Exploring Viewer Gazing Patterns for Touch-Based Mobile Gamecasting

Cong Zhang, *Student Member, IEEE*, Qiyun He, Jiangchuan Liu, *Fellow, IEEE*, and Zhi Wang, *Member, IEEE*

***Abstract*—Recent years have witnessed an explosion of gamecasting applications, in which game players (or *gamers* in short) broadcast game playthroughs by their personal devices in real time. Such pioneer platforms, such as YouTube Gaming, Twitch, and Mobcrush, have attracted a massive number of online broadcasters, and each of them can have hundreds or thousands of fellow viewers. The growing number, however, has created significant challenges to the network and end-devices, particularly considering that bandwidth- and battery-limited smartphones or tablets are becoming dominating for both gamers and viewers. Yet the unique touch operations of the mobile interface offer opportunities, too. In this paper, our measurements based on the real traces from gamers and viewers reveal that strong associations exist between the gamers' touch interactions and the viewers' gazing patterns. Motivated by this, we present a novel interaction-aware optimization framework to improve the energy utilization and stream quality for *mobile gamecasting*. Our framework incorporates a touch-assisted prediction module to extract association rules for gazing pattern prediction and a tile-based optimization module to utilize energy on mobile devices efficiently. Trace-driven simulations illustrate the effectiveness of our framework in terms of energy consumption and stream quality. Our user study experiments also demonstrate much improved (3%–13%) quality satisfaction over the state-of-the-art solution with similar network resources.**

***Index Terms*—Gazing pattern prediction, mobile gamecasting, touch interaction.**

I. Introduction

EMPOWERED by today's high-performance mobile devices and communication networks, we have witnessed an explosion of *mobile gamecasting (MGC)* as a must-have application on gamers' mobile devices. Such MGC platforms as YouTube Gaming, Twitch, and Mobcrush, have ushered in a new wave of innovations in how multimedia content is created and consumed, distributing a game playthrough from a gamer's (i.e., broadcaster's) personal device to a large population of viewers. The recent news from Twitch reveals that more than 35% of the viewership are mobile viewers in all its gamecasting channels every month. A research report from Google also indicates that a third of the U.S. mobile gamers are defined as "avid gamers", who spend more than nine hours a week on average playing mobile games on smartphones; moreover, the more time these "avid gamers" spend on YouTube, the more time they spend on gaming. These MGC applications, as the propellents in both streaming and gaming markets, are posing significant challenges to the existing live broadcasting platforms, particularly with mobile broadcasters and viewers.

Several studies [2], [3] have already investigated the opportunities on the broadcaster-side. Yet the oscillation of source quality may affect the viewers' QoE (Quality-of-Experience) greatly. Recently, the rapid development of eye-tracking research makes foveated-aware optimization of viewers' watching experience possible, which has seen use cases for content distribution and live streaming [4], [5]. They are not targeting MGC applications, and typically need eye-tracking peripherals to collect the viewers' gazing data in real-time, which in turn produces extra energy consumption on mobile devices.

To explore the opportunities in the MGC context, we measure the data traces collected from gamers and viewers. The results show that strong correlations exist between the gamers' interactions on touch screens and the viewers' gazing patterns. Motivated by this observation, we propose a novel interaction-aware optimization framework that guides mobile gamecasting in advance, even before the source encoding step. The target and key challenges towards designing the framework lie in three aspects: (1) We need to understand the characteristics of the gamers' interactions and the viewers' gazing patterns with dynamic game strategies and eye movements. (2) We need online prediction to find the correlations with no assistance from eye-gazing peripherals preferably. (3) We need to design an optimization strategy to improve the energy efficiency and adjust the stream quality using the predicted gazing patterns.

To address the above problems, we first classify the users' behaviors into distinct groups, including single-touch, press-drag, pan, and zoom for touch interactions (as shown in Table I), and area-fixation, smooth-pursuit, and scene-saccade for gazing patterns (as shown in Table II), corresponding to the steady, slow, and fast movements of human eyes. Our framework then

Manuscript received February 11, 2017; revised June 21, 2017 and August 7, 2017; accepted August 21, 2017. Date of publication August 24, 2017; date of current version September 15, 2017. This work was supported by the National Priorities Research Program under Grant #[8-519-1-108] from the Qatar National Research Fund (a member of Qatar Foundation). The work of J. Liu was supported by the Natural Sciences and Engineering Research Natural Sciences and Engineering Research Council of Canada. The work of Z. Wang was supported by Shenzhen University and Science and Technology Innovation under Grant CXZZ20150323151850088. This paper was presented in part at IEEE INFOCOM, Atlanta, GA, May 2017. The guest editor coordinating the review of this manuscript and approving it for publication was Prof. Hermann Hellwagner. *(Corresponding author: Jiangchuan Liu.)*

C. Zhang, Q. He, and J. Liu are with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: congz@cs.sfu.ca; qiyunh@cs.sfu.ca; jcliu@cs.sfu.ca).

Z. Wang is with the Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China (e-mail: wangzhi@sz.tsinghua.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2017.2743987

TABLE I
CLASSIFICATION AND DEFINITION OF GAMER'S TOUCH INTERACTIONS

Touch Interaction	Definition	Examples
Single-touch (ST)	press once and release quickly	press a button or activate an object in a game
Press-drag (PD)	first press a few seconds to activate/select a game element and then drag it to a target	move a card onto a target area or deploy attacking path using soldiers
Pan (PA)	omnidirectional one-finger swipe in mobile games	expand the field of view when game scene is larger than screen size
Zoom (ZM)	a double-touch interaction, two fingers are used to scale up/down the game view	display (or hide) details before attacking enemy camps

TABLE II
CLASSIFICATION AND DEFINITION OF VIEWER'S GAZING PATTERNS

Gazing Pattern	Definition	Examples
Area-fixation (AF)	gaze on a fixed area	spend more time to watch the center of gamecasting if there is no player's touch interaction
Smooth-pursuit (SP)	gaze smoothly follows the movement of an object	focus on the moving game cards or buildings when gamers change strategies or deployments
Scene-saccade (SS)	move eyes between two or more fixation areas quickly	read a notice board or item descriptions

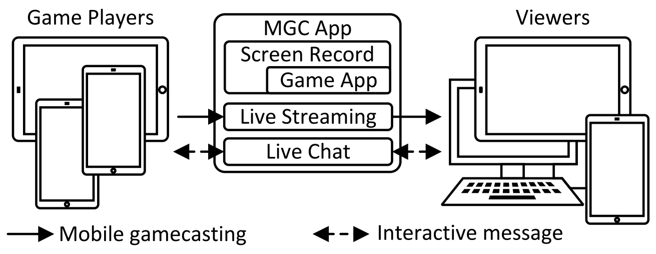


Fig. 1. Generic architecture of the MGC platform.

incorporates a touch-assisted prediction (TAP) module and a tile-based optimization (TBO) module. The former achieves offline training and online prediction by building association rules [6], and the latter improves the energy efficiency in mobile devices using a tile-based quality selection with bandwidth and QoE constraints. Trace-based simulations and a user study demonstrate that our framework achieves noticeably better QoE under similar network constraints.

II. BACKGROUND AND RELATED WORK

Recent years have witnessed an explosion of gamecasting applications, in which gamers broadcast their game playthroughs in real-time [7]. Such pioneer platforms as YouTube Gaming, Twitch, and Mobcrush have attracted a massive number of online broadcasters, and each of them can have hundreds or thousands of fellow viewers. As shown in Fig. 1, a typical MGC platform maintains two services: (1) *a live streaming service*, which not only fulfills the encoding, ingesting, transcoding, and distribution of live streams, but also implements the screen recording functionality on mobile devices; and (2) *a live chat service*, which exchanges the users' messages through IRC (Internet Relay Chat) or proprietary chatting protocols. Some recent studies have already focused on gamecasting platforms by proposing novel frameworks [7] or optimizing the live streaming service [8]. Compared with these studies, we focus on a novel branch of gamecasting services: mobile gamecasting. We therefore investigate its unique feature, i.e., gamers' touch interactions, to optimize the gamecasting transmission.

