Wireless Multicast for Zoomable Video Streaming

HUI WANG, MUN CHOON CHAN, and WEI TSANG OOI, National University of Singapore

Zoomable video streaming refers to a new class of interactive video applications, where users can zoom into a video stream to view a selected region of interest in higher resolutions and pan around to move the region of interest. The zoom and pan effects are typically achieved by breaking the source video into a grid of independently decodable tiles. Streaming the tiles to a set of heterogeneous users using broadcast is challenging, as users have different link rates and different regions of interest at different resolution levels. In this article, we consider the following problem: Given the subset of tiles that each user requested, the link rate of each user, and the available time slots, at which resolution should each tile be sent, to maximize the overall video quality received by all users. We design an efficient algorithm to solve this problem and evaluate the solution on a testbed using 10 mobile devices. Our method is able to achieve up to 12dB improvements over other heuristic methods.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; H.4.3 [**Information Systems Applications**]: Communications Applications; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems—*Video (e.g., tape, disk, DVI)*

General Terms: Algorithms, Design, Experimentation, Performance, Measurement

Additional Key Words and Phrases: Wireless multicast, zoomable video, optimal multicast algorithm, dynamic programming

ACM Reference Format:
Hui Wang, Mun Choon Chan, and Wei Tsang Ooi. 2015. Wireless multicast for zoomable video streaming. ACM Trans. Multimedia Comput. Commun. Appl. 12, 1, Article 5 (August 2015), 23 pages.
DOI: http://dx.doi.org/10.1145/2801123

1. INTRODUCTION

Video streaming is a major Internet service that has been widely used to carry both daily and major events (e.g., news, TV, sports, etc.). With the proliferation of mobile devices, streaming services continue to permeate into our daily lives further. Meanwhile, as technology evolves, videos with increasingly higher resolution are becoming available (e.g., 8K UHD supports 33 Megapixels). Due to screen size constraints (especially on mobile devices) and bandwidth constraints, however, video streaming playback is still limited in resolution. As a result, high-resolution videos are typically scaled down before transmission, leading to a loss in information.

This research was conducted under the NExT Search Center, supported by the Singapore National Research Foundation and the Interactive Digital Media R&D Program Office of Media Development Authority under research grant WBS:R-252-300-001-490. This research was supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.
Authors' addresses: H. Wang, M. C. Chan, and W. T. Ooi, School of Computing, National University of Singapore, Computing 1, 13 Computing Drive, Singapore 117417; email: {wanghui, ooiwt, chanmc}@comp.nus.edu.sg.

ACM 1551-6857/2015/08-ART5 $15.00
DOI: http://dx.doi.org/10.1145/2801123

To address the mismatch of video resolution between the capture device and playback, zoomable video streaming has recently been proposed [Mavlankar et al. 2007a, 2007b; van Brandenburg et al. 2011; Quang Minh Khiem et al. 2010; Shafiei et al. 2012]. A zoomable video supports zoom and pan as two new operations for a user to interact with the video. In particular, a user is able to zoom into a selected region of interest (RoI) in the video, to view the RoI with higher resolution. The user essentially views the video through a viewport that defines a rectangular region in the high-resolution video, from which the displayed video is cropped. While zooming in, users can pan around by moving the viewport to view different regions in the video.

In this article, we are concerned with wireless multicasting of zoomable video streams, which can arise in scenarios such as interactive TV or live events such as broadcasting lectures in campus [Mavlankar et al. 2010; Halawa et al. 2011; Shafiei et al. 2012], stage performances in concert, and sports in stadium (including eSports for spectating RTS games). Multicast is a natural operation for transmitting these contents, as existing studies have reported that users tend to zoom into a small clusters of regions in the video [Quang Minh Khiem et al. 2010] with substantial overlaps in their views.

In live zoomable video-streaming system [Shafiei et al. 2012], multiple resolution levels are available for each video stream. For a given screen pixel size, the desired resolution level of a user depends on the size of the selected region of interest (RoI). To stream efficiently, the video is broken into a grid of small, independently decodable regions, each is termed as a *tile* in this article. Instead of transmitting the whole frame, a minimum set of tiles covering the selected RoI with the desired resolution level is delivered.

The problem that we consider in this article is the following: Given the available time slots for video transmission and the selected RoI regions, how to determine, for each tile, at which resolution level should it be multicasted to maximize the overall utility of all users? There are two challenges in the aforementioned problem. First, the scheme has to deal with changes in both RoI and the wireless channel that affects the supported link rates. Second, the solution has to be computational efficient and scalable (with respect to number of users/sessions, video qualities, link rate, and time horizon).

In this work, we propose a novel and efficient algorithm to optimally solve this zoomable multicast problem. Our algorithm is inspired by several recent works [Li et al. 2009; Yoon et al. 2012] that look into the design of optimal algorithm for video multicast allocation with a focus on heterogeneous link rates. To evaluate our algorithm, we implemented the algorithm on a testbed that consists of the following key components: (i) mobile clients that support zoomable video functions, (ii) video server that supports streaming of zoomable video, and (iii) a proxy that collects client RoI requests and wireless link conditions, runs the resource allocation algorithm, and multicasts the videos obtained from the server to the clients.

The major contributions of our work are as follows.

(i) We model the zoomable video multicast problem as an optimization problem and develop an optimal algorithm that decides which resolution of which tile should be transmitted at which link rate. The proposed optimal multicast improves the average video quality by up to 12dB, 6dB, and 3dB in terms of PSNR compared with three baseline schemes, *adaptive unicast*, *adaptive multicast*, and *approximate multicast*, respectively.
(ii) If we consider each tile as an individual video session, our proposed algorithm can be applied to the optimal allocation of multi-sessions adaptive video streaming as well, and has a lower, more practical, running time (grows linearly with the number of time slots) than the existing optimal allocation algorithms [Li et al. 2009; Yoon et al. 2012].

(iii) We evaluate our solution on a wireless streaming testbed with up to 10 Android phones.

The rest of the article is structured as follows. Section 2 discusses the related work. In Section 3, we review the background of tiled zoomable streaming and mixed resolutions tiling scheme. Section 4 states our maximization problem. We present our optimal algorithm in Section 5. The system implementations are detailed in Section 6 and performance evaluation results of our algorithm on Android platform are presented in Section 7. The conclusion is made in Section 8.

2. RELATED WORK

A tremendous amount of previous work aims at improving video multicast streaming system by dynamically adapting video data rates and multicast link rates. In this section, we discuss only the most relevant work which can be broadly classified into three categories: adaptive video multicast, multicast link rate adaptation, and adaptive video with multicast link rate adaptation.

Adaptive Video Multicast. Many adaptive video multicast streaming approaches have been proposed to improve the performance of video streaming system [Chou and Miao 2006; Liu et al. 2003]. Liu et al. [2003] present an overview of existing studies and illustrate the advantages of adaptive streaming over nonadaptive streaming. The problem of rate-adaptive optimized streaming is reduced to the error-cost optimized transmission problem in the following study [Chou and Miao 2006]. This work also derives a fast practical algorithm to solve the formulated optimization problem. Most of these works focus on adapting the video rate (quality) with the fixed low multicast link rate, which may underutilize the networking resources.

Multicast Link Rate Adaptation. Recently, multicast link rate adaptation mechanisms have been suggested [Wong et al. 2013; Piamrat et al. 2009; Chandra et al. 2009; Sen et al. 2010]. Instead of using the basic rate, a relatively high broadcast rate is used for packet delivery, and FEC schemes are leveraged to protect the data from packet losses [Wong et al. 2013; Piamrat et al. 2009].

Among these work, the most relevant works are DirCast [Chandra et al. 2009] and Medusa [Sen et al. 2010]. DirCast multicasts packet at the link rate of the worst client for each access point (AP) and takes into account the rate anomaly problems. We adopt similar mechanisms. Medusa prioritizes the frames according to their importance and transmits the less important frames at higher link rates. By utilizing this frame level rate assignment heuristic, Medusa achieves higher video quality with limited resources.

Adaptive Video with Multicast Rate Adaptation. To further improve streaming performance, the last category of research jointly adapts video data rate and multicast link rate. This category is the most related to our work. Deb et al. [2008] investigate the utility optimization problem of scalable video multicast and prove that this problem is NP-Hard. A greedy algorithm is then proposed to schedule the transmissions of layers and determine the corresponding modulation and coding scheme (MCS) assigned for each transmission. Li et al. [2009] suggest a pseudo-polynomial algorithm with dynamic programming to solve the optimization problem. Most recently, MuVi [Yoon et al. 2012] has been designed to investigate the optimal multicast scheduling problem for videos encoded with I, P, and B frames. As the computational complexity of the suggested algorithm [Li et al. 2009] grows quadratically with the number of available time slots, it fails to efficiently solve the optimization problem with multiple multicast sessions. To reduce the computational complexity especially for the case of multiple sessions, a fully polynomial time approximation algorithm is presented [Li et al. 2010].

