Depth Personalization and Streaming of Stereoscopic Sports Videos

KIANA CALAGARI, Simon Fraser University
TAREK ELGAMAL, Qatar Computing Research Institute
KHALED DIAB, Simon Fraser University
KRZYSZTOF TEMPLIN, Massachusetts Institute of Technology
PIOTR DIDYK, MMCI, Saarland University
WOJCIECH MATUSIK, Massachusetts Institute of Technology
MOHAMED HEFEEDA, Qatar Computing Research Institute

Current three-dimensional displays cannot fully reproduce all depth cues used by a human observer in the real world. Instead, they create only an illusion of looking at a three-dimensional scene. This leads to a number of challenges during the content creation process. To assure correct depth reproduction and visual comfort, either the acquisition setup has to be carefully controlled or additional postprocessing techniques have to be applied. Furthermore, these manipulations need to account for a particular setup that is used to present the content, for example, viewing distance or screen size. This creates additional challenges in the context of personal use when stereoscopic content is shown on TV sets, desktop monitors, or mobile devices. We address this problem by presenting a new system for streaming stereoscopic content. Its key feature is a computationally efficient depth adjustment technique which can automatically optimize viewing experience for videos of field sports such as soccer, football, and tennis. Additionally, the method enables depth personalization to allow users to adjust the amount of depth according to their preferences. Our stereoscopic video streaming system was implemented, deployed, and tested with real users.

CCS Concepts: • **Information systems → Multimedia streaming**; • **Computing methodologies → Image processing**

Additional Key Words and Phrases: 3D Video streaming, 3d video, stereoscopic retargeting, depth personalization, video storage systems

ACM Reference Format:
Kiana Calagari, Tarek Elgamal, Khaled Diab, Krzysztof Templin, Piotr Didyk, Wojciech Matusik, and Mohamed Hefeeda. 2016. Depth personalization and streaming of stereoscopic sports videos. ACM Trans. Multimedia Comput. Commun. Appl. 12, 3, Article 41 (March 2016), 23 pages.
DOI: http://dx.doi.org/10.1145/2890103

1. INTRODUCTION

Comparing to standard two-dimensional (2D) videos, stereoscopic three-dimensional (S3D) videos provide more entertaining and immersive experience [Freeman and Avons 2000]. Due to the significant interest in such content, most of the recent big movie

This research was supported in part by Qatar Computing Research Institute (QCRI).

Authors' addresses: K. Calagari and K. Diab, School of Computing Science, Simon Fraser University, Burnaby, BC, Canada; emails: {kcalagar, kdiab}@sfu.ca; T. Elgamal and M. Hefeeda, Qatar Computing Research Institute, Doha, Qatar; emails: {tgamal, mhefeeda}@qf.org.qa; P. Didyk, Saarland University, Saarbrücken, Germany; email: pdidyk@mmci.uni-saarland.de; K. Templin and W. Matusik, Massachusetts Institute of Technology, Cambridge, MA, USA; emails: {ktemplin, wojciech}@csail.mit.edu. Most of this work was done while K. Calagari and K. Diab were interns at QCRI.

DOI: http://dx.doi.org/10.1145/2890103

productions are either shot in 3D or converted to their stereoscopic versions in post-production. At the same time, S3D displays have become a commodity. Most of the off-the-shelf TV sets are 3D ready. Also, several laptops and mobile devices are equipped with stereoscopic displays [Displaybank Co. 2010]. This trend of incorporating stereoscopic technology into home entertainment systems may be easily hampered by the challenges in streaming S3D videos. The main problem is that S3D streaming systems must be capable of serving stereoscopic content to a wide range of display devices that are used in uncontrolled conditions. As both depth perception and visual comfort highly depend on viewing conditions (for example, display size and viewing distance) [Tam et al. 2011; Thorpe and Russell 2011; Shibata et al. 2011], streaming one version of the content (e.g., a movie theater copy) is suboptimal. In addition, users have different preferences in terms of how much of the stereoscopic effect should be present in the content. This suggests that additional methods that enable easy content personalization are a necessary feature of any streaming system.

To address these issues, we propose *Anahita*—a system for online stereoscopic 3D video streaming. In contrast to previous systems, Anahita allows users to personalize S3D content based on their own preferences, which significantly improves their viewing experience. To the best of our knowledge, this is the first S3D streaming system with such capabilities. In particular, the contributions of this article are as follows:

- A new system design for adaptive S3D video streaming: The goal of this system is to optimize stereoscopic 3D videos for a wide range of display sizes, video representations, viewers' preferences, and network conditions. The system efficiently organizes the creation of various versions of 3D videos using a structure that we refer to as the 3D version tree. The system uses the Dynamic Adaptive Streaming over HTTP (DASH) to dynamically switch among different versions and optimize the quality and depth perception for different viewers.
- A novel method for depth expansion and compression for stereoscopic 3D videos: Our method performs simple image processing operations, it does not require creating accurate depth maps, and it does not introduce visual artifacts. The main target application for our method is sports videos, for example, soccer, football, tennis, and cricket. In such applications, preserving the scene structure, for example, the straightness of the field plane and the white lines, is extremely important. Our method guarantees preservation of the scene structure, because we employ linear operations to map the original image coordinates to new ones.
- A method for depth personalization, which allows users to control the depth in S3D videos to suit their comfort level. This is an important feature as not all users prefer the same amount of depth: some users do not like depth at all, while others are fascinated by 3D content with striking and aggressive depths.
- A complete end-to-end implementation and user studies to evaluate the benefits of the proposed system: We implement the server side of our system and deploy it on the Amazon cloud. The server implements our depth customization method as well as several off-the-shelf video processing operations to adapt S3D videos. In addition, we provide multiple clients on 3D-enabled mobile phones, tablets, desktop displays, and large TV displays. Our subjective studies show that significant gain in the depth quality can be achieved by the proposed system.

A preliminary version of this work was published in Calagari et al. [2014]. Here we propose an extended version of that work which (i) proposes the concept of depth personalization and (ii) provides more rigorous analysis and experimental results on different system aspects.

The rest of this article is organized as follows. In Section 2, we summarize previous efforts in academia and industry in designing 3D video streaming systems. We also describe different 3D content customization methods and how our method differs from them. Then we present the design of our system (Section 3) and the depth customization method (Section 4). Section 5 discusses details of the implementation. In Section 6, we present our user studies which evaluate our system. Section 7 concludes the article.

2. RELATED WORK

There has been significant interest both in academia and in industry in 3D video processing. However, as discussed below, the problem of depth customization for different display sizes and types has not received much attention. In addition, in contrast to previous methods, our depth manipulation method does not rely on accurate depth maps, it does not introduce any significant distortion to the videos, and it is computationally inexpensive.

2.1. 3D Streaming Systems

Multiple systems for 3D content streaming have been proposed in the literature. The Advanced three-dimensional television system technologies (ATTEST) project [Redert et al. 2002] aims to provide the full pipeline of 3D-TV broadcasting (e.g., content generation, coding and transmission, displays, and perceptual evaluation) while being backward compatible with 2D. ATTEST considered certain input formats (i.e., V+D) captured by special devices, and it employed DIBR methods for depth customization [Fehn 2003]. Xin et al. [2012] describe a 3D video streaming system, but their main focus is on the 3D media encoder and decoder. Carballeira et al. [2012] also focus on encoding and propose a framework to analyse and reduce the encoding latency of multiview videos. Gurler et al. [2011] discuss 3D formats and coding for different streaming architectures and consider rate adaptation for P2P multiview streaming and selective-view streaming. Vetro et al. [2011] focus on compression and representation of multiview video. Diab et al. [2014] focus on optimizing the storage in 3D streaming systems. 3D teleconferencing has been also proposed. For example, Johanson [2001] focuses on extending the transport protocol to associate left and right views. In addition, multiview client-server systems, where a scene can be displayed from different viewpoints, has been considered, for example, in Lou et al. [2005], Kimata et al. [2011], and Hamza and Hefeeda [2014].

In addition to the academic works mentioned above, there has been significant interest from the industry, including YouTube, 3DVisionLive, Trivido, and 3DeeCentral. YouTube [YouTube] supports multiple 3D formats including side by side, anaglyph (red-cyan, blue-yellow, or green-magenta), and row and column interleaved. However, unlike our system, it does not customize or change the video depth for different displays. 3DVisionLive [2015] is a web channel for 3D photo sharing and 3D video streaming, but it is limited to only one display technology. Trivido [2015] is a 3D Internet video platform which supports side by side, anaglyph, row interleaved, and the 3D NVIDIA Vision format. However, the problem of content customization for different displays has not been addressed in this platform. 3DeeCentral [2015] supports 3D content on multiple 3D-enabled devices. However, depth adjustment for different displays is not provided.

In summary, previous works on 3D streaming have addressed various problems including content representation, content acquisition, encoding, and transmission of both stereoscopic and multiview content. However, we are not aware of any 3D streaming system that adaptively customizes the depth based on the display technology, display size, and viewer's preferences.