A representative MGC scenario is illustrated as follows: a gamer first launches a mobile game using an MGC application, e.g., YouTube Gaming App, on a smartphone. The top bar on the screen provides the controllers for mobile cameras, microphone, screen recording, and other configurations. After clicking the screen recording button, the gamer can use this MGC application to encode the recorded game scenes as a live stream and transmit it to an ingesting server. After multi-version transcoding, the segments of this live stream are delivered to a large number of heterogeneous viewers. During the gamecasting, the gamer and the viewers can closely interact by lively discussing game strategies via a chat service. Yet the high-performance mobile devices suffer from energy constraints with the built-in batteries, but also enjoy opportunities with novel operation interfaces, in particular, the touch screens. Several works have been devoted to analyzing the touch behaviors for specific applications, e.g., recognizing users [9]. Zhang *et al.* [10] examined the instant video clip scheduling problem based on the unique scrolling behaviors on mobile devices. Our touch-assisted prediction is motivated by these works, but we focus on the gamers' touch interactions to predict the viewers' gazing patterns for MGC applications.

Our tile-based optimization is motivated by the works in [11], [12]. They mainly employ saliency models to predict Region-of-Interest (ROI) in live streams, but can hardly capture the patterns of human gaze in real-time. To overcome these challenges, recent works have designed content transmission based on the viewers' gazing information in live streaming services [4] and cloud gaming systems [5]. These works need extra support from eye-tracking devices, e.g., a web-camera [13], to capture the gazing data during the whole process. Our work differs from them in that we predict the viewers' gazing patterns based on the gamers' interactions in MGC applications.

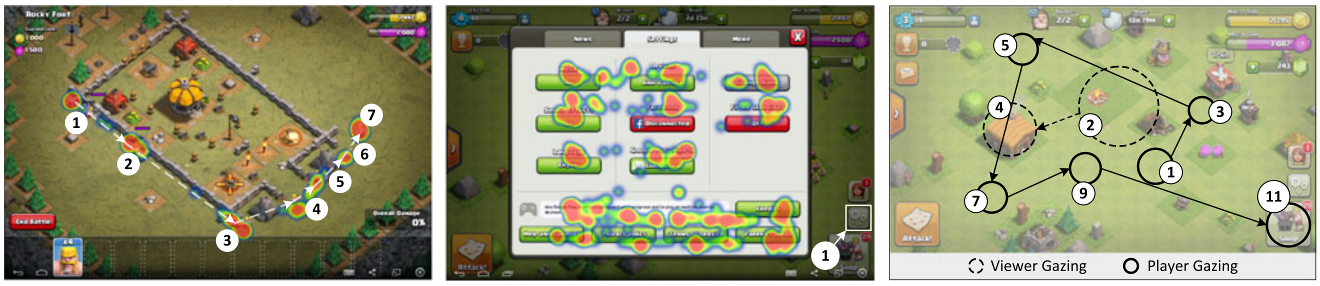


Fig. 2. Motivations of our study. (a) Example A. (b) Example B. (c) Gazing pattern comparison.

III. Motivation

In this paper, we optimize MGC applications from a new perspective. That is, through analyzing the gamers' touch interactions on touch screens, we predict the viewers' gazing patterns towards energy-efficient streaming. We first answer the following question: *How do a gamer's interactions affect the viewer's gazing patterns (including the focusing regions and movements)?* To investigate the associations between them, we capture the gamers' touch data and the viewers' gazing data through our testbed, which consists of a smartphone, an eye-tracking device, and a desktop PC. We connect the smartphone with the desktop PC to record the gamers' touch data and deploy the eye-tracking device to capture the viewers' gazing data. The details about the testbed configurations will be introduced in Sections V and VI, respectively. Here, we first highlight our findings.

Our testbed experiments show that the gamers' touch interactions change the mobile game scenes and objects, which in turn pilots the viewers' gazing patterns. Fig. 2 gives two examples to illustrate their associations using the viewers' gazing heatmap. In Fig. 2(a), a gamer designs an attacking path to deploy soldiers from region #1 to #7. Consequently, a viewer's gazing points also follow this path. This example implies that the gazing regions correspond to the touch interaction regions but have a temporal delay. In Fig. 2(b), a gamer touches the button in region #1 and activates the system setting options. Afterward, a viewer focuses on reading the information of each button. This example shows that the gazing regions do not always have spatial correlations with the touch regions, but their associations still can be found by analyzing the gamers' touch interactions and the viewers' gazing patterns together.

Yet there is a new question: if a gamer considers the game strategies through investigating a game scene first, and then decides a touch interaction in the next step, *why cannot we directly rely on the gamer's gazing data for the viewer's gazing prediction?* Such a strategy, however, needs to capture the gamer's eye movements in real-time, which needs either plugin supplements (e.g., a mobile camera) with higher energy consumption or expensive eye-tracking glasses. There are also discrepancies among different gamer and viewer groups. To illustrate it, we use an example[1] to compare the gazing differences between a gamer and a viewer in Fig. 2(c). In this figure, we plot the circle regions to exhibit the gazing movements of a gamer and a viewer, respectively. We can find that the gamer proactively focuses on lots of areas to determine the next action (i.e., click a button and open a shopping list). The gazing sequence is: the empty fields (regions #1, #3, #5, #7 and #9), and then the SHOP button (region #11). On the other hand, the viewer pays much attention to the center area in region #2, and then focuses on the Town Hall in region #4. Since a gamer must observe the game objects carefully to determine the next game strategy, s/he has more complex and unpredictable gazing patterns compared with a viewer. From these examples, we also see an association sequence between the gamers' and viewers' behaviors: gamers' gazing behaviors → gamers' game strategies → gamers' touch interactions → viewers' gazing patterns. A gamer first observes the game scenes and thinks about the game strategies; different strategies then generate the corresponding touch interactions. When a viewer watches this game video, the gazing patterns are based on two factors. First, the human eyes prefer to gaze on the center area of a scene; second, when a gamer touches or activates any object, the video scene will be changed significantly, attracting the viewer's attention. Two video examples[2] show the relationships among the touch interactions and gazing patterns of a gamer and the gazing patterns of a viewer.

IV. Interaction-Aware Design

Motivated by these observations, we design an interaction-aware optimization framework for MGC platforms, as shown in Fig. 3(a). Our design incorporates two new modules: Touch-Assisted Prediction (TAP) and Tile-Based Optimization (TBO). The TAP module predicts the viewer's gazing patterns through ingesting the gamers' touch interactions and relays the prediction results to the TBO module, which then accordingly optimizes the energy and bandwidth consumption.

A. Touch-Assisted Prediction Design

As shown on the left part in Fig. 3(b), the TAP module involves three steps: Data Collection, Data Classification, and Gazing Prediction. We highlight their design concepts here and present the details of each step in the following sections.

1) *Data Collection:* We recruit a set of gamers and viewers for training. We first collect these gamers' touch events

[1]Example link: https://youtu.be/EP2v9m9d15E.

[2][Online]. Available: https://goo.gl/2WsdtP, 2. https://goo.gl/XNS8Bu

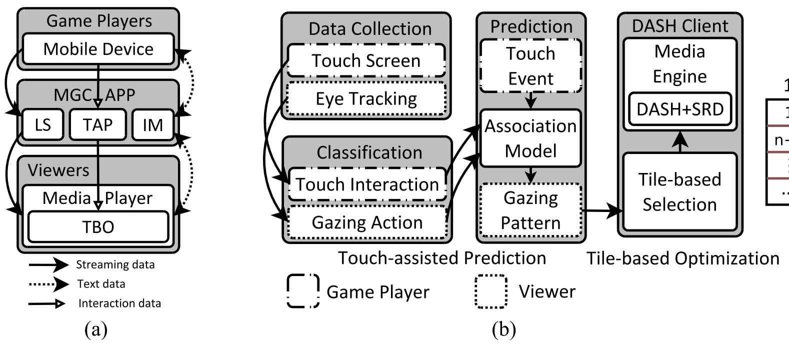


Fig. 3. Interaction-aware optimization framework in MGC. (a) Generic framework. (b) Design details in our framework.

and record the game videos when they play a mobile game. We then capture these viewers' gazing points when they watch the selected videos. These training data will be formatted and processed towards the next step.