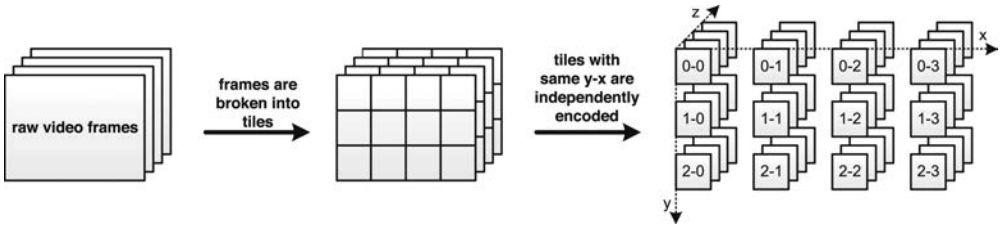


Fig. 1. Tile video.

The approximation factor, however, linearly decreases with the number of multicast sessions.

Our approach falls under the category of adaptive video with multicast link rate adaptation. In contrast to previous work, we focus on a scenario where each user is interested in a subset of video tiles and user interests may partially overlapped. Our algorithm can also be easily applied to the optimization problems with multiple sessions.

3. BACKGROUND OF MIXED-RESOLUTIONS TILED STREAMING

In this section, we review the background of *mixed resolutions tiling scheme*, which is proposed by Wang et al. [2014]. Moreover, to evaluate the perceptual quality of this scheme, the conducted psychophysical experiment is presented as well.

3.1. Mixing Tile Resolutions in Tiled Video

Zoomable video streaming is typically achieved using a technique called *tiled streaming*, where video frames are broken into a grid of tiles (Figure 1). We can view the video as a three-dimensional matrix of tiles. Tiles at the same y-x position in the matrix are temporally grouped and coded along z axis. The video is encoded into different resolutions to support zooming. The zoom-out view corresponds to the lowest resolution. As the user zooms in, a minimum set of tiles from the higher resolution video covering the RoI region is streamed. The location of RoI can be changed by panning, while the resolution can be changed by zooming.

The tiles in the same y-x are decoded together by the zoomable player at the client side. The tile groups with different y-x positions can be decoded in parallel, each frame is formed by the uncompressed tiles with same z position. The frame will be displayed in the original order by the zoomable player when all the corresponding tiles are uncompressed.

In existing works [Quang Minh Khiem et al. 2010, 2011; Feng et al. 2011], at the server side, an original video is normally encoded into different versions (streams): frames of a low-resolution stream are constructed from a smaller number of tiles; and frames of higher-resolution streams are constructed from a larger number of tiles. At the client side, the number of tiles required to cover the physical screen resolution is fixed, therefore, the bandwidth consumption for each user will be mostly constant. Initially, a low-resolution version of the video will be sent to users. When a user zooms into a RoI within the video, the server will first determine a suitable high-resolution stream based on the requested RoI size (zoom level). It then selects tiles covering the requested RoI from this stream. This mechanism allows users to see their regions of interest in detail without consuming more bandwidth.

The aforementioned RoI cropping technique performs well in small-scale networks by unicasting video stream. In one of the use cases we consider, the video stream is consumed by a large number of users within one location (e.g., in a concert hall or stadium). To overcome the scalability issues with such a large number of users

Table I. The Number of Pixels in Each Frame and Each Tile at Different Resolution Levels

Level	Frame	16 × 9 Tiles	80 × 45 Tiles
5	1920 × 1080	120 × 120	24 × 24
4	1600 × 900	100 × 100	20 × 20
3	1280 × 720	80 × 80	16 × 16
2	960 × 540	60 × 60	12 × 12
1	640 × 360	40 × 40	8 × 8

and RoI requests, wireless multicast scheme is employed. When the RoI regions from multiple users partially overlap, tiles from the overlapped regions could be potentially multicasted to all interested users to save bandwidth consumption. In zoomable video, different users, however, may have different zoom levels (i.e., different RoI sizes) and will need tiles from different versions encoded at different resolutions, which prevents the potential benefits of a wireless multicast.

Instead of fixing tile size, using a fixed number of tiles to encode and decode videos could be more effective. At the server side, an original video will be encoded into different resolution versions, but all versions consist of the same number of tiles. The same amount of tiles is required at the client side to decode each video frame. Within a frame, however, different tiles could come from different resolution streams. If a tile comes from a stream with a resolution lower/higher than the requested level, it will be scaled up/down accordingly. In zoomable video, when a user zooms into a region of interest (RoI) within the video, the server will first determine the tiles covering this RoI, and then associate each tile with an appropriate stream version, depending on their popularity and the resource constraints.

The proposed *mixed resolutions tiling* scheme has the following two essential advantages in tiled video streaming. First, benefiting from the scaling up/down operations for each tile, the multicast transmissions are considerably reduced. Next, by intelligently allocating a resolution version to each tile, the mixing resolutions approach may considerably reduce bandwidth consumption without impairing much perceived video quality. For instance, the popular regions/tiles requested by many users could come from high-resolution streams; while tiles requested by one or few users could come from a low-resolution stream under limited bandwidth condition.

Although this proposed scheme saves bandwidth, the impairment to the perceived quality is still unclear. Thus, to understand if, and at what thresholds, users could notice and/or accept the difference between original video and tiled video with mixed resolutions, we conducted a psychophysical study with 50 participants, which is presented in the remainder of this section.

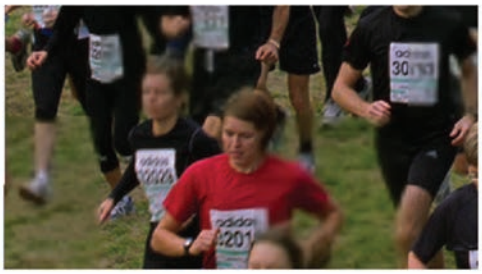
3.2. Perceptual Quality Assessment

Using the *method of limits* from psychophysics [Gescheider 1997], we measure two perceptual thresholds, Just Noticeable Difference (JND) and Just Unacceptable Difference (JUD), to understand the user perception about the quality of mixed-resolution tiled video. The two identified difference thresholds partition the quality degradation level (introduced by mixing tile resolutions) into the following three intervals: without noticeable quality degradation, with noticeable (but acceptable) quality degradation, and with unacceptable quality degradation.

3.2.1. Setup. The experiments assess the quality of mixed-resolution tiled video using three standard HD (1920 × 1080p) test video files, *Crowd-Run* (dense motion, 50fps),

(a) Tiles from HD stream

(b) Mixing tiles from stream levels 5 and 3



(c) Mixing tiles from stream levels 5 and 1

Fig. 2. Mixing tile resolutions of Crowd-Run.

Old-Town-Cross (medium motion, 50fps), and *Rush-Hour* (low motion, 25fps).[1] We have five resolution levels for each video file, these levels are labeled from 5 to 1 (Table I). The pixels of the original video frame at five resolution levels are: 1920 × 1080, 1600 × 900, 1280 × 720, 960 × 540, and 640 × 360.

In the experiments, we construct mixed-resolution tiled video by mixing two resolution levels, where the higher-resolution level is denoted as R_H and the lower-resolution level is denoted as R_L. Specifically, given a pair of R_H and R_L, we randomly allocate resolution level R_H or R_L to each tile with equal probability. For any particular pair of R_H and R_L, we restrict the range of R_H as $3 \leq R_H \leq 5$ and the range of R_L as $1 \leq R_L \leq R_H$. Figures 2, 3, and 4 show the screenshots of mixed-resolution tiled video.

Since the aspect ratio of the test HD video frame sequences is 16:9, we break the video frames into 16 × 9 tiles by default. As a result, each tile size (view region size) is $\frac{1}{16\times 9}$ of the entire view region. To evaluate the impact of tile size, in addition to the default configuration, we generate another set of videos where each video frame is broken into 80 × 45 tiles. The number of pixels for a tile at each resolution level is shown in Table I.

3.2.2. Procedures. Fifty adult participants were invited to participate in our assessment, primarily graduate students and research staffs from National University of Singapore. The sample consisted of 16 women and 34 men; all had normal vision. They were asked to watch the mixed-resolution tiled videos online[2] using a monitor with full HD display resolution.

For configurations with 16 × 9 tiles, we vary the high resolution level R_H from 5 to 3, nine stimuli series are generated over three test videos. For configurations with 80 × 45 tiles, we generate stimuli series with $R_H = 5$. As a result, we have 12 stimuli series in total, which are shuffled in a random order and played.

[1]http://media.xiph.org/video/derf/.
[2]http://liubei.ddns.comp.nus.edu.sg/resMix.

