2021 IEEE 18th International Conference on Mobile Ad Hoc and Smart Systems (MASS) | 978-1-6654-4935-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/MASS52906.2021.00040

Energy-Efficient Interactive 360° Video Streaming with Real-Time Gaze Tracking on Mobile Devices

Linfeng Shen*, Yuchi Chen*, Jiangchuan Liu*
*School of Computing Science, Simon Fraser University, Canada

***Abstract*—360° videos are becoming one of the major media in recent years, providing immersive experience for viewers with more interactions compared to traditional videos. Most of today's implementations rely on bulky Head-Mounted Displays (HMDs) or require touch screen operations for interactive display, which are not only expensive but also inconvenient for viewers. In this paper, we demonstrate that interactive 360° video streaming can be done with hints from gaze movement detected by the front camera of today's mobile devices (e.g., a smartphone). We design a lightweight real-time gaze point tracking method for this purpose. Using only the front camera, our solution detects the users' faces by a lightweight Haar-like cascaded classifier, measures the user's face-to-screen distance and sight angle, and then derives the location of the user's gaze point following a customized triangularity model. We integrate it with streaming module and apply a dynamic margin adaption algorithm to minimize the overall energy consumption for battery-constrained mobile devices. Our experiments on state-of-the-art smartphones show the feasibility of our solution and its energy efficiency toward cost-effective real-time 360° video streaming.**

***Index Terms*—360° video, gaze tracking, mobile devices, energy efficiency, machine learning.**

I. Introduction

Recorded by an omnidirectional camera or a collection of cameras, a 360° video contains the view in every direction at the same time. Viewers can control their *Field-of-View (FoV)* and interact with the scene through such devices as a head-mounted display (HMD) to have a highly immersive experience. In particular, a viewer can freely change both the position and orientation in a virtual world, with a 6-degrees of freedom (6DoF). This new type of media has been increasingly popular in recent years, and major media service platforms such as YouTube and Facebook have started offering 360° videos. In the meantime, nearly 7.1 million HMDs are shipped in 2020, and it is predicted this number will grow to 76.7 million by 2024 [1].

With the development of both software and hardware design, mobile devices such as smartphones or tablets are also ready for 360° videos. Compared with desktops or dedicated hardware, mobile devices have much constrained computation power and battery reservoir. Unfortunately, 360° videos are known to be power- and energy-hungry given their high data rate and real-time demands [2][3]. For a 360° video, the region that the viewer is currently watching only counts for a small portion of the whole video. Hence, its overall resolution of the 360° video has to be at least 8K to achieve a reasonably immersive experience [4]. As such, graphic rendering and displaying on the screen impose heavy workloads for mobile devices, not to mention the streaming module that needs the real-time response to FoV changes. Many HMDs have been developed to work with off-the-shelf mobile devices, either using a mobile device as the video feed or enclose it as the display. Today's HMD solutions however remain bulky (particularly if with an extra battery) and come with extra costs, and also a certain user population (around 10-20%) have reported experiencing sickness or nausea symptoms from the use of HMDs [5]. There have also been mobile video services allowing users to move their FoV by touching and scrolling the screen of the device [6][7]. This however requires synergization between eyes and fingers, which can be challenging, particularly for those with motor disabilities. To the best of our knowledge, there have been few works on interactive mobile 360° video streaming that are hands-free and without HMDs.

In this paper, we demonstrate that interactive 360° video streaming can be done with hints from gaze movement detected by the built-in front camera of today's off-the-shelf mobile devices. Without an HMD or touch operations, our solution incurs no extra cost and is truly hands-free. Yet the computation- and energy-demands for gaze detecting and its integration with 360° video must be well planned. As a matter of fact, today's smartphones can consume as high as 2.8 Watt for high-quality motion picture recording along [8], which is non-negligible. To this end, we present a lightweight FoV controlling solution for a mobile device to track and make use of the user's gaze point on the screen. Using only the front camera, our solution detects the users' faces by a Haar-like cascaded classifier, measures the user's face-to-screen distance and sight angle, and then derives the location of the user's gaze point following a customized triangularity model. We integrate it with the streaming module and apply a dynamic margin adaption algorithm to further reduce the overall energy consumption for battery-constrained mobile devices. We have implemented a prototype of our solution, which simulates the process of gaze tracking on a laptop and displays the gaze-controlled 360° video on a smartphone. Our experiments on state-of-the-art smartphones confirm its feasibility and energy efficiency toward cost-effective real-time 360° video streaming.

The rest of the paper is organized as follows. Section 2 summarizes the related works about gaze tracking and 360° streaming. Section 3 introduces our spatial gaze point detection solution based on lightweight machine learning. Section 4 proposes our algorithm to minimize the energy con-

978-1-6654-4935-9/21/$31.00 ©2021 IEEE
DOI 10.1109/MASS52906.2021.00040

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

sumption by carefully selecting the region to stream. Section 5 shows our experiment results. We conclude our work and provide some discussions about future work in Section 6.

II. Related Work

In this section, we introduce some previous work on 360° video streaming and gaze tracking methods on mobile devices. All the steps of our method potentially consume a significant amount of energy and we need novel algorithms to improve energy efficiency. Thus we also introduce related work about improving energy efficiency on smartphones.

