Energy-Efficient Multicasting of Multiview 3D Videos to Mobile Devices

AHMED HAMZA and MOHAMED HEFEEDA, Simon Fraser University

Multicasting multiple video streams over wireless broadband access networks enables the delivery of multimedia content to large-scale user communities in a cost-efficient manner. Three dimensional (3D) videos are the next natural step in the evolution of digital media technologies. In order to provide 3D perception, 3D video streams contain one or more views that greatly increase their bandwidth requirements. Due to the limited channel capacity and variable bit rate of the videos, multicasting multiple 3D videos over wireless broadband networks is a challenging problem. In this article, we consider a 4G wireless access network in which a number of 3D videos represented in two-view plus depth format and encoded using scalable video coders are multicast. We formulate the optimal 3D video multicasting problem to maximize the quality of rendered virtual views on the receivers' displays. We show that this problem is NP-complete and present a polynomial time approximation algorithm to solve it. We then extend the proposed algorithm to efficiently schedule the transmission of the chosen substreams from each video in order to maximize the power saving on the mobile receivers. Our simulation-based experimental results show that our algorithm provides solutions that are within 0.3 dB of the optimal solutions while satisfying real-time requirements of multicast systems. In addition, our algorithm results in an average power consumption reduction of 86%.

Categories and Subject Descriptors: C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

General Terms: Design

Additional Key Words and Phrases: Multimedia networking, energy saving, wireless networks, 3D video, WiMAX, LTE

ACM Reference Format:

Hamza, A. and Hefeeda, M. 2012. Energy-efficient multicasting of multiview 3D videos to mobile devices. ACM Trans. Multimedia Comput. Commun. Appl. 8, 3s, Article 45 (September 2012), 25 pages.
DOI = 10.1145/2348816.2348824 http://doi.acm.org/10.1145/2348816.2348824

1. INTRODUCTION

Recently, 3D video has gained a wide interest both in the research community and in the market. Market studies predict that over 20 million TV homes globally will be watching 3D TV by 2015 [Informa Telecoms and Media 2010]. The IFA 2010, Europe's largest consumer electronics show, has witnessed a surge in the number of 3D-capable devices on display [Wearden 2010]. Many TV networks are starting to realize the potential of the 3D TV market. American sports broadcaster ESPN has launched a

This work was partially supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada and in part by the British Columbia Innovation Council (BCIC).

Authors' address: A. Hamza and M. Hefeeda, Simon Fraser University, 102nd Avenue, Survey, BC V3T 0A3, Canada; email: {aah10, mhefeeda}@cs.sfu.ca.

DOI 10.1145/2348816.2348824 http://doi.acm.org/10.1145/2348816.2348824

dedicated 3D channel in 2010 broadcasting popular sports events, such as the FIFA World Cup, in 3D [Pilkington 2010]. In Canada, CBC has also broadcast some NHL hockey games in 3D [Harrison 2010].

Advances in 3D video acquisition and display technologies have paved the way for many emerging 3D applications, such as free-viewpoint video, 3D TV, and immersive teleconferencing. Such applications expand the user experience beyond what is offered by traditional media. In free-viewpoint video, the viewer can interactively choose his/her viewpoint in 3D space to observe a real-world scene from preferred perspectives [Kimata et al. 2004]. 3D TV is the extension of the traditional 2D TV to displays that are capable of 3D rendering, where more than one view is decoded and displayed simultaneously [Vetro et al. 2004]. While free-viewpoint video focuses on the free navigation functionality and 3D TV emphasizes 3D experience, immersive teleconference participants may prefer both interactivity and virtual reality [Chen et al. 2009].

As mobile devices such as cell phones, tablets, personal gaming consoles and video players, and personal digital assistants become more powerful, their ability to handle 3D content is becoming a reality. According to ABI Research, 3D devices, including smartphones, notebooks, mobile Internet devices and portable game players, will comprise more than 11 percent of the total mobile devices market by 2015 [ABI Research 2010]. However, there are still many challenges that need to be addressed before commercializing 3D mobile services. A mobile 3D TV solution would require low bit rate, high quality views, low power and bandwidth consumption, and low complexity.

In this article, we consider the problem multicasting 3D videos over 4G broadband access networks, such as Long Term Evolution (LTE) and WiMAX. In particular, we address two main challenges: (i) maximizing the video quality of rendered views in auto-stereoscopic displays [Dodgson 2005; Urey et al. 2011] of mobile receivers such as smartphones and tablets; and (ii) minimizing the energy consumption of the mobile receivers during multicast sessions. Auto-stereoscopic displays provide 3D perception without the need for special glasses. For such displays, 3D scenes need to be efficiently represented using a small amount of data that can be used to generate arbitrary views not captured during the acquisition process. The multiview-plus-depth representation has proven to be both efficient and flexible to provide good quality synthesized views. However, the quality of synthesized views is affected by the compression of texture videos and depth maps. Given the limitations on the wireless channel capacity, it is important to efficiently utilize the channel bandwidth such that the quality of all rendered views at the receiver side is maximized.

We note that the focus of this article is optimizing the quality of 3D TV applications that use *multicast* for content delivery. These applications are different from the free viewpoint video (FVV) applications in which user interactivity is important to determine appropriate views of the 3D videos to be transmitted to different users. User interactivity is mostly achieved in unicast systems where a feedback channel exists. Our article focuses on multicast sessions where such feedback channels are not practical for large-scale user communities, which is the target of our work.

We consider multicasting multiview video streams in which the textures and depth maps of the views are simulcast coded using the scalable video coding extension of H.264/AVC. Two views of each multiview-plus-depth video are chosen for multicast and all chosen views are multiplexed over the wireless transmission channel. Joint texture-depth rate-distortion optimized substream extraction is performed in order to minimize the distortion in the views rendered at the receiver. We mathematically formulate the problem of selecting the set of substreams from each of the two chosen views for all video sequences being transmitted. We show that this problem is NP-hard, and thus cannot be optimally solved in real-time for an arbitrary number of input video streams. We propose a substream selection scheme that enables receivers to render the best possible quality for all views given the bandwidth constraints of the transmission channel and the variable nature of the video bit rate.

In 4G multimedia services, the subscribers are mainly mobile users with energy-constrained devices. Therefore, an efficient multicast solution should minimize the power consumption of the receivers to provide a longer viewing time experience. We extend our algorithm to perform energy-efficient radio frame scheduling of the selected substreams. The allocation algorithm attempts to find a burst transmission schedule that minimizes the energy consumption of the receivers. Transmitting the video data in bursts enables the mobile receivers to turn off their wireless interfaces for longer periods of time, thereby saving on battery power. The extended algorithm first determines the best substreams to transmit for each of the multicast sessions based on the current network capacity. It then allocates the video data to radio frames and constructs a burst schedule that does not result in buffer overflow or underflow instances at the receivers.

We developed a simulation system that implements the proposed algorithms and conducted several experiments using 3D video segments from the MPEG 3DV ad-hoc group data set. The performance of the proposed algorithm is compared against best possible results represented by the optimal solution of the problem. Our experimental results show that the proposed substream selection algorithm produces near optimal results (less than optimal solution by at most 0.3 dB) and terminates in a few milliseconds. Moreover, our energy-efficient allocation algorithm results in an average power saving of 86% across all multicast sessions without any buffer violation instances.

The rest of this article is organized as follows. Section 2 summarizes the related work in the literature. We provide a system overview in Section 3 and formally state the optimal 3D video multicasting problem. The proposed scalable 3D video multicasting algorithm is described in Section 4. An extension to the proposed algorithm to perform energy-efficient radio frame allocation is presented in Section 5. We present our experimental evaluation in Section 6, and we conclude the paper in Section 7. For the unfamiliar reader, necessary background material on 3D display technologies and representation formats, and the concept of scalable video coding are provided in Appendix A.

2. RELATED WORK

2.1 3D Video Transmission Over Wireless Networks

De Silva et al. [2010] studied the performance of 3D-TV transmission over an error prone WiMAX broadband wireless network. The study included three 3D video formats: video-plus-depth, MVD2, and MVD4. Using the MVD2 format two video streams are transmitted along with their depth map streams, while in the MVD4 format four video streams are transmitted along with their depth maps. The authors stated that multiview plus depth 3D video formats such as MVD2 and MVD4 demand much higher bandwidth than what a WiMAX wireless channel can afford (around 19 Mbps and 43 Mbps, respectively, according to Multimedia Scalable 3D for Europe (MUSCADE) Project [2010]). Thus, the channel will not be capable of transmitting them with acceptable delay and quality. However, the sequences used in their evaluation had resolutions of 1024×768 and 1280×720. Most mobile device displays currently available have resolutions around 640×360. LTE Release 9 mobile devices, for example, support QVGA (320×240), VGA (640×480), and WVGA (800×480) resolutions [Järvinen et al. 2010]. To the best of our knowledge, only the iPad tablet's display supports a 1024×768 resolution. Consequently, transmitting 3D video over WiMAX should be feasible for current mobile devices configurations. Moreover, future standards such as LTE-Advanced promise to push the channel capacity even further using techniques such as *carrier aggregation* [Yuan et al. 2010].

2.2 Modeling the Quality of Synthesized Views

There has been some recent work on modeling the quality of synthesized views based on the qualities of two reference views from which the target view is synthesized. Liu et al. [2009] attempted to solve the no-reference evaluation problem by proposing a distortion model to characterize the view synthesis quality without requiring the original reference image. In this model, the distortion of a synthesized virtual view is composed of three additive distortions: video coding-induced distortion, depth quantization-induced distortion, and inherent geometry distortion. The practicality of the presented model is however restricted due to its high complexity. Yuan et al. [2011] proposed an alternative and concise low-complexity distortion model for the synthesized view. Kim et al. [2010] also attempted to overcome the no-reference evaluation problem when coding depth maps by approximating the rendered view distortion from the reference texture video that belongs to the same viewpoint as the depth map. However, the model does not jointly consider both texture and depth map distortions. For our work, we validate the model relation presented in Yuan et al. [2011] and use it to solve the multiple 3D video multicasting problem.

2.3 Optimal Texture-Depth Bit Allocation

Bosc et al. [2011] studied the impact of bit rate of the texture and depth components on the quality of an intermediate synthesized view. The goal of the study was to find the optimal ratio between depth and texture bit rate to maximize the quality of the synthesized view. Results indicate that this ratio is dependent on the acquisition configuration of the video sequence, such as the camera baseline. As part of their asymmetric 3D video coding framework, Shao et al. [2012] proposed a bit allocation model to characterize the view rendering distortion and a chrominance reconstruction model to characterize the binocular suppression [Shao et al. 2012]. The aim of the proposed framework is to allocate the bit rates for texture videos and depth maps appropriately such that the objective performance of view rendering is enhanced while keeping the same perceptual visual quality. The works most related to ours are Petrovic et al. [2010] and Cheung et al. [2011]. Petrovic et al. [2010] perform virtual view adaptation for selective streaming of 3D multiview video. However, the proposed adaptation scheme requires empirically constructing the rate-distortion function for the 3D multiview video. Moreover, exhaustively searching the space of possible quantizers is computationally expensive. Cheung et al. [2011] addressed the problem of selecting the best views to transmit and determining the optimal bit allocation among texture and depth maps of the selected views, such that the visual distortion of synthesized views at the receiver is minimized. Contrary to our work, the bit allocation optimization problem presented in Cheung et al. [2011] is applicable in scenarios where the selected views are encoded on-the-fly and the coding parameters can be adjusted based on the available bandwidth. Coding 3D videos in real-time is however challenging. Our work assumes that the views are pre-encoded using scalable video coders and bit rate adaptation is performed via substream extraction, which is expected to be the common case in practice due to the flexibility it provides.