(a) Tiles from HD stream

(b) Mixing tiles from stream levels 5 and 3

(c) Mixing tiles from stream levels 5 and 1

Fig. 3. Mixing tile resolutions of Old-Town-Cross.

(a) Tiles from HD stream

(b) Mixing tiles from stream levels 5 and 3

(c) Mixing tiles from stream levels 5 and 1

Fig. 4. Mixing tile resolutions of Rush-Hour.

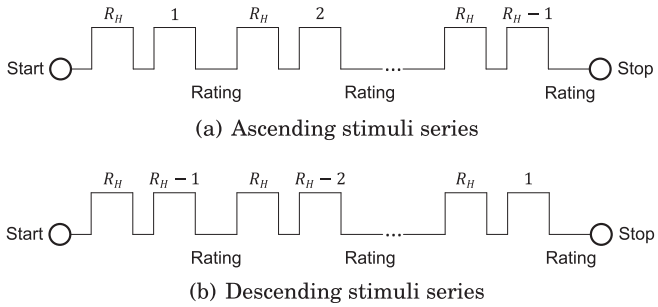


Fig. 5. Experiment procedure. The video is composed by tiles with resolution level R_H and R_L. The numbers in this figure represent the value of R_L, the first video in each pair is a standard tiled video where $R_L = R_H$ and the second video is a mixed-resolution tiled video.

Table II. The average Just Noticeable Difference Threshold (Number within Parenthesis is the 95% Confidence Interval Value)

R_H	Crowd-Run	Old-Town-Cross	Rush-Hour
5	3.68 ($\pm$0.52)	3.25 ($\pm$0.47)	0.81 ($\pm$0.23)
4	2.74 ($\pm$0.39)	2.31 ($\pm$0.34)	0.24 ($\pm$0.10)
3	2.09 ($\pm$0.30)	1.73 ($\pm$0.26)	0.11 ($\pm$0.06)

For each series, the stimuli is randomly manipulated in either an ascending or a descending order, the procedures are depicted in Figure 5. In a stimuli series, we fix the high-resolution level R_H and vary the low-resolution level R_L. As shown in the figures, each pair presents a standard video where $R_L = R_H$ and a mixed-resolution tiled video. After watching the videos in a pair (10s per video), the participant is asked to rate the level of the difference between two videos. In particular, two questions are asked: (i) Is the quality difference noticeable; and (ii) Is the quality difference unacceptable. In the case of ascending series, we increase R_L from 1. On each successive trial, we increase R_L by 1 until the participant eventually reports the difference is unnoticeable or $R_L = R_H - 1$. If the series is descending, the stimuli operates in an opposite direction. We start from $R_L = R_H - 1$ and gradually decrease R_L until the participant reports the difference is unacceptable or $R_L = 1$.

Using this procedure, the obtained results fall into the following three categories: (i) The noticeable difference threshold and unacceptable difference threshold are both detected; (ii) Only the noticeable difference threshold is detected; and (iii) Neither the noticeable difference threshold nor the unacceptable difference threshold can be detected. Assuming that we have detected the noticeable difference threshold and the unacceptable threshold, denoted by T_{ND} and T_{UD}, respectively, then according to the method of limits [Gescheider 1997], we estimate the Just Noticeable Difference threshold as $(T_{ND} + (T_{ND} + 1))/2 = T_{ND} + 0.5$. Similarly, we express Just Unacceptable Difference threshold as $(T_{UD} + (T_{UD} + 1))/2 = T_{UD} + 0.5$. For the cases where we failed to detect the difference threshold, we set the corresponding Just Noticeable/Unacceptable Difference threshold to 0.

3.2.3. Results. We first examine the configuration with 16 $\times$ 9 tiles. Figure 6 depicts the CDF distribution of participants that cannot notice any difference between mixed-resolution tiled video (5, R_L) and standard tiled HD video (5, 5). The CDF distribution of participants that accept the quality difference is present in Figure 7. The average measured thresholds of Just Noticeable Difference and Just Unacceptable Difference for R_H in the range from 5 to 3 are shown in Table II and Table III, respectively.

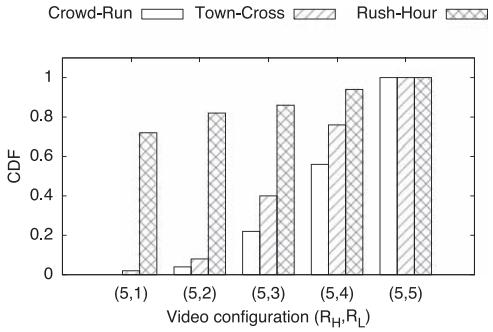


Fig. 6. CDF distribution of participants that cannot notice any difference between mixed-resolution tiled video (5, R_L) and standard HD tiled video (5, 5).

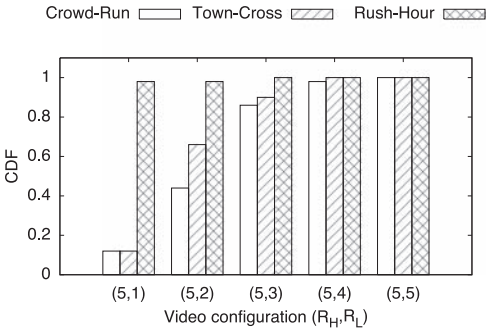


Fig. 7. CDF distribution of participants that accept the quality difference between mixed-resolution tiled video (5, R_L) and standard HD tiled video (5, 5).

Table III. The Average Just Unacceptable Difference Threshold (Number within Parenthesis is the 95% Confidence Interval Value)

R_H	Crowd-Run	Old-Town-Cross	Rush-Hour
5	2.03 ($\pm$0.31)	1.76 ($\pm$0.27)	0 (0)
4	1.64 ($\pm$0.26)	1.28 ($\pm$0.21)	0 (0)
3	1.28 ($\pm$0.21)	0.69 ($\pm$0.14)	0 (0)

From the results, we can draw the following observations.

Feasibility of Mixing Tile Resolutions. The measured thresholds confirm the feasibility of mixed-resolution tiled video. The CDF distribution from Figure 6 implies that we can mix tiles with resolution levels 5 and 4 without being noticed in most cases. Further, the depicted result from Figure 7 indicates that more than 85% participants accept the quality difference with configurations where $3 \leq R_L \leq R_H = 5$; under these configurations, up to 30% bandwidth can be saved by mixing tile resolutions. When we construct video from tiles at resolution level 5 and 2, almost all participants noticed the difference for video *Crowd-Run* and *Old-Town-Cross*. 40% to 65% of the participants, however, still accept the quality difference.

Impact of Content. With the same configuration, the results from Tables II and III show a great disparity in the measured average JND and JUD across three test videos. Overall, video *Crowd-Run*, which has the highest amount of motion among the three test videos, is most sensitive to the resolution mixing, as the highest average threshold and the greatest variation are detected. Interestingly, video *Rush-Hour*, which has the lowest amount of motion among the three test videos, performs remarkably different from others. It is difficult to notice the quality difference between the mixed-resolution tiled video and the standard version, thus the average measured thresholds and the variations are much smaller compared with other test videos.

Gap between JND and JUD Thresholds. For many cases, although participants could notice the difference, it is still acceptable. Generally, a greater gap value indicates a higher video quality tolerance degree when the quality difference is noticeable. From the Tables II and III, we observe a significant gap between the average measured JND and JUD thresholds, especially for $R_H = 5$. In particular, the average gap quantities for video *Crowd-Run* and *Old-Town-Cross* with $R_H = 5$ are 1.65 and 1.49, respectively. As the tolerance space is reduced with smaller R_H value, the quantity of the threshold gap between JND and JUD will be reduced as well, as can be seen in both tables.

Table IV. The Average Just Noticeable Difference Threshold Where $R_H = 5$ (Number within Parenthesis is the 95% Confidence Interval Value)

	Crowd-Run	Old-Town-Cross	Rush-Hour
16×9	3.68 ($\pm$0.52)	3.25 ($\pm$0.47)	0.81 ($\pm$0.23)
80×45	3.30 ($\pm$0.48)	3.04 ($\pm$0.44)	0.76 ($\pm$0.20)

Table V. The Average Just Unacceptable Difference Threshold Where $R_H = 5$ (Number within Parenthesis is the 95% Confidence Interval Value)

	Crowd-Run	Old-Town-Cross	Rush-Hour
16×9	2.03 ($\pm$0.31)	1.76 ($\pm$0.27)	0 (0)
80×45	1.76 ($\pm$0.29)	1.63 ($\pm$0.25)	0 (0)

Impact of Tile Size. The comparison between the configurations with 16×9 tiles and 80×45 tiles is present in Tables IV and V. The threshold values with 80×45 tiles is slightly smaller than the corresponding threshold values with 16×9 tiles, which indicates that the quality degradation introduced by mixing resolutions is slightly less obvious for the finer-grained tile size (80×45) compared with the coarse-grained tile size (16×9). The finer-grained tiles, however, are generally less efficient in terms of encoding and transmission bandwidth. Therefore, we need to balance the trade-off between the video quality and the efficiency to obtain an appropriate configuration.