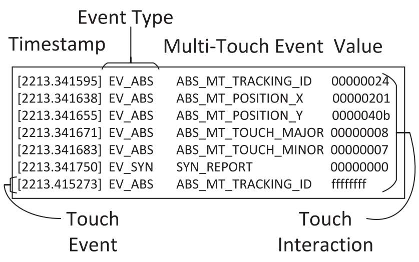
2) *Data Classification:* According to game-specific rules, we classify the touch events and gazing points into predefined groups. The gamers' touch interactions include single-touch, press-drag, pan, and zoom. The viewers' gazing patterns consist of area-fixation, scene-saccade, and smooth-pursuit.
3) *Gazing Prediction:* The main part of this step is to build an association model, which is derived from association rules learning. The prediction module receives the gamers' touch interactions and obtains the predicted viewers' gazing patterns during mobile gamecasting.

B. Tile-Based Optimization Design

In the TBO module, as shown on the right part in Fig. 3(b), our framework is based on the Spatial Relationship Description (SRD) feature in the recent MPEG-DASH (Dynamic Adaptive Streaming over HTTP) amendment [14]. SRD works to stream a subset of spatial sub-parts of a video to viewers' devices. Every frame of a short content in a video is first partitioned into multiple frames of smaller resolution. Then, these neighboring smaller frames in the same region are combined into an HTTP-based tiled content (*tile* in short). Finally, a viewer's media player renders a sequence of tiles to reconstruct this video. Every tile, therefore, contains a part of the video during a short interval. A tile-based optimization algorithm is then designed to adjust the quality of every tile according to the predicted viewers' gazing patterns. We partition every short stream into $n - by - n$ tiles following the works in [14], [15]. In practical scenarios, the tile size can be adjusted to meet different requirements [15], for example, 4×2 in HD videos [14]. In our study, the default n is set to be 5.

V. Understanding Game Touch Interactions

In this section, we investigate the gamers' touch interactions based on real-world data traces and classify game-specific touch interactions.


Event Type
Timestamp
Multi-Touch Event Value
Touch Event
Touch Interaction

```
[2213.341595] EV_ABS   ABS_MT_TRACKING_ID   00000024
[2213.341638] EV_ABS   ABS_MT_POSITION_X    00000201
[2213.341655] EV_ABS   ABS_MT_POSITION_Y    0000040b
[2213.341671] EV_ABS   ABS_MT_TOUCH_MAJOR   00000008
[2213.341683] EV_ABS   ABS_MT_TOUCH_MINOR   00000007
[2213.341750] EV_SYN   SYN_REPORT           00000000
[2213.415273] EV_ABS   ABS_MT_TRACKING_ID   ffffffff
```

Fig. 4. Sample of touch screen events.

A. Touch Data Collection

To collect the gamers' touch data, we install the Android Debug Bridge (ADB)[3] on a desktop PC (DELL Optiplex 7010) that connects to a mobile phone (Samsung Galaxy S5 with Android 6.0.1).

Fig. 4 shows a sample of touch screen events, where each line is an event with four fields: timestamp, event type, multi-touch event, and value. According to the multi-touch events and the event values, we can distinguish different touch interactions. This sample represents a single-touch interaction, which means a gamer quickly touches the screen center once. The dynamics of *touch position* (ABS_MT_POSITION_X/Y) and *touch area* (ABS_MT_TOUCH_MAJOR/MINOR) are recorded in a trace file. We write a Python script to extract the event properties of every touch interaction from the original data traces. The formatted touch interaction is a 5-tuple: {ID, start_timestamp, position_array, area_dynamics, duration}.

To understand the characteristics of distinct gamers, we recruited ten volunteers.[4] Each of them individually plays a game on S5 for two minutes. To explore the impact of game genre, we select three popular games: G1-Clash of Clans, G2-HearthStone, G3-Clash Royale and capture the screens during game playing. The selected three games not only attract a huge number of audiences in various gamecasting platforms, but also achieve high revenue in the mobile gaming market over the world.[5] Table III presents the details of our data traces. We

[3]ADB is a versatile command line tool that allows users communicate with connected Android devices.

[4]Gender, female/male: 2/8; Age, (20–25)/(26–30)/(>30): 3/5/2; Game experience, expert/beginner: 7/3, https://eyegazing.github.io/

[5][Online]. Available: https://www.superdataresearch.com/market-data/

TABLE III
STATISTICS OF TOUCH DATA

Game ID	Game Name	# of Events	# of Interactions	Game Genre
G1	Clash of Clans	65940	684	Multiplayer online strategy game
G2	HearthStone	33422	250	Collectible card game
G3	Clash Royale	37151	421	Multiplayer online battle arena, collectible card games, tower defense

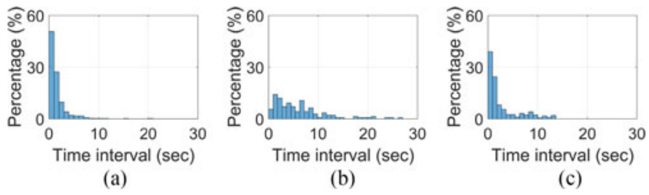


Fig. 5. Time intervals between consecutive touch interactions. (a) G1. (b) G2. (c) G3.

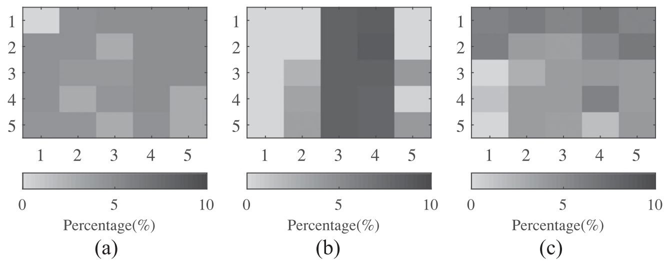


Fig. 6. Characteristics of touch regions. (a) G1. (b) G2. (c) G3.

find that the number of touch interactions is mostly determined by the gamers' preferences. There is no strict proportion between the number of events and the number of interactions. We also investigate the time interval between two consecutive touch interactions. As shown in Fig. 5, the time intervals in G2 are higher than those in G1 and G3. We will explore the impact of time intervals in Section IX-A.

B. Interaction Classification

According to the official description of touch actions in the Android Design Documentation,[6] we define the following four game-specific interactions: *Single-Touch* (ST), *Press-Drag (PD)*, *Pan* (PA), and *Zoom* (ZM), as shown in Table I. These touch interactions are frequently used in mobile games. We use a decision tree [6] to classify them with five key features extracted from the data: duration, position, direction, maximum touch area, and minimum touch area. Corresponding to the touch interactions, this decision tree has four outputs: single-touch, press-drag, pan, and zoom. To train the decision tree and test the accuracy, we use 300 touch interactions labeled by the gamers. The decision tree achieves an average accuracy of 97% for the classification of the four touch interactions, which is adequate for the association learning in our framework.

As shown in Fig. 6, we divide the touch screen into 25 regions, the vertical color-bar exhibits the mapping of the percentages

[6][Online]. Available: https://material.google.com/patterns/gestures.html

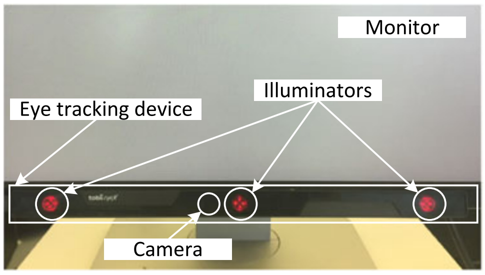


Fig. 7. Eye tracking device in our testbed.

into the color-map on the left part. We can find that (1) G1 gamers touch almost all the areas, except for area #1, because the gamers have to frequently carry out strategies in G1; (2) G2 gamers seldom touch the left-side regions, because most of the objects controlled by the gamers are located on the right-side and the bottom-side of the touch screen; (3) G3 gamers prefer the top-side and right-side areas,[7] which is also determined by the game design.

VI. INSIGHTS INTO VIEWERS' GAZING PATTERNS

In this section, we first propose the data collection method in the viewers' gazing investigation. Then, we analyze the characteristics of the viewers' gazing data.

A. Gazing Data Collection

We choose Tobii eyeX[8] as the eye-tracking device to collect the viewers' gazing data due to its affordable price, suitable sampling rate, and high accuracy. It is connected to a desktop PC (DELL Optiplex 7010) through a USB 3.0 port and attached to the frame of a 27-inch monitor (DELL U2715H), as shown in Fig. 7. The eye-tracking device consists of three illuminators and one camera. Fig. 8 illustrates that a viewer's gazing data is collected by the eye-tracking device. When the viewer watch a video, the illuminators create patterns of near-infrared light on his/her eyes. Then, the camera captures high-resolution images of the eyes. Finally, the built-in algorithms analyze these images and calculate the corresponding gazing coordinates on the monitor. The ten volunteers mentioned earlier have also assisted us to collect gazing data. Each of them have personal profiles to calibrate the eye-tracking device before the data collection. As a

[7]Because G3 is a portrait-oriented game, the top-side and right-side in Fig. 6(c) are the right-side and bottom-side of the portrait-oriented touch screen, respectively.

