wu, T. Zhang, A. A. Ramli, and X. Liu. 2016. Shooting a moving target: Motion-prediction-based transmission for 360-degree videos. In *2016 IEEE Int. Conf. on Big Data (Big Data)*. IEEE, Washington, DC, USA, 1161–1170.

[5] X. Chen, A. T. Z. Kasgari, and W. Saad. 2020. Deep Learning for Content-Based Personalized Viewport Prediction of 360-Degree VR Videos. *IEEE Networking Letters* 2, 2 (2020), 81–84.
[6] X. Corbillon, F. De Simone, and G. Simon. 2017. 360-Degree Video Head Movement Dataset. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. ACM, New York, NY, USA, 199–204.
[7] M. Ester, H. Kriegel, J. Sander, and X. Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *KDD-96*. AAAI, Portland, Oregon, USA, 226–231.
[8] C. Fan, J. Lee, W. Lo, C. Huang, K. Chen, and C. Hsu. 2017. Fixation Prediction for 360° Video Streaming in Head-Mounted Virtual Reality. In *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'17)*. Association for Computing Machinery, New York, NY, USA, 67–72.
[9] X. Feng, Z. Bao, and S. Wei. 2019. Exploring CNN-Based Viewport Prediction for Live Virtual Reality Streaming. In *2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*. IEEE, San Diego, CA, USA, 183–1833.
[10] X. Feng, Y. Liu, and S. Wei. 2020. LiveDeep: Online Viewport Prediction for Live Virtual Reality Streaming Using Lifelong Deep Learning. In *IEEE Conf. on Virtual Reality and 3D User Interfaces (VR)*. IEEE, Atlanta, GA, USA, 800–808.
[11] J. Heyse, M. Torres Vega, F. De Backere, and F. De Turck. 2019. Contextual Bandit Learning-Based Viewport Prediction for 360 Video. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, Osaka, Japan, 972–973.
[12] X. Jiang, Y. Chiang, Y. Zhao, and Y. Ji. 2018. Plato: Learning-based Adaptive Streaming of 360-Degree Videos. In *2018 IEEE 43rd Conference on Local Computer Networks (LCN)*. IEEE, Chicago, IL, USA, 393–400. https://doi.org/10.1109/LCN.2018.8638092
[13] C. Li, W. Zhang, Y. Liu, and Y. Wang. 2019. Very Long Term Field of View Prediction for 360-Degree Video Streaming. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, San Jose, CA, USA, 297–302.
[14] A. T. Nasrabadi, A. Samiei, A. Mahzari, R. P. McMahan, R. Prakash, M. C. Q. Farias, and M. M. Carvalho. 2019. A Taxonomy and Dataset for 360° Videos. In *Proceedings of the 10th ACM Multimedia Systems Conference (MMSys '19)*. Association for Computing Machinery, New York, NY, USA, 273–278.
[15] A. Y. Ng, M. I. Jordan, and Y. Weiss. 2001. On Spectral Clustering: Analysis and an algorithm. *NIPS* 2, 14 (2001), 8.
[16] A. Nguyen, Z. Yan, and K. Nahrstedt. 2018. Your Attention is Unique: Detecting 360-Degree Video Saliency in Head-Mounted Display for Head Movement Prediction. In *Proceedings of the 26th ACM International Conference on Multimedia (MM '18)*. Association for Computing Machinery, New York, NY, USA, 1190–1198.
[17] S. Petrangeli, G. Simon, and V. Swaminathan. 2018. Trajectory-Based Viewport Prediction for 360-Degree Virtual Reality Videos. In *Int. Conf. on Artificial Intelligence and Virtual Reality (AIVR)*. IEEE, Taichung, Taiwan, 157–160.
[18] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan. 2016. Optimizing 360 Video Delivery over Cellular Networks. In *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges (ATC '16)*. Association for Computing Machinery, New York, NY, USA, 1–6.
[19] J. Van der Hooft, M. Torres Vega, S. Petrangeli, T. Wauters, and F. De Turck. 2019. Optimizing Adaptive Tile-Based Virtual Reality Video Streaming. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, Arlington, VA, USA, 381–387.
[20] J. Van der Hooft, M. Torres Vega, S. Petrangeli, T. Wauters, and F. De Turck. 2019. Tile-Based Adaptive Streaming for Virtual Reality Video. *ACM Trans. Multimedia Comput. Commun. Appl.* 15, 4, Article 110 (Dec. 2019), 24 pages.
[21] J. Vielhaben, H. Camalan, W. Samek, and M. Wenzel. 2019. Viewport Forecasting in 360° Virtual Reality Videos with Machine Learning. In *2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*. IEEE, San Diego, California, USA, 74–747. https://doi.org/10.1109/AIVR46125.2019.00020
[22] C. Wu, Z. Tan, Z. Wang, and S. Yang. 2017. A Dataset for Exploring User Behaviors in VR Spherical Video Streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. Association for Computing Machinery, New York, NY, USA, 193–198.
[23] C. Wu, R. Zhang, Z. Wang, and L. Sung. 2020. A Spherical Convolution Approach for Learning Long Term Viewport Prediction in 360 Immersive Video. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. AAAI, New York, New York, USA, 14003–14040.
[24] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo. 2017. 360ProbDASH: Improving QoE of 360 Video Streaming Using Tile-Based HTTP Adaptive Streaming. In *Proceedings of the 25th ACM International Conference on Multimedia (MM '17)*. Association for Computing Machinery, New York, NY, USA, 315–323.
[25] L. Xie, X. Zhang, and Z. Guo. 2018. 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 (MM '18)*. Association for Computing Machinery, New York, NY, USA, 564–572.
[26] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao. 2018. Gaze Prediction in Dynamic 360° Immersive Videos. In *Proceedings of the Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Salt Lake City, Utah, USA, 5333–5342.
[27] L. Zelnik-Manor and P. Perona. 2004. Self-Tuning Spectral Clustering. , 8 pages.