3.3. Summary

The subjective assessment demonstrated that in most cases, the perceptual quality loss of mixing resolutions in tiled video is insignificant, as long as the variance of mixed resolution levels is low. From the evaluation results, we can draw the following two important observations:

—in most cases, tiles from the 1920×1080p stream and the 1600×900p stream could be mixed together without being noticed;
—even when participants could notice quality degradation in videos combined with tiles from the 1920×1080p stream and tiles from the 960×540p stream, greater than 80% of participants still accept the quality difference for low and medium-motion videos; and more than 40% of participants accept the quality difference for the dense motion video.

This section confirms the feasibility of the *mixed resolution tiling* scheme, which will be applied to wireless multicast of tiled video streams in the rest of this article. Instead of randomly mixing resolutions of tiles, we are looking into how to optimally allocate resolution versions to each tile to better utilize the wireless bandwidth and improve overall utilities of users.

4. PROBLEM DEFINITION

We now describe an optimization problem to determine which tile should be sent at which resolution and at which link rate, given the wireless network constraint. Let T be the number of slots available on average for the delivery of a single frame, where a slot refers to a minimum transmission time unit in 802.11 network (e.g., $9\mu s$ in 802.11a). The wireless network supports N_r different link rates. Let n be the number of users in our system; the physical link rates of these n clients are: $r_1, r_2, \ldots, r_n$. Without loss of generality, we assume that link rate r_i is a nondecreasing function of index i.

We generate M resolution versions (or levels) for each frame, and every frame is broken into N_g view regions, each view region is termed as a tile (or grid). Instead of

using the y-x notation in Figure 1, we simply number the tiles $1, 2, \ldots, N_g$ when we discuss the algorithm. A tile is considered as a logical entity – when transmitted, a tile has to have a specific resolution level. A tile g with resolution level m ($1 \leq m \leq M$) is denoted by g_m, the size of which is s_g^m. The sequence of $s_g^1, s_g^2, \ldots, s_g^M$ is strictly increasing.

The set of tiles in the RoI of user i is denoted as $\mathbf{G}(i)$. Let R_i be the request resolution level from user i. With restricted bandwidth condition, we may not be able to satisfy all the user requests. As a result, some tiles may be streamed with resolution levels lower than the desired resolution level (R_i). To avoid significant perceptual quality loss introduced by downgrading tile resolution levels, for user i, we have a lower bound L_i of the tile resolution levels, which is guaranteed to be satisfied. More specifically, for every tile in $\mathbf{G}(i)$, the resolution level to be decoded (the highest received level) by user i should be at least L_i.

Receiving g_m at user i yields utility $u_{g,i}^m$, which follows these rules.

—if $g \notin \mathbf{G}(i)$, then $u_{g,i}^m = 0$ (for all $1 \leq m \leq M$);
—if $g \in \mathbf{G}(i)$ and $m < L_i$, we have $u_{g,i}^m = -\infty$;
—if $g \in \mathbf{G}(i)$ and $L_i \leq m < m' \leq R_i$, we have $u_{g,i}^m < u_{g,i}^{m'}$;
—if $g \in \mathbf{G}(i)$ and $m > R_i$, we have $u_{g,i}^m = u_{g,i}^{R_i}$.

For simplicity, we use a tile-size-based utility assignment mechanism. In particular, $u_{g,i}^{R_i}$ is the maximum achievable utility at user i by receiving tile g, the utility assignments of receiving other levels are proportional to the corresponding tile sizes. The utility function, however, can be any general function (e.g., the PSNR of tiles) subject to these rules.

Given the RoI selection and the corresponding utility assignment of tiles with each resolution level at each user, the objective is to maximize the total utility received by all users subject to the total transmission slot constraint.

Last, we discuss the parameter settings for the average available time slots T and the tile size with a specific resolution level. All pixels belonging to the same tile across different frames will be encoded as a group of picture (GOP). Due to the dependency in a GOP, if we pick a resolution level m for a tile, we have to transmit this tile at the same resolution m for all frames within the same GOP. In our model, we therefore model s_g^m as the average tile size in a GOP and model the average number of time slots needed per frame as T. In the implementation, however, the time slots allocated to frames in a GOP is proportionally distributed according to the actual frame sizes, as there is a considerable diversity in the sizes of I, B, and P frames.

The key notations used in this article are summarized in Table VI.

5. OPTIMAL BROADCAST ALGORITHM

This section presents a dynamic programming algorithm to solve the utility maximization problem defined in the previous section. The solution consists of three major components: (i) an algorithm that determines an appropriate quality lower bound for each user; (ii) an optimal algorithm for determining the link rate and resolution level of a single tile; and (iii) an efficient algorithm for determining the link rate and resolution level over multiple tiles.

5.1. Adaptive Utility Assignment

The mixture of resolution levels could result in two potential issueswhen the available bandwidth is insufficient to meet the requirements from all users. First, as discussed in Section 3, the significant disparity of resolution levels between tiles for a user may severely impair the visual perception. Next, the utility-oriented optimization algorithm

Table VI. Key Notations in the Algorithm

Notation	Definition
T	Average available time slots (802.11 slots) for delivering one frame
n	Number of users (or clients)
r_i	The link rate of user i
N_r	Number of different link rate levels
M	The number of available resolution levels
$\mathbf{G}(i)$	The set of tiles in the RoI of user i
R_i	Resolution level requested by user i ($R_i \geq 1$)
L_i	Resolution level guaranteed to be satisfied for user i ($L_i \geq 1$)
N_g	Total number of tiles (or grids)
g_m	Tile g at resolution level m
s_g^m	The size (in bytes) of tile g at resolution level m
$u_{g,i}^m$	The utility of tile g at resolution level m assigned to user i
$\mathcal{M}(g,i)$	The highest received resolution level of tile g at user i

could result in severe unfairness. To address these issues, we suggest an algorithm to tune adaptively the lower bound $L_i (1 \leq i \leq n)$ of resolution level that is guaranteed to be satisfied.

Recall that R_i is the requested resolution level from user i, and L_i is the resolution level guaranteed to be satisfied for user i among its interested tiles. Given R_i and L_i, the rules for utility assignment are specified in Section 4. It is clear that when all requests of users are satisfied, we have $L_i \geq R_i$ for any $1 \leq i \leq n$, and the overall utility is optimal. Hence, we set $L_i = R_i$ at the beginning, then we validate the feasibility of current configuration for L_i and adapt accordingly.

We define an indicator variable $x_{g,i}^m$, which takes the value of 1 if resolution level m of tile g is transmitted at link rate r_i, and 0 otherwise. Let $\mathcal{M}(g,i)$ be the maximum resolution level of tile g to be received by user i. Since user i can only receive the transmissions with link rates not higher than r_i, the expression of $\mathcal{M}(g,i)$ can be written as $\mathcal{M}(g,i) = \max\{m | x_{g,i'}^m = 1 \text{ and } 1 \leq i' \leq i\}$. Now we can formulate the feasibility validation problem as

$$\sum_{g=1}^{N_g} \sum_{i=1}^{n} u_{g,i}^{\mathcal{M}(g,i)} \geq 0, \tag{1}$$

$$\text{subject to } \sum_{g=1}^{N_g} \sum_{m=1}^{M} \sum_{i=1}^{n} \left(x_{g,i}^m \cdot \left\lceil \frac{s_g^m}{r_i} \right\rceil \right) \leq T, \tag{2}$$

where $u_{g,i}^{\mathcal{M}(g,i)} = -\infty$ if $g \in \mathbf{G}(i)$ and $\mathcal{M}(g,i) < L_i$; the unit of expression s_g^m / r_i is a 802.11 time slot. To obtain an appropriate setting of L_i, we keep decreasing L_i by 1 for all i until Inequality (1) is feasible subject to time limit constraint (2).

To solve this feasibility problem, we first independently calculate the minimum required time slots for every tile g ($1 \leq g \leq N_g$) and then simply integrate the required time slots across all N_g tiles. The total required slots should be less or equal than T, if the current lower bound requirement (L_i) is achievable. The following paragraph presents an algorithm to calculate the minimum required time slot for any single tile g.