2.2. Depth Customization

There are two basic requirements that stereoscopic content should meet in order to be comfortable to watch. First, every object in the scene should fit within the "comfort

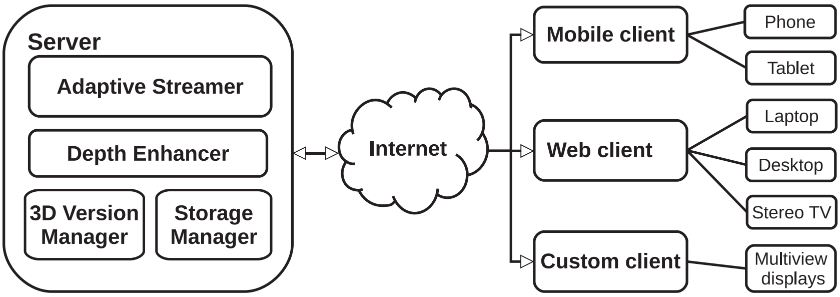


Fig. 1. Design of the proposed 3D streaming system (Anahita), which offers depth optimization of 3D videos viewed on different displays. It also introduces the concept of *depth personalization* in DASH 3D streaming systems, in which users can increase/decrease the depth based on their personal preferences.

zone," that is, it cannot be too far from the screen plane [Shibata et al. 2011]. A rule of thumb used by stereographers (known as the *percentage rule*) suggests that the pixel disparity should not be greater than 2–3% of the screen width [Shibata et al. 2011]. Second, nearby objects (in the xy plane) cannot be too distanced from each other in the z direction, that is, the disparity should be within the limit known as Panum's fusional area [Burt and Julesz 1980]; otherwise, the observer will not be able to fuse one of the objects. Meeting these constraints is dependent on two stereoscopic parameters: the camera separation and the convergence. Camera separation influences the *range* of depth (and thus distances between objects), while convergence influences the *placement* of that range relative to the screen plane. Oskam et al. [2011] developed a method that optimizes these parameters in real time. However, it is limited to synthetic content and cannot be applied as a postprocess.

Modifying the convergence in postproduction is relatively easy and can be accomplished by simply shifting either one or both views. Fixing the camera separation, however, is more difficult, since it requires synthesizing new camera views. Lang et al. [2010] show how this, and even more sophisticated operations, can be accomplished via nonlinear disparity mapping. Unfortunately, their method relies to a large extent on stereo correspondences, and it requires recovering pixel values for points that have not been registered by the camera. Therefore, it is difficult to reliably generate clean, artifact-free output without manual intervention, not to mention real-time performance. Furthermore, nonlinear disparity mapping can severely degrade the quality of videos of field sports such as soccer, due to objectionable curving of the lines. Depth can also be manipulated as a consequence of depth compression performed to limit the bandwidth of 3D content. Although some of these techniques can adapt to different viewing conditions [Wu et al. 2011; Pajak et al. 2014], their primary goal is to maintain the original depth. In contrast, our technique intentionally modifies the depth to enhance viewer experience.

3. SYSTEM ARCHITECTURE

Anahita,[1] the proposed 3D streaming system, provides depth-optimized videos to clients with different 3D display types and sizes. As depicted in Figure 1, the proposed 3D streaming system consists of (i) a server that processes the stereoscopic content and creates an optimized version for each display based on its size and technology and

[1] In ancient times, Anahita was the source of all clean water streams that flowed through golden channels to all lands and oceans on Earth. In the Internet age, Anahita is the source of all high-quality 3D video streams that flow through network channels to all types of displays.

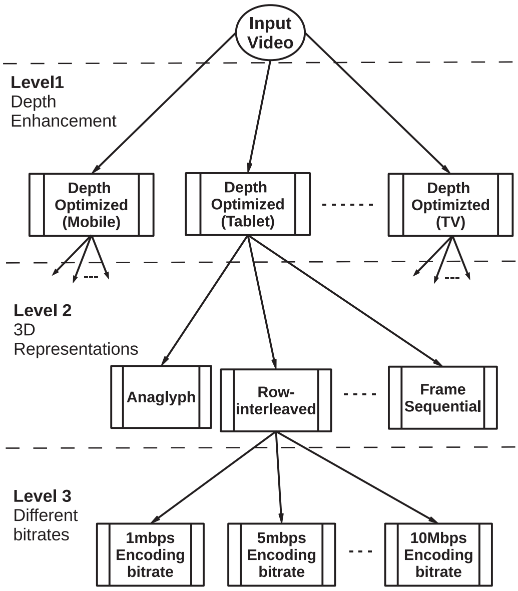


Fig. 2. 3D Version Tree. It serves as an execution or master plan to create different versions of each 3D video.

(ii) multiple clients from which 3D devices request display-optimized versions and receive the corresponding ones. The following subsections provide more details.

3.1. Server Side

The server adjusts the depth of 3D videos and adaptively streams them to clients upon their requests. It also creates and stores multiple versions of the 3D videos to cater different displays requirements. In particular, the server has four main components: (i) Depth Enhancer, (ii) 3D Version Manager, (iii) Storage Manager, and (iv) Adaptive Streamer.

The Depth Enhancer customizes the depth of original 3D videos based on the target displays. It can either increase or decrease the amount of perceived depth in the video by employing lightweight image processing methods. This is described in more detail in Section 4.

The 3D Version Manager creates different versions of the same 3D video. This is done through what we call a 3D Version Tree, which is shown in Figure 2. We note that current 2D streaming systems, for example, YouTube, typically store about 20 versions of the same 2D video but with different bitrates and formats to accommodate the variability in network bandwidth of different clients. The proposed 3D streaming system, however, renders more complexity with creation of customized 3D versions. The 3D Version Tree consists of more levels than typical 2D streaming systems. This is because, in addition to handling network bandwidth variations, clients requesting 3D videos use different displays in terms of size, depth rendering method, and the amount of depth that can be perceived. As a result, the 3D versions count in the proposed system is much larger than traditional 2D streaming systems.

The 3D version tree in Figure 2 shows the creation order of all different 3D versions. Specifically, the input video is assumed to be in the side-by-side format, which is currently the most commonly used 3D representation. The proposed depth enhancement method creates up to D versions with different depth values, where D is the number of depth levels supported by the system. These D versions represent the first level of the version tree. In this level, all versions are still in the side-by-side format. In the second level of the tree, various video conversion methods are applied in order to support displays with different depth rendering technologies, which include anaglyph, frame interleaved, row interleaved, column interleaved, and video-plus-depth. For each of the D versions in level 1, up to R versions are created in level 2, where R is the number of different 3D video representations supported by the system. In the third level of the 3D version tree, we create up to B versions for each version in level 2 to accommodate clients with heterogeneous and varying network bandwidth, where B is the number of different bitrates (qualities) offered by the system.

We note that the depth enhancement is more computationally expensive than other operations. We minimize the number of depth enhancement invocations by applying it only in the first level in the tree. Further, the depth enhancement method should be applied on the original, side-by-side, video in order to create consistent and optimized depth perception regardless of the different 3D video representations.

The third component of Anahita is the Storage Manager, which manages the storage of different versions. The proposed 3D version tree can create different 3D versions up to $D \times R \times B$ in order to support optimized 3D video quality on displays with different sizes, 3D video representations, and dynamic network conditions. With current display technologies and sizes, the parameters D, R, and B are roughly in the ranges [5–20], [5–7], and [10–20]. That is, more than 200 versions of the same 3D video are required to support the large diversity of clients. Creating and storing all versions for all videos may waste storage and processing resources of the system, especially for videos with low popularity.

Adaptive Streamer, the last component of our system, adopts the Dynamic and Adaptive Streaming over the HTTP (DASH) protocol [ISO/IEC 2012; Sodagar 2011; Stockhammer 2011]. DASH enables the server to scale to many concurrent sessions, uses commodity HTTP servers, facilitates switching among different versions, and provides wide client accessibility because it uses the standard HTTP protocol, which is allowed in most networks. The server side of the DASH protocol divides a video into a sequence of small video segments. Current 2D streaming systems that use DASH create few versions of each segment at different bitrates. In our 3D streaming system, however, we create different versions of each segment using the 3D version tree to adapt not only to the network bandwidth but also to different display sizes and technologies.

Finally, using the proposed depth expansion/compression method (Section 4) and the adaptive nature of DASH streaming, we propose the feature of *depth personalization*, which allows adjustment of the depth of a video based on the preferences of individual viewers. This is a desirable feature, as it improves the engagement of viewers and allows them to choose the most visually comfortable depth perception.

3.2. Client Side

As shown in Figure 1, Anahita supports multiple client platforms, including (i) mobile, (ii) web-based, and (iii) custom. These clients implement the client side of the DASH protocol to adaptively request segments of the different versions of 3D videos stored at the streaming server. Specifically, the client starts by requesting the manifest file from the server. The manifest file contains metadata about the requested 3D video and its available versions. Then, based on the display characteristics and the current network conditions, the client decides on the most suitable 3D version. The client then starts requesting video segments from the server and the media player is notified to start playing these segments.

For the mobile client, we developed an application that works on various mobile devices such as smartphones and tablets. Most 3D-enabled mobile devices use

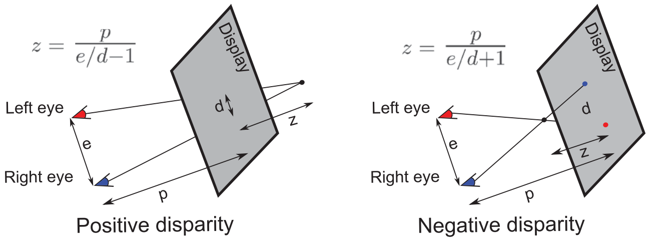


Fig. 3. Disparity (d) vs. perceived depth (z). p is the viewing distance, and e is the inter-ocular distance.

autostereoscopic displays, which do not require glasses. Rendering 3D content on these displays depend on five main parameters including pitch (spatial frequency or the period of the parallax barrier), angle (barrier degree of rotation from vertical), duty cycle (parallax barrier ratio of opacity to transparency), optical thickness (distance between the parallax barrier and the display), and shift (horizontal offset of the parallax barrier relative to the display). Different manufacturers can choose different values for these parameters. Therefore, our mobile client application is designed in two parts. The first (lower) part calls specific APIs supplied by the manufacturer of the mobile device, while the second part is independent of the manufacturer and implements all high-level functions such as the adaptive streaming.