A. 360° Video Streaming

There have been many recent works addressing the high resource demands of 360° videos streaming and displaying. Tile-based streaming methods [9][10][11] have been proposed to save bandwidth, which divide the whole video into several tiles and then select the tiles to stream based on the current FoV. These methods generally require HMDs, and some also need the prediction of a saliency map, which introduces extra computation to mobile devices. Dambra et al. [12] present an approach with content manipulations (film editing) to limit and even control the user's motion in order to improve the streaming. Experiments show that this method can reduce head velocity up to 30% and save the bandwidth with more than 25% reduction. Sassatelli et al. [13] introduce a new additional degree of freedom for streaming, known as Virtual Walls (VWs). VWs are designed to save bandwidth along with preserving the visual quality by subtly limiting the user's freedom in well-chosen periods. Experiments with 18 users confirm that if the VMs are positioned carefully, only a substantial fraction of users seldom perceive them.

B. Gaze Tracking on Mobile Devices

Gaze tracking has always been an attractive area, but most of the related research was done on PCs with dedicated hardware, e.g., a Tobii eye tracker. Li et al. [14] propose a solution for users to type on a common mobile device by manipulating their gaze point. It employs a Gaussian regression model that is coordinated with the camera on a Mac laptop. Since the model delivers only coarse gaze point detection on the laptop screen, they propose two techniques, namely, a group centroid estimator and a transitional gaze remover, to calibrate and prune the result for mobile devices. Their solution achieves a typing accuracy of 97% and 89% in static and dynamic environments, respectively, with latencies from 2s to 2.6s, which is far from being real-time. A later solution, *Gazture* [15], explores the usage of gaze point detection to define a user's gaze point movement as gestures for the interaction purpose. *Gazture* achieves an accuracy of 1.8-2.4 cm at a rate of 12.54 fps for gaze point detection but requires active calibration from the user. Its average gesture recognition accuracy is 82.5% at a user-device distance of 50 cm and 75% at 70 cm. These solutions do not target 360° videos, nor have been integrated into 360° video streaming systems with global energy optimization.

C. Energy Optimization for Mobile Media

Yan et al. [16] explore eye adaptation in HMDs, shifting the default fixed full brightness to compromise the human eyes in dark HMD. This can achieve 25% system energy reduction with limited impact on the brightness perception. In [2], Jiang et al. propose a mechanism *QuRate* to model the correlation between the perceivable video quality and the user behavior. *QuRate* builds on top of the user's reduced level of concentration on the video frames during view switching and dynamically adjusts the frame rate with minimum impact to the perceivable video quality. This extends the smartphone battery life by up to 1.24 times. Yan et al. [17] propose a customized energy management policy that dynamically configures the mobile platform. The proposed online personality-guided user satisfaction prediction model for individual can improve the user experience by around 36% compared with state-of-the-art energy management policies. He et al.[18] propose a novel traffic redundancy elimination (TRE) system named *TailoredRe*. It clusters the clones of smartphones in the cloud and cooperatively conduct the redundancy detection task to reduce cache resource consumption in the cloud. Experiment results show that *TailoredRe* can achieve better cache hit rate, end-to-end throughput, bandwidth saving and energy efficiency compared with previous TRE methods.

III. Spatial Gaze Point Detection with Mobile Devices

In this section, we focus on detecting the viewer's spatial gaze point solely using the built-in front camera of a mobile device. We will then integrate our lightweight solution into 360° video streaming and optimize the energy efficiency in the next section.

Fig. 1 illustrates the key steps of our solution, which starts from the basic viewer feature detection for extracting useful measurement of the viewer's gaze and head pose and then derives the viewer's gaze point, i.e. the visual focus in the space around the viewer. This spatial gaze point will then be projected into a virtual space around the viewer, which will be an informative hint for gaze detection and FoV control in the 360° video playback, as will be described in the next section. All these steps are carefully designed and optimized for resource-constrained mobile devices.

A. Face and Landmark Detection

Most current mobile devices have their front cameras over the top of their screens to acquire the images where their users' faces are included. We first extract the viewer's face from the image captured by the front camera. To this end, we apply a Haar-like cascaded classifier [19], a lightweight yet efficient model for face detection. While the model can yield all faces present from the input image, we assume that only the viewer's face is included in the typical 360° video playback scenario.

With the viewer's face detected from the image, the proposed method then extracts the face landmarks from the cropped face image. For reliable face landmark detection, we designed a deep learning-based detection from the MobileNet