For user i, the lower-bound requirement of resolution level L_i can be satisfied by either transmitting at link rate r_i or at lower link rate $r_{i'}$, where $1 \leq i' \leq i$. Define

$\Re_g(i,l)$ as the minimum required time slots satisfying nonnegative utility requirement with users up to i and with resolution level l has not been satisfied from users with indexes larger than i. The recursive equation for $\Re_g(i,l)$ can be written as

$$\Re_g(i,l) = \begin{cases} \min\left\{\Re_g(i-1,H), \Re_g(i-1,0) + \left\lceil \frac{s_g^H}{r_i} \right\rceil\right\}, & \text{if } g \in \mathbf{G}(i); \\ \Re_g(i-1,l), & \text{if } g \notin \mathbf{G}(i), \end{cases} \tag{3}$$

where $H = \max\{l, L_i\}$. The minimum time slots required for delivering tile g while satisfying the quality lower bound is $\Re_g(n,0)$, which could be easily calculated by leveraging recursion (3). Now we are able to simplify the feasibility validation problem to $\sum_{g=1}^{N_g} \Re_g(n,0) \le T$.

5.2. Optimal Allocation for a Single Tile

For ease of analysis, we begin with designing an optimal resource allocation algorithm for a single tile. We denote this particular tile as g. The optimal allocation approach determines the resolution levels of tile g to be transmitted and the link rate for each transmission.

5.2.1. Optimal Allocation Algorithm. Let t $(0 \le t \le T)$ be the total slots available for the transmissions of tile g. The utility optimization problem can be formulated as

$$\begin{aligned} &\text{maximize } \sum_{i=1}^{n} u_{g,i}^{\mathcal{M}(g,i)}, \\ &\text{subject to } \sum_{m=1}^{M}\sum_{i=1}^{n}\left(x_{g,i}^{m} \cdot \left\lceil \frac{s_g^m}{r_i} \right\rceil\right) \le t. \end{aligned} \tag{4}$$

As we assume that the users with higher link rate can receive all transmissions at lower rates, we have the following important observation: for any tile, a higher resolution version is always transmitted with higher link rate. By utilizing this observation, we have the following definition of the maximum utility function. For tile g, define $\mathcal{U}_g(i,m,t)$ as the optimal utility with users $u_1, u_2, \ldots, u_i$, with resolution levels up to m, and within transmission time limit t.

Every state $\mathcal{U}_g(i,m,t)$ falls into one of these categories: either user i is not interested in tile g or user i is interested in tile g. If user i is not interested in tile g $(g \notin \mathbf{G}(i))$, the state transition equation could be simply written as

$$\mathcal{U}_g(i,m,t) = \mathcal{U}_g(i-1,m,t). \tag{5}$$

It is slightly more complicated to analyze the transitions of state $\mathcal{U}_g(i,m,t)$ when user i is interested in tile g. There are two transition possibilities for this state.

(i) If the resolution level m of g is not transmitted, the recursive function is

$$\mathcal{U}_g(i,m,t) = \mathcal{U}_g(i,m-1,t). \tag{6}$$

(ii) If the resolution level m is transmitted at link rate level $r_{i'}$ $(i' \le i)$, the recursive function is

$$\mathcal{U}_g(i,m,t) = \max_{1 \le i' \le i}\left\{\mathcal{U}_g\left(i'-1, m-1, t - \left\lceil \frac{s_g^m}{r_{i'}} \right\rceil\right) + \sum_{c=i'}^{i} u_{g,c}^{m}\right\}. \tag{7}$$

The terminating conditions for the recursion and the corresponding value assignments are

$$\begin{aligned} \mathcal{U}_g(i, m, t) &= -\infty, && \text{if } t < 0 \text{ or } m < 0; \\ \mathcal{U}_g(0, m, t) &= 0, && \text{if } t \geq 0 \text{ and } m \geq 0. \end{aligned}$$

We start the recursion from state $\mathcal{U}_g(n, M, t)$ with the given available time slots t, the highest resolution level M, and user n with the highest link rate. The recursion can be solved by applying Eqs. (5), (6), and (7). The transition complexity for Eqs. (5) and (6) are both $\mathcal{O}(1)$. Eq. (7) enumerates the user link rate for every transmission to attain the optimal transition. As a result, the transition complexity for Eq. (7) is $\mathcal{O}(n)$. The overall computational complexity of our optimization algorithm is $\mathcal{O}(n^2tM)$, which grows quadratically with n.

5.2.2. Virtual Clustering. This section applies a clustering method to make our optimal algorithm scalable with n (number of users). Since Eq. (7) is the most time-consuming operation, we will concentrate on analyzing this equation.

Assuming that a specific link rate $r_{i'}$ is used for transmitting resolution level m in Eq. (7), all clients with no smaller than link rate $r_{i'}$ are able to receive this resolution level. Instead of enumerating user i', only the distinct link rates are required to be considered. As a consequence, we could cluster the users with identical link rate to a virtual user in the algorithm. The clustering process is achieved by simply integrating the corresponding utility values. Specifically, the utility of tile g at resolution level m for a virtual user at link rate r could be defined as $\sum_{i=1}^{n} u_{g,i}^m$, where $r_i = r$.

By clustering, the number of users n is reduced to at most N_r, which is the maximum number of distinct link rates. As the number of link rate levels is noticeably small (8 in 802.11a [802.11-2007]), with user clustering, our algorithm scales with any number of users without considering the frame losses and retransmissions.

5.3. Optimal Allocation for Multiple Tiles

This section presents an algorithm that is able to achieve the maximum utility by optimally allocating resources over all N_g tiles. First, we extend the algorithm in Section 5.2 to incorporate multiple tiles. Next, we analyze the computational complexity of the algorithm and demonstrate its inefficiency. Finally, we reduce the computational overhead of the algorithm to make it more efficient and practical for deployment.

Given time limit $t(g)$ for tile g, the optimal utility is $\mathcal{U}_g(n, M, t(g))$, which is calculated in Section 5.2. The overall system utility is the integrated utility over all N_g tiles, the optimization problem can be represented as

$$\begin{aligned} &\text{maximize} \sum_{g=1}^{N_g} \mathcal{U}_g(n, M, t(g)), \\ &\text{subject to} \sum_{g=1}^{N_g} t(g) \leq T. \end{aligned} \tag{8}$$

From the formulas, we observe that optimization problem (8) is to optimally distribute the total time slots T to all tiles.

Define function $\mathcal{U}(g, t)$ as the maximum utility achieved with tiles from 1 to g within time limit t. Enumerating the allocated time slots t' for transmissions of tile g yields

$$\mathcal{U}(g, t) = \max_{0 \leq t' \leq t} \left\{ \mathcal{U}(g-1, t-t') + \mathcal{U}_g(n, M, t') \right\}. \tag{9}$$

The maximum system utility is $\mathcal{U}(N_g, T)$. This equation is employed by Li et al. [2009, 2010] as well to incorporate the allocation of multiple multicast sessions into their optimal algorithm.

We now discuss the complexity of this multiple tiles allocation algorithm. We precomputed all $\mathcal{U}_g(n, M, t)$, where $1 \le g \le N_g$ and $0 \le t \le T$, the complexity is $\mathcal{O}(n^2TMN_g)$. As shown in Eq. (9), the transition complexity for each state is $\mathcal{O}(T)$, the complexity of the recursion procedure to calculate $\mathcal{U}(N_g, T)$ is $\mathcal{O}(T^2N_g)$. Combining the precomputing and the recursion complexity gives $\mathcal{O}(n^2TMN_g + T^2N_g)$ in total.

The parameters of n (reduced to N_r), M, and N_g are constants for a given video, so the computational cost depends on T. Assuming that the video frame rate is 25fps, the slots available on average for a single frame is 40 ms $\approx$ 4444 slots (9 μs per slot in 802.11a). When this value of T is substituted into $\mathcal{O}(n^2TMN_g + T^2N_g)$, the overhead is clearly too large to be practical. Therefore, it is essential to further reduce the computational complexity.

The key idea of reducing computational overhead is to trade space for algorithm running time. Define the optimal utility function $\mathcal{U}^*(g, i, m, t)$ as

$$\mathcal{U}^*(g, i, m, t) = \max_{0 \le t' \le t} \left\{ \mathcal{U}(g-1, t-t') + \mathcal{U}_g(i, m, t') \right\}. \tag{10}$$

Same as the analysis for the allocation algorithm of a single tile, the category that each state $\mathcal{U}^*(g, i, m, t)$ falls into depends on whether user i is interested in tile g.