Our web-based client employs standard web interfaces to render the encoded 3D video, while the 3D display along with its associated glasses (if any) creates the 3D perception for the viewer. Finally, there are displays that require special handling of 3D videos, such as multiview displays. Such displays usually require the 3D video in the form of video-plus-depth (2D images and their associated depth map). View synthesis methods are then used to create multiple virtual views from the video-plus-depth input. In our system, we implemented a custom client for such displays, which asks for video-plus-depth segments from the server and performs view synthesis using the manufacturer APIs.

4. DEPTH CUSTOMIZATION AND PERSONALIZATION

We need to show a different view to each eye to perceive binocular stereopsis. These two views show the same scene but from two slightly different viewing angles. Specifically, for each pixel in the left view, its corresponding pixel in the right view is located a few pixels away. Given a pair of corresponding pixels, the signed distance $d = x_r - x_l$ between their x-positions is called *disparity*. Disparities are detected by the human visual system and interpreted as depth. The depth perception from the display plane depends on the disparity value. An object is perceived behind or in front of the display plane if the disparity is positive or negative respectively as depicted in Figure 3. The object appears on the display plane in the special case of zero disparity. The amount of perceived depth z is a function of disparity d, viewing distance p, and interocular distance e. Perceived virtual depth, both for positive and negative disparities, is commonly approximated as in Figure 3, with larger disparities resulting in perception of larger distances [Holliman 2004].

In addition to viewing conditions, the depth perception varies from person to person [Coutant and Westheimer 1993; Didyk et al. 2011]. Thus, there is a need for techniques enabling customization of the depth distribution in 3D videos. In the context of streaming sports videos, such techniques need to meet three requirements: (i) they need to work as a postprocess, since we do not have influence on the recording process; (ii) they need to be fast; and (iii) they need to automatically produce high-quality results, without objectionable artifacts. In the following subsection, we propose a new

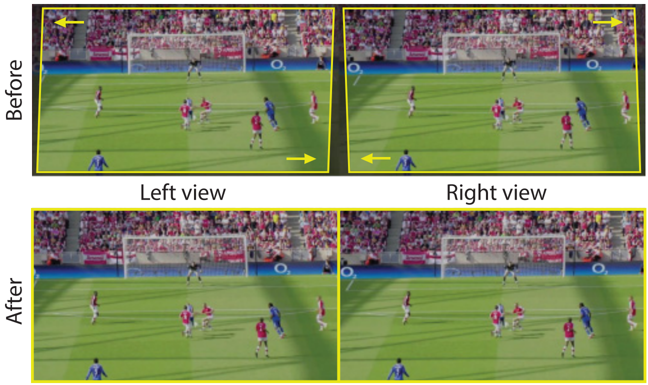


Fig. 4. Example of the slant operation. A parallelogram-shaped region is mapped back onto the whole image area, modifying the vertical component g_y of the depth gradient g. Note how the opposing slants in the left and right view complement each other to minimize the distortion.

method for depth expansion/compression that meets these requirements. It targets videos of various field sports such as soccer, football, and tennis. In Section 4.2, we utilize this method to enable depth personalization.

4.1. Structure Preserving Scene Shifting

Depth adjusting is a nontrivial task. As discussed in Section 2 , it can be achieved using a disparity remapping. This is, however, a viable option only in offline applications with some form of supervision. The only safe automatic adjustment for general scenes is convergence manipulation, which can be easily performed using a horizontal shift of the two views. We observed, however, that for some scenes, especially in sports videos, the geometry has approximately planar structure. In such cases, depth maps can be well described by a single depth gradient $g = (g_x, g_y)$.

Based on the previous observation and discussion, we propose the Structure Preserving Scene Shifting (SPSS) method for depth expansion/compression, which adjusts the depth range of the scene by means of 2D geometric transformations. The basic idea behind the SPSS method is to estimate the depth gradient g and adjust its magnitude. The gradient is estimated by fitting a plane to the scene's depth map which is obtained using a stereocorrespondence algorithm. In contrast to the disparity remapping approach, we use stereo correspondences only to estimate the single depth gradient, hence the accuracy of the correspondences is not critical.

Modification of the gradient is achieved via a remapping operation, in which a parallelogram-shaped region of the input image is selected and mapped back onto the whole image area. Such a mapping can be applied to one of the views or both, and in the latter case, the depth modifications caused by each transformation will add up. To minimize visibility of the distortion, we split the desired depth adjustment between the two views, so each mapping of one view is accompanied by a complementary mapping of the other. The top and bottom edges of the mapping parallelogram are always kept horizontal, and its shape is described by a combination of two parameters: the *slant* and the *stretch*.

The *slant* operation regulates the skewness of the mapping region by horizontally moving its top and bottom edges in opposite directions. This operation modifies g_y. An example is given in Figure 4. The *stretch* operation rescales the mapping region horizontally. This transformation modifies g_x. An example is given in Figure 5.

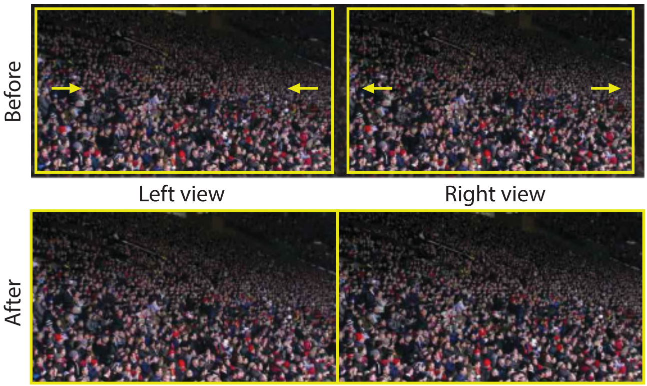


Fig. 5. Example of the stretch operation. The left and right views are horizontally scaled. In effect, the horizontal component g_x of the depth gradient g is changed.

The slant and stretch operations are done linearly to assure that any plane structure remains planar. This becomes especially important in sports videos since deformations such as curvatures in the field plane or the white lines can be very visible and disturbing.

We denote the amount of slant and stretch as σ_{sl} and σ_{st}, respectively. Assuming that the image x- and y-coordinates are in the range $[-\frac{1}{2}, \frac{1}{2}]$, the two operations are combined into one operator, mapping a pair of old coordinates (x, y) to a pair of new coordinates $(\hat{x}, \hat{y})$ as shown in Eq. (1). To accommodate slanting and stretching, the mapping region has to be slightly smaller than the input image, therefore the factor r (typically 0.95) is used to rescale the coordinates. Recall that the depth transformation is split equally between the two views of the stereo image, hence the factor ±0.5. In addition to slanting and stretching, a *shift* operation that moves the mapping region to the left or right can be used to adjust the convergence. Depending on the direction of the shift, the scene disparities are uniformly increased or decreased, and as a result, the scene appears to pop out of the display (i.e., negative disparity) or go deep behind the display plane (i.e., positive disparity). We refer to the amount of shift as the *pop-out factor*, and we denote it by β. The default value for β is zero. In Section 4.2 we discuss the effect of assigning different values to β,

$$(\hat{x}, \hat{y}) = \big(x \pm 0.5 \cdot (\sigma_{\mathrm{sl}} \cdot y + \sigma_{\mathrm{st}} \cdot x + \beta),\ \ y\big) \cdot \frac{1}{r}. \tag{1}$$

From the equation above, we can infer that the maximum added disparity is

$$\max_{\substack{-\frac{1}{2} \le x \le \frac{1}{2} \\ -\frac{1}{2} \le y \le \frac{1}{2}}} \sigma_{\mathrm{sl}} \cdot y + \sigma_{\mathrm{st}} \cdot x = \frac{|\sigma_{\mathrm{sl}}| + |\sigma_{\mathrm{st}}|}{2}. \tag{2}$$

This value is the maximum added disparity in the positive direction and the negative of this value is the maximum added disparity in the negative direction.

SPSS Coverage: Ekin et al. distinguish four types of camera shots in soccer games: long shots, medium shots, close-ups, and out-of-field shots. Long shots provide a global view, in which the field takes most of the display space, and multiple small player silhouettes are visible. Medium shots show a smaller portion of the field, usually with a couple larger silhouettes, while close-ups zoom on one or few players only. Finally, out-of-field shots show the audience, coaches, and so on [Ekin et al. 2003].

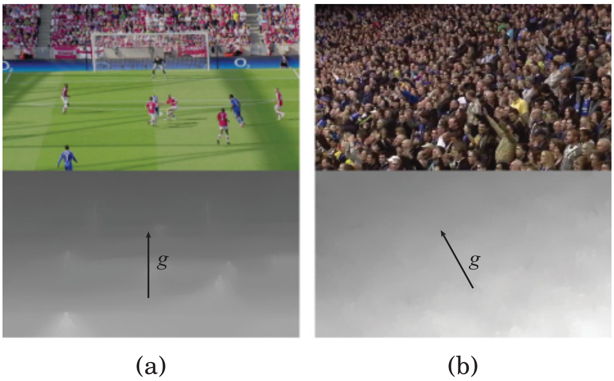


Fig. 6. Examples of shots in a soccer video that have planar depth structure. The depth maps were determined using optical flow estimation methods [Ugo Capeto 2013; Brox et al. 2004], and were further enhanced by cross bilateral filtering [Paris and Durand 2009]. Note that they are provided for visualization purposes, and our method does not require computationally expensive estimation of accurate, dense stereo correspondences.

