Multicasting of Multiview 3D Videos over Wireless Networks

Ahmed Hamza
School of Computing Science
Simon Fraser University
Surrey, BC, Canada
aah10@cs.sfu.ca

Mohamed Hefeeda
Qatar Computing Research Institute
Qatar Foundation
Doha, Qatar
mhefeeda@qf.org.qa

ABSTRACT

In this paper, we consider a 4G wireless network in which a number of 3D videos represented in two-view plus depth format and encoded using scalable video coders are multicasted to a group of users. 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. Our simulation-based experimental results show that our algorithm provides solutions which are within 0.3 dB of the optimal solutions while satisfying real-time requirements of multicast systems.

1. INTRODUCTION

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 (3-D) videos are the next natural step in the evolution of digital media technologies. We address the problem of maximizing the video quality of rendered views in auto-stereoscopic displays [1], [2] for mobile receivers such as smartphones and tablets. 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. 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 consider multicasting multiview video streams in which the textures and depth maps of the views are independently coded using the scalable video coding extension of H.264/AVC. We perform joint texture-depth rate-distortion optimized substream extraction in order to minimize the distortion in the views rendered at receivers. 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. We conduct experiments using 24 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 results show that the proposed algorithm produces near optimal results and terminates in a few milliseconds.

The rest of this paper is organized as follows. In Section 2, we provide a brief background on 3D display technologies and representation formats, and the concept of scalable video coding. Section 3 summarizes the related work in the literature. We state the optimal 3D video multicasting problem and describe the proposed algorithm to solve the problem in Section 4. We present our experimental evaluation in Section 5, and we conclude the paper in Section 6.

MoVid'12 , February 24, 2012, Chapel Hill, North Carolina, USA.

2. BACKGROUND

Autostereoscopic displays. Autostereoscopic 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 autostereoscopic displays divide the horizontal resolution of the display into two sets. The two displayed images are visible in multiple zones in space. To prevent incorrect *pseudoscopic* viewing, multiview autostereoscopic displays increase 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 [1]. 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.

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* [3]. Using only texture information to drive multiview displays requires transmitting large amounts of data which can exceed the network capacity. Implicit representations overcome 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 [4]. Given the scene geometry information, a high quality view synthesis technique such as *depth image-based rendering (DIBR)* [5] can generate any number of views, within a given range, using

a fixed number of received views as input. 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 are interpolated from surrounding areas using a filling algorithm. The 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 [6]. This is possible because areas which are occluded in one of the reference views will not be occluded in the other one.

Scalable Video Coding. In this paper 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 [7]. 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.

3. RELATED WORK

Liu et al. [8] propose a distortion model to characterize the view synthesis quality without requiring the original reference image. In this model, the distortion of a synthesized view is composed of 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. [9] propose an alternative and concise low-complexity distortion model for the synthesized view. Kim et al. [10] also attempt 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 [9] and use it to solve the multiple 3D video multicasting problem.

The works most related to ours are [11] and [12]. In [11], Petrovic et al. 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 can be computationally expensive. In [12], Cheung et al. address 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 [12] 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.

4. PROBLEM FORMULATION AND SOLUTION

4.1 Problem Formulation

We consider a wireless multicast/broadcast service in 4G wireless networks, such as evolved multicast broadcast multimedia services (eMBMS) in LTE networks and multicast broadcast service (MBS) in WiMAX, streaming multiple 3D videos in MVD2 representation. 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, i.e., 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, e.g., 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 symbols used in the following formulation are listed in Table 1. Assuming there are S multiview-plus-depth video streams where two reference views are picked for transmission from each video. All the 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. We assume 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 L be the number of layers for each stream. 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 $l = 1, 2, ..., L$. 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 [7] and the base station is media-aware, this information can be obtained directly

Table 1: List of Symbols Used in this Paper.

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
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

from the encoded video stream itself using the Supplementary Enhancement Information (SEI) messages.

Let $\mathbf{I}$ be the set of possible intermediate views which can be synthesized at the receiver for a given 3D video. 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 blocks of size $b_{sl} = \lceil r_{sl} \cdot \delta / F \rceil$. We use a recent linear virtual view distortion model presented in [9] to represent the quality of the synthesized view in terms of the qualities of reference views. 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, e.g., using MATLAB's Surface Fitting Toolbox.