If user i is not interested in tile g, substituting Eq. (5) into Eq. (10) yields

$$\begin{aligned} \mathcal{U}^*(g, i, m, t) &= \max_{0 \le t' \le t} \left\{ \mathcal{U}(g-1, t-t') + \mathcal{U}_g(i-1, m, t') \right\} \\ &= \mathcal{U}^*(g, i-1, m, t). \end{aligned} \tag{11}$$

On the other hand, if user i is interested in tile g, by substituting Eqs. (6) and (7) into Eq. (10), we attain Eq. (12).

$$\begin{aligned} &\mathcal{U}^*(g, i, m, t) \\ &= \max_{0 \le t' \le t} \left\{ \mathcal{U}(g-1, t-t') + \max \left[\mathcal{U}_g(i, m-1, t'), \max_{1 \le i' \le i} \left(\mathcal{U}_g\left(i'-1, m-1, t' - \left\lceil \frac{s_g^m}{r_{i'}} \right\rceil \right) + \sum_{c=i'}^{i} u_{g,c}^m \right) \right] \right\} \\ &= \max \left\{ \mathcal{U}^*(g, i, m-1, t), \max_{1 \le i' \le i} \left[\mathcal{U}^*\left(g, i'-1, m-1, t - \left\lceil \frac{s_g^m}{r_{i'}} \right\rceil \right) + \sum_{c=i'}^{i} u_{g,c}^m \right] \right\}. \end{aligned} \tag{12}$$

The initial conditions and recursive transitions at boundaries for $\mathcal{U}^*(g, i, m, t)$ are

$$\begin{aligned} \mathcal{U}^*(g, i, m, t) &= -\infty, && \text{if } t < 0 \text{ or } m < 0; \\ \mathcal{U}^*(g, 0, m, t) &= \mathcal{U}^*(g-1, n, M, t), && \text{if } g \ge 1, t \ge 0, m \ge 0; \\ \mathcal{U}^*(0, i, m, t) &= 0, && \text{if } t \ge 0, m \ge 0. \end{aligned}$$

The recursive Eqs. (11) and (12) clearly illustrate the procedure to solve the optimal multiple tiles allocation problem. The maximum utility is $\mathcal{U}^*(N_g, n, M, T)$.

The transition Eq. (11) consumes $\mathcal{O}(1)$ complexity. Eq. (12) enumerates user id i' instead of time slots, thus the transition complexity is $\mathcal{O}(n)$. Taking all transitions into consideration, we have a total computational complexity of $\mathcal{O}(n^2TMN_g)$. Here, n can be replaced by N_r by clustering users according to the available link rate levels. Compared with previous multiple-tiles allocation algorithm, the computational complexity

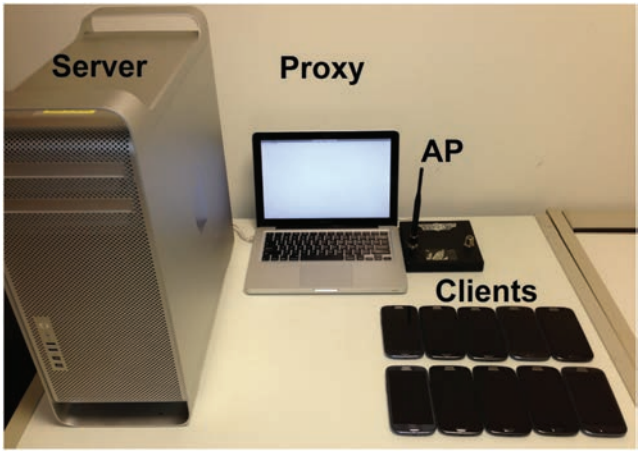


Fig. 8. System setup.

of current algorithm is significantly reduced by a factor of T. In the evaluation section, we will demonstrate the effectiveness of our optimal algorithm.

6. EXPERIMENTAL SETUP

To evaluate our algorithm, we set up the following experimental system.

6.1. System Setup

Our system uses a zoomable video streaming server that runs on a Mac Pro with a 3.2GHz Quad-Core processor and 8GB memory. The proxy runs on a MacBook with a 2.9GHz dual-core processor and 8GB memory. The video server, proxy, and WiFi AP used for multicast are all connected through wired Ethernet. The mobile devices, all Samsung Galaxy SIII, communicate with the AP using IEEE 802.11a operating at 5GHz.

The AP used supports two Complex IEEE 802.11abg adapters featuring the Atheros AR5414 chipset and runs OpenWRT Kamikaze 7.09 with kernel version 2.6.25.16. The driver of the wireless adapter used is MadWifi (version 0.9.4). To enable packet level rate assignment, we use the Click modular router [Kohler et al. 2000] (version 1.6.0). For each video packet transmission, we extract the rate value that is specified by the proxy in the header of every video packet, then passes the assigned rate value to the MadWifi driver. The setup is shown in Figure 8.

6.2. Rate Adaptation

As the WiFi SNR values on the mobile devices are not available, we use frame loss as a basis for rate adaptation [Bicket 2005; Wong et al. 2006; Pefkianakis et al. 2013]. In particular, we implement History-Aware Robust Rate Adaptation Algorithm (HA-RRAA) [Pefkianakis et al. 2013] that extends the work of RRAA [Wong et al. 2006].

RRAA uses two parameters, *Maximum Tolerable loss* (MTL) and *Opportunistic Rate Increase* (ORI), for rate adaptation. The corresponding threshold for these parameters are denoted by P_{MTL} and P_{ORI}, where $P_{ORI} < P_{MTL}$. RRAA measures the frame loss rate P over a period of *Estimation Window* and adapts the link rate as follows. The rate decreases to next lower one if P is greater than P_{MTL}. If P is smaller than P_{ORI}, the rate is increased to next higher one. When P is between P_{MRL} and P_{ORI}, the current rate is retained.

To limit transmissions at the adjacent high-loss rates, HA-RRAA is suggested [Pefkianakis et al. 2013]. HA-RRAA exponentially increases the window size of next lower rate upon transmission failure of current rate ($P > P_{MTL}$) and reset the window size when transmissions of current rate are successful ($P < P_{MTL}$). To be responsive to fast channel deterioration as RRAA, the algorithm additionally computes the loss over a *small window*. When the loss rate over this small window is greater than P_{MTL}, the current rate is directly moved to the next lower rate.

From our experiments, we observe that the HA-RRAA tuning mechanism may still result in the oscillation between two adjacent rates. We slightly modify the algorithm so that the window size is halved instead of being reset when transmissions of the current rate is successful. Furthermore, since we may broadcast packets at different rates under heterogeneous links, a client may receive packets sent at a rate higher than its current rate – these packets serve as "free" probes that prevent a client from increasing its rate unnecessarily. As a result, our rate adaptation is stable and responsive.

For tractability, packet losses and frame retransmissions are not incorporated into our algorithm. Therefore, conservative threshold parameters are used in our work. In particular, we set $P_{MTL} = 10\%$ and $P_{ORI} = 3\%$. The minimum *Estimation Window* size equals the interval between two consecutive allocation algorithm runs, this interval is also used as the *small window* to maintain responsiveness.

6.3. Video Coding and Streaming

In the evaluation, we do not need to play the video on the mobile devices and hence do not send actual video data. Instead, the following is done.

As depicted in Figure 1, each raw video frame from the test video is broken into N_g tiles, and the tiles with same y-x are encoded using FFmpeg tool (version 1.2.1) with H264 codec at the server. During our experiments, instead of transmitting the corresponding tiles from the test video, the server simply transmits the same number of arbitrary bits as the actual video tile. The metadata containing the tile size, y-x position, resolution level, and the frame ID for identification, is embedded. A client running on the mobile device extracts these fields from each received tile and periodically provide the reception bitmap to the server. When the transmission is over, we gather the reception bitmaps from all the clients, and reconstruct the mixed-resolutions video frames with decoded tiles at the server side. Here, the lost tiles (indicated by bitmaps) in a group of pictures (GOP) are concealed by the default method in FFmpeg.

7. EVALUATION

In this section, we present the evaluation results of our proposed optimal multicast algorithm through extensive experiments using up to 10 mobile devices.

Compared Algorithms. We compare performance of optimal multicast against the following baseline schemes. These schemes use HA-RRAA link adaptation as well.

Adaptive Unicast (aUnicast). This scheme transmits packets using wireless unicast only. To ensure the lowest quality (resolution level 1) is received by every user, the algorithm calculates the number of time slots required to transmit every tile at resolution level 1. The algorithm then loops through each user, and if there is sufficient available time slot remaining, the resolution of the tiles transmitted to the user is replaced by the desired resolution level. The loop terminates when the requests from all users are satisfied or the remaining time slots are insufficient for any user.

Adaptive Multicast (aMulticast). Similar to aUnicast, the lowest resolution level 1 is guaranteed for each user and the remaining available slots are utilized to upgrade the resolution level tile by tile. As in DirCast [Chandra et al. 2009], the assigned link rate for a particular tile is the lowest supported link rate among all interested users. As multicast is used, at most one multicast transmission is required for any tile.

Table VII. The Data Rate of Different Resolution Levels

Level	Resolution Size	# Tiles	Low Rate[a]	Medium Rate[b]	High Rate[c]
5	1920×1080	16×9	6.2	10.9	20.2
4	1600×900	16×9	4.5	6.6	11.1
3	1280×720	16×9	3.2	4.6	8.4
2	960×540	16×9	2.2	2.9	5.0
1	640×360	16×9	1.2	1.5	2.5

[a]*Rush-hour*, compressed using FFmpeg with parameter $qp = 25$.
[b]*Controlled-burn*, compressed with $qp = 25$.
[c]*Controlled-burn*, compressed with $qp = 22$.