Close-ups, out-of-field, and medium shots usually have quite complicated geometry. Long shots, however, differ, because their geometry can be very well approximated by a plane, and therefore they are perfect candidates for the SPSS (see Figure 6(a) for an example of a long shot). Occasionally, some medium or out-of-field shots, such as the one shown in Figure 6(b), are also well fitted by a plane and thus can benefit from the proposed depth adjustment. In the evaluation section, we analyze the shot types in different sports videos and show that the proposed SPSS method can cover a significant portion (60–70%) of the shots in field sports.

Our depth adjustment technique assumes that the scene can be well approximated by a plane. As a result, to detect shots[2] suitable for our technique, we should estimate how well a scene can be approximated by a single plane. To this end, we first recover depth using [Yang et al. 2010]. Next we fit a plane to it using the least-squares method and compute the coefficient of determination (R^2) to measure its goodness. We then construct a binary classifier, which classifies a scene based on the R^2 value, that is, if the goodness is above a certain threshold q, the scene is considered suitable for our manipulations. Otherwise, it remains unchanged. In order to determine a good threshold q, we processed 1,015 randomly chosen frames from one soccer game and classified them manually. Then we analyzed how our binary classifier performs for different threshold values using the relation between true positive and false positive ratios. Based on the analysis we chose $q = 0.693$, which gives a true-positive ratio = 0.8981 and a false-positive ratio = 0.1925.

Main Steps of SPSS: For any scene classified as suitable for the SPSS method, the following steps are taken:

(1) Compute the gradient $g = (g_x, g_y)$ of the scene's disparity map.
(2) Compute the slant and stretch as follows, where $\hat{d}$ is the *target disparity*, which depends on the viewing conditions and the content:

$$\sigma_{\mathrm{sl}} = \hat{d} \cdot \frac{g_y}{|g_x| + |g_y|} - g_y, \quad \sigma_{\mathrm{st}} = \hat{d} \cdot \frac{g_x}{|g_x| + |g_y|} - g_x, \tag{3}$$

[2] A shot in our definition is just a pair of images, and we do not need any shot boundary detection algorithms.

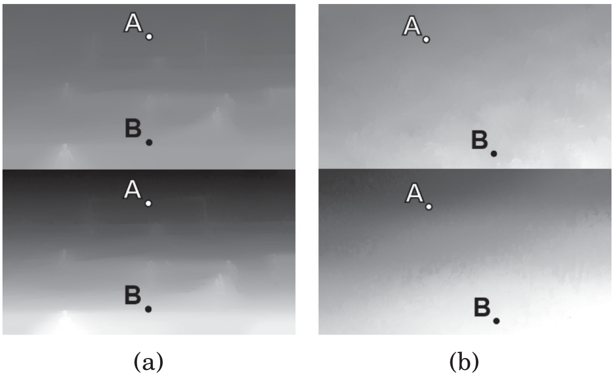


Fig. 7. Our method has increased the disparity range while preserving the depth structure and orientation (the top images are the original disparity maps, while the bottom ones show depth after our optimization.)

(3) Temporally smooth σ_{sl} and σ_{st} using the history of their values in the n previous frames in order to maintain temporal coherency.
(4) Remap the views according to Eq. (1) using linear interpolation.

Figure 7 shows the depth maps of the two samples from Figure 6 before and after SPSS. It can be seen that SPSS enhances the depth contrast, while preserving the original scene structure. In both samples, the depth difference between points A and B has increased, while, the direction of the depth gradient has remained unchanged.

Remarks and limitations of SPSS: We note that SPSS preserves the straight lines in the scene and does not introduce artifacts, because Eq. (1) is a linear remapping of input coordinates. Although linear remapping operations such as scaling and stretching can cause quality degradation due to pixel averaging, in SPSS this degradation is negligible. This is because the target disparity and thus the amount of slanting, stretching, and scaling performed in SSPS is small. Increasing the disparity to greater values will violate the comfort zone and cause discomfort long before pixel averaging artifacts become noticeable. In Section 6, we describe a perceptual study, in which optimal target disparities are found.

Furthermore, as mentioned, SPSS is only suitable for scenes with planar depth structure. Since planar structures can be linearly modeled, their depth can be accurately enhanced using linear remapping. We do not apply SPSS on nonplanar structures since visual distortion can happen. For example, players may look tilted towards the ground. However, since most of the shots in field sports have a planar structure, our SPSS method can improve depth for a significant portion of shots. In Section 6.4, we analyze the coverage of our method and show that 60–70% of the shots can benefit from SPSS.

4.2. Depth Personalization

The perception of depth varies from person to person, as it is a matter of personal preference. Thus, using the same content in all situations is suboptimal, in terms of both depth perception and visual comfort. In this section, we describe a depth personalization method that allows a viewer to choose his/her preferred depth perception level. Depth personalization is realized as follows. The user interface at the client side displays multiple depth options. The default selection is chosen based on the viewer's display size. However, the viewer is allowed to choose other versions with more or less depth. After selection, the client translates this request to the corresponding segment ID and submits it to the DASH server, which streams the requested segments.

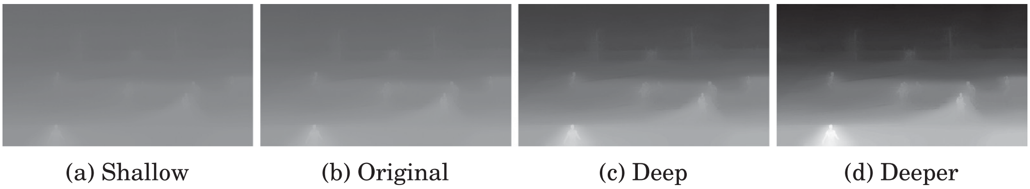

(a) Shallow (b) Original (c) Deep (d) Deeper

Fig. 8. Our system can produce personalized depth, based on viewer's preferences.

In order to achieve depth personalization, we perform three main operations: (i) create multiple versions with different depth values, (ii) control the pop-out effect, and (iii) integrate versions with DASH architecture. These operations are discussed in the following.

Creating Multiple Versions with Different Depth Values: We start by determining the upper and lower bounds at which the depth can be perceived. The upper bound is the maximum depth that can be fused by the human visual system. The lower bound is the depth at which no object appears in or out of the display (i.e., almost like 2D). As shown in Figure 3, the amount of perceived depth (z) is defined by $z = \frac{p}{e/d-1}$, where d is the disparity between the left and right images and e is the interocular distance. Therefore, the point where z has the maximum value happens when $d = e$ [Mendiburu 2012]. This is the point where $z = \infty$. On the other hand, the point where there is no depth perception ($z = 0$) happens when $d = 0$. We adjust the target disparity $\hat{d}$ using the upper and lower bounds on disparity. We set $\hat{d}_{max} = e$ for maximum depth expansion and $\hat{d}_{min} = 0$ for maximum depth compression. To convert $\hat{d}$ to pixels we should multiply it by the pixel density of the display, where pixel density is the diagonal resolution of the display in pixels divided by the length of the diagonal. The maximum disparity can then be expressed in pixels as:

$$\hat{d}_{max} = pixel\_density * e,$$

where e is a constant value equal to *2.5 inches*, which is the typical distance between human eyes. If the disparity exceeds this amount the human visual system will not be able to fuse the two views. The above formula suggests that the maximum disparity allowed on a display depends on the pixel density of that display. This means that in order to have an effective depth personalization, the maximum disparity presented to a tablet user should differ from the one presented to a TV user. For example, a HTC Evo 3D mobile phone has a pixel density of 256 pixels per inch, and therefore the $\hat{d}_{max}$ for this phone is 640 pixels. On the other hand, the pixel density for a 55″ Philips TV set is 40 pixels per inch, and therefore the $\hat{d}_{max}$ of this TV set is only 100 pixels.

Next, using our depth customization method in Section 4.1, we create multiple versions between the minimum and maximum depth values. Each of these versions represents a different level of depth. For simplicity of implementation, we create the versions (20 in our case) at equal distance in the range of $(\hat{d}_{min}, \hat{d}_{max})$. For $\hat{d}_{max}$, we choose the largest $\hat{d}_{max}$ among all supported devices. All versions are stored in a manifest file that contains metadata about the video and its available versions. Each version has an attribute in the manifest file called *target_disparity*, which is the $\hat{d}$ value used to compute the version. This value is used to prevent clients from requesting versions that exceed the maximum allowed disparity on their displays. Figure 8 shows some samples of expanded and compressed depths for the sample image in Figure 6(a). The figures from left to right show how the depth range gradually increases so it can match various users' preferences.

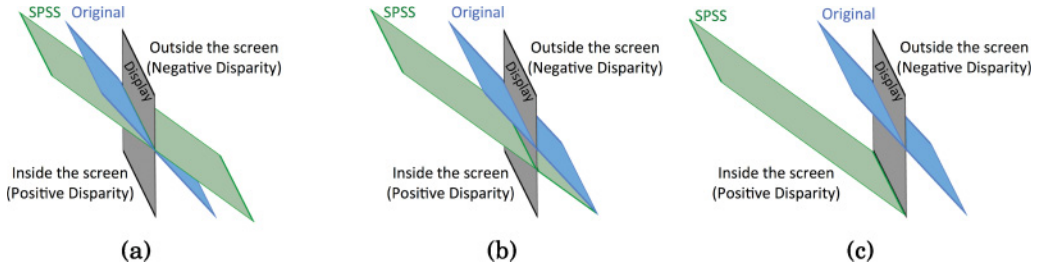


Fig. 9. Our system provides users with three options for controlling the pop-out effect: (a) Increasing the pop-out effect, (b) increasing the depth range while not increasing the pop-out effect, and (c) removing the pop-out effect.