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

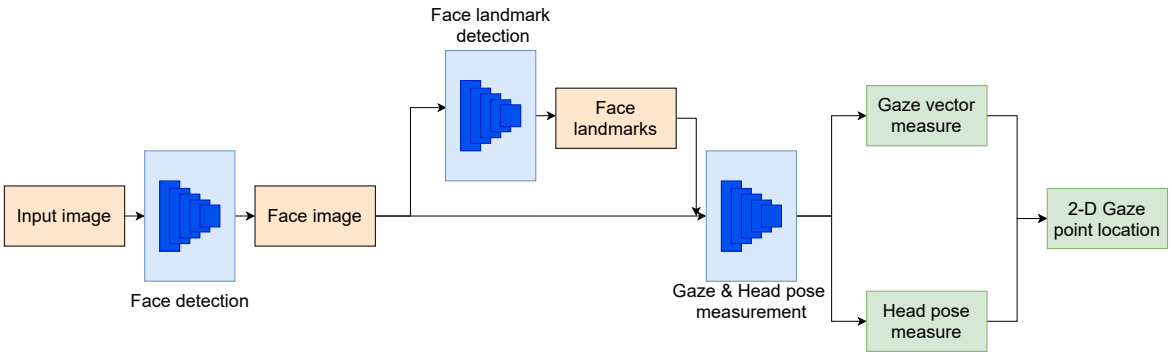

Fig. 1. Overview of the lightweight gaze tracking method for mobile devices

V2 model [20]. MobileNet is a general-purpose convolutional neural network (CNN) for feature extraction tasks, designed in a way optimized for mobile devices. Different from the CNNs that apply traditional convolution layers, MobileNet splits the layer into two sub-tasks: a depth-wise convolution for filtering inputs, and a point-wise convolution for the combination of the filtered values. The two sub-tasks together form a depth-wise separable convolution block, which operates faster than traditional convolution layers. The MobileNet V2 model we use takes one channel 64 × 64 image inputs cropped from the last step, and yields locations of 68 face landmarks [21], which locate the viewer's eyebrows, eyelids, nose top, mouth, and the contour of the face.

From the derived face image and face landmarks, a series of useful measurements can then be obtained for gaze point detection, in particular, the gaze vector and the head pose. We next introduce the gaze vector and head pose measurement from face landmarks and further spatial gaze point detection.

B. Gaze Vector Measurement

The gaze vector is defined as the line of sight sourced from the viewer's pupil. The viewer's head pose includes two key features: the transform and rotation. The transform indicates the 3-D location of the viewer's head from the camera, including the Euler distance between the head and the screen. The rotation, on the other hand, indicates the impact of the head rotation on the gaze vector.

For measuring the gaze vector of each eye, we use the locations of the inner and outer corner of the eye region, as well as the location of the pupil, as the hints. More specifically, taking the middle of the two corners as the initial location of the pupil, the deviation of the current location from the initial location, either on azimuth or altitude, indicates the rotation of the gaze vector on two dimensions. Formally, denote the gaze vector from the left and right eyes as $\boldsymbol{g_l}$ and $\boldsymbol{g_r}$ respectively. Assume that the azimuth deviation of the pupil in the left eye causes an angle shift of θ, then the transformed gaze vector $\boldsymbol{g'_l}$ is determined as follows:

$$\boldsymbol{g'_l} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \boldsymbol{g_l} \tag{1}$$

Similarly, assume that the azimuth deviation of the pupil in the right eye causes an angle shift of ϕ, the transformed gaze vector $\boldsymbol{g'_r}$ from the right eye can be determined on both azimuth and altitude:

$$\boldsymbol{g'_r} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \boldsymbol{g_r} \tag{2}$$

C. Head Pose Measurement

For head pose measurement, we take the 3-D location of the face landmarks on a generic head model as a reference and use the 2-D location of the viewer's face landmarks in the image to compute the 6 Degree of Freedom (6-DOF) head pose, i.e. the transform on three dimensions (x, y, z) and the rotation (α, β, γ) along the 3 dimensions. Specifically, denote matrices, $\boldsymbol{R}_x(\alpha)$, $\boldsymbol{R}_y(\beta)$ and $\boldsymbol{R}_z(\gamma)$ as the rotation matrix on three dimensions, then the general rotation matrix $\boldsymbol{R}_r$ is calculated as follows:

$$\boldsymbol{R}_r = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha) \tag{3}$$

Denote the transform vector $T = (t_x, t_y, t_z)^T$, then the relationship between the 2D location of a face landmark (u, v) and its 3D location (x, y, z) is as follows:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{I}(\boldsymbol{R}_r|T) \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{4}$$

where s is the scale factor. The matrix $\boldsymbol{I}$ is the intrinsic matrix of the camera, defined as follows:

$$\boldsymbol{I} = \begin{bmatrix} f_x & 0 & x_o \\ 0 & f_y & y_o \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

where f_x and f_y are the focus length of the front camera in terms of pixels, and (x_o, y_o) is the principle point, which is also the center of the screen.

By solving this Perspective-N-Points (PNP) problem, the rotation matrix $\boldsymbol{R}_r$ and transform vector T can be calculated, and then the 6-DOF head pose is derived.

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

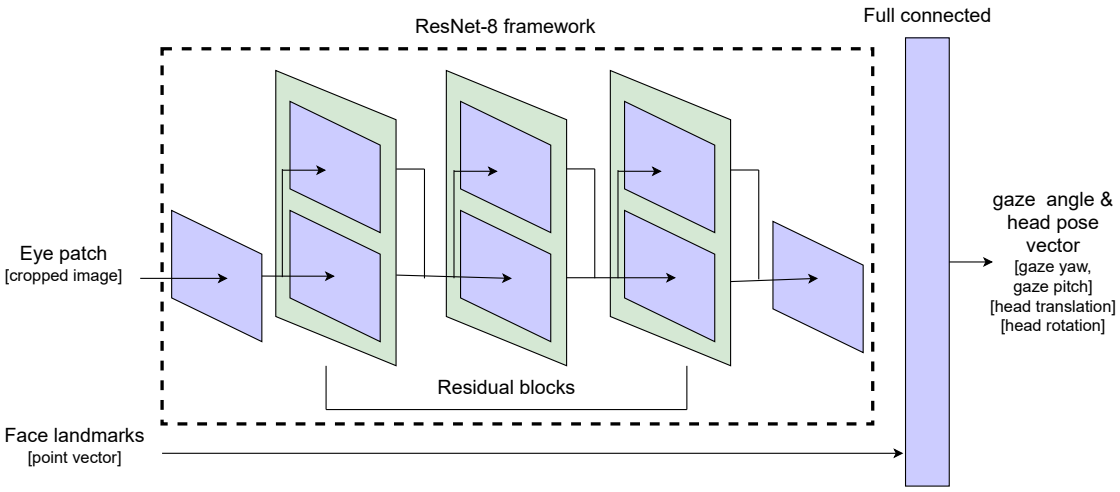

Fig. 2. Model for gaze vector and head pose measurement

Fig. 2 illustrates the model for measuring the gaze vector and the head pose. The model is built from the ResNet architecture [22], with a variation that 6 convolution layers form three stacked residual blocks.