Table VIII. The Achieved Link Rates of Mobile Users

# Users	Min Rate	Max Rate	Average Rate
1	6	6	6
3	6	36	20.0
5	6	36	21.6
8	6	36	22.5
10	6	36	21.0

Approximation. We apply the approximation method in Li et al. [2010] to our maximization problem, where the utility slots instead of the time slots is used as a state dimension in the dynamic programming. The approximation factor bound of this approach is $1 - \varepsilon N_g$. A better approximation factor is obtained with a finer-grained utility unit (a smaller ε). As the computational complexity of the approximation algorithm grows quadratically with the number of utility units, the finer-grained utility unit significantly increases the computational complexity. In our experiment, the same $\varepsilon = 0.2$ is used, the running time is close to our optimal multicast.

In our article, all of these algorithms collect the RoI requests and run the allocation algorithm every 2 seconds. The average running time of our optimal algorithm is 49.18ms, which only incurs 2.5% overhead.

We measure the peak signal-to-noise ratio (PSNR), a standard metric for measuring the video quality, and *goodput* of the system to compare the performance of the algorithms.

Video Setup. We evaluate the algorithms using two standard HD (1920×1080p) test video files, *controlled-burn* (dense motion) and *rush-hour* (low motion).[3] Table VII presents the video configurations and data rates.

Wireless Channels: We place the mobile devices at different locations and distances from the AP, to vary the channel conditions between the mobile devices and the AP. Table VIII shows the minimum, maximum, and average achieved link rates when there are up to 10 mobile devices.

RoI Variation. User requests and RoI used in the evaluations are based on the real interaction logs from 10 users who have used zoomable video system [Quang Minh Khiem et al. 2010].

7.1. Baseline Comparison

The average PSNR with error bars (standard deviation) across different users streaming at medium video rate are depicted in Figure 9. The corresponding achieved average goodput is present in Table IX. As the unicast scheme cannot fit the lowest resolution level requirement for more than five clients, no data point is presented in this range in the results. From the results, we can draw the following observations.

[3]Available at http://media.xiph.org/video/derf/.

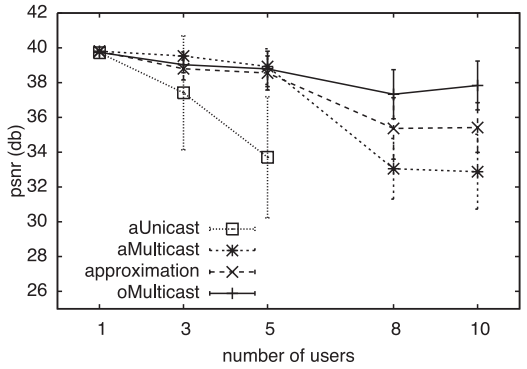


Fig. 9. Average PSNR with medium video rate.

Table IX. Average Goodput Achieved with Heterogeneous Link Qualities at Medium Video Rate

# Users	aUnicast	aMulticast	Approximation	oMulticast
1	3.83	3.79	3.81	3.82
3	2.95	3.45	3.46	3.41
5	1.8	3.07	3.05	3.07
8	\	2.1	2.27	2.56
10	\	1.99	2.25	2.67

(i) *PSNR Gains*. The multicast algorithms are able to satisfy up to five users requests without notable PSNR degradation. On the other hand, the video quality with unicast dramatically decreases beyond three users, and only up to five users can be supported by adaptive unicast. With more than five users, all three multicast schemes experience some PSNR loss. The optimal multicast, however, considerably outperforms approximation and adaptive multicast under heavy load, with the improvements of about 3dB and 5dB in PSNR, respectively.

(ii) *Goodput Gains*. Due to zooming, the demands between different clients are not identical. Hence, the trend in average goodput does not strictly follows that of video quality (Table IX). As predicted from Figure 9 and Table IX, the multicast algorithms outperform unicast when there are more than three users in terms of both PSNR and goodput. When there are more than five users, the improvements of optimal multicast over approximation and adaptive multicast with 10 users are 19% and 34%, respectively.

(iii) *Fairness Gains*. The error bars in Figure 9 indicate that our optimal multicast achieves the best fairness among all algorithms, due to adaptive utility assignment (Section 5.1) in our algorithm. Although a similar allocation method is used by adaptive multicast and adaptive unicast, they performs remarkably different in terms of fairness. While multicast transmission can benefit multiple users, unicast transmission does not, which may lead to less fairness among the users.

7.2. Impact of Video Rate

To evaluate the impact of video data rate (and thus the traffic load), we repeat the experiments using a different video with a lower rate and the same video encoded with a higher rate. We generate low-rate and high-rate videos in addition to the previously used medium rate. The configurations are detailed in Table VII. Wireless link quality

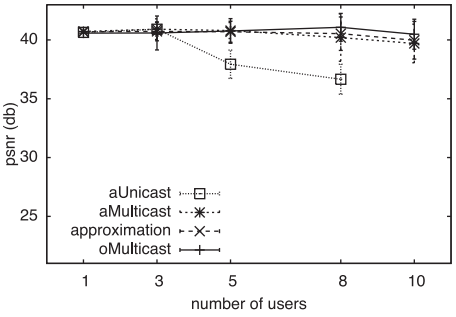


Fig. 10. Average PSNR with low video rate.

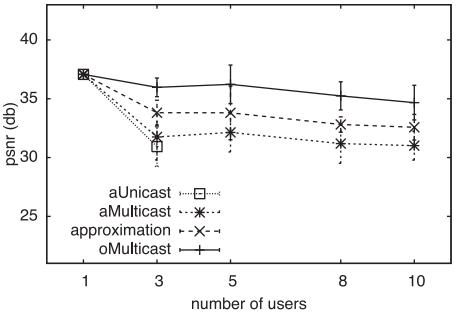


Fig. 11. Average PSNR with high video rate.

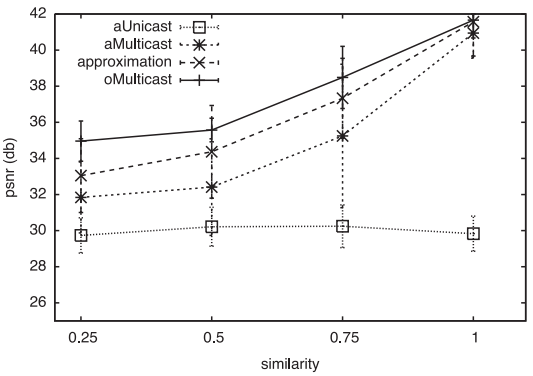


Fig. 12. Average PSNR with different similarity.

settings are the same to the previous section. Figure 10 and Figure 11 depict the average achieved PSNR for low-rate and high-rate videos, respectively.

Figure 10 demonstrates that all four algorithms perform better with lighter workload as expected. Specifically, the multicast algorithms scale up to 10 users without significant quality degradation, and the unicast scheme is able to support more clients.

For higher traffic load, all algorithms perform worst. Compared with other schemes, our optimal algorithm, however, still provide relatively fair quality under the higher load. In general, if a client does not induce lower link rate or request higher resolution level, no additional multicast traffic will be introduced. Thus, the video qualities are only slightly reduced even as more clients are added to the multicast sessions.

7.3. Impact of RoI Similarity

Intuitively, larger amount of RoI overlapping increases the relative performance gap of multicast over unicast. The impact of RoI overlapping is evaluated in this section. In order to control the amount of overlap, we do not use collected traces to simulate RoI variation. Instead, we manually vary the RoI sizes and positions so that they can change in a uniform and controlled manner. Here, the RoI sizes and the request resolution levels of all clients are identical. We vary the positions of RoI to generate different similarity.

To measure the degree of overlapping, we first define the *popularity* of a tile g, p_g as the fraction of the number of users interested in it. The degree of overlapping for user i is then the total popularity of all tiles in $\mathbf{G}(i)$, excluding the tiles only interested by user i, divided by the number of tiles in $\mathbf{G}(i)$. We then define *similarity* as the average

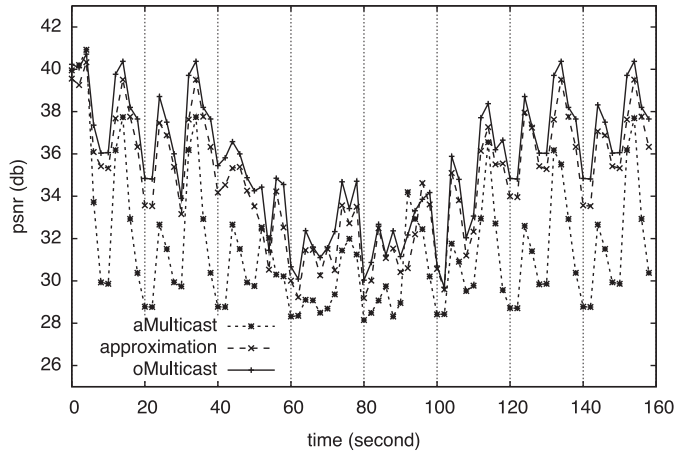


Fig. 13. Average PSNR of the mobile client.

overlapping degree across all users. We present how PSNR changes with different similarity, for eight users, in Figure 12.