[8]The refreshing rate: > 60 Hz; the operating range: 50–90 cm.

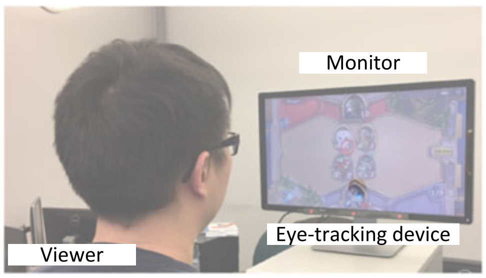


Fig. 8. Illustration of collecting a viewer's gazing data.

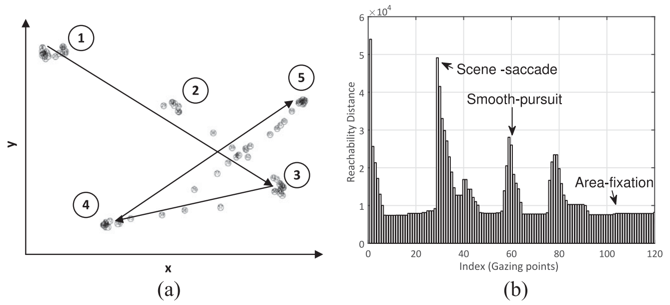


Fig. 9. Example of gazing points classification. (a) Gazing points. (b) OPTICS result.

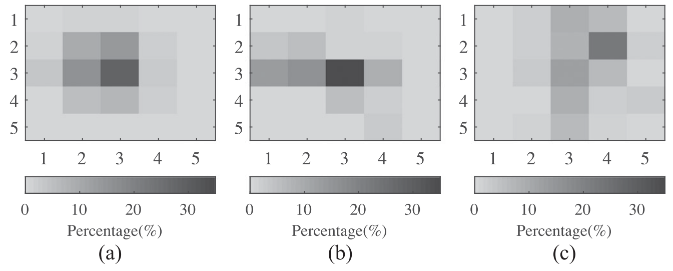


Fig. 10. Characteristics of gazing regions. (a) G1. (b) G2. (c) G3.

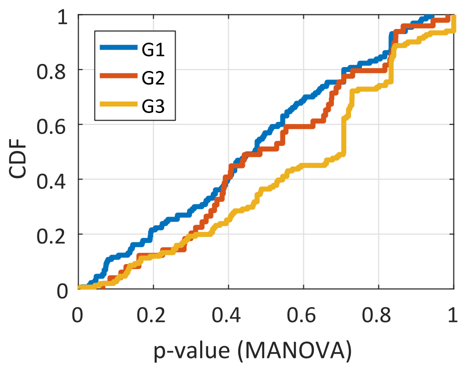


Fig. 11. CDF of p-value.

viewer, every volunteer watches three two-minute game videos selected from a gamer in Section V.

B. Gazing Classification

We define the following three gazing patterns: *Area-Fixation* (AF), *Smooth-Pursuit* (SP), and *Scene-Saccade* (SS), which are based on state-of-the-art eye-tracking research [16], as shown in Table II. Fig. 9(a) shows several examples of these patterns. In this figure, we use the labeled circles to indicate the AF patterns of a viewer. The viewer's attention is quickly changed from region #1 to region #2, which corresponds to an SS pattern. Then, we observe three SP patterns among areas #2, #3, #4, and #5.

After investigating the gazing points, we find that perfectly classifying gazing points into the three patterns is impossible due to data noises. As such, we first pre-process the gazing points to improve the accuracy. As shown in Fig. 9(a), if a viewer gazes on a fixed area, the gazing points are clustered in a two-dimensional space. Moreover, every gazing point has a temporal dimension, i.e., its timestamp, so we employ the OPTICS (Ordering Points To Identify the Clustering Structure) algorithm [17] to pre-process these gazing points. The OPTICS algorithm finds the density-based clusters (i.e., the AF patterns in our data traces) through calculating the distance between two gazing points. This distance also reflects the speed of the movement of human eyes from one area to another. Fig. 9(a) shows a pre-processing example, which depicts five AF patterns, one SS pattern (from area #1 to #2), and three SP patterns (#2→#3, #3→#4, and #4→#5). Fig. 9(b) shows the corresponding pre-processing results using the OPTICS algorithm. According to the results, we extract the features of the viewers' gazing patterns, including the time interval and the reachability distance.[9] Similar to the classification of the touch interactions, 100 labeled patterns are used to train the decision tree, which achieves an average accuracy of 96% for the classification of the three gazing patterns.

To further investigate the viewers' gazing patterns, we use a similar approach as in Section V-B to examine the characteristics of the gazing regions. According to the setting of divided tiles, we define "gazing region" to illustrate which tile is gazed by a viewer. Fig. 10 plots the percentages of the gazing regions in the three games. Note that the viewers exhibit distinct gazing preferences in different games: (1) G1 viewers mostly gaze on the left part of gamecasting; (2) G2 viewers focus on the middle part; (3) G3 viewers prefer the top part. This implies that strong correlations exist between the gamers' touch interactions and the viewers' gazing patterns in the MGC context.

To further examine the associations between the touch interactions and the gazing patterns, we choose Multivariate ANalysis Of VAriance (MANOVA) [18] to statistically analyze different viewers' gazing regions between the start of a touch interaction and the start of the next one. In MANOVA, a high p-value means a high similarity of the gazing regions among different viewers. Fig. 11 shows the Cumulative Distribution Function (CDF) of the p-values in the three games. The mean for each game is higher than 0.45. Besides, more than 60% of results are higher than 0.4, which implies that the viewers' gazing patterns are similar after the same touch interactions.

[9]The reachability distance from point g_1 to g_2 is equal to the maximum value between two types of distances: (1) the distance from point g_1 to g_2; and (2) the distance from point g_1 to the core point in its cluster.

VII. Touch-Gaze Association Learning

Association rules describe strong relations of the items that occur frequently together in a data set. In this section, we first introduce the preliminaries of association rule learning, and then build up such associations in the MGC scenario.

A. Preliminaries

The inputs of association rules learning contain: (1) *items data*, A_i, where item $A_i \in I, i = \{1, \cdots, M\}$, and (2) a *transaction set*, T, which consists of a set of transactions $< T_i, \{A_p, \ldots, A_q\} >$, where T_i is a transaction identifier and $A_i \in I, i = \{p, \cdots, q\}$. A collection of zero or more items is defined as an itemset. If an itemset contains k items, it is called k-itemset. An association rule is defined as an implication expression of form $X \rightarrow Y$, where $X \bigcap Y = \emptyset$ and $X, Y \subseteq I$. The *item support count* $\delta(X)$ of itemset X gives the number of the transactions that contain a particular itemset. $\delta(X) = |\{T_i | X \in T_i, T_i \in T\}|$. To find the frequently occurred rules, *support* and *confidence* also need to be defined. Support determines how often a rule is applicable to a given data set, while confidence determines how frequent items in Y appear in transactions that contain X. Support $s(X \rightarrow Y)$ is defined as follows: $\delta(X \bigcup Y)/M$. Confidence of a rule $X \rightarrow Y$ is accordingly defined as $c(X \rightarrow Y) = \delta(X \bigcup Y)/\delta(X)$.

To discover frequent itemsets and build reasonably strong associations, we must specify two thresholds: *minimum support*, s', and *minimum confidence*, c'. The first step is to find all the itemsets that satisfy threshold s'. These itemsets are called *frequent itemsets*. The second step is to extract all the rules from the frequent itemsets found in the previous step such that the confidence of these rules is no less than threshold c'. The second step is straightforward, while the first step needs more attention since it involves searching all possible itemsets. The classical Apriori algorithm [6] employs a bottom-up approach by identifying the frequent individual items in the transaction data set and extending them to larger itemsets while the items satisfy the minimum support threshold. The frequent itemsets returned by the Apriori algorithm are then used to determine the association rules.