Controlling Popping-out Effects: Objects that pop out of the display create a nice, catchy effect for many viewers. However, if the popping out is excessive or not properly adjusted, it can cause visual discomfort and often headaches [Smolic et al. 2011; Shibata et al. 2011]. In order to manage the tradeoff between creating catchy effects and maintaining visual comfort, our system provides the user with three options to control the pop-out effects, as follows:

(1) *Increasing the pop-out effect:* This is the default case for our method and it is done by setting $\beta = 0$ in Eq. (1). Applying this equation expands the depth in both directions, meaning that the objects that are originally popping out of the display pop out more, and the objects originally inside the display go deeper inside. (See Figure 9(a).)
(2) *Increasing the depth range while not increasing the pop-out effect:* In this case, the scene is adjusted in a way that the objects that used to pop out in the original scene will maintain the same depth perception and the objects inside the display plane go deeper inside. This avoids any discomfort caused by having some objects strongly popping out of the display and at the same time it maintains the catchy effects that appear in the original scene. Hence, we set the pop-out factor in a way that the scene is pushed back inside until it reaches the maximum negative disparity of the original scene. In Eq. (2), it is shown that the maximum negative disparity added to the original image is given by $-\frac{|\sigma_{\mathrm{sl}}|+|\sigma_{\mathrm{st}}|}{2}$. Therefore, we set $\beta = \frac{|\sigma_{\mathrm{sl}}|+|\sigma_{\mathrm{st}}|}{2}$ to cancel the added negative disparity. Since the value of β is added uniformly to all pixels, such adjustment does not affect the expanded depth range achieved by Eq. (1). It, however, ensures that the maximum negative disparity is maintained at the original value. (See Figure 9(b).)
(3) *Removing the pop-out effect:* Some viewers do not like the pop-out effect. For such viewers, we adjust the scene so that all objects are perceived behind the display plane. In order to do this, we set the pop-out factor in a way that the pixels with maximum negative disparity are shifted to be on the display plane. Therefore, we set $\beta = \frac{|\sigma_{\mathrm{sl}}|+|\sigma_{\mathrm{st}}|}{2} + |d_{max\_negative}|$, where $d_{max\_negative}$ is the maximum negative disparity in the original scene. This value for β removes all negative disparities in the scene. (See Figure 9(c).)

Integrating Versions with DASH Architecture: DASH is an adaptive streaming protocol that uses HTTP for streaming purposes. Videos under DASH are divided into small segments and encoded with different bitrates. This enables DASH clients to adapt to variable network conditions smoothly. Further, DASH defines a manifest file called Media Presentation Description (MPD) that contains the available segments bitrates, codecs, resolutions, and timing information. Typically, the DASH client requests MPD and parses its content. When the streaming session starts, it sends an HTTP request to the server with the chosen segment, and the server replies back with the corresponding segment. As described earlier in Section 3.2, our client first requests the manifest file from the server and decides on the most suitable 3D version offered by the server based on current network conditions and display characteristics. The client then starts requesting video segments from the server. During the video playback, the viewer is allowed to manually expand/compress the depth of the video. If the user decides to expand the depth of the video, then the client looks up the manifest file and requests the next segment from the version with the expanded depth perception. However, if the *target_disparity* attribute of the requested version is greater than the display's $\hat{d}_{max}$, the client will not switch to the expanded version and the user will be notified that the current version has the maximum depth suitable for his/her display. In case the *target_disparity* attribute of the requested version is less than the display's $\hat{d}_{max}$, the client continues fetching subsequent segments from the expanded version. At any point of time, the user can return back to the previous setting by choosing to compress the depth, and, similarly, the user will be notified when the current version has the most compressed depth. Once the user is satisfied with the depth perception, he/she is allowed to adjust the *pop-up factor* to one of the three options described earlier.

5. SYSTEM IMPLEMENTATION

We have developed a complete end-to-end prototype of our 3D streaming system, in which we implemented all of the proposed algorithms. We use this prototype system in the experimental evaluation in Section 6. We provide below a brief description of our prototype.

In our prototype, we implemented the following operations: (1) anaglyph, (2) row interleaving, (3) column interleaving, (4) frame sequential, (5) depth optimization, and (6) depth estimation. Additionally, we implemented some auxiliary operations, such as split, which splits a side-by-side image into separate left and right images, and scale, which resizes the images uniformly. These server-side operations have been developed using C++ and OpenCV. The system is implemented in an extensible and modular manner. Specifically, all video processing operations, for example, depth optimization, frame interleaving, and so on, are implemented as independent components with standard interfaces. This enables us to create different 3D versions by chaining a sequence of these video operations. The sequence of operations needed to create a 3D version is predefined using our 3D version tree as illustrated in Figure 2. Only one 3D version tree is maintained in the system, which is consulted on the creation of any 3D version. For example, to serve a viewer using a stereoscopic 3D TV with polarized glasses, the corresponding version is created by performing three operations: depth optimization, row interleaving, and scaling. We note that to create a version in Level 3 of the version tree (Figure 2), the ancestor versions at Levels 1 and 2 have to be created first. The system is designed to have the most time-consuming operations in Level 1. Therefore, these time-consuming operations are executed once instead of being executed in every branch of the tree. Further, when the depth optimization operation does not apply for a nonplanar depth structure frame, the original frame is included instead, so the client will not notice missing frames.

During initialization, the system decides which versions to create from each 3D video in the system. This decision depends on the storage requirements and the available processing capacity to create any missing version on demand. The chosen versions can be at any level of the 3D version tree and are not necessarily leaf nodes. To create a version, the system recursively traverses up the 3D version tree starting from the node corresponding to that version until it reaches an already-created version or it hits the root node (the original 3D video). The system pushes the noncreated versions to a stack.

Then it creates these versions using the order in the stack. We note that this version creation process is highly parallelizable, and a separate thread can be spawned for each version. However, multiple versions may share an ancestor that should be created. To avoid multiple creations of the same version, we chose to parallelize Level-1 branches only. With this setup, one thread is spawned for each Level-1 branch, and versions inside each branch are generated in series. We encoded the resulted versions with a frame rate of 25fps and bitrate of 2Mbits/s. We used H.264/AVC encoder and an mp4 container for nonweb clients and VP8 encoder and a webm container for web clients. We divided each version to DASH segments of length 2s.

At the client side, we have implemented three applications: First, a web application using HTTP and DASH-JS, which is a JavaScript DASH library for the Google Chrome browser. Second, a mobile application for the autostereoscopic mobile devices using Java and the Android SDK. We tested the application on HTC and LG 3D smartphones and on a Gadmei 3D tablet. Third, we implemented a MS Windows application for a multiview display (Dimenco 55″ TV). We implemented the DASH client using Java. The application has been developed to perform view synthesis and present the content on the multiview display. We note that there is no fundamental limitation of SPSS to be implemented at the client side. However, we choose to implement SPSS at the server to relieve the client from extra processing, especially for mobile clients. Also, if SPSS is implemented at the client side, the client needs to invoke SPSS every time it requires a 3D version. We minimize the number of SPSS invocations by executing it at level 1 of 3D version tree. Also, the server processing and preparation are aligned with DASH rate adaptation where the client is concerned with network conditions.

6. EVALUATION

We use the prototype described in the previous section to conduct subjective studies evaluating the depth quality in 3D videos. Our subjective studies in Section 6.3 show that up to 25% improvement in the depth quality can be achieved by the proposed system.

We start our evaluation by showing the need for depth customization and personalization through subjective studies. Next, we analyze the impact of depth personalization and show that increasing the depth level improves depth perception but can cause visual discomfort if performed excessively. We then measure the improvements achieved by our system for different viewing conditions. Finally, we analyze two full soccer games and a 10min tennis game and show that our method can enhance between 60% to 70% of the shots in 3D videos of field sports, such as soccer and tennis, while keeping the rest of the shots unchanged. In addition, we analyze the processing times of different video processing operations implemented in the proposed system.

6.1. The Need for Depth Customization and Personalization

In this study, we manipulated the depth of a 3D video with different degrees and displayed them to multiple subjects to discover their preferences. We conducted two experiments. In the first experiment, we focus on the need of depth customization for different displays and viewing conditions. In the second experiment, we focus on the need for depth personalization to satisfy the preference of different viewers. We note that early works, for example, ATTEST [Redert et al. 2002], showed the need for depth customization but for specific types of experimental displays and video-plus-depth format. Our experiments show this need for various modern, commodity, 3D displays, including mobile phones and active and passive TV displays.