The relatively stable performance in terms of video quality shows that the unicast scheme is not affected by the amount of RoI overlap. As expected, the improvement of multicast over unicast increases with the increasing RoI similarity. When the RoIs are identical (all users want the same regions), the improvement is about 12dB in PSNR. Interestingly, with increased similarity value, the PSNR quantities of three multicast algorithms converge to an identical point. Such convergence is caused by both the decrease in traffic demand and the fact that the same data is requested.

7.4. Client Mobility

The previous sections demonstrate the effectiveness of our optimal multicast algorithm with stationary clients. In this section, we evaluate the performance of our optimal algorithm with client mobility. In particular, we keep two clients static, the obtained link rates for them are 6Mb/s and 36Mb/s. One additional mobile client starts from a location close to the AP, moves away from it, and then moves back. Figure 13 plots the average PSNR of the mobile client for every two seconds. The movement period is from 40 s to 120 s.

In the experiment, the high-rate video is used and a segment of 20 s is played repeatedly. Although, the RoI of each user and the allocations are fixed under static condition, the PSNR of different frames are different. This disparity is due to the fact that sensitivity of different frames with mixed resolution tiles are different. The same trend of PSNR variations under static conditions can be observed between different playbacks.

From the figure, we observe that our optimal algorithm consistently out performs two baseline algorithms. The average enhancements of our optimal multicast over approximation and adaptive multicast are about 1dB and 4.5dB, respectively. Moreover, our algorithm can quickly adapt to the link rate and the video quality returns quickly to the level similar to the static period after the movement (at 120 seconds).

8. CONCLUSION

We have developed and implemented an efficient algorithm for multicasting mixed resolution tiles to heterogeneous users, for interactive video applications that support zoom and pan. Our algorithm optimizes the total utility of all clients and achieves

significant improvements in video quality: up to a 3dB improvement over approximation multicast approach, 6dB improvement over an adaptive multicast scheme, and 12dB improvement over adaptive unicast scheme in our experiment settings. Additionally, our approach can be directly applied to design an optimal allocation algorithm for a general multisessions video multicast. In the future, we shall extend this work to the scenarios with multiple access points (APs), where the AP association mechanism could be exploited to further enhance the multicast performance.

REFERENCES

IEEE 802.11-2007. 2007. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. IEEE 802.11 LAN Standards 2007.

John Charles Bicket. 2005. Bit-rate selection in wireless networks. Ph.D. dissertation, Massachusetts Institute of Technology.

Ranveer Chandra, Sandeep Karanth, Thomas Moscibroda, Vishnu Navda, Jitendra Padhye, Ramachandran Ramjee, and Lenin Ravindranath. 2009. Dircast: A practical and efficient wi-fi multicast system. In *Proceedings of the 17th IEEE International Conference on Network Protocols (ICNP)*. IEEE, 161–170.

Philip A. Chou and Zhourong Miao. 2006. Rate-distortion optimized streaming of packetized media. *IEEE Trans. Multimed.* 8, 2, 390–404.

Supratim Deb, Sharad Jaiswal, and Kanthi Nagaraj. 2008. Real-time video multicast in WiMAX networks. In *Proceedings of the 27th IEEE International Conference on Computer Communications (INFOCOM)*. IEEE.

Wu-chi Feng, Thanh Dang, John Kassebaum, and Tim Bauman. 2011. Supporting region-of-interest cropping through constrained compression. *ACM Trans. Multimed. Comput. Commun. Appl.* 7, 3, 17.

George A. Gescheider. 1997. *Psychophysics: The Fundamentals*. Psychology Press.

Sherif Halawa, Derek Pang, Ngai-Man Cheung, and Bernd Girod. 2011. ClassX: An open source interactive lecture StreamingSystem. In *Proceedings of the 19th ACM International Conference on Multimedia*. (AZ). ACM, 719–722.

Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. 2000. The Click modular router. *ACM Trans. Comput. Syst.* 18, 3, 263–297.

Peilong Li, Honghai Zhang, Baohua Zhao, and Sampath Rangarajan. 2009. Scalable video multicast in multicarrier wireless data systems. In *Proceedings of the 17th IEEE International Conference on Network Protocols (ICNP)*. IEEE, N.J., 141–150.

Peilong Li, Honghai Zhang, Baohua Zhao, and Sampath Rangarajan. 2010. Scalable video multicast with joint layer resource allocation in broadband wireless networks. In *Proceedings of the 18th IEEE International Conference on Network Protocols (ICNP)*. IEEE, 295–304.

Jiangchuan Liu, Bo Li, and Ya-Qin Zhang. 2003. Adaptive video multicast over the Internet. *IEEE MultiMedia* 10, 1, 22–33.

Aditya Mavlankar, Piyush Agrawal, Derek Pang, Sherif Halawa, Ngai-Man Cheung, and Bernd Girod. 2010. An interactive region-of-interest video streaming system for online lecture viewing. In *Proceedings of the 18th International Packet Video Workshop (PV)*. IEEE, 64–71.

Aditya Mavlankar, Pierpaolo Baccichet, David Varodayan, and Bernd Girod. 2007a. Optimal slice size for streaming regions of high resolution video with virtual pan/tilt/zoom functionality. In *Proceedings of 15th European Signal Processing Conference (EUSIPCO)*.

Aditya Mavlankar, David Varodayan, and Bernd Girod. 2007b. Region-of-interest prediction for interactively streaming regions of high resolution video. In *Proceedings of Packet Video 2007*. IEEE, 68–77.

Ioannis Pefkianakis, S. Wong, Hao Yang, S. Lee, and Songwu Lu. 2013. Towards history-aware robust 802.11 rate adaptation. *IEEE Trans. Mobile Comput.* 12, 3, 502–515.

Kandaraj Piamrat, Adlen Ksentini, J.-M. Bonnin, and César Viho. 2009. Q-DRAM: QoE-based dynamic rate adaptation mechanism for multicast in wireless networks. In *Proceedings of the Global Telecommunications Conference (GLOBECOM'09)*. 1–6.

Ngo Quang Minh Khiem, Guntur Ravindra, Axel Carlier, and Wei Tsang Ooi. 2010. Supporting zoomable video streams with dynamic region-of-interest cropping. In *Proceedings of the 1st Annual ACM SIGMM Conference on Multimedia Systems*. (AZ). ACM, 259–270.

Ngo Quang Minh Khiem, Guntur Ravindra, and Wei Tsang Ooi. 2011. Adaptive encoding of zoomable video streams based on user access pattern. In *Proceedings of the 2nd Annual ACM Conference on Multimedia systems*. (A). ACM, 211–222.

Sayandeep Sen, Neel Kamal Madabhushi, and Suman Banerjee. 2010. Scalable WiFi media delivery through adaptive broadcasts. In *Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation (NSDI)*. USENIX Association, 13.

Arash Shafiei, Quang Minh Khiem Ngo, Ravindra Guntur, Mukesh Kumar Saini, Cong Pang, and Wei Tsang Ooi. 2012. Jiku live: A live zoomable video streaming system. In *Proceedings of the 20th ACM International Conference on Multimedia*. ACM, 1265–1266.

Ray van Brandenburg, Omar Niamut, Martin Prins, and Hans Stokking. 2011. Spatial segmentation for immersive media delivery. In *Proceedings of the 15th International Conference on Intelligence in Next Generation Networks (ICIN)*. IEEE, 151–156.

Hui Wang, Vu-Thanh Nguyen, Wei Tsang Ooi, and Mun Choon Chan. 2014. Mixing tile resolutions in tiled video: A perceptual quality assessment. In *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop (NOSSDAV)*. 25–30.

Starsky H. Y. Wong, Ramya Raghavendra, Yang Song, and Kang-Won Lee. 2013. X-WING: A high-speed wireless broadcasting framework for IEEE 802.11 networks. In *Proceedings of the 10th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*. IEEE, 344–352.

Starsky H. Y. Wong, Hao Yang, Songwu Lu, and Vaduvur Bharghavan. 2006. Robust rate adaptation for 802.11 wireless networks. In *Proceedings of the 12th ACM Annual International Conference on Mobile Computing and Networking*. 146–157.

Jongwon Yoon, Honghai Zhang, Suman Banerjee, and Sampath Rangarajan. 2012. MuVi: A multicast video delivery scheme for 4G cellular networks. In *Proceedings of the 18th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*. 209–220.

Received October 2014; revised March 2015; accepted April 2015