B. Touch-Gaze Association

To map our data into the corresponding transactions T_i, we treat the touch interactions and the gazing patterns as items and each transaction in our study as a combination of several touch items and gazing items from the start time of one touch to the start time of the next one. To simplify the discussion and fit the tile-based optimization, we partition the touch items and the gazing items into N groups according to their positions, where $N = n^2, n \in \mathbb{Z}^+$. Each item is then encoded based on the tree structure in Fig. 12, where the fourth layer represents the order of groups. Thus, given a touch interaction or a gazing pattern, we can encode it based on its classification and position. Table IV shows one encoding example. In this example, we set $N = 25$. $\{ST, (365, 632)\}$ and $\{AF, (250, 430), (255{,}439)\}$ mean that (1) a gamer touches (365, 632) once before the next interaction;

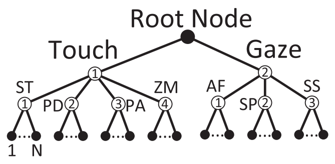


Fig. 12. Encoding tree.

TABLE IV
ENCODING EXAMPLE

Data traces	Encoded transactions
{ST, (365, 632)};{AF, (250,430),(255,439)}	T_1. {1113, 217}
{PD, (375, 700), (300, 1000)}; {SP, (280, 500),(255,900)}	T_2. {1213, 1218, 227, 2217}

Algorithm 1: Touch-Gaze Association Learning

```
Input:
(1) $A$: the list of encoded touch interactions;
(2) $B$: the list of encoded gazing patterns
Output: $G$, Association rules
1  create an empty transaction set $T$;
2  for $i$ from 1 to $|A|$ do
3      create an empty transaction $T_i$;
4      add touch interaction $A_i$ in $T_i$;
5      if $A_i$ is not the last touch interaction then
6          add the gazing patterns that occur between $A_i$ and $A_{i+1}$ in $T_i$;
7      else
8          add the gazing patterns that occur after $A_i$ in $T_i$;
9      add $T_i$ in $T$;
10 generate frequent itemsets $G$ from $T$ using the Apriori Algorithm;
11 for each itemset $g$ in $G$ do
12     if all items in $g$ belong to a type of behavior then
13         remove $g$ from $G$;
14 return $G$;
```

and (2) a viewer's gazing points contain (250, 430) and (255, 439), which is an area-fixation pattern. According to the setting of tiles, two gazing points are in a region, thus they are encoded in an item $\{217\}$. The encoded transaction T_1 includes two items $\{1113\}$ (Touch-ST-13) and $\{217\}$ (Gaze-AF-7). The rationale of this encoding method is that each encoded item contains all key information we need.

We summarize the touch-gaze association learning in Algorithm 1. Because lots of gazing patterns may occur after a touch interaction, the algorithm first adds the encoded gazing items into a sequence of transactions according to the time intervals between consecutive touch items (lines 2 to 9). Then, we use the Apriori algorithm to find all frequent itemsets for mining strong association rules in the encoded transactions (line 10). Since we aim to find the association rules between the touch items and the gazing items, we remove the frequent itemsets that only contain touch items or gazing items (lines 11 to 13). For example, we cannot find association rules from frequent itemset $\{211, 212, 235\}$, because all items in it are gazing items.

Given the frequent items, we can acquire the association rules to predict the viewers' gazing patterns. Through employing association rules for new touch data, the TAP module will predict the gazing patterns before the next touch interaction. As mentioned, the gazing region (i.e., the tile gazed by viewers) is related to the viewers' QoE. We thus define four gazing pattern sets to represent the importance of gazing regions: (1) if an AF pattern exists in a gazing region, we add the order of this region into set L_1; (2) similarly, let L_2 denote the set of the regions that contain SP patterns; (3) L_3 is the set of the regions that include SS patterns; (4) L_4 is the set of other regions that do not have gazing patterns. Finally, the result L of the predicted gazing patterns is sent to the TBO module, where $L = \{L_1, L_2, L_3, L_4\}$.

VIII. Tile-Based Optimization

In this section, we first model the tile-based optimization problem with bandwidth and QoE constraints and transform it into an equivalent problem with an efficient solution.

A. Problem Formulation

For ease of exposition, we assume that the duration D of a short stream varies from a few seconds to several minutes (e.g., two seconds in our experiments). Every short stream is reconstructed by N tiles and each tile has V quality versions. We thus denote each tile as $t_{i,j}$, $i \in \{1, \cdots, N\}, j \in \{1, \cdots, V\}$. Let $d_{i,j}$ and $e_{i,j}$ be the size and downloading energy consumption of tile $t_{i,j}$, respectively. We denote the quality of a tile by $q_{i,j}$, which is a concave increasing function of the encoding bitrate. The size and encoding bitrate of a tile can be acquired from the streaming servers, and the downloading energy consumption can be estimated [19]. We use S to denote the duration of the buffered stream on the viewer-side. The tile-based optimization can thus be formulated as to maximize the tile quality per energy consumption.

$$\text{Maximize}: \sum_{i=1}^{N}\sum_{j=1}^{V}\frac{q_{i,j}}{e_{i,j}}x_{i,j} \tag{1}$$

subject to:

Streaming Availability Constraints (2) and (3)

$$\frac{\sum_{i=1}^{N}\sum_{j=1}^{V} d_{i,j}x_{i,j}}{B} \leq S \tag{2}$$

$$\sum_{j=1}^{V} x_{i,j} = 1, i \in \{1, \cdots, N\}, x_{i,j} = \{0, 1\}. \tag{3}$$

Foveated Quality Constraints (4), (5) and (6)

$$v(t_{i,a}) \geq \alpha, t_{i,a} \in L_1 \tag{4}$$

$$0 \leq v(t_{i,a}) - v(t_{i,b}) \leq \beta, t_{i,a} \in L_k, t_{i,b} \in L_{k+1} \tag{5}$$

$$v(t_{i,a}) - v(t_{i,b}) = 0, t_{i,a}, t_{i,b} \in L_k, k \in \{1, \cdots, 4\} \tag{6}$$

where B is the average bandwidth, which is estimated according to the current gamecasting session on the viewer-side, and α and β are two tunable parameters. The rationale of the Streaming Availability Constraints is as follows: (1) all tiles should be downloaded completely before the buffer becomes empty, which guarantees smooth playback of a gamecasting; (2) the clients only need to download one quality version for every tile,

Algorithm 2: Tile-based Selection Optimization

```
Input:
(1) $T_{info}$, tile information, including the version info., size info., and energy info. for all tiles;
(2) $L$, the result of the predicted gazing patterns;
(3) $\alpha$, the minimum version for the tiles in $L_1$;
(4) $\beta$, the maximum version gap between the tiles in $L_k$ and $L_{k+1}$.
Output: $Q$, the version selection for all tiles
1  $index = 0$;
2  for each gazing pattern set $l$ in $L$ do
3      for each tile $t$ in $l$ do
4          for each quality version $v$ of $t$ do
5              remove $v$ if it does not meet the Foveated Quality Constraints;
6          update $T_{info}$;
7  if the time of downloading the lowest versions of all tiles is less than the duration of buffered stream then
8      acquire optimal version selection $Q$ using a dynamic programming solution of the MCK problem;
9  else
10     //update $\beta$ and $\alpha$ alternately to enlarge the solution space;
11     if $index \bmod 2 == 0$ then
12         $\beta \leftarrow \beta + 1$;
13     else
14         $\alpha \leftarrow \alpha - 1$;
15         $\beta \leftarrow \beta - 1$;
16     $index \leftarrow index + 1$;
17     goto line 2;
18 return $Q$;
```

which avoids extra bandwidth consumption. The rationale of the Foveated Quality Constraints is as follows: (1) to improve the quality of the tiles in gazing pattern set L_1, we denote the minimum quality version of these tiles by a parameter α; (2) we can select the version of the tiles in different gazing pattern sets to meet the streaming availability constraints and achieve the optimization target, but the version gap between two neighboring pattern sets cannot be larger than β considering that a large value will impact viewers' QoE; (3) the tiles in a gazing pattern set should have the same version.