D. Spatial Gaze Point Detection

With the gaze vector and the head pose measured, the spatial gaze point can be determined as the intersection of two lines that are sourced from both of the eyes and stretching along with the gaze. The translation of the head pose determines the spatial location of the head, which is then translated as the 3D location of the two eyes. Formally, denote the centers of the two eyes as P_l and P_r respectively, and denote the vector from one to two another as $\boldsymbol{v}_{lr}$. Similarly, denote the gaze vector of the both eyes as $\boldsymbol{g}_l$ and $\boldsymbol{g}_r$. The spatial gaze point P_g, which is the intersection point of the two lines sourced from the two eyes, is then determined as follows:

$$P_g = P_l \pm \frac{||\boldsymbol{g}_r \times \boldsymbol{v}_{lr}||}{||\boldsymbol{g}_r \times \boldsymbol{g}_l||}\boldsymbol{g}_l \tag{6}$$

where the sign of $\pm$ is determined by the two normal vectors $\boldsymbol{n}_1 = \boldsymbol{g}_r \times \boldsymbol{v}_{lr}$ and $\boldsymbol{n}_2 = \boldsymbol{g}_r \times \boldsymbol{g}_l$. If the dot product $\boldsymbol{n}_1 \cdot \boldsymbol{n}_2$ has a value larger than 0, the sign is taken as plus; otherwise the sign is taken as minus.

Fig. 3 illustrates that the spatial gaze point is taken as the intersection point, which is determined by both the gaze vectors and the head pose. Notably, if the two gaze vectors are identical, there will not be an intersection point for the two lines. In this case, we take the point where a line-plane collision happens as the spatial gaze point, which we will describe in the next subsection.

E. Measurement of 2D Gaze Point on Screen

The detected spatial gaze point may not be located on the same surface as the screen. Since we eventually need the viewer's gaze point on the screen, the 3D spatial location of the detected gaze point has to be projected onto a 2D coordinate

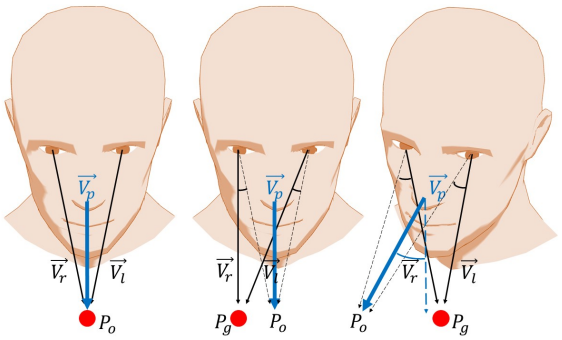

Fig. 3. Using gaze vector and head pose to determine the spatial location of the gaze point

system. Take the location of the principal point $P_o(x_o, y_o)$, which is the center point of the screen, as the initial location of the viewer's gaze point. The face-to-screen distance, denoted as d, is derivable from the transformed vector on the z-axis. Denote the Euler angle of the gaze vector on azimuth as Θ_h, and similarly, the Euler angle on altitude as Θ_e, the location of the gaze point $P_g(x_g, y_g)$ can be calculated as:

$$\begin{cases} x_g = x_o + d\tan\Theta_h \\ y_g = y_o + \frac{d\tan\Theta_e}{\cos\Theta_h} \end{cases} \tag{7}$$

where the viewer's head rotates, the deviation of the Euler angle $\Delta\Theta_h$ and $\Delta\Theta_e$ on azimuth and altitude can be acquired from the rotation matrix respectively. The location of the gaze point $P_g(x_g, y_g)$ is then given by:

$$\begin{cases} x_g = x_o + d\tan(\Theta_h + \Delta\Theta_h) \\ y_g = y_o + \frac{d\tan(\Theta_e + \Delta\Theta_e)}{\cos(\Theta_h + \Delta\Theta_h)} \end{cases} \tag{8}$$

When the viewer's distance to the screen change, both the rotation and transform of the head impacts the movement of

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

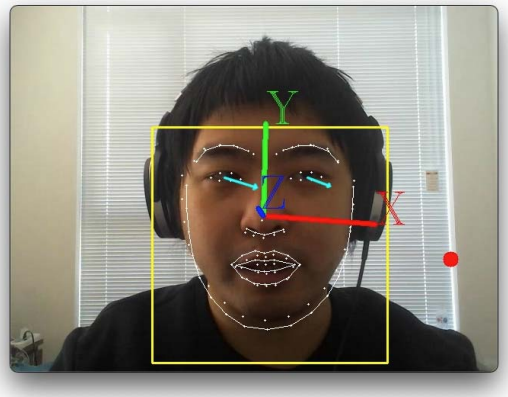

Fig. 4. An example frame of our method: gaze point is plotted by the red point; gaze vectors are shown by blue arrows; facial landmarks are shown by white lines

the gaze point. Denote the new face-to-screen distance as d', the azimuth and altitude Euler angle deviation of the head as $\Delta\Phi_h$ and $\Delta\Phi_e$, the location of the gaze point $P_g(x_g, y_g)$ is then calculated as:

$$\begin{cases} x_g = x_o + d' \tan(\Theta_h + \Delta\Theta_h - \Delta\Phi_h) \\ y_g = y_o + \frac{d' \tan(\Theta_e + \Delta\Theta_e) + \Delta\Phi_e}{\cos(\Theta_h + \Delta\Theta_h - \Delta\Phi_h)} \end{cases} \tag{9}$$