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

We have experimentally validated this relation using both the luminance component Peak Signal to Noise Ratio (Y-PSNR) and structural similarity (SSIM) [13] video quality metrics. Details are omitted due to space limitations. We now 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. It can be seen that the substream selection problem is equivalent to the Multiple Choice Knapsack Problem (MCKP), which is known to be NP-Complete [14]. 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. 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.

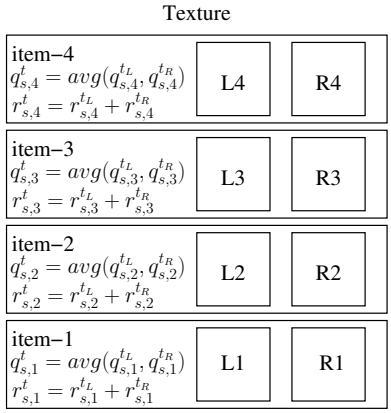


Figure 1: Calculating profits and costs for texture component substreams of the reference views.

$$\text{Maximize} \quad \frac{1}{S}\sum_{s\in\mathbf{S}}\frac{1}{I}\sum_{i\in\mathbf{I}}\left(\alpha^i_s\sum_{l=1}^{L}x^t_{sl}q^t_{sl} + \beta^i_s\sum_{l=1}^{L}x^d_{sl}q^d_{sl}\right) \tag{P1}$$

$$\text{such that} \quad \sum_{s=1}^{S}\left(\sum_{l=1}^{L}x^t_{sl}b^t_{sl} + \sum_{l=1}^{L}x^d_{sl}b^d_{sl}\right) \le P \tag{P1a}$$

$$\sum_{l=1}^{L}x^t_{sl} = 1, \quad s = 1, ..., S, \tag{P1b}$$

$$\sum_{l=1}^{L}x^d_{sl} = 1, \quad s = 1, ..., S, \tag{P1c}$$

$$x^t_{sl}, x^d_{sl} \in \{0, 1\} \tag{P1d}$$

4.2 Proposed Solution

We propose 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 Figure 2. First, we calculate a single coefficient for the decision variables in the objective function. For variables associated with the texture component we have $\hat{q}^t_{sl} = q^t_{sl}\sum_{i\in\mathbf{I}}\alpha^i_s$. Similarly, the coefficient for depth component variables is $\hat{q}^d_{sl} = q^d_{sl}\sum_{i\in\mathbf{I}}\beta^i_s$. We then find an upper bound on the optimal solution value in order to reduce the search space. This

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 **S**
Input: Model parameters for each virtual view position of each video α_s^i, β_s^i
Input: Approximation factor ϵ
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 $(1-\epsilon)z^*$

Figure 2: Proposed S3VM algorithm.

is achieved by solving the linear program relaxation of the MCKP. A linear time partitioning algorithm for solving the LP-relaxed MCKP exists. It is based on the works of Dyer [15] and Zemel [16] and does not require any pre-processing of the classes, such as expensive sorting operations. We note that a class in Dyer-Zemel [14] represents one of the two components (texture or depth) of a given 3D video in our problem. Dyer-Zemel is an iterative algorithm and the number of classes available at the beginning of an iteration changes from one iteration to another as the algorithm proceeds. Thus, at the beginning of S3VM we have $2S$ classes.

An optimal solution x^{LP} to the linear relaxation of the MCKP satisfies the following properties: (1) x^{LP} has at most two fractional variables; and (2) if x^{LP} has two fractional variables, they must be from the same class. When there are two fractional variables, one of the items 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 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 (the stream to which the split substream belongs) and the sum of the qualities of the substreams with the smallest number of blocks in each of the other streams [14]. 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 $1,...,g$, where $1 \leq g \leq 2S$, such that the total quality of the selected substreams is q. For all components $g \in \{1,...,2S\}$ and all quality values $q \in \{0,...,2z^h\}$, we construct a table where the cell values are $B(g,q)$ for the corresponding g and q values. If no solution with total quality of q exists, $B(g,q)$ is set to ∞. Initializing $B(0,0) = 0$ and $B(0,q) = \infty$ for $q = 1,...,2z^h$, the values for classes $1,...,g$ are calculated for $g = 1,...,2S$ and $q = 1,...,2z^h$ using the recursion shown in Eq. (2).