B. Solution

If we only consider the Streaming Availability Constraints, the tile-based optimization problem can be transformed into a Multiple Choice Knapsack (MCK) problem, which is NP-hard with practically efficient solutions available (e.g., pseudopolynomial-time dynamic programming) [20]. It is worth noting that the optimal solution of this MCK problem may not meet the Foveated Quality Constraints. We therefore propose Algorithm 2. The inputs include (1) parameters α and β; (2) the prediction result L; and (3) the version of tiles, and the corresponding size and estimated energy consumption. The algorithm first narrows down the solution space (lines 2 to 6) according to the Foveated Quality Constraints before employing dynamic programming. If there exists a feasible solution (line 7), the algorithm solves the MCK problem using a dynamic programming approach; otherwise, it uses a parameter $index$ to alternately adjust β and α to enlarge the solution space (lines 10 to 16), which is based on the trace-driven simulations in Section IX-B. Finally, we can obtain the version selection of every tile (line 18).

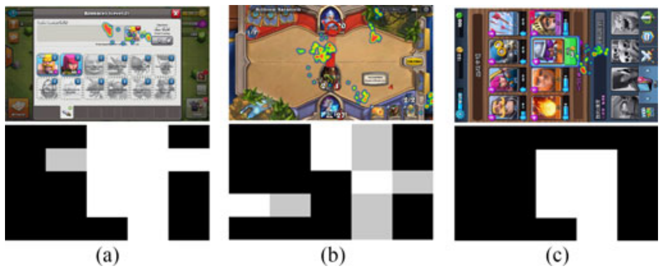


Fig. 13. Eye gazing patterns versus Prediction results. (a) G1. (b) G2. (c) G3.

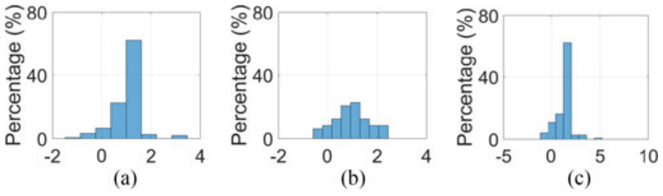


Fig. 14. Normalized scanpath saliency. (a) G1. (b) G2. (c) G3.

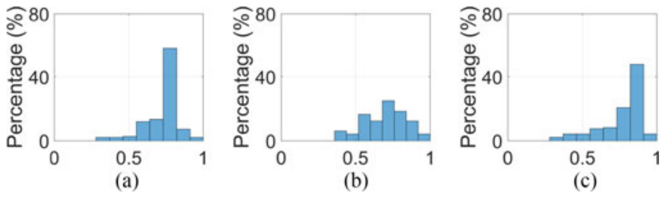


Fig. 15. Area under the curve (Borji). (a) G1. (b) G2. (c) G3.

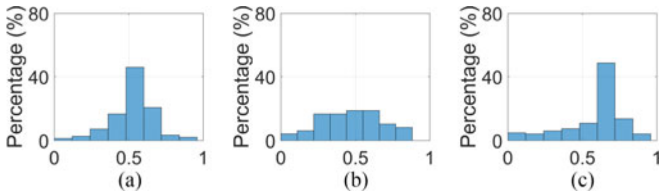


Fig. 16. Area under the curve (Judd). (a) G1. (b) G2. (c) G3.

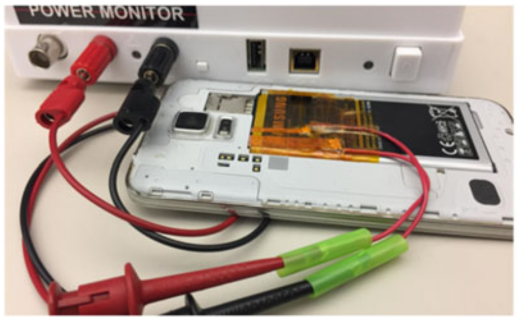


Fig. 17. Energy measurement platform.

IX. Performance Evaluation

In this section, we examine the performance of the proposed framework by the following three steps: we first evaluate the performance of the touch-assisted prediction module based on three state-of-the-art similarity metrics; then, we collect real traces to examine the performance of the tile-based optimization; finally, we conduct a user study to compare the viewers' QoE under a variety of configurations.

A. *Performance of Touch-Assisted Prediction*

In computer vision research, saliency models generate saliency maps to predict where humans look at images. Similarly, our touch-assisted prediction module can be used to generate saliency maps to reflect the gazing areas of viewers. To examine the performance of this module, we choose three state-of-the-art metrics [21]: Normalized Scanpath Saliency (NSS), Area Under the Curve (Borji implementation, *AUC-Borji* in short), and Area Under the Curve (Judd implementation, *AUC-Judd* in short). We use the three metrics to compare the similarity between the predicted saliency maps and the ground truth collected by our eye-tracking device. Higher NSS, AUC-Borji, and AUC-Judd values indicate high-valued predictions of viewers' gazing areas. The theoretical ranges of NSS, AUC-Borji, and AUC-Judd are $[-\infty, \boldsymbol{\infty}]$, $[0, \mathbf{1}]$, and $[0, \mathbf{1}]$, respectively (best score in bold). Based on the setting of tiles in this paper, every predicted saliency map consists of white/grey/black tiles, as shown in Fig. 13. In these examples, the white tiles show that these areas include AF patterns, the grey ones contain SP and SS patterns, and the black ones do not have any gazing patterns. We further plot the percentage of all metrics to show the prediction performance in Figs. 14–16. From these figures, we observe that the touch-assisted prediction achieves good performance for the three games with all the three metrics. The prediction performance in G2 is worse than others, because the time intervals between two touch interactions are longer than those in G1 and G3. That is, the time intervals between consecutive touch interactions can affect the similarities of gazing patterns among different viewers.

B. *Trace-Driven Simulation*

We evaluate our interaction-aware optimization framework through trace-driven simulations and a user study under various settings. We first connect a Samsung Galaxy S5 with Android 4.4.2 to a PC (DELL Optiplex 7010) and gain privileged control using rooting tools, CF-Auto-Root. Then, we collect the communication data using *Android tcpdump* and retrieve the tile files using *wget* on this S5 in a campus network. To measure the energy consumption, we supply the power of this S5 using a Monsoon power monitor, which connects to a PC through USB and feeds back the energy consumption of the viewer's S5 to the PC in real-time, as shown in Fig. 17.

We first investigate the impact of parameters α and β in terms of energy consumption, data transmission, and stream quality. We divide a 2-second video into 25 tiles, which are encoded at eight versions. Based on the predicted gazing pattern sets, as shown in Fig. 18(a), we conduct the tile-based optimization to determine which tile should be obtained and collect the corresponding results, i.e., the energy consumption, data transmission and stream quality, under different parameter settings. For ease of comparison, the results are normalized by the respective

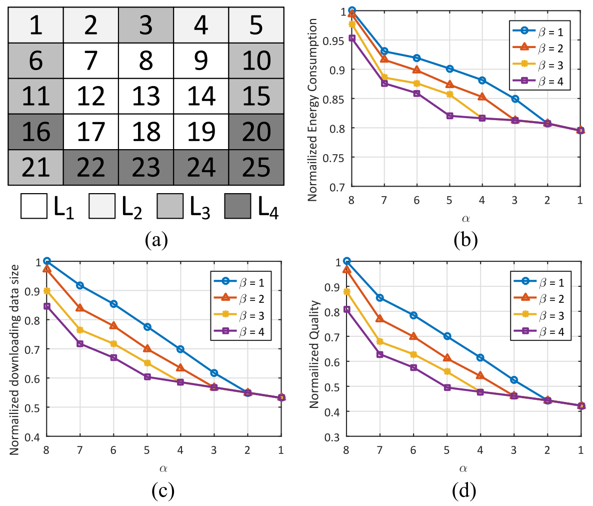


Fig. 18. Impacts of α and β . (a) Gazing pattern sets. (b) Energy consumption. (c) Data transmission. (d) Stream quality.

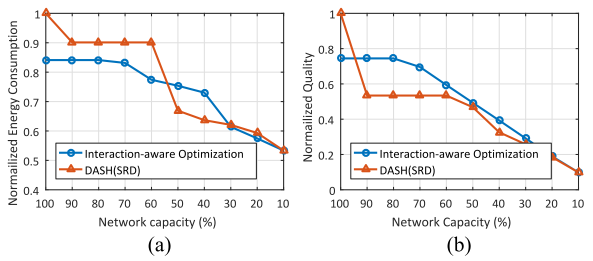


Fig. 19. Comparison of our approach and DASH (SRD). (a) Energy consumption. (b) Stream quality.