3. SYSTEM OVERVIEW AND PROBLEM STATEMENT

3.1 Overview

A wireless mobile video streaming system has four main components: content servers, access gateways, cellular base stations, and mobile receivers. Receivers periodically send feedbacks about current channel conditions, for example, signal-to-noise ratio (SNR) or link-layer buffer state, to the base station. Based on this feedback, the base station changes the modulation and coding scheme so that the SNR is increased. This consequently results in a change in channel capacity. Knowing the current capacity of the channel, a base station can adapt the bit rate of the transmitted video accordingly.

Transmitting 3D video over wireless networks faces many challenges. The main challenge is the capacity of the wireless channel, which is limited by the available bandwidth of the radio spectrum and various types of noise and interference. 3D video challenges the network bandwidth more than 2D videos as it requires the transmission of at least two video streams. These two streams can either be a stereo pair (one for the left eye and one for the right eye), or a texture stream and an associated depth stream from which the receiver renders a stereo pair by synthesizing a second view.

Receivers in a wireless network such as WiMAX and LTE are heterogeneous. A smartphone may be equipped with an auto-stereoscopic display capable of rendering only two views. Examples of such mobile displays include MasterImage's TN-LCD stereoscopic display [MasterImage 2012] and the 3D HDDP LCD produced by NEC [Uehara et al. 2008]. However, devices with a larger display size such as tablets will likely be capable of rendering more views in the near future since it is possible that more than one viewer will be watching from different angles. Rendering a large number of views also enables a more immersive experience. For example, the user can move his head in front of the display and experience (to a certain degree) a *look-around* effect by being able to see newly revealed background behind foreground objects. For such displays, a small number of views will be used to synthesize the remaining ones. To the best of our knowledge, no multiview auto-stereoscopic mobile display is currently available on the market. This is expected to change in the near future. Currently, the Philips WOWvx [Philips Electronics 2012] auto-stereoscopic display accepts a single V+D sequence and renders 9 views. However, a single V+D stream is not suitable for rendering multiple views, because the amount of disocclusions increases as the distance between the reference view and the synthesized view increases which results in a degradation of quality in views located far away from the reference view.

Transmitting two views and their depth maps enables the display to render higher quality views at each possible viewing angle [Gotfryd et al. 2008]. Although it is possible to use three or more reference views to cover most of the disocclusion holes in the synthesized view, the major concern is bandwidth consumption. Even with the texture and depth information of only two reference views, the aggregate rate of the four streams may exceed the channel capacity due to the variable bit rate nature of the video streams and the variation in the wireless channel conditions. Thus, allocation of system resources should be performed dynamically and efficiently to reflect the time varying characteristics of the channel [Su et al. 2011], which is the goal of this article.

3.2 Problem Statement

We consider a wireless multicast/broadcast service in 4G wireless networks streaming multiple 3D videos in MVD2 representation. Examples of such a service include the evolved multicast broadcast multimedia services (eMBMS) in LTE networks and the multicast broadcast service (MBS) in WiMAX. MVD2 is a multiview-plus-depth (MVD) representation in which there are only two views. Therefore, two video streams are transmitted along with their depth map streams. Each texture/depth stream is encoded using a scalable encoder into multiple quality layers.

Time is divided into a number of scheduling windows of equal duration δ, that is, each window contains the same number of time division duplex (TDD) frames. The base station allocates a fixed-size data area in the downlink subframe of each TDD frame. In the case of multicast applications, the parameters of the physical layer, for example, signal modulation and transmission power, are fixed for all receivers. These parameters are chosen to ensure an average level of bit error rate for all receivers in the coverage area of the base station. Thus, each frame transmits a fixed amount of data within its multicast area. In the following, we assume that the entire frame is used for multicast data and we refer to the multicast area within a frame as a *multicast block*. The first problem that we attempt to address in this article is the optimal substream selection problem, which can be stated as follows.

Problem 1 (*Optimal Multicasting of 3D Video Streams*). Consider a certain capacity of the wireless channel, a set $\mathbf{S}$ of 3D video streams in two-view plus depth (MVD2) format, and receivers with

Table I. List of Symbols Used in This Article

Symbol	Description
S	Number of 3D video streams
I	Number of synthesized intermediate views
L	Number of layers per view
q^t_{sl}	Average PSNR of left and right texture substream *sl*
q^d_{sl}	Average PSNR of left and right depth substream *sl*
r^t_{sl}	Sum of left and right texture substream *sl* data rates
r^d_{sl}	Sum of left and right depth substream *sl* data rates
b^t_{sl}	Number of blocks required for texture substream *sl*
b^d_{sl}	Number of blocks required for depth substream *sl*
P	Number of TDD frames within scheduling window
F	Capacity of the TDD frame
τ	Duration of the TDD frame
δ	Duration of the scheduling window
α^i_s	Quality model parameter for intermediate view i of video s
β^i_s	Quality model parameter for intermediate view i of video s
Υ^k_s	Stream s consumption buffer level at start frame of interval k
x^k_s	Start frame of interval k for stream s
z^k_s	End frame of interval k for stream s
y^k_s	Number of frames to be allocated for stream s in interval k

auto-stereoscopic displays. Each texture and depth component of every video stream is encoded into L layers using a scalable video coder. For each video stream $s \in \mathbf{S}$, select the optimal subset of layers to be transmitted over the network from each of the scalable streams representing the reference views such that: (1) the total amount of transmitted data does not exceed the available capacity; and (2) the average quality of synthesized views over all 3D video streams being transmitted is maximized.

4. PROBLEM FORMULATION AND SOLUTION

4.1 Mathematical Formulation

We now mathematically formulate the substream selection problem. The symbols used in the formulation are listed in Table I. Assuming there are S multiview-plus-depth video streams where two reference views are picked for transmission from each video. All videos are to be multiplexed over a single channel. If each view is encoded into multiple layers, then at each scheduling window the base station needs to determine which substreams to extract for every view pair of each of the S streams. Let R be the current maximum bit rate of the transmission channel. For each 3D video, we have four encoded video streams representing the two reference streams and their associated depth map streams. Each stream has at most L layers. The value of L can be different for each of the four streams. Thus, for each stream, we have L substreams to choose from, where substream l includes layer l and all layers below it. Let the data rates and quality values for selecting substream l of stream s be r_{sl} and q_{sl}, respectively, where $l = 1, 2, \ldots, L$. For example, q_{32} denotes the quality value for first enhancement layer substream of the third video stream. These values may be provided as separate metadata. Alternatively, if the scalable video is encoded using H.264/SVC [Schwarz et al. 2007] and the base station is media-aware, this information can be obtained directly from the encoded video stream itself using the Supplementary Enhancement Information (SEI) messages.

In the general case, texture or depth streams will not have the same number of layers. This provides flexibility when choosing the substreams that would satisfy the bandwidth constraints but complicates the quality model in the objective function since we will have to deal with the quality of the left view and the right view independently. Thus, we only consider an equal number of layers for left and right texture streams, as well as for the left and right depth streams. Moreover, corresponding layers in the left and right streams are encoded using the same *quantization parameter (QP)*. This enables us to treat corresponding layers in the left and right texture streams as a single item with a weight (cost) equal to the sum of the two rates and a representative quality equal to the average of the two qualities. The same also applies for left and right depth streams.

Let $\mathbf{I}$ be the set of possible intermediate views which can be synthesized at the receiver for a given 3D video that is to be transmitted. The goal is to maximize the average quality over all $i \in \mathbf{I}$ and all $s \in \mathbf{S}$. Thus, we have the problem of choosing the substreams such that the average quality of the intermediate synthesized views between the two reference views is maximized, given the constraint that the total bit rate of the chosen substreams does not exceed the current channel capacity. Let x_{sl} be binary variables that take the value of 1 if substream l of stream s is selected for transmission, and 0 otherwise. We denote with superscripts t and d the texture and depth streams, respectively. If the capacity of the scheduling window is C and the size of each TDD frame is F, then the total number of frames within a window is $P = C/F$. The data to be transmitted for each substream can thus be divided into $b_{sl} = \lceil r_{sl} \cdot \delta / F \rceil$ multicast blocks. We use a recent linear virtual view distortion model presented in Yuan et al. [2011] to represent the quality of the synthesized view in terms of the qualities of reference views. In Appendix B, we experimentally validate this distortion model using two different video quality metrics. Based on this model, the quality of a virtual view can be approximated by a linear surface in the form given in Eq. (1), where Q_v is the average quality of the synthesized views, Q_t is the average quality of the left and right texture references, Q_d is the average quality of the left and right references depth maps, and α, β, and C are model parameters. The model parameters can be obtained by either solving three equations with three combinations of Q_v, Q_t, and Q_d, or more accurately using regression by performing linear surface fitting.

$$Q_v = \alpha Q_t + \beta Q_d + C. \tag{1}$$

Consequently, we have the optimization problem (P1). In this formulation, constraint (P1a) ensures that the chosen substreams do not exceed the transmission channel's bandwidth. Constraints (P1b) and (P1c) enforce that only one substream is selected from the texture references and one substream from the depth references, respectively.

$$\text{Maximize} \quad \frac{1}{S}\sum_{s\in\mathbf{S}}\frac{1}{I}\sum_{i\in\mathbf{I}}\left(\alpha_s^i\sum_{l=1}^{L}x_{sl}^t q_{sl}^t + \beta_s^i\sum_{l=1}^{L}x_{sl}^d q_{sl}^d\right) \tag{P1}$$

$$\text{such that} \quad \sum_{s=1}^{S}\left(\sum_{l=1}^{L}x_{sl}^t b_{sl}^t + \sum_{l=1}^{L}x_{sl}^d b_{sl}^d\right) \le P \tag{P1a}$$

$$\sum_{l=1}^{L}x_{sl}^t = 1, \quad s = 1, \ldots, S, \tag{P1b}$$

$$\sum_{l=1}^{L}x_{sl}^d = 1, \quad s = 1, \ldots, S, \tag{P1c}$$

$$x_{sl}^t, x_{sl}^d \in \{0, 1\} \tag{P1d}$$

The following theorem shows that the optimal texture-depth substream selection problem given in (P1) is an NP-complete problem.