Setup: We used a series of short clips taken from the Manchester United vs. Wigan Athletic FA Community Shield game (2–0, 12 August 2013). Each clip was a few seconds

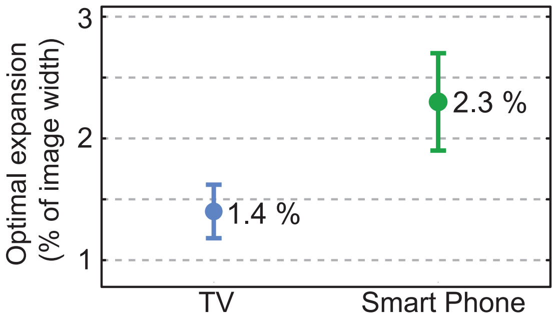


Fig. 10. Mean expansion value and standard error of the mean for two displays.

long, and the entire sequence was 54s long. The initial depth range of the test sequence was small, and the whole image appeared flat. The subjects were shown six versions of the test sequence, with increasing expansion values. The values used were as follows: −0.5% (depth compression); 0% (original sequence); and 1%, 2%, 3%, and 4% (depth expansion). In each condition, the subjects were asked to choose the preferred version, assuming that they were to watch the entire 90min soccer game.

Depth Customization: Ten subjects participated in this study. They were all computer science students and researchers, and all could perceive stereoscopic 3D effect. We tested two displays: smartphone and TV. The smartphone model was the LG Optimus 3D MAX P725 in which the stereo effect is achieved by means of a parallax barrier. The observation distance was about 30cm. The TV was a Sony Bravia XBR-55HX929 and a pair of active shutter glasses. The observation distance was about 2m. Half of the subjects saw the TV condition first, and the other half saw the smartphone condition first.

The results together with the standard error of the mean are shown in Figure 10. Results show that the average preferred values for expansion were 1.4% for the TV and 2.3% for the smartphone. The experiment shows that the original 3D videos were not the best choice for subjects. Instead, the depth-manipulated 3D videos scored higher than the original ones. In addition, the level of depth manipulation depended on the used display. Therefore, we can conclude that for optimized viewing of 3D videos on different displays, the depth of the videos needs to be adjusted.

We use the results of this experiment to set the default expansion values. That is, the expansion values for the TV and smartphone are set to 1.4% and 2.3%, respectively. We use linear interpolation of the expansion value for the other display sizes used in the experiments in Section 6.3.

Depth Personalization: For this experiment, we enlarged the number of subjects to capture various personal differences. Twenty-five subjects participated in this study. They were all computer science students and researchers, and all could perceive stereoscopic 3D effect. We used a 4K TV display (59.5″ LG 60UF8500) with a viewing distance of about 3m. The results (Figure 11) show the diverse depth preferences of the subjects.

6.2. Impact of Depth Personalization

The depth personalization feature of our system allows users to adjust depth based on their own preferences by increasing/decreasing the depth level. In this study we assess the impact of different depth levels on the 3D video perception. According to the ITU BT.2021 recommendations [ITU 2012], there are three primary perceptual dimensions for 3D video assessment: picture quality, depth quality, and visual (dis)comfort. Picture quality is mainly affected by encoding and/or transmission. Depth quality measures the

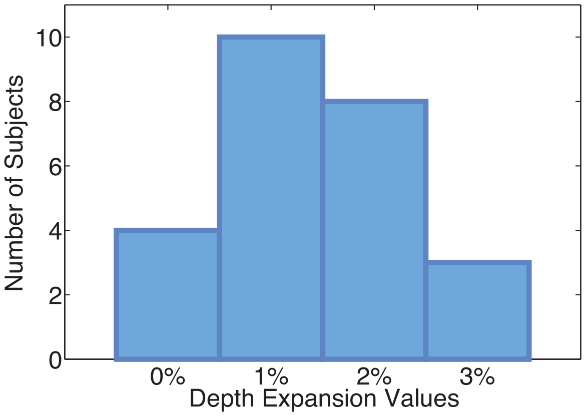


Fig. 11. Histogram of the expansion values preferred by subjects.

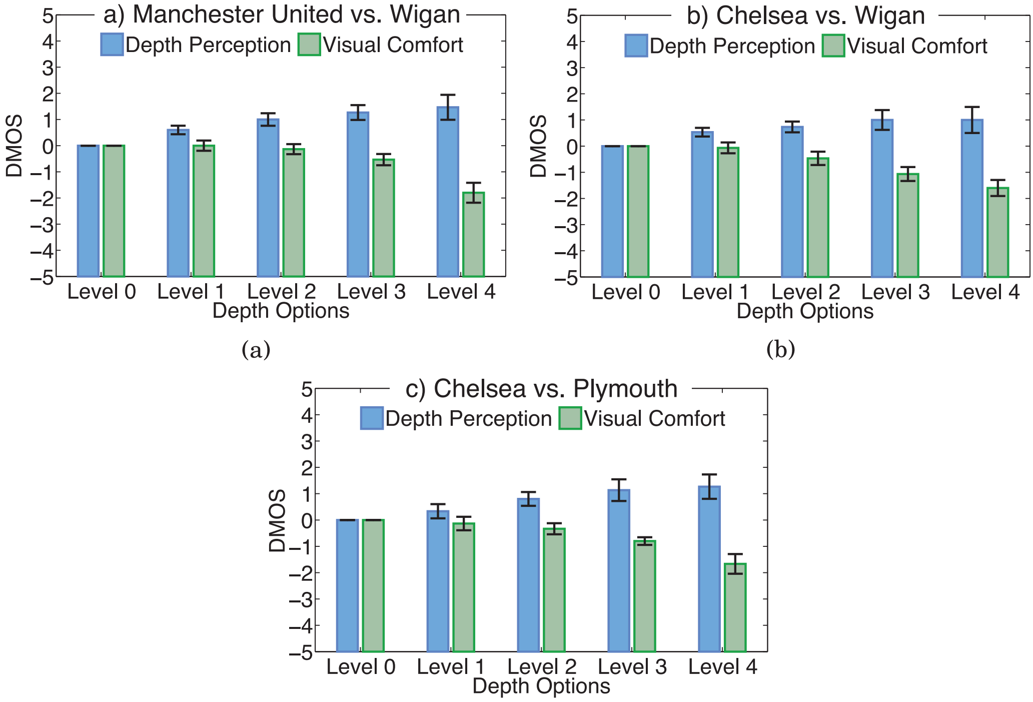
amount of perceived depth, and visual discomfort measures any form of physiological unpleasantness due to 3D perception, that is, fatigue, eye strain, headache, and so on. Such discomforts often occur due to 3D artifacts, depth alteration, comfort zone violations, and/or cross talk. In this study, we measure depth quality and visual comfort. We do not measure picture quality because we do not change any compression or encoding parameters.

We display sequences indoors on a 4K 59.5″ LG TV with passive polarized glasses in low lighting conditions. The viewing distance was around 3m. Fifteen subjects took part in this experiment. They were all computer science students and researchers. Their stereoscopic vision was tested prior to the experiment using static and dynamic random dot stereograms.

We used three soccer 3D video clips. From each clip, we used a series of shots with the following total lengths: (i) Manchester United vs. Wigan: 60s (ii) Chelsea vs. Wigan: 24s (iii) Chelsea vs. Plymouth: 20s. All shots were of long-view nature and thus suitable for our method. In our system, these shots are automatically identified by a classifier. Other shots are not subjected to our depth customization. The subjects viewed five versions of each sequence: the original version (level 0) and four versions with higher depth levels provided by our system. For all versions, β (the pop-out factor) was set to the default value of zero.

We use the single-stimulus method of the ITU recommendations to assess depth quality and visual comfort. The sequences are shown to subjects in random order. As suggested by the ITU recommendations, each sequence is preceded by a 3s midgrey field indicating the coded name of the sequence, followed by a 10s midgrey field asking subjects to vote. We use a discrete five-grade scale to rate depth quality and comfort. The depth quality labels are 5–Excellent, 4–Good, 3–Fair, 2–Poor, and 1–Bad, while the comfort labels are 5–Very Comfortable, 4–Comfortable, 3–Mildly Uncomfortable, 2–Uncomfortable, and 1–Extremely Uncomfortable. The subjects are asked to rate the amount of depth they can perceive in each sequence along with how comfortable it was to watch the sequence. We asked subjects to clarify all their questions and ensure their full understanding of the experimental procedure.

The inclusion of the original sequence allows us to compute the Difference Opinion Score (= score of each sequence − score of the original sequence). We then calculate the mean of the difference opinion scores (DMOS). A DMOS of zero implies that the sequence is judged the same as the original one, while a negative DMOS implies a depth perception/comfort lower than the original.