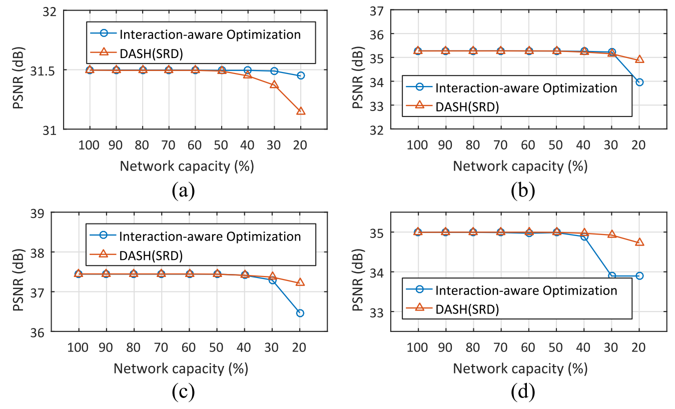


Fig. 20. PSNR experiments in four gazing pattern sets. (a) L_1. (b) L_2. (c) L_3. (d) L_4.

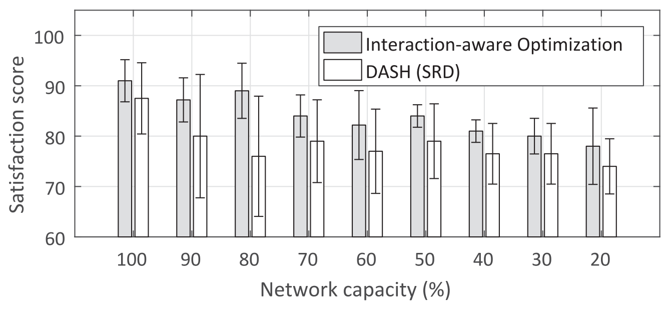


Fig. 21. Satisfaction score (error bars are 95% confidence intervals).

maximum values. We plot the normalized results under different α and β settings in Figs. 18(b)–18(d). We observe that α has a higher impact than β; therefore, if there is no feasible solution, the optimization algorithm first adjusts β, and then changes α (lines 10 to 16 in Algorithm 2). To avoid the impact of large β, we adjust it only once for different α in Algorithm 2.

To explore the effectiveness of our solution, we also examine the performance of the tile-based optimization under different network capacities through throttling the bandwidth on the mobile device. We compare our method with the original DASH (SRD) selection, in which all tiles have the same version in a short content. In our simulation, we set $\alpha = 8$ and $\beta = 1$ by default. Fig. 19 plots the results. In Fig. 19(a), we observe that our method has lower energy consumption except for two data points at 50% and 40% of the network capacity. The reason is that the original DASH adaptation may reduce the quality of tiles suddenly to accommodate a decrease of bandwidth, while our method fully utilizes the available bandwidth to optimize the tile quality per energy consumption. Fig. 19(b) shows that our solution optimizes the tile quality selection with low energy consumption except for the case of 100% of network capacity, which is determined by our optimization objective, i.e., efficient energy utilization.

C. User Study

We further conduct a user study to examine the QoE of ten viewers. We first select a 10-second video clip and generate 25 tiles at 8 versions using *ffmpeg*, *x264 encoder*, and *mp4box*.[10] According to the corresponding touch interactions, we predict the gazing pattern sets and output the video clips under different network capacities. We also produce DASH (SRD) video clips for comparisons.

We plot the PSNR for each gazing pattern set in Fig. 20. As can be seen, our approach scores a slightly higher PSNR under different network capacities in Fig. 20(a); however, PSNR decreases in the other three sets because our approach optimizes the tiles in the area-fixation pattern set. To compare the quality differences of the two video clips, we deploy a desktop PC to simultaneously play them under the same network capacity. To guarantee the fairness of comparison, we play the two clips in randomly selected windows every time. The viewers compare their quality differences with the source video clip through grading their satisfaction. Here, we use a satisfaction score to represent the viewers' evaluation about the qualities of their gazing tiles and the whole scene. The satisfaction score is from 1(worst) to 100(best). If they find the quality of one gazing region in video clip A is higher than that in video clip B, they will assign a high score to A. Because the source video clip has the best quality, we assume that its satisfaction score is 100. Fig. 21

[10][Online]. Available: https://gpac.wp.mines-telecom.fr/mp4box

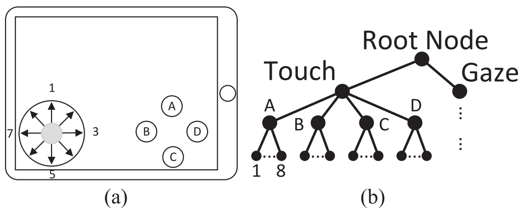


Fig. 22. Mobile games with a virtual controller. (a) Virtual gamepad. (b) Encoding tree.

shows the evaluation results under different network capacities. From this figure, we observe that our approach always achieves a higher score (3%–13%) than DASH (SRD).

X. Further Discussion

A. Impact of Game Control Approaches

A virtual controller provides a virtual gamepad to move the objects in mobile games, as shown in Fig. 22(a). A gamer controls the movements of objects by the left thumb and presses several functional buttons by the right thumb. To process the touch data in mobile games with a virtual gamepad, we can modify the structure of the encoding tree by replacing certain nodes, as shown in Fig. 22(b). The nodes in the third layer represent the buttons controlled by the right thumb, and the nodes in the fourth layer represent the different directions controlled by the left thumb. Using this encoding tree, the touch interaction in mobile games with a virtual gamepad can be transformed to transactions as shown in Section VII.

B. Impact of Game Mechanisms

Many mobile games are non-deterministic, e.g., a gamer touches and attacks the enemies that appear in different locations randomly. If a new location never appears in the collected touch data, the association rules cannot provide any feedback to the TBO module. To address this issue, we propose the following solution: after learning the rules from the original setting of tiles (e.g., the 5 × 5 setting in this paper), the association learning algorithm also adjusts the setting of tiles to learn the rules, e.g., from 5 × 5 to 4 × 4. For example, when an enemy appears at the location of tile #1 in the 5 × 5 setting, the gamer touches and attacks it. If there does not exist any rule about tile #1 in this setting, the TAP module searches the rules in the 4 × 4 setting. Because tile #1 in the 4 × 4 setting contains the contents of tiles #1, #2, #6, and #7 in the 5 × 5 setting, if the TAP module finds rules in the 4 × 4 setting, we consider these four tiles together to optimize the gamecasting.

If a game involves mostly randomly generated objects, the benefit would diminish. For example, in a Pinball game, the ball is only controlled by the player when launching it at the first step or redirecting it by two flippers. Then, the movements of this ball totally depend on the design of the playfield. In this scenario, the viewers' gazing patterns can hardly be predicted.

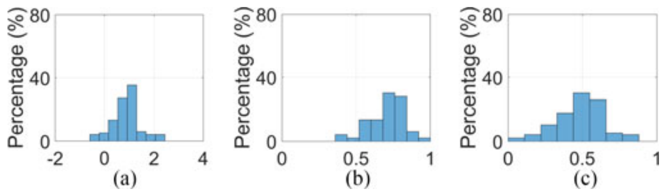


Fig. 23. Performance of new strategy (G2). (a) NSS. (b) AUC-Borji. (c) AUC-Judd.

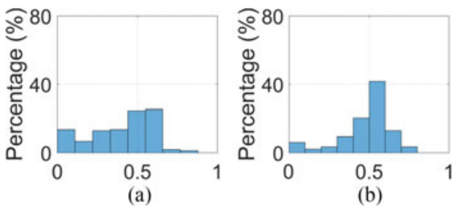


Fig. 24. Impact of viewers' game experiences (AUC-Judd, G1). (a) Beginners. (b) Experts.

C. Impact of Time Interval Between Touch Interactions

Based on the touch data, we observe that the time interval of touch interactions in G2 is longer than in G1 and G3. In Section IX-A, the touch-assisted prediction performance in G2 is worse than others. Thus, we design the following strategy to address this issue. According to all viewers' gazing data, the prediction module first learns a general pattern, in which the tiles in the center region get higher quality than other regions. If there is a long interval between two interactions, the tile-based optimization determines the quality of every tile according to the general gazing pattern. This approach not only provides the predicted gazing patterns after a touch interaction, but also improves the adaptability of our framework with long intervals between consecutive touch interactions. This modification improves the performance of touch-assisted prediction in all the three metrics, as shown in Fig. 23 for G2 (about 20% improvement).