Fig. 4 shows a sample frame of the calculated gaze point on screen. To better illustrate our method, the facial landmark and gaze vector are also plotted in the frame.

IV. Energy Efficient 360° Video Streaming and Display

With the lightweight gaze tracking module for mobile devices, we now discuss the integrated system toward real-time streaming and interaction, as shown in Fig. 5. Unlike the traditional 360° videos that are displayed by HMDs, our system only uses the front camera of the mobile device to obtain the real-time gaze point on the screen, which serves as a hint of FoV. Previous work [2] has demonstrated that screen, network overhead, video decoding and view rendering contribute to most of the energy consumption in 360° video streaming. Thus we accordingly consider the four key hardware modules in a mobile device, that is the front camera, the computing module, the screen, and the network module. The gaze point and other user information will be sent back to the server, and the server will send the proper cropped frame based on the user information. Then the follow up frames can be decoded by the mobile device and displayed to the user.

A. Gaze-aware Margin Adaption

Since only part of the videos will be displayed to the viewer, it is obvious that sending a small portion of the frames can save the network bandwidth and decoding workload, which in turn saves energy. Most existing 360° video applications take a simple tiling scheme, which divides the whole frame into several small tiles and only sends the tiles that are within or around the current FoV. This unfortunately can still waste a lot of bandwidth and consequent energy because of the inconsistent pixel density, as demonstrated in [23]. We instead base our design on the concept of VAM (Visible Area plus Margin). A VAM frame includes the current FoV and a carefully selected margin area around the FoV, which has been used in the context of HMD with full head moverment information [23]. Our gaze-tracking method however only gives the gaze points on the screen and does not have the head rotation information (the *roll* parameter). To address this issue, we use the current gaze movement speed as a hint to determine the margin area. This is based on the observation that a high gaze movement speed generally causes more deviation of the gaze tracking results. We also adopt a dynamic adaption margin area instead of a static margin area to further save bandwidth. Fig. 6 shows two examples of our method, in which only the portion of the predicted FoV plus a dynamic margin area to the client. The margin area is smaller when the gaze movement speed is low and larger when the speed increases. The experiment in the next section confirms the effectiveness of our method.

Another challenge is how to correctly predict the FoV of the next frame and the gaze movement speed. There has been significant research on the use of advanced learning to predict the FoV in the HMD context [9][24][25]. They are however not well suited for our application context. First, using learning for FoV prediction and the gaze track brings heavy computing workload for the mobile device and will definitely introduce more energy consumption. Second, most of the current training datasets for 360° videos are based on HMDs, which cannot be directly applied to our system. To this end, we use a lightweight Linear Regression [26] to approximate the predicted FoV and gaze movement speed. We assume that the gaze trace moves at a nearly uniform speed and will not change in a very short time (i.e. $0.1s$). Then we use approximate the gaze trace as a line in this short period.

B. Viewer-Attention and Blink-Aware Optimization

We have implemented a prototype of our design and have extensively tested it. During the experiments, we have identified the key performance bottlenecks as well as a series of opportunities that can be explored for performance optimization. First, we note that a viewer's attention can often be distracted from outside of the screen and their gaze will then be off from the screen. This however never happens in the HMD context since an HMD provides a fully enclosed virtual world around the viewer. Hence, in our open viewing context, we apply a 'lazy' strategy when we find the gaze point is outside the screen. In this situation, we only transmit the FoV area to the client without margin area since the viewer does not really care about the content of the video in these periods. Another improvement is based on the natural adaptation of human eyes. Blink detection has been studied for a long time in other application contexts [27][28]. The human's blinking duration time is not trivial during video playback, either, albeit

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

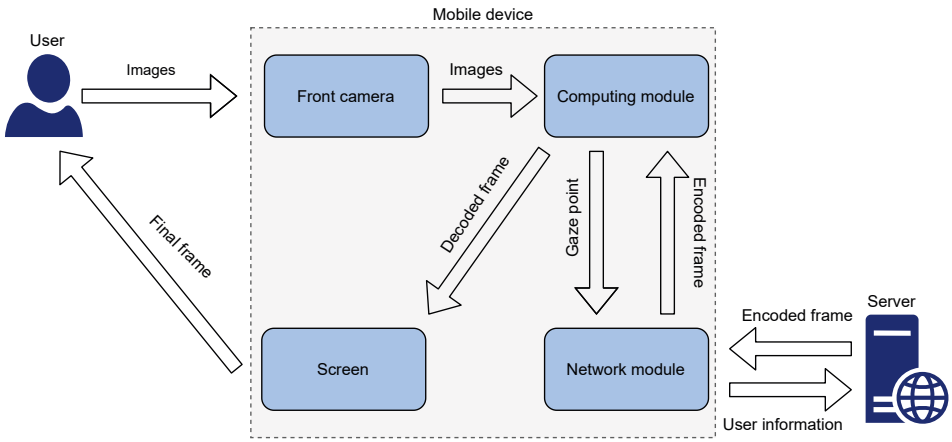

Fig. 5. System framework

Fig. 6. Dynamic margin area frames based on gaze tracking: (a) smaller margin area with low gaze movement speed; (b) larger margin area with high gaze movement speed

Algorithm 1: Dynamic Margin Selection Based on Gaze Tracking

Input : The images of the viewer's face captured by front camera.
Output: The frames showed on the screen.

