Efficient Streaming of Stereoscopic Depth-Based 3D Videos

Dogancan Temel\textsuperscript{a}, Mohammed Aabed\textsuperscript{a}, Mashhour Solh\textsuperscript{b}, and Ghassan AlRegib\textsuperscript{a}

\textsuperscript{a}School of Electrical and Computer Engineering, Georgia Institute of Technology
Atlanta, GA, 30332 USA
\textsuperscript{b}Texas Instruments, Dallas, TX, USA

ABSTRACT

In this paper, we propose a method to extract depth from motion, texture and intensity. We first analyze the depth map to extract a set of depth cues. Then, based on these depth cues, we process the colored reference video, using texture, motion, luminance and chrominance content, to extract the depth map. The processing of each channel in the $YC_R C_B$–color space is conducted separately. We tested this approach on different video sequences with different monocular properties. The results of our simulations show that the extracted depth maps generate a 3D video with quality close to the video rendered using the ground truth depth map. We report objective results using 3VQM and subjective analysis via comparison of rendered images. Furthermore, we analyze the savings in bitrate as a consequence of eliminating the need for two video codecs, one for the reference color video and one for the depth map. In this case, only the depth cues are sent as a side information to the color video.

Keywords: Three-dimensional video, 3DTV, depth image based rendering, depth map, monocular cues, stereoscopic video.

1. INTRODUCTION

Recently, the interest in Three-Dimensional Television (3DTV) and Free Viewpoint Video (FVV) have been proliferating in industry and academia. 3DTV provides a sense of depth to the video scene. Moreover, FVV allows the user to navigate freely inside the scene. Within the context of these standards for future video technology, many challenges arise in terms of bandwidth, synthesis and quality. Of all the proposed techniques to synthesize a 3D view, Depth-Image-Based Rendering (DIBR) is one of the most propitious ways to generate 3D scenes. In DIBR-based applications, a depth map and a colored reference frame (view) are required to synthesize additional views. The quality of the synthesized views are considerably affected by the quality of the depth map. However, transmitting the depth map not only requires extra bandwidth and codec at both sides but also causes the received depth map to be of lower quality due to transmission errors and losses and compression artifacts. To overcome these problems, we propose a method that eliminates transmitting the depth map and enables the reconstruction at the receiver by using depth map cues that are sent as side information within the color video stream. In this paper, we use motion, texture, luminance and chrominance content within the reference color video to reconstruct the depth map from the received depth cues.

The overall system is illustrated in Figure 1. The main components in the system are: (i) 3D video capturing and content generation; (ii) 3D content video coding; (iii) transmission; (iv) decoding the received sequence; (v) generating virtual views; (vi) displaying the stereoscopic images on the screen. In this work, we are going to mainly focus on the second block and the fifth block. Instead of encoding the depth map in the second block, we will extract depth cues and send them as a side information with the reference colored video. Thus, eliminating the need for a depth streaming channel. In the fifth block, we will reconstruct the depth map using the depth and monocular cues.

The Human Visual system (HVS) exploits a set of visual depth cues to perceive 3D scenes. These depth cues can be classified into two categories: binocular and monocular cues. Binocular cues are the disparities that exist between the two views seen by both eyes of a particular scene. The HVS extracts the depth information by comparing two views of a particular scene. The illusion of 3D is created in today’s technologies by projecting two views with a slight horizontal disparity onto the left and right eyes. It is believed that the human mind creates the illusion of 3D by exploiting the
differences between the perceived images. The HVS can also extract depth using a single eye. The depth information that can be extracted from a single view is known as monocular cue. Monocular depth cues are numerous and the following is a list of the important cues within the context of our discussion [1].

- **Motion Cue:** The HVS can distinguish depth from relative motion of objects since near objects move faster across the retina than far objects.

- **Texture Gradient Cue:** Texture is also an important depth cue. Depth can be extracted from texture by estimating the shape of a surface based on certain attributes.

- **Color and Intensity Cues:** Depth can also be estimated from luminance and color variations in the scene. By a phenomenon known as atmospheric scattering, scenes in the foreground tend to have higher contrast as compared to scenes in the background. In addition, brighter or higher luminance values are often closer to the foreground.