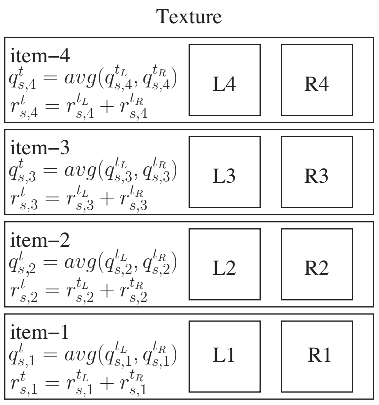

Texture
item–4
$q^t_{s,4} = avg(q^{t_L}_{s,4}, q^{t_R}_{s,4})$
$r^t_{s,4} = r^{t_L}_{s,4} + r^{t_R}_{s,4}$
L4
R4
item–3
$q^t_{s,3} = avg(q^{t_L}_{s,3}, q^{t_R}_{s,3})$
$r^t_{s,3} = r^{t_L}_{s,3} + r^{t_R}_{s,3}$
L3
R3
item–2
$q^t_{s,2} = avg(q^{t_L}_{s,2}, q^{t_R}_{s,2})$
$r^t_{s,2} = r^{t_L}_{s,2} + r^{t_R}_{s,2}$
L2
R2
item–1
$q^t_{s,1} = avg(q^{t_L}_{s,1}, q^{t_R}_{s,1})$
$r^t_{s,1} = r^{t_L}_{s,1} + r^{t_R}_{s,1}$
L1
R1

Fig. 1. Calculating profits and costs for texture component substreams of the reference views.

THEOREM 1. *Determining which layers to transmit from the texture and depth components of multiple 3D video sequences in MVD2 format over a wireless channel with a limited capacity such that the average perceived quality of all synthesized views is maximized is an NP-complete problem.*

PROOF. To show that the problem is NP-complete, we reduce a well-known NP-complete problem, the Multiple Choice Knapsack Problem (MCKP) [Kellerer et al. 2004, pp. 317], to our problem in polynomial time. We then show that a solution for our problem can be verified in polynomial time. In an MCKP instance, there are M mutually exclusive classes $N_1, \ldots, N_M$ of *items* to be packed into a knapsack of capacity W. Each item $j \in N_i$ has a profit p_{ij} and a weight w_{ij}. The problem is to choose exactly one item from each class such that the profit sum is maximized without having the total sum exceed the capacity of the knapsack.

The substream selection problem can be mapped to the MCKP in polynomial time as follows. The texture/depth streams of the reference views of each 3D video represent a multiple choice class in the MCKP. Substreams of these texture/depth reference streams represent items in the class. The average quality of the texture/depth reference views substreams represent the profit of choosing an item and the sum of their data rates represents the weight of the item. Figure 1 demonstrates this mapping for the texture component of video s in a set of 3D videos, where both the texture and the depth streams are encoded into 4 layers. For example, item-2 in the figure represents the second layer in both left and right reference texture streams with a cost equal to the sum of their data rates and a profit equal to their average quality. The 3D video is represented by two classes in the MCKP, one for the texture streams and one for the depth map streams. Finally, by making the scheduling window capacity the knapsack capacity, we have a MCKP instance. Thus, the problem is NP-hard, that is, an optimal solution to our problem would yield an optimal solution to the MCKP. Moreover, given a set of selected substreams from the components of each 3D video stream, this solution can be verified in $O(SL)$ steps. Hence, our substream selection problem is NP-complete. □

4.2 Proposed Solution

The 3D video multicasting problem can be solved optimally using enumerative algorithms such as branch-and-bound or dynamic programming. These algorithms are implemented in most of the available optimization tools. However, these algorithms have, in the worst case, running times that grow exponentially with the input size. Thus, this approach will not be suitable if the problem is large. Furthermore, optimizations tools may be too large or complex to run on a wireless base station. We propose

ALGORITHM 1: Scalable 3D Video Multicast (S3VM) Algorithm

```
Input: Scheduling window capacity $P$
Input: TDD frame capacity $F$
Input: Set of scalably simulcast coded MVD2 3D videos $\mathbf{S}$
Input: Model parameters for each virtual view position of each video $\alpha_s^i$, $\beta_s^i$
Input: Approximation factor $\epsilon$
Output: Set of substreams to transmit during the current scheduling window for texture/depth components of
    each 3D video
1: LP-relaxation: relax the integrality constraint (P1d) in the problem formulation to obtain an LP-relaxation of
   the problem.
2: SolveRelaxedLP
3: Drop fractional values, obtain split solution of value $z^{'}$
4: Calculate an upper bound ($2z^h$) on the optimal solution, where $z^h = \max(z^{'}, z^s)$
5: Calculate a scaling factor $K$
6: Scale the qualities of substreams $q_{sl}^{'} = \lfloor \hat{q}_{sl}/K \rfloor$
7: Solve the scaled down instance of the problem using dynamic programming by reaching to obtain a solution
   whose value is no less than $(1-\epsilon)z^*$
```

an approximation algorithm which runs in polynomial time and finds near optimal solutions. Given an approximation factor ϵ, an approximation algorithm will find a solution with a value that is guaranteed to be no less than $(1-\epsilon)$ of the optimal solution value, where ϵ is a small positive constant. The main steps of our proposed scalable 3D video multicast (S3VM) algorithm are given in Algorithm 1.

To solve a substream selection problem instance, we first calculate a single coefficient for the decision variables in the objective function. For variables associated with the texture component we have $\hat{q}_{sl}^t = q_{sl}^t \sum_{i\in\mathbf{I}} \alpha_s^i$, and the coefficient for depth component variables is $\hat{q}_{sl}^d = q_{sl}^d \sum_{i\in\mathbf{I}} \beta_s^i$. We then find an upper bound on the optimal solution value in order to reduce the search space. This is achieved by solving the linear program relaxation of the MCKP. A linear time partitioning algorithm for solving the LP-relaxed MCKP exists. This algorithm is based on the works of Dyer [1984] and Zemel [1984] and does not require any pre-processing of the classes, such as expensive sorting operations. This algorithm relies on the concept of *dominance* to delete items that will never be chosen in the optimal solution. We apply the Dyer-Zemel algorithm to our problem as shown in Algorithm 2. We note that a class in the context of the MCKP represents one of the two components (texture or depth) of a given 3D video in our problem, where each component is comprised of the corresponding streams from the two reference views. It should also be noted that m denotes the number of classes available at a particular iteration, since this changes from one iteration to another as the algorithm proceeds. Thus, at the beginning of the algorithm, we have $m = 2S$ classes.

An optimal solution vector $\mathbf{x}^{LP}$ to the linear relaxation of the MCKP satisfies the following properties: (1) $\mathbf{x}^{LP}$ has at most two fractional variables; and (2) if $\mathbf{x}^{LP}$ has two fractional variables, they must be from the same class. When there are two fractional variables, one of the items (substreams) corresponding to these two variables is called the *split item*, and the class containing the two fractional variables is denoted as the *split class*. A *split solution* is obtained by dropping the fractional values and maintaining the LP-optimal choices in each class (i.e., the variables with a value equal to 1). If $\mathbf{x}^{LP}$ has no fractional variables, then the obtained solution is an optimal solution to the MCKP.

By dropping the fractional values from the LP-relaxation solution, we have a split solution of value $z^{'}$ which we can use to obtain an upper bound. A heuristic solution to the MCKP with a worst-case performance equal to 1/2 of the optimal solution value can be obtained by taking the maximum of $z^{'}$ and z^s, where z^s is the sum of the split substream from the split class, that is, the stream to which the split substream belongs, and the sum of the qualities of the substreams with the smallest number

ALGORITHM 2: SolveRelaxedLP

```
Input: LP-relaxed version of (P1) after dropping integrality constraint (P1d)
Output: Solution vector $\mathbf{x}^{LP}$, having at most two fractional variables
 1: $\forall$ class (component) $N_j$, pair substreams two by two as $(jk_1, jk_2)$ and order each pair such that $b_{jk_1} \leq b_{jk_2}$,
    break ties such that $\hat{q}_{jk_1} \geq \hat{q}_{jk_2}$ and eliminate dominated substreams
 2: Set $B = 0$ and $Q = 0$
 3: $\forall$ class (component) $N_j$. If component has only one substream $k$ left, decrease capacity $P = P - b_{jk}$, set
    $Q = Q + \hat{q}_{jk}$, and remove component $N_j$
 4: $\forall$ $(jk_1, jk_2)$, derive slope $\pi_{jk_1jk_2} = \frac{\hat{q}_{jk_2} - \hat{q}_{jk_1}}{b_{jk_2} - b_{jk_1}}$
 5: Let $\gamma$ be the median of the slopes $\{\gamma_{jk_1jk_2}\}$
 6: For $j = 1, \ldots, m$, derive $M_j(\pi)$ and $\phi_j$, $\psi_j$ for $j = 1, \ldots, m$ according to:
    $$M_j(\pi) = \left\{ j \in N_j : (\hat{q}_{jk} - \pi b_{jk}) = \max_{l \in N_j}(p_{jl} - \pi b_{jl}) \right\}$$
    $$\phi_j = \arg\min_{k \in M_j(\pi)} b_{jk}$$
    $$\psi_j = \arg\max_{k \in M_j(\pi)} b_{jk}$$
 7: If $\pi$ is optimal, i.e. if $B + \sum_{j=1}^{m} b_{j\phi_j} \leq P < B + \sum_{j=1}^{m} b_{j\psi_j}$, set $B = B + \sum_{j=1}^{m} b_{j\phi_j}$, and $Q = Q + \sum_{j=1}^{m} \hat{q}_{j\phi_j}$.
    Optimal solution to LP-relaxation is $z^* = Q + (P - B)\gamma$. Stop.
 8: If $\sum_{j=1}^{m} b_{j\phi_j} \geq P$, then for all pairs $(jk_1, jk_2)$ with $\pi_{jk_1jk_2} \leq \pi$ delete substream $k_2$
 9: If $\sum_{j=1}^{m} b_{j\psi_j} < P$, then for all pairs with $\pi_{jk_1jk_2} \geq \pi$ delete substream $k_1$
10: Go to step 1.
```

of required multicast blocks in each of the other components' streams [Kellerer et al. 2004]. Since the optimal objective value z^* is less than or equal to $z^{'} + z^s$, thus $z^* \leq 2z^h$ and we have an upper bound on the optimal solution value. We use the upper bound in calculating a scaling factor K for the quality values of the layers. In order to get a performance guarantee of $1 - \epsilon$, we choose $K = \frac{\epsilon z^h}{2S}$. The quality values are scaled down to $q^{'}_{sl} = \lfloor \hat{q}_{sl}/K \rfloor$. We then proceed to solve the scaled down instance of the problem using *dynamic programming by reaching* (also known as *dynamic programming by profits*).