Initialize margin area parameter α, screen region N;
while *video playback unfinished* **do**
  Get the gaze point (x_g, y_g) on the screen by the method introduced in Section 3;
  if (x_g, y_g) *is outside* N *or Blink detected* **then**
    Transmit origin frame F without margin area to the client;
  end
  else
    Get the predicted Fov F' for the next frame and the current gaze movement speed v using linear regression;
    Transmit the appropriate frame based on the predicted FoV plus the margin area αM to the client;
  end
  Render the frames and display on the screen;
end

not being well examined, particularly for 360° videos. The duration of a blink is between 100–400 ms according to the Harvard Database of Useful Biological Numbers and an adult will have 10-20 blinks in a minute [29]. This will result in up to 8s-period when the viewer's eyes are partly closed in one-minute video playback. As in the first situation, instead of proactively updating the FoV, we only send the frame without the margin area to the client.

The whole process to select the margin area based on real-time gaze tracking is illustrated in Algorithm 1. After we obtained the images of the viewer by the front camera, we apply the method introduced in Section 3 to get the gaze point (x_g, y_g) on the screen. If the gaze point is outside the screen region or there is Blink detected, we do not update the FoV and only transmit the origin frames to the client. On the other hand, if the gaze point is located on the screen, we apply linear regression to predict the FoV for the following frames according to the previous gaze points in a short period. Then we transmitted predicted FoV plus the margin area multiplied by a coefficient α. The effectiveness of our algorithm and its contribution to energy-saving is proved in the next section.

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

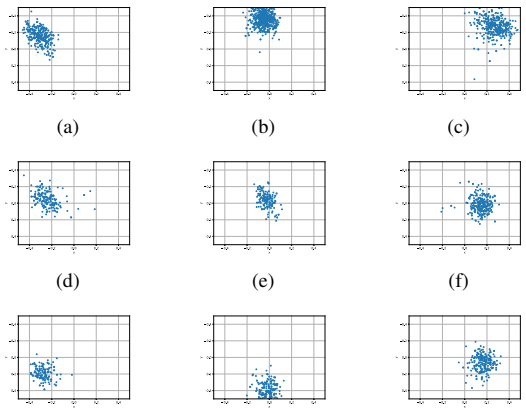

Fig. 7. Gaze point Detection on a screen divided into 3×3 tiles (left_up, up, right_up, left, middle, right, left_bottom, bottom, right_bottom)

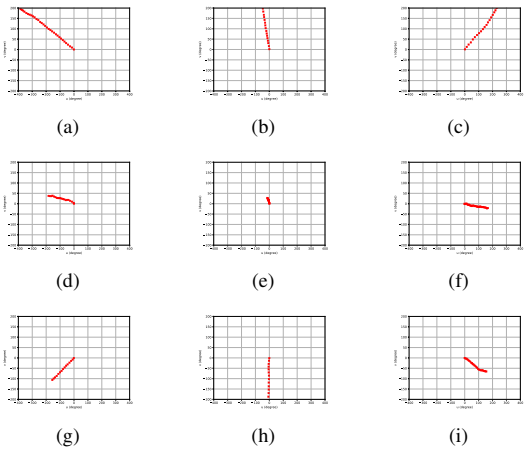

Fig. 8. FoV movement trace on a screen divided into 3×3 tiles

V. Performance Evaluation

We now present the performance evaluation results of our prototype under different configurations. On the server's side, we use a workstation with 8 CPU cores and 16GB memory, which delivers 360° videos encoded with H.264. We deploy the player on a Google Pixel 4a phone, which has a $3140mAh^2$ battery and a front camera supporting up to 1080P video recording in 30 FPS [30]. In our experiment, the FoV has a range of 90° horizontal and 60° vertical. The basic margin size is set to 10° in each direction.

A. Experiment Results

We first evaluate the performance of our lightweight gaze point tracking method. We divide the whole screen into $9(3 \times 3)$ tiles and let the viewers try their best to focus on the center of each tile for 3-5s. We record the gaze point estimated by our method and the results are shown in Fig. 7. We can see that most of the gaze points locate in the correct region but there are still some points outside the region. Given the limit of