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

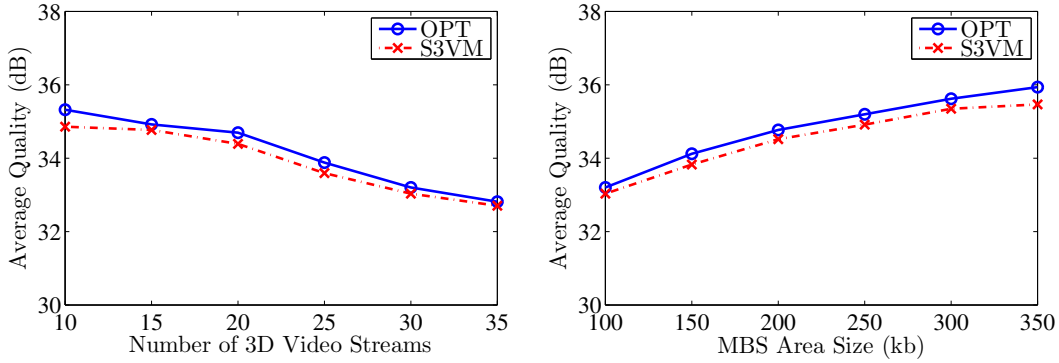
The value of the optimal solution is given by Eq. (3). To obtain the solution vector for the substreams to be transmitted, we perform backtracking from the cell containing the optimal value.

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

5. EVALUATION

5.1 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 from the MPEG 3DV ad-hoc group data set: *Champagne Tower*, *Pantomime*, *Kendo*, *Balloons*, *Lovebird1*, and *Newspaper*. We divide each sequence into four 60-frame (2-sec) 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 [17] into one base layer and four medium grain scalability (MGS) layers. The QP 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. For each texture-depth quality combination, three intermediate views are synthesized using VSRS 3.5 [18]. 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 non-compressed 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



(a) Quality vs. Number of Streams (b) Quality vs. MBS Area Size

Figure 3: Average quality of solutions obtained using the S3VM algorithm versus optimal solutions.

scheme [19]. The typical frame duration in Mobile WiMAX is 5 ms. Thus, for a 1-sec 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 CPLEX LP/MIP solver [20]. The two performance metrics used in our evaluation are: *average video quality* (over all synthesized views and all streams), and *running time*.

5.2 Results

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. Moreover, 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 to be transmitted.

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 that of the optimal solution.

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 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. Next, we fix the number of videos at 30 streams and the MBS area size was varied from 100 kb to 350 kb. Results indicate that the running time of our algorithm is still significantly less than that of the optimum solution, 6.6 times faster on average.

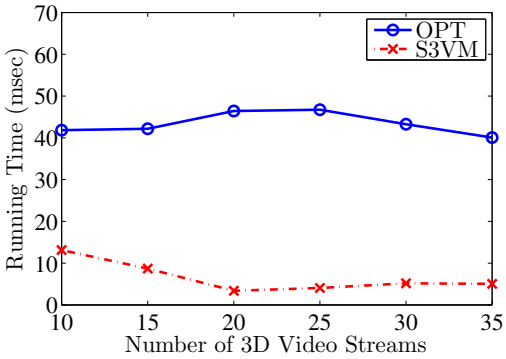


Figure 4: Average running times for a variable number of video streams.

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. Increasing the value of the approximation parameter results in significantly faster running times, 2.3 to 4.7 times faster than running time for obtaining optimal solution using CPLEX. 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.