Let $B(g, q)$ denote the minimal number of blocks for a solution of an instance of the substream selection problem consisting of stream components $1, \ldots, g$, where $1 \leq g \leq 2S$, such that the total quality of selected substreams is q. For all components $g \in \{1, \ldots, 2S\}$ and all quality values $q \in \{0, \ldots, 2z^h\}$, we construct a table where the cell values are $B(g, q)$ for the corresponding g and q. If no solution with total quality q exists, $B(g, q)$ is set to ∞. Initializing $B(0, 0) = 0$ and $B(0, q) = \infty$ for $q = 1, \ldots, 2z^h$, the values for classes $1, \ldots, g$ are calculated for $g = 1, \ldots, 2S$ and $q = 1, \ldots, 2z^h$ using the recursion in Eq. (2).

$$B(g, q) = \min \begin{cases} B(g-1, q - q_{g1}) + b_{g1} & \text{if } 0 \leq q - q_{g1} \\ B(g-1, q - q_{g2}) + b_{g2} & \text{if } 0 \leq q - q_{g2} \\ \vdots & \\ B(g-1, q - q_{gn_g}) + b_{gn_g} & \text{if } 0 \leq q - q_{gn_g} \end{cases} \tag{2}$$

The value of the optimal solution is given by Eq. (3). To obtain the solution vector for the substreams to be transmitted, we perform backtracking from the cell containing the optimal value.

$$Q^* = \max\{q | B(2S, q) \leq P\}. \tag{3}$$

4.3 Analysis

4.3.1 *Correctness.* The core component of our algorithm is solving the dynamic programming formulation based on the recurrence relation in Eq. (2). We prove the correctness of the recurrence relation using induction. For the basis step where we only consider a single component of one video stream, only the substream of maximum quality and a number of blocks requirement not exceeding the capacity of the scheduling window is selected. We assume for the induction hypothesis case of $g-1$ components that it is also the case that the selected substreams have the maximum possible quality with a total bit rate not exceeding the capacity. For filling the $B(g,q)$ entries in the dynamic programming table, we first retrieve all $B(g-1, q-q_{gl})$ entries and add the number of block requirements b_{sl} of corresponding layers to them. According to Eq. (2), only the substream with minimum number of blocks among all entries which result in quality q is chosen. This guarantees that the *exactly one substream per component* constraint is not violated. Since $B(g-1,q)$ is already minimum, then $B(g,q)$ is also minimum for all q. Therefore, based on the above and Eq. (3), the proposed algorithm generates a valid solution for the substream selection problem.

4.3.2 *Approximation Factor.* Let the optimal solution set to the problem be X^* with a corresponding optimal value of z^*. Running dynamic programming by profits on the scaled instance of the problem results in a solution set $\tilde{X}$. Using the original values of the substreams chosen in $\tilde{X}$, we obtain an approximate solution value z^A. Because we use the floor operation to round down the quality values during the scaling process, we have

$$z^A = \sum_{j \in \tilde{X}} q_j \geq \sum_{j \in \tilde{X}} K \left\lfloor \frac{q_j}{K} \right\rfloor. \tag{4}$$

The optimal solution to a scaled instance will always be at least as large as the sum of the scaled quality values of the substreams in the optimal solution set X^* of the original problem. Thus, we have the following chain of inequalities

$$\begin{aligned} \sum_{j \in \tilde{X}} K \left\lfloor \frac{q_j}{K} \right\rfloor \; &\geq \; \sum_{j \in X^*} K \left\lfloor \frac{q_j}{K} \right\rfloor \geq \sum_{j \in X^*} K \left(\frac{q_j}{K} - 1 \right) \\ &= \; \sum_{j \in X^*} (q_j - K) = z^* - 2SK. \end{aligned} \tag{5}$$

Replacing the value of K, we get

$$z^A \geq z^* - 2S \cdot \frac{\epsilon z^h}{2S} = z^* - \epsilon z^h. \tag{6}$$

Since z^h is a lower bound on the optimal solution value ($z^h \leq z^*$), we finally have

$$z^A \geq z^* - \epsilon z^* = (1-\epsilon) z^*. \tag{7}$$

This proves that the solution obtained by our algorithm is always within a factor of $(1-\epsilon)$ from the optimal solution. Therefore, it is a constant factor approximation algorithm.

4.3.3 *Time Complexity.* The dynamic programming table for the $B(g,q)$ entries contains $2S \times 2z^h$ entries. Computing each entry in the table requires $O(L)$ time according to the recurrence relation given in Eq. (2). Therefore, table construction requires $O(L \cdot 2S \cdot 2z^h)$ time or $O(nz^*)$, where n is the total number of layers for all the streams components. Calculating z^h takes $O(n)$ time using the Dyer-Zemel algorithm. This leads to a total time of $O(n + nz^*)$. We are using dynamic programming to solve a scaled down instance of the problem where $K = \frac{\epsilon z^h}{2S}$. Since $z^* \leq 2z^h$, we have $z^*/K \leq \frac{4S}{\epsilon}$. This means that computing the table entries now takes $O(nS/\epsilon)$ time. Therefore, the time complexity of the S3VM algorithm is $O(nS/\epsilon)$.

5. ENERGY EFFICIENT RADIO FRAME SCHEDULING

In the S3VM algorithm presented in Section 4.2, we have determined which substreams to transmit from the texture and depth components of each 3D video in order to maximize the quality of the synthesized views. We now turn to the problem of allocating the video data of the chosen substreams within the frames of the scheduling window. Minimizing energy consumption is a main concern in battery powered mobile wireless devices. Implementing an energy saving scheme which minimizes the energy consumption over all mobile subscribers is therefore a crucial requirement for multicasting video streams over wireless access networks. Instead of continuously sending the streams at the encoding bit rate, a typical energy saving scheme transmits the video streams in *bursts*. After receiving a burst of data, mobile subscribers can switch off their RF circuits until the start of the next burst. An optimal allocation scheme should generate a burst schedule that maximizes the average system-wide energy saving over all multicast streams. The problem of finding the optimum schedule is complicated by the requirement that the schedule must ensure that there are no receiver buffer violations for any multicast session. In fact, the problem of burst scheduling for the much simpler case of 2D video streams has been proven to be NP-complete [Hefeeda and Hsu 2008].

5.1 Proposed Allocation Algorithm

We approach the problem by leveraging a scheme known as *double buffering* in which a receiver buffer of size B is divided into two buffers, a *receiving* buffer and a *consumption* buffer, of size $B/2$ [Hsu and Hefeeda 2010]. Thus, a number of bursts with an aggregate size of $B/2$ can be received while the video data are being drained from the consumption buffer. This scheme resolves the buffer overflow problem. To avoid underflow, we must make sure that the reception buffer is completely filled by the time the consumption buffer is completely drained, and the buffers are swapped at that point in time. Unlike Hsu and Hefeeda [2010], we consider a burst composed of one or more contiguous radio frames allocated to a certain video stream because we are dealing with complete radio frames of fixed duration.

Let γ_s be the energy saving for a mobile subscriber receiving stream s. γ_s is the ratio between the amount of time the RF circuits are put in sleep mode within the scheduling window to the total duration of the window. This metric has been used in previous works in the literature [Yang et al. 2004], [de Diego Balaguer et al. 2005] to evaluate the energy saving of a burst schedule. The average system-wide energy saving over all multicast sessions can therefore be defined as $\gamma = \frac{1}{S}\sum_{s=1}^{S}\gamma_s$. The objective of an energy efficient allocation algorithm is thus a list Γ of the form $\langle n_s, \langle f_s^1, w_s^1\rangle, \ldots, \langle f_s^2, w_s^2\rangle\rangle$ for each 3D video stream. In this list, n_s is the number of bursts that should be transmitted for stream s within the scheduling window, and f_s^k and w_s^k denote the starting frame and the width of burst k, respectively. Moreover, no two bursts should overlap, that is, $[f_s^k \ldots f_s^k + w_s^k] \cap [f_{\bar{s}}^{\bar{k}} \ldots f_{\bar{s}}^{\bar{k}} + w_{\bar{s}}^{\bar{k}}]$. Here, the operator $[\ldots]$ denotes an integer interval.

Since the substreams have been already chosen by the S3VM algorithm, we omit the substream subscripts l from corresponding terms in the following for simplicity, for example, r_s^t instead of r_{sl}^t. Let r_s be the aggregate bit rate of the texture and depth component substreams of video s, that is, $r_s = r_s^t + r_s^d$. For each 3D video stream, we divide the scheduling window into a number of intervals w_s^k, where k denotes the interval index, during which we need to fill the receiving buffer with $B/2$ data before the consumption buffer is completely drained. We note that depending on the video bit rate, the length of the interval may not necessarily be aligned with the radio frames. Therefore, buffer swapping at the receiver, which occurs whenever the consumption buffer is completely drained, may take place at any point during the last radio frame of the interval. The starting point of an interval is always aligned with radio frames. Thus, it is necessary to keep track of the current level of the consumption buffer at the beginning of an interval to determine when the buffer swapping will occur and set the deadline accordingly.

Let Υ_s^k denote the consumption buffer level for stream s at the beginning of interval k, and x_s^k and z_s^k are the start and end frames for interval k of stream s, respectively. The end frame for an interval represents a deadline by which the receiving buffer should be filled before a buffer swap occurs. Within each interval for stream s, the base station schedules y_s^k for transmission before the deadline. Except for the last interval, the number of frames to be transmitted is $\lceil \frac{B/2}{F} \rceil$. We note that the last of the scheduled frames within an interval may not be completely filled with video data. For the last interval, the end time is always set to the end of the scheduling window. The amount of data to be transmitted within this interval is calculated based on how much data will be drained from the consumption buffer by the end of the window.

$$\Upsilon_s^k = \begin{cases} B/2 & \text{if } k = 0 \\ \frac{B}{2} - \left(1 - \frac{\Upsilon_s^{k-1} \mod r_s\tau}{r_s\tau}\right) & \text{if } \Upsilon_s^{k-1} \mod r_s\tau \neq 0 \\ B/2 & \text{otherwise} \end{cases} \tag{8}$$

$$x_s^k = \begin{cases} 0 & \text{if } k = 0 \\ z_s^{k-1} & \text{if } \Upsilon_s^{k-1} \mod r_s\tau = 0 \\ z_s^{k-1} + 1 & \text{otherwise} \end{cases} \tag{9}$$

$$z_s^k = \begin{cases} P & \text{if } k \text{ is last interval} \\ x_s^k + \lfloor \frac{\Upsilon_s^k}{r_s\tau} \rfloor & \text{otherwise} \end{cases} \tag{10}$$

$$y_s^k = \begin{cases} \lceil (\frac{B}{2} - \Upsilon_s^k) + r_s\tau(P - x_s^k) \rceil & \text{if } k \text{ is last interval} \\ \lceil \frac{B/2}{F} \rceil & \text{otherwise.} \end{cases} \tag{11}$$

Based on this formulation, a complete energy-efficient scalable 3D video multicast (eS3VM) algorithm is given in Algorithm 3. Assuming that the consumption buffer is initially full, the proposed allocation extension proceeds as follows. The start frame number for all streams is initially set to zero. Decision points are set at the start and end frames for each interval of each frame as well as the frame at which all data to be transmitted within the interval has been allocated. At each decision point, the algorithm picks the interval with earliest deadline, that is, closest end frame, among all outstanding intervals. It then continues allocating frames for the chosen video until the next decision point or the fulfillment of the data transmission requirements for that interval. We demonstrate the concepts of transmission intervals and decision points in Figure 2 for a two stream example. Stream-2 in the figure has a higher data rate. Thus, the consumption buffer for the receivers of the second multicast session is drained faster than consumption buffer of the receivers of the first stream. Consequently, the transmission intervals for stream-2 are shorter. The set of decision points within the scheduling window is the union of the decision points of all streams being transmitted.