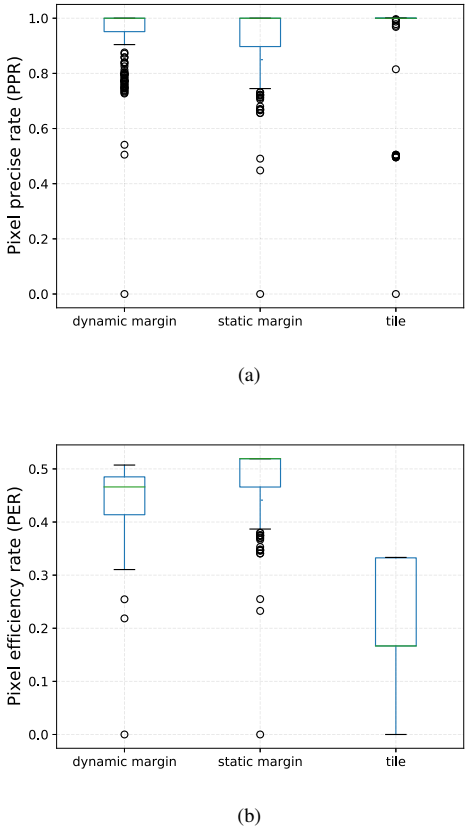

Fig. 9. Comparison between our method and baselines (a) Boxplots of PPR ; (b) Boxplots of PER

our current Google Pixel 4a, the precision of the gaze tracking implementation on it remain low. To address this problem, we also simulate the tracking using a laptop with a GeForce RTX 2060 Graphics Card, which confirms the effectiveness of our solution. Given the rapid evolution of today's smartphones, particularly the embedded neurocomputing chips, we believe comparable high precision can soon be realized on off-the-shelf smartphones. In our prototype, the update of FoV is determined by the estimation of gaze point. We move the FoV towards the region where the estimated gaze point is the spatial center. In the experiments of FoV update, the viewers also try their best to focus on the center of each tile. The traces of the FoV movement are shown in Fig. 8. We can see that the FoVs are generally moving towards the spatial point where the viewer is staring.

Next, we evaluate the performance of our Dynamic Margin Selection Algorithm and its contribution to energy saving. To evaluate the feasibility of our algorithm, we define two new evaluation matrices named "pixel-precise rate" (PPR) and "pixel-efficiency rate" (PER). PPR means the percentage of the pixels in the FoV that are included in the predicted FoV plus margin area. When there are some pixels not included in the received frames, we can only render the view on the screen

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

with nearby pixels and this will definitely impact the watching experience of viewer. Thus a higher PPR will guarantee a good watching experience of viewer during the 360° video playback. PER means the percentage of pixels that are used to render and display in the whole transmitted frame. A higher PER means that fewer extra pixels are transmitted to the client and save the network bandwidth in the server. To evaluate our method, we also implement two traditional methods as baselines:

- **Tile:** Tile-based method which divides the whole frame into 4×4 tiles.
- **Static Margin:** Static margin area method in which the margin size is set to 10° in each direction.

We compare the PPR and PER of our method to **Tile** and **Static Margin**. The results are shown in Fig. 9. Our method has a higher PPR compared to **Static Margin** which confirms the feasibility of our design. Although **Tile** has a relatively higher PPR, its PER is much lower than the other two methods. This means much energy and bandwidth are wasted in its process of 360° video display, which is intolerable in our energy-intensive system. Our method also has better PER compared to **Static Margin**. Although most of the frames in **Static Margin** have high PER, there are still some frames that have very low PER. The reason is that the PER will decrease significantly when the FoV prediction is not precise. Our method effectively tackles this challenge by applying a larger margin area in such situations.

To evaluate the contribution of our method to energy efficiency, we deploy our method and the baseline methods **Tile** and **Static Margin** on our experimental mobile phone. To ensure consistency, all the other applications on the smartphone are terminated during the playback of 360° videos. We plot the remaining battery percentage during one-hour 360° video playback in Fig. 10. We can see that our method and **Static Margin** consume less energy compared to **Tile**. Yet the energy consumption of **Static Margin** will increase significantly when the size of the margin region increases. It is hard for **Static Margin** to maintain a low energy consumption and high PPR/PER simultaneously. We can conclude that our method has the best balance of energy efficiency and quality of display among the three methods.

VI. Conclusion and Future Work

We designed a new real-time gaze point tracking method in this paper and applied it to interactive 360° video streaming. This method releases the user from the inconvenience of HMDs and hands operations. To tackle the energy challenge of our system, we designed a dynamic margin adaption algorithm to improve the energy efficiency of the power-hungry system. The experiments on a real smartphone showed the feasibility and efficiency of our system. Although its accuracy is still low now and we have to simulate some parts on a laptop, we believe it brings a new orientation for future applications and devices. In future work, we will continue improving the accuracy of gaze point tracking using pupil center estimation algorithms and explore other levers such as frame coding to reduce more energy consumption. We will also explore advanced learning tools such as model compression and work toward a fully fledged implementation on advanced smartphones.

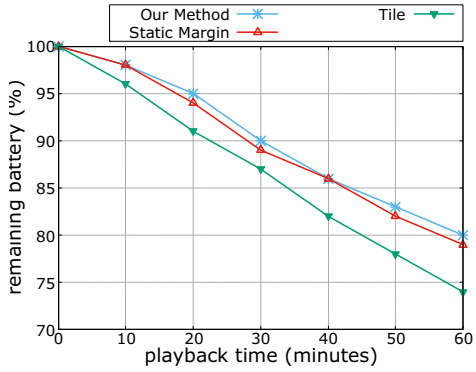

Fig. 10. Evaluation of energy consumption among three methods

Acknowledgments

This research is supported by a Huawei-SFU Joint Lab grant.