(a)

(b)

(c)

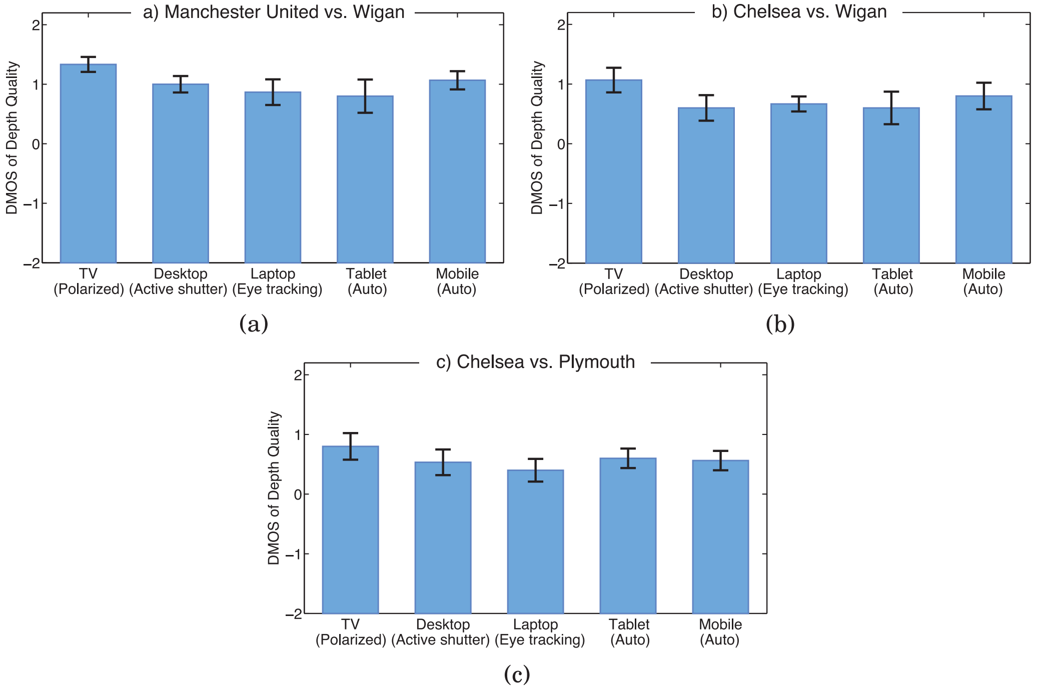
Fig. 12. Difference Mean Opinion Score (DMOS) between different depth levels and the original 3D sequence showing the impact of each level on depth perception and visual comfort. Zero implies that the sequence is the same as the original. Error bars represent the standard error of the mean. Higher levels translate to higher depth values. Level 0 is the original video. Our depth personalization allows users to choose the level they prefer the most.

Figure 12 shows the DMOS values along with the standard error of the mean. The figure shows that for the first few levels, increasing the depth level enhances depth perception without any noticeable degradation in comfort (level 1, level 2). However, when the depth exceeds a certain amount, visual comfort will drop dramatically (level 4). In other words, increasing the depth level has a positive impact on the 3D quality, but if increased excessively it will cause degradation in visual comfort and thus the 3D quality. Since the optimal depth is highly subjective, depth personalization enables users to choose the level that maximizes their depth perception while being comfortable to watch the content.

6.3. Depth Improvement

We conduct a subjective study to measure the perceptual depth enhancements achieved by our system for different videos and viewing conditions. The system was deployed on the Amazon cloud and fully tested over the Internet. For our subjective studies, however, the experiments were done over a LAN to assess the effectiveness of the proposed depth adjustment method. For handling network dynamics, our system supports encoding the video in different bitrates and it can switch among them in real time using the standard DASH protocol. Our depth adjustment method does not increase the bitrate and it is independent of the rate adaptation method.

We use the double-stimulus method for this experiment. Subjects view each pair of sequences at least twice before voting to assess their differences properly. The



(a)

(b)

(c)

Fig. 13. Difference Mean Opinion Score (DMOS) between our optimized version and the original 3D sequence. Our method improves depth quality for all tested videos on all displays. Error bars represent the standard error of the mean.

sequences are shown in random order without the subjects knowing which is which. We use the three video sequences in Section 6.2 and display them under five different viewing conditions: (i) smartphone: LG Optimus 3D MAX P725 phone (autostereoscopic), with observation distance about 30cm; (ii) tablet: GADMEI tablet (autostereoscopic), with observation distance about 40cm; (iii) laptop: 15.6″ Toshiba Qosmio F755-3D350 laptop (autostereoscopic with eye tracking), with observation distance about 60cm; (iv) desktop: 27″ Samsung desktop (active shutter glasses), with observation distance about 1 m; and (v) big TV: 55″ Philips (passive polarized glasses), with observation distance about 3m. Fifteen subjects took part in this experiment. All subjects viewed the five 3D displays. For each display, the subjects were shown the original sequence and the optimized version. Both the order in which participants saw the displays and the video sequences were randomized. The subjects were asked to rate the depth quality of each version on a five-grade discrete scale as follows: 5–Excellent, 4–Very Good, 3–Good, 2–Fair, and 1–Poor.

As suggested by ITU, we compute and report the DMOS values. The DMOS results for all 3D videos are shown in Figure 13. The figure shows the difference between ranking values of the depth-optimized 3D videos created by our method and the original 3D videos. The error bars visualize the standard error of the mean. The results demonstrate that our method improves the depth quality in all tested cases.

We note that the current stereo videos for sports are shot with stereo cameras with fixed parameters, and typically these parameters are set conservatively so the produced 3D video can be viewed on many displays. In other words, typical depth ranges in original 3D videos are small, which leaves large room for improvements for our method. For example, Figure 13(a) shows that the MOS of the depth quality for TV improved

Table I. Shot Analysis of the Milan vs. Barcelona Full 3D Soccer Game on 20/2/2013 in the UEFA Champions League

Time/Shot Type	Long	Medium	Close-up	Out-of-field
00:00:00–00:02:10	Introduction			
00:02:10–00:48:17	73.1%	24.4%	1.3%	1.2%
00:48:17–1:03:20	Half-time			
1:03:20–1:53:52	67.1%	28.4%	1.4%	3.1%

Table II. Shot Analysis of the Manchester United vs. Liverpool Full 3D Soccer Game on 16/3/2014 in the English Premiere League

Time/Shot Type	Long	Medium	Close-up	Out-of-field
00:00:00–00:04:30	Introduction			
00:04:30–00:55:10	68.25%	26.25%	4.1%	1.4%
00:55:10–1:06:56	Half-time			
1:06:56–1:58:10	61.3%	29.4%	6.6%	2.7%

Table III. Shot Analysis of the R. Nadal vs. N. Djokovic 3D Tennis Clip on 3/7/2011 in the Wimbledon Competition

Time/Shot Type	Long	Medium	Close-up	Out-of-field
00:00:00–00:00:28	Introduction			
00:00:28–00:09:41	64.4%	16.8%	2%	16.8%

by up to 1.4 points compared to the original sequence. This means that more than 25% improvement was achieved using our method.

Finally, we mention that we checked the statistical significance of our results by performing the Wilcoxon signed-rank test. All differences reported are statistically significant (p-value < 0.05) except the tablet version of the second video (Figure 13(b)), and the desktop and laptop versions of the third video (Figure 13(c)).

6.4. Coverage of the Depth Enhancement Method

To demonstrate the percentage of shots that can benefit from our depth customization method, we analyze two full 3D soccer games (each is more than 90min) and a 10min segment of 3D tennis game. The two soccer games were from different broadcasters and different competitions. We watched each video and manually classified shots. We summarized our analysis of the first full 3D soccer game, Milan vs. Barcelona, in Table I, which shows the percentage of long, medium, close-ups, and out-of-field shots. Similarly to any usual soccer game, the game included goals, player changes, injuries, offsides, and so on. It can be seen that the percentage of shots is similar throughout the two halves of the game. During the second half, two player changes, two injuries, and two goals occurred, which decreased the percentage of long shots. This is because these events typically have more close-up shots. However, on average, over 70% of the full 3D soccer game is long shots. Table II shows the analysis of another full 3D soccer game (Manchester United vs. Liverpool). The main difference between the two games is the percentage of close-up shots. Nevertheless, the shot percentage is still dominated by long shots in both games.

In Table III we summarized our analysis of the 3D segment from the tennis game between Rafael Nadal and Novak Djokovic. It can be seen that our method can enhance more than 64% of the game.

In summary, our analysis of various 3D games shows that our method can enhance between 60% to 70% of the shots in 3D videos in field sports, such as soccer and tennis.

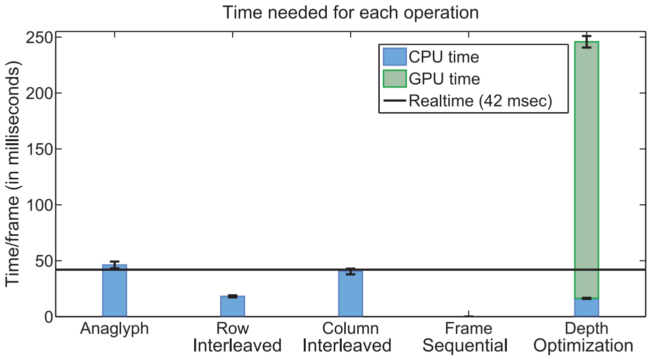


Fig. 14. Time required to process each operation in Anahita. The horizontal line is at $t = 42$ msec, representing real time processing at 24 fps.

6.5. Running Times

In this experiment, we measure the processing time of each operation implemented in the system. We measure the time on one server with an Intel Xeon CPU E5620 (2.4GHz) with 12GBs of memory and an Nvidia Tesla C2075 GPU with 6GB GPU of memory. All operations are implemented using C++ and OpenCV. Figure 14 shows the time required to perform each operation. The figure shows that most of the operations' processing time is below real time which is 42ms (assuming 24fps). The anaglyph operation is slightly higher than real time because it requires upscaling the image to double the resolution. This operation takes 70% of the time needed to run the anaglyph operation. Nevertheless, with some slight optimizations the anaglyph operation can run in real time.

In order to speed up the computation, we use GPU to run our depth optimization operation. The figure shows that the total running time for this operation is 245ms with a standard deviation of ±5ms. Although, the operation runs in 6× real time it is much faster than other methods that rely on creating the depth map before customizing the depth perception. In order to create the depth map in a good quality it takes from 10s to 2min for each frame, which is at least 40 times slower than our method.