If no feasible allocation satisfying the buffer constraints is returned, the selected substreams cannot be allocated within the scheduling window. Thus, the problem size needs to be reduced by discarding one or more layers from the input video streams and a new set of substreams needs to be recomputed. To prevent severe shape deformations and geometry errors, we initially restrict the layer reduction

ALGORITHM 3: Energy-efficient Scalable 3D Video Multicast (eS3VM)

```
Input: Scheduling window capacity $P$
Input: TDD frame capacity $F$
Input: Buffer size $B$
Input: Set of scalably simulcast coded MVD2 3D videos $\mathbf{S}$
Input: Model parameters for each virtual view position of each video $\alpha_s^i$, $\beta_s^i$
Input: Approximation factor $\epsilon$
Output: Video data burst allocation to radio frames of current scheduling window
 1: Run S3VM algorithm to select the substreams for the texture and depth components of the videos
 2: for $s = 1 \to S$ do
 3:     $k \leftarrow 0$
 4:     Calculate $x_s^k$ using (9)
 5:     while $x_s^k < P$ do
 6:         Calculate $y_s^k$ and $z_s^k$ using (11) and (10)
 7:         $k \leftarrow k + 1$
 8:     end while
 9: end for
10: Let $\Lambda = \phi$
11: foreach decision point do
12:     $t_{\text{current}} \leftarrow$ current time
13:     $t_{\text{next}} \leftarrow$ next decision point time
14:     Get interval $w_s^k$ with earliest deadline $z_s^k$ among all outstanding intervals
15:     Allocate frames between $t_{\text{current}}$ and $t_{\text{next}}$
    end
16: $e_s^k \leftarrow$ actual completion time for bursts in interval $k$
17: if $\max\{e_s^k - z_s^k\} \leq 0$ then
18:     return $\Lambda$
19: end if
20: No feasible solution
```

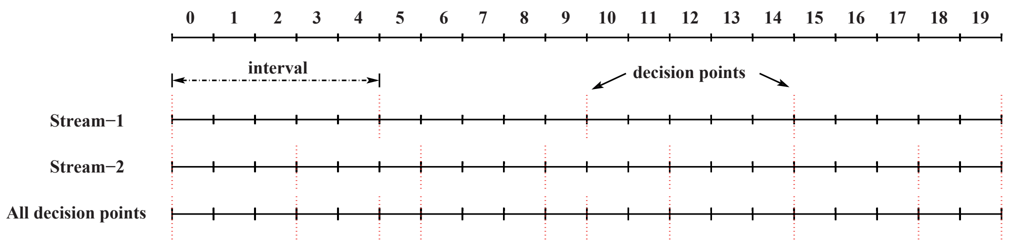


Fig. 2. Transmission intervals and decision points for two streams ($r_2 > r_1$) in a scheduling window of 20 TDD frames.

process to the texture components of the 3D videos. This process is repeated until a feasible allocation is obtained or all enhancement layers of texture components have been discarded. If a feasible solution is not obtained after discarding all texture component enhancement layers, we proceed with reducing layers from the depth components. Given only the base layers of all components, if no feasible solution is found, the system should reduce the number of video streams to be transmitted. Deciding on the video stream from which an enhancement layer is discarded is based on the ratio between the average quality synthesized views and size of the video data being transmitted within the window. We calculate the average quality given by the available substreams of each video over all synthesized views. We then divide this value by the amount of data being transmitted within the scheduling window. The video stream with the minimum quality to bits ratio is chosen for enhancement layer reduction.

6. EVALUATION

6.1 Setup

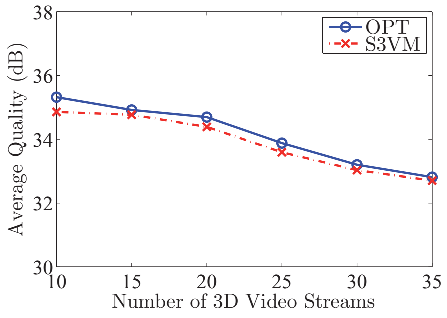
We implemented the proposed substream selection algorithm in Java and evaluated its performance using scalable video trace files. To generate the video traffic, we used six 3D video sequences. We divide each sequence into four 60-frame (2 seconds) segments to obtain 24 multiview-plus-depth video streams. The texture and depth streams were then encoded using the JSVM reference software version 9.19 [JSVM 2011] into one base layer and four medium grain scalability (MGS) layers. The quantization parameter values used in the encoding process are 36, 34, 30, 28, and 26. We then extract and decode each of the substreams from the encoded bitstreams and calculate the average quality and total bit rate for the corresponding layers of the left and right reference views. We summarize this information for the first segment of each of the original six video sequences in Table III in Appendix C.

For each texture-depth quality combination, three intermediate views are synthesized using VSRS 3.5 [Tanimoto et al. 2008]. We synthesize virtual views by using the general synthesis mode with half-pel precision. The quality of the synthesized views are compared against the quality of views synthesized from the original noncompressed references. These values are then used along with average qualities obtained for the compressed reference texture and depth substreams to obtain the model parameters at each synthesized view position. We consider a 20-MHz Mobile WiMAX channel, which supports data rates up to 60 Mbps depending on the modulation and coding scheme [Kumar 2008]. The typical frame duration in Mobile WiMAX is 5 ms. Thus, for a 1-second scheduling window, there are 200 TDD frames. We assume that the size of the MBS area within each frame is 100 Kb. The initial multicast channel bit rate is therefore 20 Mbps. To assess the performance of our algorithm, we run several experiments, as described in the sequel, and compare our results with the optimal substream selection solution obtained using the CPLEX LP/MIP solver [CPLEX 2011]. All experiments were run on a dual 2.66-GHz Intel Xeon processor machine with two cores in each physical processor (for a total of 4 cores) and 8 GB of physical memory. The two performance metrics used in our evaluation are: *average video quality* (over all synthesized views and all streams), and *running time*.

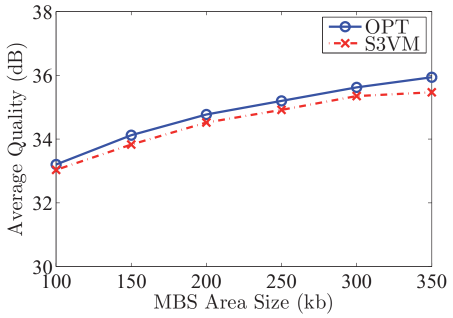
6.2 Substream Selection Results

6.2.1 *Video Quality.* In the first experiment, we study the performance of our algorithm in terms of video quality. We first fix the MBS area size at 100 Kb and vary the number of 3D video streams from 10 to 35 streams. The approximation parameter ϵ is set to 0.1. We calculate the average quality across all video streams for all synthesized intermediate views. We compare the results obtained from our algorithm to those obtained from the absolute optimal substream set returned by the CPLEX optimization software. The results are shown in Figure 3(a). As expected, the average quality of a feasible solution decreases since more video data need to be allocated within the scheduling window. However, it is clear that our algorithm returns a near optimal solution with a set of substreams that results in an average quality that is less than the optimal solution by at most 0.3 dB. Morever, as the number of videos increases, the gap between the solution returned by the S3VM algorithm and the optimal solution decreases. This indicates that our algorithm scales well with the number of streams.

We then fix the number of video streams at 30 and vary the capacity of the MBS area from 100 Kb to 350 Kb, reflecting data transmission rates ranging from 20 Mbps to 70 Mbps. As can be seen from the results in Figure 3(b), the quality of the solution obtained by our algorithm again closely follows the optimal solution.

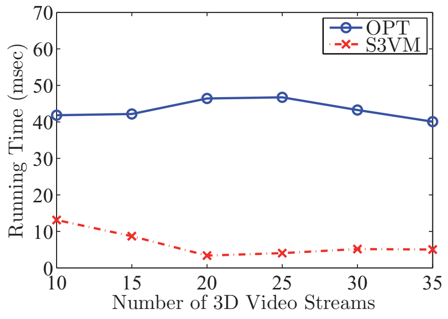


(a) Quality vs. Number of Streams

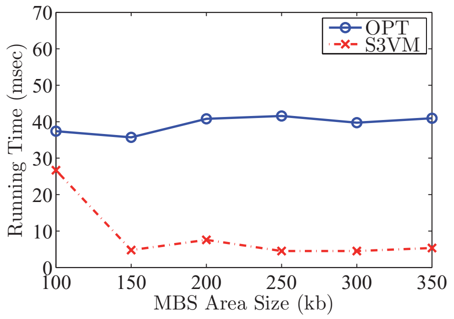


(b) Quality vs. MBS Area Size

Fig. 3. Average quality of solutions obtained using proposal (taken over all video sequences) for: (a) variable number of video streams; (b) different MBS area sizes.



(a) Number of Streams vs. Running Time



(b) MBS Area Size vs. Running Time

Fig. 4. Average running times for: (a) variable number of video streams; (b) different MBS area sizes.

6.2.2 *Running Time.* In the second set of experiments, we evaluate the running time of our algorithm against that of finding the optimum solution. Fixing the approximation parameter at 0.1 and the MBS area size at 100 Kb, we measure the running time of our algorithm for a variable number of 3D video streams. Figure 4(a) compares our results with those measured for obtaining the optimal solution. As shown in the figure, the running time of the S3VM algorithm is almost a quarter of the time required to obtain the optimal solution for all samples. In Figure 4(b), we show the results for a second experiment where the number of videos was fixed at 30 streams and the MBS area size was varied from 100 Kb to 350 Kb. From the figure, it is clear that the running time of our algorithm is still significantly less than that of the optimum solution.

6.2.3 *Approximation Parameter.* In the last experiment, we study the effect of the approximation parameter value ϵ on the running time of our algorithm. We use 30 video streams with an MBS area size of 100 Kb, and vary ϵ from 0.1 to 0.5. As shown in Figure 5(a), increasing the value of the approximation parameter results in faster running time. In the description of the S3VM algorithm in Section 4.2, the scaling factor K is proportional to the value of ϵ. Therefore, increasing ϵ results in smaller quality values which reduces the size of the dynamic programming table and consequently the running time of the algorithm at the cost of increasing the gap between the returned solution and optimal solution, as illustrated in Figure 5(b).