The goal of the depth estimation from monocular cues is to convert depth cues contained in video sequences into actual depth values of a captured scene. The extraction of depth from monocular depth cues for 2D-to-3D conversion is a complex challenge, one that has attracted a lot of attention in the last decade [2]. We focus here on the studies most relevant to this work, namely, motion, texture and intensity based depth estimation and extraction. Authors in [3] proposed using keyframe pairs and their corresponding depth maps to interpolate a depth map at both ends of a scene. This approach uses forward/backward motion for alignment. [4] proposed a content-adaptive depth assignment scheme. Video shots are segmented and broken into three classes based on conversion schemes. The foreground is separated from the background for independent processing. Depth assignment is performed by analyzing several cues, such as motion parallax, atmospheric perspective, texture gradient, linear perspective, and relative height. In [5], depth maps are obtained from motion features and used for stereoscopic video synthesis. This work takes into account multi-user cases and display device specifications. In a stereoscopic characterization stage, minimum and maximum depth values and view field were calculated. These depth values and view field were then applied to various types of 3D displays. The conversion of motion to depth was done relying on three cues: motion magnitude, camera movements and scene complexity. In [6], the \( C_R \) component in the \( YC_RCB \)-color space is mapped to a given range of depth values after adjusting the very bright red segments.

In this paper, we propose a depth extraction method based on the depth monocular cues from texture, motion and intensity. At first, we process a given depth map to extract a set of depth cues. These depth cues are used as a side information with motion and texture to reconstruct the depth map. These depth maps are then combined to obtain a single depth map. Luminance and chrominance channels are also used alongside depth cues to reconstruct the depth map. We match the statistical features of the ground truth depth and these extracted color channels to get the depth map estimation. The rest of this paper is organized as follows. Section 2 provides the details of our proposed depth-map cues extraction and estimation at the sender and receiver. In Section 3, we show the results and analysis of quality assessment and changes in the bitrates using a set of well-known video sequences. Finally, the concluding remarks and future directions are given in Section 4.

![Figure 1. A block diagram demonstrating the proposed framework for depth maps extraction. The proposed algorithm has two components labeled as (A) and (B), respectively.](image-url)
This section explains the proposed depth extraction methodology based on intensity, motion and texture. Our framework is based on the following idea. We argue that depth can be estimated from the variations in luminance and chrominance among other monocular features such as motion and texture. The depth map at the sender side can be captured by an active or a passive sensor with a depth estimation algorithm. We first extract depth cues by estimating a set of parameters that can be sent along the colored video for depth-map reconstruction or estimation at the receiver. Second, depth maps at the receiver side will be reconstructed from motion, texture and intensity cues of the received colored video and the received depth cues.

2. INTENSITY, TEXTURE, AND MOTION BASED DEPTH EXTRACTION

The goal of this operation (Process (A) in Figure 1) is to process the depth map at the sender to extract the number of depth planes \( N \) and the relative (or absolute) depth value for each plane \( D_n \) where \( n = 0, \cdots, N - 1 \). We extract these depth values by analyzing the histogram of the depth map, as shown in the example given in Figure 2 (a) for the Balloons sequence. Let \( H_D(k) \) for \( k = 0, 1, \cdots, 255 \) be the histogram of the depth map where \( H_D(k) \) is the number of pixels with intensity value \( k \). First, for each sequence, a threshold value, \( h_{th} \), corresponding to the minimum number of pixels in a plane is defined heuristically or automatically. The value for \( h_{th} \) is chosen to be the average between the largest maxima and lowest minima in the histogram.

After obtaining \( N, h_{th} \) and \( H_D(k) \), the values of \( D_n \) are found in \( N/2 \) iterations. In every iteration \( i \), we calculate the values of two depth planes: \( D_i \) and \( D_{N−1−i} \). In other words, in every iteration we process a depth level where we calculate the far and near depth values of that level. The calculations of \( D_i \) and \( D_{N−1−i} \) are done by a twofold scan process in every iteration. \( D_i \) is determined by scanning the histogram from left-to-right. Similarly, the histogram is scanned from right-to-left to find \( D_{N−1−i} \). Furthermore, in every iteration we determine the width of the depth plane. For the \( n \)th depth plane with depth value \( D_n \), the width of that depth plane is denoted by \( p_n \). This process is explained in details in the following pseudo code:

1. for \( i = 0, \cdots, (N/2) − 1 \) do
2. if \( i = 0 \) then
3. Initialize \( D_0 = 0 \) and \( D_{N−1} = 255 \)
4. else
5. Initialize \( D_0 = D_{i−1}+p_{i−1} \) and \( D_{N−1−i} = D_{N−i}−p_i \)
6. end if
7. Initialize \( p_i = 0 \) and \( p_{N−i−1} = 0 \)
8. for \( j = D_i, \cdots, D_{N−1−i} \) do
9. if \( H_D(j) > h_{th} \) then
10. Set \( D_i = j \)
11. BREAK
12. end if
13. end for
14. for \( j = D_{N−1−i}, \cdots, D_i \) do
15. if \( H_D(j) > h_{th} \) then
16. Set \( D_{N−1−i} = j \)
17. BREAK
18. end if
19. end for
20. for \( j = D_i, \cdots, D_{N−1−i} \) do
21. if \( H_D(j) > h_{th} \) then
22. \( p_i = p_i + 1 \)
23. else
24. BREAK
25. end if
26. end for
27. for \( j = D_{N−1−i}, \cdots, D_i \) do
28. if \( H_D(j) > h_{th} \) then
29. \( p_{N−1−i} = p_{N−1−i} + 1 \)
30. else
31. BREAK
32. end if
33. end for
34. \( D_0 = D_0 + p_0 \)
35. \( D_{N−1} = D_{N−1} + p_{N−1} \)
36. \( \forall n = 1, \cdots, N − 2; D_n = \text{Median of } \{D_n, \cdots, D_n + p_n\} \)
37. end for

This algorithm extracts the number of planes, \( N \), and the depth values for each plane, \( D_0, \cdots, D_{N−1} \). Figure 2(a) shows the extracted parameters using this depth analysis process. Based on the experiments we conducted, we have found that the number of extracted depth planes on average tends to be less than or equal to four \( (N \leq 4) \). Hence, the discussion from this point will assume \( N = 4 \) for the simulated sequences. Nevertheless, this approach is generic for any number of depth plans.
2.2 Luminance-Based Depth Extraction

At the receiver, the parameters extracted in Section 2.1 are used with the luminance channel to reconstruct the depth map. For each plane \( n \) and its corresponding depth values \( D_n \), a corresponding range of depth values is extracted from the luminance component of the colored video. Thus, a set of planes \( L_n \) are extracted from the luminance channel. This can be done by following similar steps to the one used in depth cues extraction. A luminance to depth mapping is then performed by replacing the high luminance planes with the near depth, the low luminance pixels with far depth and middle luminance plans with the corresponding intermediate plane-depth values. The output of this mapping is the estimated depth-map, \( Z_Y \). Figure 2(b) shows a histogram matching example for one of the frames in the Balloons sequence after applying the algorithms in Sections 2.1 and 2.2 to the depth map and luminance channel, respectively. The estimation of the depth map is done by replacing luminance values between 0 and \( L_0 \) with \( D_0 \), values between \( L_0 \) and \( L_1 \) with \( D_1 \), values between \( L_1 \) and \( L_2 \) with \( D_2 \) and values between \( L_2 \) and \( L_3 \) with \( D_3 \).

2.3 Chrominance-Based Depth Extraction

To estimate the depth from chrominance cues, we use the depth cues extracted in Section 2.1 to map the chrominance to depth. We obtain two depth estimates for \( C_R \) and \( C_B \) channels. The estimated depth values, denoted by \( Z_{C_R} \) and \( Z_{C_B} \), are calculated as follows:

\[
Z_{C_R} = C_R I \times (D_{N-1} - D_0)/255 + D_0 \quad (1a)
\]

\[
Z_{C_B} = C_B I \times (D_{N-1} - D_0)/255 + D_0 \quad (1b)
\]

In this paper, we do not address the issue of fusing or combining \( Z_Y, Z_{C_R} \) and \( Z_{C_B} \). We limit the scope to analyzing each of the three estimated depth maps independently.