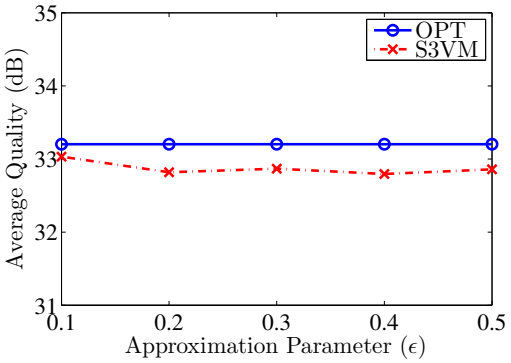


Figure 5: Impact of approximation parameter value on obtained video quality.

6. 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 evaluated the performance of our algorithm by trace-based simulations using traces from six 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 returned set of substreams yields an average synthesized views quality that is within 0.3 dB of the optimal.

7. ACKNOWLEDGEMENTS

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

8. REFERENCES

[1] N. Dodgson, "Autostereoscopic 3D displays," *Computer*, vol. 38, no. 8, pp. 31–36, Aug. 2005.

[2] H. Urey, K. V. Chellappan, E. Erden, and P. Surman, "State of the art in stereoscopic and autostereoscopic displays," *Proceedings of the IEEE*, vol. 99, no. 4, 2011.

[3] A. Vetro, T. Wiegand, and G. Sullivan, "Overview of the stereo and multiview video coding extensions of the H.264/MPEG-4 AVC standard," *Proceedings of the IEEE*, vol. 99, no. 4, pp. 626 –642, April 2011.

[4] G. Akar, A. Tekalp, C. Fehn, and M. Civanlar, "Transport methods in 3DTV - a survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1622–1630, Nov. 2007.

[5] C. Fehn, "Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV," *Stereoscopic Displays and Virtual Reality Systems XI*, vol. 5291, no. 1, pp. 93–104, 2004.

[6] M. Gotfryd, K. Wegner, and M. Domański, *View synthesis software and assessment of its performance*, ISO/IEC JTC1/SC29/WG11, MPEG 2008/M15672, Hannover, Germany, July 2008.

[7] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, Sept. 2007.

[8] Y. Liu, Q. Huang, S. Ma, D. Zhao, and W. Gao, "Joint video/depth rate allocation for 3D video coding based on view synthesis distortion model," *Signal Processing: Image Communication*, vol. 24, no. 8, pp. 666–681, Sept. 2009.

[9] H. Yuan, Y. Chang, J. Huo, F. Yang, and Z. Lu, "Model-based joint bit allocation between texture videos and depth maps for 3-D video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 4, pp. 485–497, April 2011.

[10] W. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map coding with distortion estimation of rendered view," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 7543, January 2010.

[11] G. Petrovic, L. Do, S. Zinger, and P. H. N. de With, "Virtual view adaptation for 3d multiview video streaming," *Stereoscopic Displays and Applications XXI*, vol. 7524, no. 1, p. 752410, 2010.

[12] G. Cheung, V. Velisavljevic, and A. Ortega, "On dependent bit allocation for multiview image coding with depth-image-based rendering," *IEEE Transactions on Image Processing*, vol. 20, no. 12, Dec. 2011, to appear.

[13] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[14] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Springer-Verlag, 2004.

[15] M. Dyer, "An O(n) algorithm for the multiple-choice knapsack linear program," *Mathematical Programming*, vol. 29, pp. 57–63, 1984.

[16] E. Zemel, "An O(n) algorithm for the linear multiple choice knapsack problem and related problems," *Information Processing Letters*, vol. 18, pp. 123–128, March 1984.

[17] "Joint Scalable Video Model (JSVM) - Reference Software," http://ip.hhi.de/imagecom_G1/savce/downloads/SVC-Reference-Software.htm.

[18] M. Tanimoto, T. Fujii, K. Suzuki, N. Fukushima, and Y. Mori, *Reference Softwares for Depth Estimation and View Synthesis*, ISO/IEC JTC1/SC29/WG11, MPEG2008/M15377, Archamps, France, April 2008.

[19] A. Kumar, *Mobile Broadcasting with WiMAX: Principles, Technology, and Applications*. Elsevier Inc., 2008.

[20] "IBM ILOG CPLEX Optimizer," http://www.ibm.com/software/integration/optimization/cplex-optimizer/.