7. CONCLUSIONS AND FUTURE WORK

We presented a novel system for adaptive streaming of stereoscopic content. The key feature of our solution is the capability of providing high-quality stereoscopic 3D videos to heterogeneous receivers in terms of display sizes and type, viewing conditions, viewers' preferences, and network conditions. To support such capabilities, our system employs the DASH protocol for adaptive streaming and switching among different versions of 3D videos. In addition, we proposed a new method for depth customization of 3D sports videos such as soccer, football, and tennis. The technique is computationally inexpensive and maintains the scene structure. Together with the adaptability of DASH streaming, it enables depth personalization to allow users to adjust the depth of a video based on their own preferences. This is a key feature of our system, which not only improves the viewers' engagement but also provides a comfortable experience.

To evaluate the performance of our system, we conducted a series of user experiments with different content and display devices showing that our technique can significantly improve the perceived quality.

This work can be extended in multiple directions. For example, in order to support a wider range of 3D content, depth customization methods for nonsports videos need to be designed. Another example is designing rate adaptation methods for streaming 3D videos in dynamic networks where bandwidth and packet loss rate frequently change.

REFERENCES

3 DeeCentral. 2015. 3DeeCentral Web site. Retrieved from http://www.3deecentral.com/.

3 DVisionLive. 2015. 3DVisionLive Web site. Retrieved from https://www.3dvisionlive.com/.

Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. 2004. High accuracy optical flow estimation based on a theory for warping. In *Proc. of European Conference on Computer Vision (ECCV'04)*. 25–36.

Peter Burt and Bela Julesz. 1980. Modifications of the classical notion of Panum's fusional area. *Perception* 9, 6 (1980), 671–682.

Kiana Calagari, Krzysztof Templin, Tarek Elgamal, Khaled Diab, Piotr Didyk, Wojciech Matusik, and Mohamed Hefeeda. 2014. Anahita: A system for 3d video streaming with depth customization. In *Proc. of the ACM International Conference on Multimedia (ACM MM'14)*. 337–346.

Pablo Carballeira, Julián Cabrera, Antonio Ortega, Fernando Jaureguizar, and Narciso García. 2012. A framework for the analysis and optimization of encoding latency for multiview video. *IEEE J. Select. Top. Signal Process.* 6, 5 (2012), 583–596.

Ben E. Coutant and Gerald Westheimer. 1993. Population distribution of stereoscopic ability. *Ophthal. Physiol. Opt.* 13, 1 (1993), 3–7.

Khaled Diab, Tarek Elgamal, Kiana Calagari, and Mohamed Hefeeda. 2014. Storage optimization for 3d streaming systems. In *Proc. of ACM Conference on Multimedia Systems (MMSys'14)*. 59–69.

Piotr Didyk, Tobias Ritschel, Elmar Eisemann, Karol Myszkowski, and Hans-Peter Seidel. 2011. A perceptual model for disparity. *ACM Trans. Graphics* 30, 4 (2011), 96:1–96:9.

Displaybank Co. 2010. 3D TV industry trend and market forecast. Special Report. Retrieved from http://www.displaybank.com/research_file/710527.pdf.

Ahmet Ekin, A. Murat Tekalp, and Rajiv Mehrotra. 2003. Automatic soccer video analysis and summarization. *IEEE Transactions on Image Processing* 12, 7 (2003), 796–807.

Christoph Fehn. 2003. A 3d-tv system based on video plus depth information. In *Asilomar Conference on Signals, Systems and Computers* 2, 1529–1533.

Jonathan Freeman and Steve E. Avons. 2000. Focus group exploration of presence through advanced broadcast services. In *Proc. of SPIE Human Vision and Electronic Imaging Conference*. 530–539.

C. Göktuğ Gurler, Burak Görkemli, Görkem Saygili, and others. 2011. Flexible transport of 3-d video over networks. *Proc. IEEE* 99, 4 (2011), 694–707.

Ahmed Hamza and Mohamed Hefeeda. 2014. A DASH-based free-viewpoint video streaming system. In *Proc. of ACM NOSSDAV'14 Workshop, in Conjunction with ACM Multimedia Systems (MMSys'14) Conference*. 55–60.

Nicolas S. Holliman. 2004. Mapping perceived depth to regions of interest in stereoscopic images. In *Proc. of SPIE Stereoscopic Displays and Virtual Reality Systems Conference*. 117–128.

ISO/IEC 23009-1:2012. 2012. Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats.

ITU-R BT.2021. 2012. Subjective methods for the assessment of stereoscopic 3D TV systems. International Telecommunication Union, Geneva, Switzerland.

Mathias Johanson. 2001. Stereoscopic video transmission over the internet. In *Proc. of IEEE Workshop on Internet Applications (WIAPP'01)*. 12–19.

Hideaki Kimata, Katsuhiko Fukazawa, Akio Kameda, Yoshie Yamaguchi, and Norihiko Matsuura. 2011. Interactive 3D multi-angle live streaming system. In *Proc. of IEEE International Symposium on Consumer Electronics (ISCE'01)*. 576–579.

Manuel Lang, Alexander Hornung, Oliver Wang, Steven Poulakos, Aljoscha Smolic, and Markus Gross. 2010. Nonlinear disparity mapping for stereoscopic 3D. *ACM Trans. Graphics* 29, 3 (2010), 75:1–75:10.

Jian-Guang Lou, Hua Cai, and Jiang Li. 2005. A real-time interactive multi-view video system. In *Proc. of ACM International Conference on Multimedia (ACMMM'05)*. 161–170.

Bernard Mendiburu. 2012. *3D Movie Making: Stereoscopic Digital Cinema from Script to Screen*. CRC Press.

Thomas Oskam, Alexander Hornung, Huw Bowles, Kenny Mitchell, and Markus Gross. 2011. OSCAM - optimized stereoscopic camera control for interactive 3D. *ACM Trans. Graphics* 30, 6 (2011), 189:1–189:8.

Dawid Pajak, Robert Herzog, Radosław Mantiuk, Piotr Didyk, Elmar Eisemann, Karol Myszkowski, and Kari Pulli. 2014. Perceptual depth compression for stereo applications. *Comput. Graphics Forum* 33, 2 (2014), 195–204.

Sylvain Paris and Frédo Durand. 2009. A fast approximation of the bilateral filter using a signal processing approach. *Int. J. Comput. Vision* 81, 1 (2009), 24–52.

Selen Pehlivan, Anil Aksay, Cagdas Bilen, Gozde Bozdagi Akar, and M. Reha Civanlar. 2006. End-to-end stereoscopic video streaming system. In *Proc. of IEEE International Conference on Multimedia and Expo (ICME'06)*. 2169–2172.

André Redert, Marc Op de Beeck, Christoph Fehn, Wijnand Jsselsteijn, Marc Pollefeys, Luc Van Gool, Eyal Ofek, Ian Sexton, and Philip Surman. 2002. ATTEST: Advanced three-dimensional television system technologies. In *Proc. of International Symposium on 3D Data Processing Visualization and Transmission*. 313–319.

Takashi Shibata, Joohwan Kim, David M. Hoffman, and Martin S. Banks. 2011. The zone of comfort: Predicting visual discomfort with stereo displays. *J. Vision* 11, 8 (2011), 11:1–11:29.

Aljoscha Smolic, Peter Kauff, Sebastian Knorr, Alexander Hornung, Matthias Kunter, Marcus Mller, and Manuel Lang. 2011. Three-dimensional video postproduction and processing. *Proc. IEEE* 99, 4 (2011), 607–625.

Iraj Sodagar. 2011. The MPEG-DASH standard for multimedia streaming over the internet. *IEEE Multimed. Mag.* 18, 4 (2011), 62–67.

Thomas Stockhammer. 2011. Dynamic adaptive streaming over HTTP: Standards and design principles. In *Proc. of ACM Conference on Multimedia Systems (MMSys'11)*. 133–144.

Wa James Tam, Filippo Speranza, Sumio Yano, Koichi Shimono, and Hiroshi Ono. 2011. Stereoscopic 3D-tv: Visual comfort. *IEEE Trans. Broadcast.* 57, 2 (2011), 335–346.

Jonathan R. Thorpe and Mark J. Russell. 2011. Perceptual effects when scaling screen size of stereo 3D presentations. In *Proc. of Society of Motion Picture and Television Engineers Conferences (SMPTE'11)*. 1–10.

Trivido. 2015. Trivido Web site. Retrieved from http://www.trivido.com/.

Ugo Capeto. 2013. Depth Map Automatic Generator (DMAG). Retrieved from http://3dstereophoto.blogspot.com/2013/04/depth-map-automatic-generator-dmag.html.

Anthony Vetro, Thomas Wiegand, and Gary J. Sullivan. 2011. Overview of the stereo and multiview video coding extensions of the H. 264/MPEG-4 AVC standard. *Proc. IEEE* 99, 4 (2011), 626–642.

Wanmin Wu, Ahsan Arefin, Gregorij Kurillo, Pooja Agarwal, Klara Nahrstedt, and Ruzena Bajcsy. 2011. Color-plus-depth level-of-detail in 3D tele-immersive video: A psychophysical approach. In *Proc. of ACM International Conference on Multimedia (ACM MM'11)*. 13–22.

Baicheng Xin, Ronggang Wang, Zhenyu Wang, Wenmin Wang, Chenchen Gu, Quanzhan Zheng, and Wen Gao. 2012. AVS 3D video streaming system over internet. In *Proc. of IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC'12)*. 286–289.

Qingxiong Yang, Liang Wang, and Narendra Ahuja. 2010. A constant-space belief propagation algorithm for stereo matching. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10)*. 1458–1465.

YouTube. Retrieved from http://www.youtube.com/.

Received June 2015; revised January 2016; accepted January 2016