D. Impact of Viewers' Game Experiences

In our data traces, we have three beginners and seven experts. Compared with the game experts, what are the differences of the beginners when watching mobile gamecastings? To answer this question, we further recruit five game beginners and one expert. As shown in Fig. 24, when we calculate the AUC-Judd values between the predicted saliency maps and the gazing data in different groups, the experts' gazing data achieve a higher performance than the beginners' ones. Through further investigating the characteristics of the gazing data, we find that the SS and SP patterns in the beginners' gazing data account for more than 27%, while these two patterns only account for about 20% in the experts' data. That is, beginners who are not familiar with games are easily distracted by the animation of game scenes.

XI. Conclusion and Future Work

In this paper, we explored the emerging mobile gamecasting systems in which both stream sources and receivers are mobile devices. Through collecting traces from gamers and viewers in the real world, we identified the relations between the touch interactions of the gamers (i.e., broadcasters) and the gazing patterns of the viewers. Motivated by this, we proposed an interaction-aware optimization framework that includes two novel designs: (1) a touch-assisted prediction module to build association rules offline and perform viewers' gazing pattern prediction online; and (2) a tile-based optimization module for energy consumption and quality selection under limited network capacity. The experimental results showed that our solution effectively utilizes the available bandwidth with better tile quality and less energy consumption. The user study also proved that the approach improves the viewers' satisfaction (from 3% to 13%). We are currently implementing the whole framework to further evaluate the performance from different perspectives, such as the parameter selection in the optimization algorithms, the impact of the number of tiles, and the energy utilization under different wireless networks.

Acknowledgment

We would like to thank the guest editor and anonymous reviewers for their valuable comments and suggestions. The findings achieved herein are solely the responsibility of the authors.

References

[1] C. Zhang, Q. He, J. Liu, and Z. Wang, "Beyond the touch: Interaction-aware mobile gamecasting with gazing pattern prediction," in *Proc. IEEE INFOCOM*, 2017, pp. 1027–1035.

[2] S. Wilk, R. Zimmermann, and W. Effelsberg, "Leveraging transitions for the upload of user-generated mobile video," in *Proc. 8th Int. Workshop Mobile Video*, 2016, pp. 5:1–5:6.

[3] S. V. Rajaraman, M. Siekkinen, V. Virkki, and J. Torsner, "Bundling frames to save energy while streaming video from LTE mobile device," in *Proc. 8th ACM Int. Workshop Mobility Evolving Internet Architecture*, 2013, pp. 35–40.

[4] J. Ryoo, K. Yun, D. Samaras, S. R. Das, and G. Zelinsky, "Design and evaluation of a foveated video streaming service for commodity client devices," in *Proc. 7th Int. Conf. Multimedia Syst.*, 2016, pp. 6:1–6:11.

[5] H. Ahmadi, S. Zad Tootaghaj, M. R. Hashemi, and S. Shirmohammadi, "A game attention model for efficient bit rate allocation in cloud gaming," *Multimedia Syst.*, vol. 20, no. 5, pp. 485–501, 2014.

[6] X. Wu *et al.*, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, no. 1, pp. 1–37, 2008.

[7] Q. He, J. Liu, C. Wang, and B. Li, "Coping with heterogeneous video contributors and viewers in crowdsourced live streaming: A cloud-based approach," *IEEE Trans. Multimedia*, vol. 18, no. 5, pp. 916–928, May 2016.

[8] R. Aparicio-Pardo, K. Pires, A. Blanc, and G. Simon, "Transcoding live adaptive video streams at a massive scale in the cloud," in *Proc. 6th ACM Multimedia Syst. Conf.*, 2015, pp. 49–60.

[9] Y. Chen, J. Sun, R. Zhang, and Y. Zhang, "Your song your way: Rhythm-based two-factor authentication for multi-touch mobile devices," in *Proc. IEEE Conf. Comput. Conf.*, Apr.-May 2015, pp. 2686–2694.

[10] L. Zhang, F. Wang, J. Liu, and X. Ma, "On mobile instant video clip sharing with screen scrolling," in *Proc. IEEE/ACM Int. Symp. Quality Service*, 2016, pp. 1–10.

[11] Y. Sun, I. Ahmad, D. Li, and Y.-Q. Zhang, "Region-based rate control and bit allocation for wireless video transmission," *IEEE Trans. Multimedia*, vol. 8, no. 1, pp. 1–10, Feb. 2006.

[12] Z. Li, S. Qin, and L. Itti, "Visual attention guided bit allocation in video compression," *Image Vis. Comput.*, vol. 29, no. 1, pp. 1–14, 2011.

[13] S. S. Mukherjee and N. M. Robertson, "Deep head pose: Gaze-direction estimation in multimodal video," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 2094–2107, Nov. 2015.

[14] O. A. Niamut *et al.*, "MPEG DASH SRD: Spatial relationship description," in *Proc. 7th Int. Conf. Multimedia Syst.*, 2016, pp. 5:1–5:8.

[15] J. Le Feuvre and C. Concolato, "Tiled-based adaptive streaming using MPEG-DASH," in *Proc. 7th Int. Conf. Multimedia Syst.*, 2016, pp. 5:1–5:8.

[16] A. Duchowski, *Eye Tracking Methodology: Theory and Practice*, vol. 373. New York, NY, USA: Springer, 2007.

[17] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "OPTICS: Ordering points to identify the clustering structure," *ACM SIGMOD Rec.*, vol. 28, no. 2, pp. 49–60, 1999.

[18] J. H. Bray and S. E. Maxwell, *Multivariate Analysis of Variance* (Sage Univ. Paper Ser. Qualitative Res. Methods), vol. 54. Beverly Hills, CA, USA: Sage, 1985.

[19] W. Hu and G. Cao, "Energy optimization through traffic aggregation in wireless networks," in *Proc. IEEE Conf. Comput. Commun.*, Apr.-May 2014, pp. 916–924.

[20] H. Kellerer, U. Pferschy, and D. Pisinger, *Introduction to NP-Completeness of Knapsack Problems*. New York, NY, USA: Springer, 2004.

[21] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand, "What do different evaluation metrics tell us about saliency models?" *CoRR*, 2016. [Online]. Available: http://arxiv.org/abs/1604.03605

Cong Zhang (S'14) received the M.Sc. degree in information engineering from Zhengzhou University, Zhengzhou, China, in 2012, and is currently working toward the Ph.D. degree in computing science at Simon Fraser University, Burnaby, BC, Canada. His research interests include multimedia communications, cloud computing, and crowdsourced live streaming.

Qiyun He received the B.Eng. degree from Zhejiang University, Zhejiang, China, and the BSc. degree from Simon Fraser University, Burnaby, BC, Canada, in 2015, and the M.Sc. degree from Simon Fraser University, in 2017, all in the field of computer science. His research interests include cloud computing, social media, multimedia systems and networks. He was the recipient of the 2017 Transactions on Multimedia Honorable Mention Award.

Jiangchuan Liu (S'01–M'03–SM'08–F'17) received the B.Eng. (*cum laude*) from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from The Hong Kong University of Science and Technology, Hong Kong, China, in 2003, both in computer science. He is currently a Full Professor (with University Professorship) with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada.

He is an NSERC E.W.R. Steacie Memorial Fellow. He is a Steering Committee Member of the IEEE TRANSACTIONS ON MOBILE COMPUTING, and an Associate Editor of the IEEE/ACM TRANSACTIONS ON NETWORKING, the IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, the IEEE TRANSACTIONS ON BIG DATA, and the IEEE TRANSACTIONS ON MULTIMEDIA. He is a co-recipient of the Test of Time Paper Award of IEEE INFOCOM (2015), the ACM TOMCCAP Nicolas D. Georganas Best Paper Award (2013), and the ACM Multimedia Best Paper Award (2012).

Zhi Wang (S'10–M'13) received the B.E. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 2008 and 2014, respectively. He is currently an Assistant Professor with the Graduate School at Shenzhen, Tsinghua University. His research interests include multimedia big data, edge computing, and large-scale multimedia systems. He was the recipient of the China Computer Federation Outstanding Doctoral Dissertation Award (2014), the ACM Multimedia Best Paper Award (2012), and the MMM Best Student Paper Award (2015).