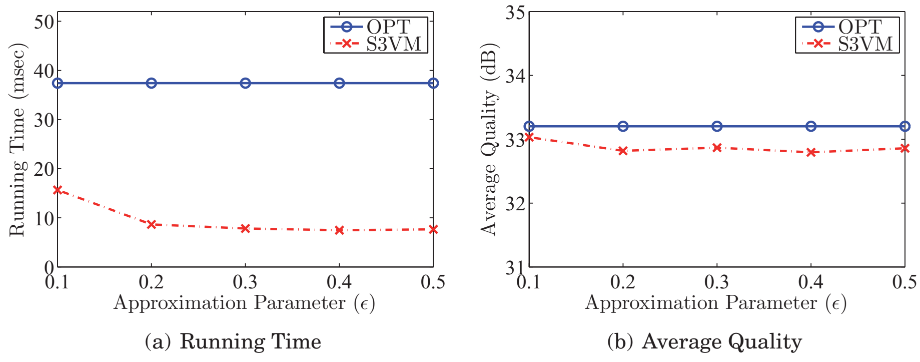


Fig. 5. Average running times for different approximation parameter values.

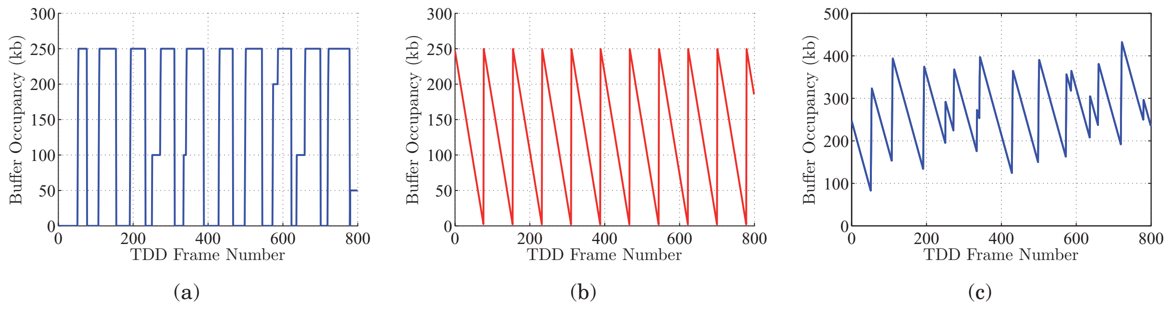


Fig. 6. Allocation algorithm performance in terms of receiver buffer occupancy levels of selected substreams using a 4 second scheduling window: (a) receiving buffer; (b) consumption buffer; (c) overall buffer level.

6.3 Radio Frame Allocation Results

To study the performance of our allocation algorithm, we generate a 500 second workload from each 3D video. We do this by taking the 8-second video streams, starting from a random initial frame, and then repeating the frame sequences. The resulting sequences are then encoded as discussed in Section 6.1. The experiments are performed over a period of 50 consecutive scheduling windows.

6.3.1 *Buffer Level Validation.* In this experiment, we validate that the output schedule from the proposed allocation algorithm does not result in buffer violations for receivers. We set the scheduling window duration to 4 seconds and the size of the receivers' buffers to 500 kb. We then plot the total buffer occupancy for each multicast session at the end of each TDD frame within the scheduling window. The total buffer occupancy is calculated as the sum of the receiving buffer level and the consumption buffer level. Figure 6 demonstrates the buffer occupancy for the two buffers as well as the total buffer occupancy for one multicast session. As can be seen from Figure 6(a), the receiver buffer occupancy never exceeds the buffer size, indicating no buffer overflow instances. For the consumption buffer, we observe that its occupancy jumps directly to the maximum level as soon as the buffer becomes empty due to buffer swapping, as shown in Figure 6(b). Similar results were obtained for the rest of the multicast sessions. This indicates that no buffer underflow instances occur.

6.3.2 *Energy Saving.* In the last experiment, we evaluate the energy saving performance of our radio frame allocation algorithm. For this evaluation, we use the power consumption parameters of

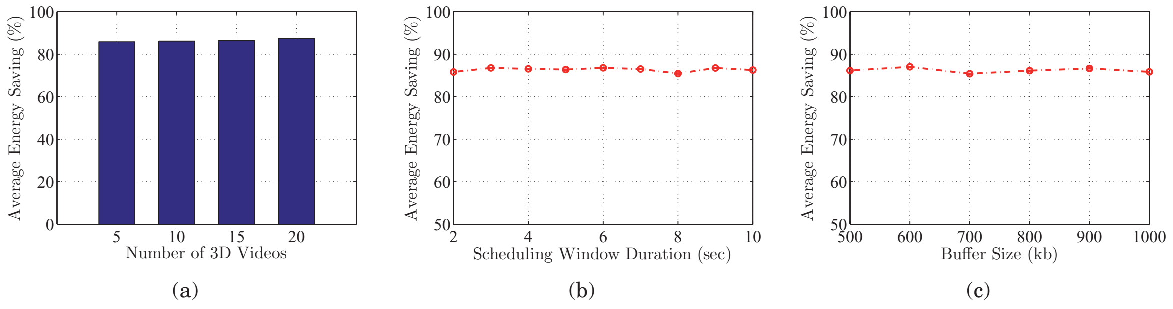


Fig. 7. Average energy saving for: (a) variable number of videos streams; (b) variable scheduling window duration; and (c) variable receiver buffer size.

an actual WiMAX mobile station [Sequans Communications 2007]. The power consumption during the sleep mode and listening mode is 10 mW and 120 mW, respectively. This translates to an energy consumption of 0.05 mJ and 0.6 mJ, respectively, for a 5-ms radio frame. In addition, the transition from the sleep mode to the listening mode consumes 0.002 mJ. We set the TDD frame size to 150 kb and the receiver buffer size to 500 kb. Using a 2-second scheduling window, we vary the number of multicasted videos from 5 to 20 and measure the average power saving over all streams, as shown in Figure 7(a). Next, keeping all other parameters the same, we set the number of videos to 5 and vary the duration of the scheduling window from 2 to 10 seconds. We plot the average energy savings along with the variance in Figure 7(b). Finally, in Figure 7(c), we evaluate the energy saving at different buffer sizes. We set the number of video to 10, the duration of the window to 2 seconds, and vary the receiver buffer size from 500 to 1000 kb. As can be seen from Figure 7, our eS3VM algorithm maintains a high average energy saving value, around 86%, over all transmitted streams. In all cases, the measured variance was very small and hardly noticeable in the figures.

7. CONCLUSIONS

We formulated the 3D video multicasting problem in wireless environments. In this problem, it is required to select the reference representation that maximizes the quality of the synthesized views rendered on the receiver's display given the bandwidth limitations of the channel. We showed that the problem is NP-complete. We presented an approximation algorithm for solving the problem in multicast services over 4G wireless networks. Our algorithm leverages scalable coded multiview-plus-depth 3D videos and performs joint texture-depth rate-distortion optimized substream extraction to maximize the average quality of rendered views over all 3D video streams. We proved that our algorithm has an approximation factor of $(1-\epsilon)$ and a running time complexity of $O(nS/\epsilon)$, where n is total number of layers, S is the total number of streams, and ϵ is the approximation parameter. A radio frame allocation algorithm was then presented as an extension to our algorithm to efficiently schedule the chosen substreams such that the power consumption of the receiving mobile devices is minimized without introducing any buffer overflow or underflow instances. We evaluated the performance of our algorithm using trace-based simulations of 3D videos that have different characteristics. Each of these videos is encoded into 5 quality layers. Results show that our algorithm runs much faster than enumerative algorithms for finding the optimal solution. And the returned set of substreams yields an average quality for the synthesized views that is within 0.3 dB of the optimal. Moreover, our energy efficient radio frame allocation results in schedules that reduce the average power consumption of the receivers by 86% on average.

APPENDIXES

A. BACKGROUND

A.1 Auto-Stereoscopic 3D Displays

Traditional 3D displays, including almost all commercially available 3DTV displays nowadays, require the viewer to wear special glasses that present two different images to each eye. Auto-stereoscopic displays relief the viewer from the discomfort of wearing specialized glasses by dividing the viewing space into a finite number of viewing slots where only one image (view) of the scene is visible. Each of the viewer's eyes sees a different image, and those images change as the viewer moves or changes his head position. Two-view auto-stereoscopic displays divide the horizontal resolution of the display into two sets. Every second column of pixels constitutes one image of the left- and right-image pair, while the other image consists of the rest of the columns. The two displayed images are visible in multiple zones in space. However, the viewer will perceive a correct stereoscopic image only if standing at the ideal distance and in the correct position. Moving much forward or backward from the ideal distance greatly reduces the chance of seeing a correct image.

If the two-view stereoscopic display is equipped with a head-tracking device, it can prevent incorrect *pseudoscopic* viewing by displaying the right and left images in the appropriate zones. One main disadvantage of head-tracking stereoscopic displays is that they only support a single-viewer. Moreover, they should be designed to have minimal lag so that the user does not notice the head tracking. Multiview autostereoscopic displays overcome the limitations of two-view and head-tracking stereoscopic displays by increasing the number of displayed views. Thus, they have the advantage of allowing viewers to perceive a 3D image when the eyes are anywhere within the viewing zone. This enables multiple viewers to see the 3D objects from their own point of view, which makes these displays more suitable for applications such as computer games, home entertainment, and advertising.

A.2 3D Video Representation

Multiview 3D videos can be represented explicitly or implicitly. In an explicit representation, all possible views are either coded separately (*simulcast coding*) or jointly using *multiview coding* [Vetro et al. 2011]. Using only texture information to drive multiview displays requires providing a large number of views to the display. This is not efficient since it requires transmitting a large amount of data which can exceed the network capacity. Even if multiview coding is used to encode the views, the resulting bit rate will still be proportional to the number of views [Müller et al. 2011]. Moreover, it has been shown that for common multiview coding conditions only up to 30% of the macroblocks are predicted from inter-view reference pictures [Merkle et al. 2007].