References

[1] I. D. C. (IDC). (2020) Ar and vr headsets will see shipments decline in the near term due to covid-19, but long-term outlook is positive, according to idc. [Online]. Available: https://www.idc.com/getdoc.jsp?containerId=prUS46143720

[2] N. Jiang, Y. Liu, T. Guo, W. Xu, V. Swaminathan, L. Xu, and S. Wei, "Qurate: power-efficient mobile immersive video streaming," in *Proc. of ACM Multimedia Systems Conference*, 2020.

[3] H. Chen, Y. Dai, H. Meng, Y. Chen, and T. Li, "Understanding the characteristics of mobile augmented reality applications," in *Proc. of IEEE International Symposium on Performance Analysis of Systems and Software*, 2018.

[4] J. Chen, M. Hu, Z. Luo, Z. Wang, and D. Wu, "Sr360: boosting 360-degree video streaming with super-resolution," in *Proc. of ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2020.

[5] D. Saredakis, A. Szpak, B. Birckhead, H. A. Keage, A. Rizzo, and T. Loetscher, "Factors associated with virtual reality sickness in head-mounted displays: a systematic review and meta-analysis," *Frontiers in human neuroscience*, vol. 14, 2020.

[6] Youtube VR. [Online]. Available: https://vr.youtube.com/

[7] Facebook 360. [Online]. Available: https://facebook360.fb.com/

[8] X. Chen, K. W. Nixon, and Y. Chen, "Practical power consumption analysis with current smartphones," in *Proc. of IEEE International System-on-Chip Conference*, 2016.

[9] S. Lee, D. Jang, J. Jeong, and E.-S. Ryu, "Motion-constrained tile set based 360-degree video streaming using saliency map prediction," in *Proc. of ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2019.

[10] J. He, M. A. Qureshi, L. Qiu, J. Li, F. Li, and L. Han, "Rubiks: Practical 360-degree streaming for smartphones," in *Proc. of ACM Annual International Conference on Mobile Systems, Applications, and Services*, 2018.

[11] F. Qian, L. Ji, B. Han, and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proc. of Workshop on All Things Cellular: Operations, Applications and Challenges*, 2016.

[12] S. Dambra, G. Samela, L. Sassatelli, R. Pighetti, R. Aparicio-Pardo, and A.-M. Pinna-Déry, "Film editing: New levers to improve vr streaming," in *Proc. of ACM Multimedia Systems Conference*, 2018.

[13] L. Sassatelli, M. Winckler, T. Fisichella, R. Aparicio, and A.-M. Pinna-Déry, "A new adaptation lever in 360° video streaming," in *Proc. of ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2019.

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.

[14] Z. Li, M. Li, P. Mohapatra, J. Han, and S. Chen, "itype: Using eye gaze to enhance typing privacy," in *Proc. of IEEE Conference on Computer Communications*, 2017.
[15] Y. Li, Z. Cao, and J. Wang, "Gazture: Design and implementation of a gaze based gesture control system on tablets," *Proc. of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 1, no. 3, pp. 1–17, 2017.
[16] Z. Yan, C. Song, F. Lin, and W. Xu, "Exploring eye adaptation in head-mounted display for energy efficient smartphone virtual reality," in *Proc. of ACM International Workshop on Mobile Computing Systems & Applications*, 2018.
[17] K. Yan, X. Zhang, J. Tan, and X. Fu, "Redefining qos and customizing the power management policy to satisfy individual mobile users," in *2016 49th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE, 2016, pp. 1–12.
[18] S. He, H. Shen, V. Soundararaj, and L. Yu, "Cloud assisted traffic redundancy elimination for power efficiency in smartphones," in *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 2018, pp. 371–379.
[19] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.
[20] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2018.
[21] G. Amato, F. Falchi, C. Gennaro, and C. Vairo, "A comparison of face verification with facial landmarks and deep features," in *Proc. of International Conference on Advances in Multimedia*, 2018.
[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of IEEE conference on computer vision and pattern recognition*, 2016.
[23] S. Shi, V. Gupta, and R. Jana, "Freedom: Fast recovery enhanced vr delivery over mobile networks," in *Proc. of ACM Annual International Conference on Mobile Systems, Applications, and Services*, 2019.
[24] C.-L. Fan, J. Lee, W.-C. Lo, C.-Y. Huang, K.-T. Chen, and C.-H. Hsu, "Fixation prediction for 360 video streaming in head-mounted virtual reality," in *Proc. of ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2017.
[25] S. Gül, D. Podborski, T. Buchholz, T. Schierl, and C. Hellge, "Low-latency cloud-based volumetric video streaming using head motion prediction," in *Proc. of ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2020.
[26] S. Weisberg, *Applied linear regression*. John Wiley & Sons, 2005, vol. 528.
[27] A. Królak and P. Strumiłło, "Eye-blink detection system for human–computer interaction," *Universal Access in the Information Society*, vol. 11, no. 4, pp. 409–419, 2012.
[28] T. Soukupova and J. Cech, "Eye blink detection using facial landmarks," in *21st computer vision winter workshop, Rimske Toplice, Slovenia*, 2016.
[29] Blinking, "Blinking — Wikipedia, the free encyclopedia," 2021, [Online; accessed 20-February-2021]. [Online]. Available: https://en.wikipedia.org/wiki/Blinking
[30] G. LLC. (2019) Pixel 4a hardware specs. [Online]. Available: https://store.google.com/ca/product/pixel_4a_specs

Authorized licensed use limited to: SIMON FRASER UNIVERSITY. Downloaded on January 15,2022 at 12:19:23 UTC from IEEE Xplore. Restrictions apply.