Implicit representation of multiview videos overcomes this by transmitting scene geometry information, such as depth maps, along with the texture data. This is known as the *video-plus-depth (V+D)* representation [Akar et al. 2007]. Given the scene geometry information, a high-quality view synthesis technique such as *depth image-based rendering* (DIBR) [Kauff et al. 2007; Fehn 2004] can generate any number of views, within a given range, using a fixed number of received views as input. This significantly reduces the bandwidth requirements for transmitting the 3D video, as the receiver would only need to receive a subset of the views along with their depth maps and generate the remaining views. Video-plus-depth representations also have the advantage of providing the flexibility of adjusting the depth range so that the viewer does not experience eye discomfort [Arican et al. 2009]. In addition, the video can be displayed on a wide variety of auto-stereoscopic displays with a different number of rendered views. Because of the aforementioned advantages, the free-viewpoint television working group in MPEG has chosen V+D as the 3D scene representation format. It is possible to reduce the transmitted video data even more by exploiting the redundancies between the views of the multiview

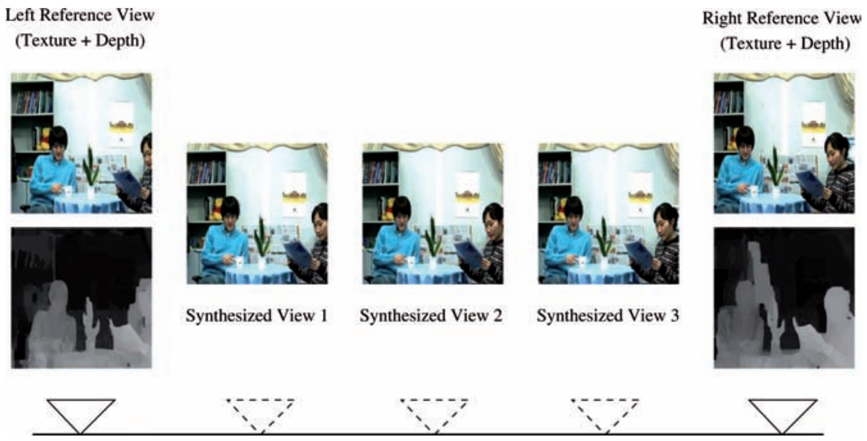


Fig. 8. Synthesizing three intermediate views using two reference views and associated depth maps. Newspaper video sequence, © GIST (used with permission).

texture streams, as well as the redundancies between the multiview depth map streams, using the multiview coding (MVC) profile of H.264/AVC [Vetro et al. 2011]. This, however, will only be acceptable for non-real-time streaming scenarios due to the high coding complexity of such encoders.

Rendering a virtual view from a single reference view and its associated depth map stream suffers from the *disocclusion* or *exposure* problem, where some regions in the virtual view have no mapping because they were invisible in the reference view. These regions are known as *holes* and require applying a filling algorithm that interpolates the value of the unmapped pixels from surrounding areas. This disocclusion effect increases as the angular distance between the reference view and the virtual view increases. Virtual views may be synthesized more correctly if two or more reference views, from both sides of the virtual view, are used [Gotfryd et al. 2008]. This is possible because areas which are occluded in one of the reference views may not be occluded in the other one. An illustrative example of this process is given in Figure 8. Each of the reference views on the left and the right have two components: a texture image and a depth map. Using the depth map and the corresponding camera parameters of the reference view, as well as the camera parameters of the target view, the texture components of the references are individually warped using a DIBR algorithm to the target view. By blending the two resulting images, an synthetic image is generated where texture information for occluded regions in either reference are obtained from the other.

A.3 Scalable Video Coding

In this article, we assume that the 3D video content is represented using multiple texture video streams, captured from different viewpoints of the scene, and their respective depth map streams. The streams are simulcast coded in order to support real-time service. We leverage scalable video coders (SVCs) that encode video content into multiple layers [Schwarz et al. 2007]. These scalable coded streams can then be transmitted and decoded at various bit rates. This can be achieved using an extractor that adapts the stream for the target rate and/or resolutions. The extractor can either be at the streaming server side, at a network node between the sender and the receiver, or at the receiver-side. In the context of this paper, the base station in a wireless video broadcasting service will be responsible for extracting the substreams to be transmitted. Each extracted substream can be rendered at a lower quality than the original (complete) stream.

B. MODELING SYNTHESIZED VIEWS DISTORTION

In order to efficiently utilize available channel bandwidth such that the quality of all rendered views at the receiver side is maximized, we need to understand the sources of distortion in a synthesized view. Four factors contribute to the distortion of a synthesized view: (i) compression of reference texture video streams and depth map streams; (ii) performance of the view synthesis algorithm; (iii) inherent inaccuracy of depth maps; and (iv) whether or not captured texture videos are well rectified, that is, have been transformed such that the two image planes are co-planar and pairs of conjugate epipolar lines become collinear and parallel to one of the image axes. Assuming a well-rectified camera system and fixing the view synthesis algorithm and depth maps, the distortion of a virtual view will only depend on the compression of texture video and depth map streams.

The quality of synthesized views can be evaluated by the mean-squared error between the synthesized image and the original image at that viewpoint. However, it is not practical in many 3D video applications to assume that the original view at the synthesized position exists. For example, autostereoscopic displays may render views at arbitrary positions along the baseline distance between any two views in the original multiview video. Thus, it is necessary to have an accurate model that represents the distortion of a synthesized image based on the distortions of reference images.

We chose the model proposed in Yuan et al. [2011] for its simplicity and low complexity. In the chosen model, the distortion of a synthesized view is represented as given in Eq. (12), where D_t^L and D_t^R are the distortions of the left texture video and the right texture video, respectively, and D_d^L and D_d^R are the distortions of the left and right depth map streams, respectively. f is the focal length of the camera, and l is baseline distance between the cameras. Parameters ρ, ζ, η, and θ are view-dependent model parameters, ω_L and ω_R are two coefficients satisfying $\omega_L + \omega_R = 1$, and M and C are constants. Ψ_L and Ψ_R are two parameters determined by the contents of the left and the right images, respectively [Mathew and Taubman 2010].

$$\begin{aligned} D_v &= \omega_L^2 D_t^L + \omega_R^2 D_t^R + \omega_L^2 \left(\frac{fl}{M}\right)^2 \Psi_L D_d^L \\ &\quad + \omega_R^2 \left(\frac{fl}{M}\right)^2 \Psi_R D_d^R + C \\ &= \rho D_t^L + \zeta D_t^R + \eta D_d^L + \theta D_d^R + C. \end{aligned} \tag{12}$$

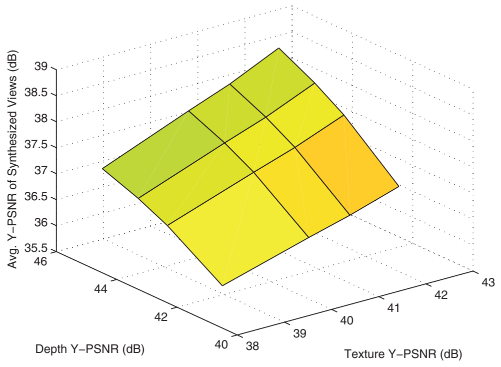
Experimental results in Yuan et al. [2011] show that D_t^L and D_t^R are similar when the two texture views are compressed by the same quantization parameter and they can be approximated by D_t, which is the average distortion of the left and right views' texture streams. Similarly, D_d^L and D_d^R can be approximated by the average distortion of the left and right depth maps, D_d. Thus, if the same quantization parameter is used for compressing the reference textures and another quantization parameter value for compressing both the left and right reference depth map streams, we get the simplified model given by Eq. (13).

$$\begin{aligned} D_v &= (\omega_L^2 + \omega_R^2) D_t + (\omega_L^2 + \omega_R^2) \left(\frac{fl}{M}\right)^2 \bar{\Psi} D_d + C \\ &= \alpha D_t + \beta D_d + C. \end{aligned} \tag{13}$$

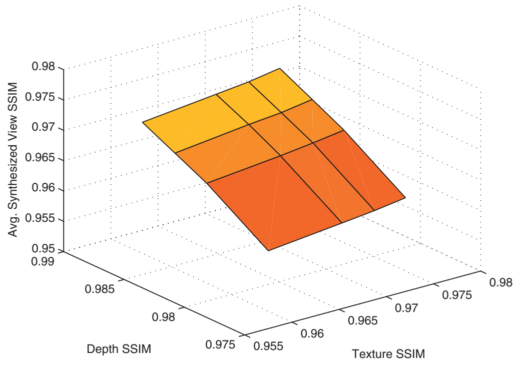
To validate that the relation between the quality of the synthesized views and the quality of the texture and depth maps of the reference views can be approximated by a linear plane, we perform the following experiment. For a set of 6 multiview video sequences and their associated depth maps, we chose two reference views which are 4 baseline distances apart from each video. We encoded 30 frames

Table II. 3D Video Sequences Used in 3D Distortion Model Validation Experiments

Sequence	Resolution	Number of Views	Reference Views	Synthesized Views
Champagne	1280 × 960	80	37, 41	38, 39, 40
Pantomime	1280 × 960	80	37, 41	38, 39, 40
Kendo	1024 × 768	7	1, 5	2, 3, 4
Balloons	1024 × 768	7	1, 5	2, 3, 4
Lovebird1	1024 × 768	12	4, 8	5, 6, 7
Newspaper	1024 × 768	9	2, 6	3, 4, 5



(a) PSNR



(b) SSIM

Fig. 9. Average quality of 3 synthesized views from decoded substreams of the Balloons sequence with respect to views synthesized from uncompressed references.

(about 1 second) of the texture and depth map streams of the chosen views using the Joint Scalable Video Model (JSVM) software [JSVM 2011]. JSVM is the reference software for scalable video coding (SVC). The encoder was configured such that the generated scalable streams contain 4 layers (one base layer and three MGS enhancement layers). A description of the sequences used in our experiment is given in Table II.

We use two video quality metrics to calculate the quality for each layer of the left and right reference streams. This is done for both texture and depth streams. The two metrics used are luminance component Peak Signal to Noise Ratio (Y-PSNR) and structural similarity (SSIM) [Wang et al. 2004]. We then synthesize three intermediate views between the two reference views using the view synthesis reference software (VSRS) [Tanimoto et al. 2008]. VSRS uses two reference views, left and right, to synthesize an intermediate virtual view by using the two corresponding reference depth maps. The distortion in views synthesized from reconstructed references is a summation of both the distortion resulting from compression and the distortion resulting from the view synthesis process. We calculate the average quality for each texture and depth map substream combination against views synthesized at the same camera position but from the uncompressed reference streams. This will demonstrate only the effect of reference views compression on the view synthesis process. A sample from our results is shown in Figure 9; results for other video sequences are similar. The figure illustrates the relationship between the average quality of the left and right reference textures, the average quality of the left and right depth maps, and average quality of the 3 synthesized views in terms of PSNR and SSIM values. As shown in the figure, all surfaces can indeed be approximated by a linear surface in the form given in Eq. (1).

C. BIT RATES AND QUALITY VALUES FOR EVALUATION TEST SEQUENCES

Table III. Data Rates (kbps) and Y-PSNR Values (dB) Representing Each Layer of the Salable Encodings of the Texture and Depth Streams

Sequence		1 Layer		2 Layers		3 Layers		4 Layers		5 Layers	
		r_1	q_1	r_2	q_2	r_3	q_3	r_4	q_4	r_5	q_5
Champagne	t	157	35.3511	284	36.4732	653	39.5048	971	40.6820	1493	42.2360
	d	64	40.2629	104	40.9074	238	43.2668	386	44.3530	650	45.6268
Pantomime	t	517	34.5877	674	35.4403	1183	38.2229	1670	39.5058	2398	41.3435
	d	119	40.6100	180	41.2614	352	42.9276	554	43.8257	896	44.8836
Kendo	t	295	35.9112	415	36.8813	771	39.4434	1121	40.6169	1697	42.0590
	d	203	38.5026	294	39.3620	546	41.5120	819	42.6939	1264	44.0096
Balloons	t	217	35.3134	342	36.4006	716	39.1762	1068	40.3533	1665	41.8603
	d	101	38.7728	162	39.6052	348	41.7705	564	42.8321	952	44.0922
Lovebird1	t	137	33.8922	282	34.9077	739	37.9702	1108	39.1853	1710	40.7982
	d	30	43.1877	52	43.7545	116	45.3736	207	46.4544	370	47.5168
Newspaper	t	168	34.7678	295	35.7765	694	38.7524	1035	39.9533	1587	41.4886
	d	79	39.1052	126	39.7935	288	41.8723	466	42.9496	787	44.2220

ACKNOWLEDGMENTS

The authors would like to thank Nagoya University/Tanimoto Laboratory, the Electronics and Telecommunications Research Institute (ETRI), and Gwangju Institute of Science and Technology (GIST) for providing the 3D video sequences used in this article.

REFERENCES

ABI RESEARCH. 2010. 3D Mobile devices. http://www.abiresearch.com/research/1006094.

AKAR, G., TEKALP, A., FEHN, C., AND CIVANLAR, M. 2007. Transport methods in 3DTV—a survey. *IEEE Trans. Circ. Syst. Video Tech. 17,* 11, 1622–1630.

ARICAN, Z., YEA, S., SULLIVAN, A., AND VETRO, A. 2009. Intermediate view generation for perceived depth adjustment of stereo video. *Appl. Digital Image Proc. XXXII 7443,* 1, 74430U.

BOSC, E., JANTET, V., PRESSIGOUT, M., MORIN, L., AND GUILLEMOT, C. 2011. Bit-rate allocation for multi-view video plus depth. In *Proceedings of the 3DTV Conference: The True Vision—Capture, Transmission and Display of 3D Video (3DTV-CON)*. 1–4.

CHEN, Y., WANG, Y.-K., UGUR, K., HANNUKSELA, M., LAINEMA, J., AND GABBOUJ, M. 2009. The emerging MVC standard for 3D video services. *EURASIP J. Adv. Sig. Proc. 2009*.

CHEUNG, G., VELISAVLJEVIC, V., AND ORTEGA, A. 2011. On dependent bit allocation for multiview image coding with depth-image-based rendering. *IEEE Trans. Image Proc. 20,* 11, 3179–3194.

CPLEX 2011. IBM ILOG CPLEX Optimizer. http://www.ibm.com/software/integration/optimization/cplex-optimizer/.

DE DIEGO BALAGUER, E., FITZEK, F., OLSEN, O., AND GADE, M. 2005. Performance evaluation of power saving strategies for DVB-H services using adaptive MPE-FEC decoding. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'05)*. Vol. 4. 2221–2226.

DE SILVA, D., EKMEKCIOGLU, E., ABDUL-HAMEED, O., FERNANDO, W., WORRALL, S., AND KONDOZ, A. 2010. Performance evaluation of 3D-TV transmission over WiMAX broadband access networks. In *Proceedings of the International Conference on Information and Automation for Sustainability (ICIAFs)*. 298–303.

DODGSON, N. 2005. Autostereoscopic 3D displays. *IEEE Computer 38,* 8, 31–36.

DYER, M. 1984. An O(n) algorithm for the multiple-choice knapsack linear program. *Mathemat. Prog. 29*, 57–63.

FEHN, C. 2004. Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV. *Stereoscop. Displ. Virt. Reality Syst. XI 5291,* 1, 93–104.

GOTFRYD, M., WEGNER, K., AND DOMAŃSKI, M. 2008. *View Synthesis Software and Assessment of Its Performance*. ISO/IEC JTC1/SC29/WG11, MPEG 2008/M15672.

HARRISON, D. 2010. Hockey Night in Canada offers games in 3D. http://www.cbc.ca/sports/hockey/story/2010/09/29/sp-hockey-night-panasonic.html.

HEFEEDA, M. AND HSU, C.-H. 2008. Energy optimization in mobile TV broadcast networks. In *Proceedings of the International Conference on Innovations in Information Technology*. 430–434.

HSU, C.-H. AND HEFEEDA, M. 2010. Achieving viewing time scalability in mobile video streaming using scalable video coding. In *Proceedings of the 1st ACM SIGMM Conference on Multimedia Systems (MMSys'10)*. 111–122.

INFORMA TELECOMS AND MEDIA. 2010. 22.5 million homes will tune into 3D by 2015. http://shop.informatm.com/itmgcontent/icoms/s/press-releases/20017774108.html.

JÄRVINEN, K., BOUAZIZI, I., LAAKSONEN, L., OJALA, P., AND RÄMÖ, A. 2010. Media coding for the next generation mobile system LTE. *Comput. Comm. 33,* 16, 1916–1927.

JSVM 2011. Joint Scalable Video Model (JSVM)—Reference Software. http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm.

KAUFF, P., ATZPADIN, N., FEHN, C., MÜLLER, M., SCHREER, O., SMOLIC, A., AND TANGER, R. 2007. Depth map creation and image-based rendering for advanced 3DTV services providing interoperability and scalability. *Sig. Proc. Image Comm. 22,* 2, 217–234.

KELLERER, H., PFERSCHY, U., AND PISINGER, D. 2004. *Knapsack Problems*. Springer-Verlag.

KIM, W., ORTEGA, A., LAI, P., TIAN, D., AND GOMILA, C. 2010. Depth map coding with distortion estimation of rendered view. *Proc. SPIE 7543*.

KIMATA, H., KITAHARA, M., KAMIKURA, K., YASHIMA, Y., FUJII, T., AND TANIMOTO, M. 2004. System design of free viewpoint video communication. In *Proceedings of the IEEE International Conference on Computer and Information Technology*. 52–59.

KUMAR, A. 2008. *Mobile Broadcasting with WiMAX: Principles, Technology, and Applications*. Elsevier Inc.

LIU, Y., HUANG, Q., MA, S., ZHAO, D., AND GAO, W. 2009. Joint video/depth rate allocation for 3D video coding based on view synthesis distortion model. *Signal Proc. Image Comm. 24,* 8, 666–681.

MASTERIMAGE. 2012. MasterImage 3D Autostereoscopic 3D LCD. http://masterimage3d.com/products/3d-lcd.

MATHEW, R. AND TAUBMAN, D. 2010. Quad-tree motion modeling with leaf merging. *IEEE Trans. Circ. Syst. Video Tech. 20,* 10, 1331–1345.

MERKLE, P., SMOLIC, A., MÜLLER, K., AND WIEGAND, T. 2007. Efficient prediction structures for multiview video coding. *IEEE Trans. Circ. Syst. Video Tech. 17,* 11, 1461–1473.

MÜLLER, K., MERKLE, P., AND WIEGAND, T. 2011. 3-D video representation using depth maps. *Proc. IEEE 99,* 4, 643–656.

MULTIMEDIA SCALABLE 3D FOR EUROPE (MUSCADE) PROJECT. 2010. Mission scenarios and service requirements. http://www.muscade.eu/deliverables/MUS.RP.00001.EBU%20D1.1.1.pdf.

PETROVIC, G., DO, L., ZINGER, S., AND DE WITH, P. H. N. 2010. Virtual view adaptation for 3d multiview video streaming. *Stereoscopic Displays and Appl. XXI 7524,* 1, 752410.

PHILIPS ELECTRONICS. 2012. Philips WOWvx Autostereoscopic Display. http://www.business-sites.philips.com/3dsolutions/home/index.page.

PILKINGTON, E. 2010. ESPN viewers can watch World Cup matches in 3D—at a price. http://www.guardian.co.uk/technology/2010/jan/06/espn-3d-world-cup-ces.

SCHWARZ, H., MARPE, D., AND WIEGAND, T. 2007. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Trans. Circ. Syst. Video Tech. 17,* 9, 1103–1120.

SEQUANS COMMUNICATIONS. 2007. Datasheet: SQN1130 System-on-Chip for WiMAX Moblie Stations. http://www.sequans.com/products-solutions/mobile-wimax/sqn1130/.

SHAO, F., JIANG, G., YU, M., CHEN, K., AND HO, Y.-S. 2012. Asymmetric coding of multi-view video plus depth based 3-D video for view rendering. *IEEE Trans. Multimed. 14,* 1, 157–167.

SU, G.-M., LAI, Y.-C., KWASINSKI, A., AND WANG, H. 2011. 3D video communications: Challenges and opportunities. *Int. J. Comm. Syst. 24,* 10, 1261–1281.

TANIMOTO, M., FUJII, T., SUZUKI, K., FUKUSHIMA, N., AND MORI, Y. 2008. *Reference Softwares for Depth Estimation and View Synthesis*. ISO/IEC JTC1/SC29/WG11, MPEG2008/M15377.

UEHARA, S., HIROYA, T., KUSANAGI, H., SHIGEMURA, K., AND ASADA, H. 2008. 1-inch diagonal transflective 2D and 3D LCD with HDDP arrangement. In *Proc. SPIE. 6803*. 68030O.

UREY, H., CHELLAPPAN, K. V., ERDEN, E., AND SURMAN, P. 2011. State of the art in stereoscopic and autostereoscopic displays. *Proc. IEEE 99,* 4.

VETRO, A., MATUSIK, W., PFISTER, H., AND XIN, J. 2004. Coding approaches for end-to-end 3D TV systems. In *Proceedings of the 23rd Picture Coding Symposium (PCS'04)*. 319–324.

VETRO, A., WIEGAND, T., AND SULLIVAN, G. 2011. Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 AVC standard. *Proc. IEEE 99,* 4, 626–642.

WANG, Z., BOVIK, A., SHEIKH, H., AND SIMONCELLI, E. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Proc. 13,* 4, 600–612.

WEARDEN, G. 2010. 3D TV dominates IFA electronics show. http://www.guardian.co.uk/technology/blog/2010/sep/02/3d-television-ifa-2010.

YANG, X., SONG, Y., OWENS, T., COSMAS, J., AND ITAGAKI, T. 2004. Performance analysis of time slicing in DVB-H. In *Proceedings of the Joint Ist Workshop on Mobile Future, and the Symposium on Trends in Communications (SympoTIC'04)*. 183–186.

YUAN, G., ZHANG, X., WANG, W., AND YANG, Y. 2010. Carrier aggregation for LTE-advanced mobile communication systems. *IEEE Comm. Mag. 48,* 2, 88–93.

YUAN, H., CHANG, Y., HUO, J., YANG, F., AND LU, Z. 2011. Model-based joint bit allocation between texture videos and depth maps for 3-D video coding. *IEEE Trans. Circ. Syst. Video Tech. 21,* 4, 485–497.

ZEMEL, E. 1984. An O(n) algorithm for the linear multiple choice knapsack problem and related problems. *Inf. Proc. Lett. 18*, 123–128.

Received January 2012; revised May 2012; accepted May 2012