2.4 Texture-Based Depth Extraction

Texture structure of an image provides information about the depth of a scene. [7] used texture information for the depth-matching experiments and they showed that it provides information about the depth of a scene by itself. In our case, we construct an estimate of the depth map from the texture given a colored reference frame. First the texture randomness index at macroblock \([x, y]\) of the frame \(I\) is computed by

\[
t[x, y] = EI[x, y] \times MI[x, y]
\]

where \( EI \) is an edge intensity binary image with 1’s where the function finds edges in \(I\) and 0’s elsewhere, and \( MI \) is the mean intensity of \(I\). Both \( EI \) and \( MI \) are computed on non-overlapping squared macroblocks. The texture randomness index is then mapped to the object index as follows [8]:

\[
T[x, y] =
\begin{cases}
K_1 + \left(0.5 \times K_1 \times \frac{\log_2(t[x, y])}{\log_2{\beta_1}}\right) & \beta_1 \leq t[x, y] < \beta_2 \\
K_2 + \left(0.5 \times K_2 \times 2^{-(t[x, y] - \beta_2)}\right) & t[x, y] > \beta_2 \\
K_1 & \text{otherwise}
\end{cases}
\]
where $K_1$ and $K_2$ are constant parameters chosen to control the weights assigned to the structured regions and randomly
textured regions, respectively. By setting $K_1 \gg K_2$, higher weights are assigned to the structured regions. $\beta_1$ and $\beta_2$ are
edge detector threshold parameters. After processing all the macroblocks using Equations (2) and (3), this process results
in a texture map for the frame denoted by $IT$. Due to constant block size of the macroblocks, the same object closer to the
camera will have a higher texture randomness index compared to the one at the background so that it enables extracting
depth information from the texture structure.

This texture index is calculated for every pixel in the reference frame to give a depth estimate, $Z_T$:

$$Z_T = \frac{IT - \min(IT)}{\min(IT) - \max(IT)} \times (D_{N-1} - D_0) + D_0 \quad (4)$$

### 2.5 Motion-Based Depth Extraction

For the purpose of this work, motion information is calculated by taking the difference of the pixel values between the
reference colored frame and the $i^{th}$ subsequent frame. This motion information is denoted by, $IM$. It should be noted that
$i$ is a parameter that is chosen heuristically and fixed for the entire sequence. Block based motion estimation methods such
as exhaustive and three step search were also performed to extract the motion information. However, they did not provide
an accurate estimation of depth due to extensive amount of noise. The depth estimate from motion is then calculated as
follows:

$$Z_M = \frac{IM - \min(IM)}{\min(IM) - \max(IM)} \times (D_{N-1} - D_0) + D_0 \quad (5)$$

### 2.6 Combining Motion- and Texture-based Depths

The first stage of this process estimates the depth value for the pixel unless motion and texture are both zeros. For any pixel
$[m, n]$

$$S_1 [m, n] = \begin{cases} 
Z_M [m, n], & Z_M [m, n] \neq 0 \text{ and } Z_T [m, n] = 0 \\
Z_T [m, n], & Z_M [m, n] = 0 \text{ and } Z_T [m, n] \neq 0 \\
A_1 [m, n], & Z_M [m, n] = 0 \text{ and } Z_T [m, n] = 0 \\
sZ_M [m, n] + (1 - s)Z_T [m, n], & \text{otherwise}
\end{cases} \quad (6)$$

If both the motion and texture have zero values, the auxiliary function $A_1 [m, n]$ is used which is defined as follows:

$$A_1 [m, n] = \begin{cases} 
A_{2M} [m, n], & A_{2M} [m, n] \neq 0 \text{ and } A_{2T} [m, n] = 0 \\
A_{2T} [m, n], & A_{2M} [m, n] = 0 \text{ and } A_{2T} [m, n] \neq 0 \\
sA_{2M} + (1 - s)A_{2T} [m, n], & A_{2M} [m, n] \neq 0 \text{ and } A_{2T} [m, n] = 0
\end{cases} \quad (7)$$

where $0 < s < 1$ is a scaling factor determined empirically and varies from one sequence to another. The auxiliary
functions $A_{2M}$ is computed as follows:

$$A_{2M} [m, n] = \begin{cases} 
Z_M [m, n + k], & Z_M [m, n + k] > \epsilon \\
0, & \text{otherwise}
\end{cases} \quad (8)$$

where $\Gamma$ is the block size processing parameter that is set to 20 and epsilon is a threshold value set to 5; these values
were determined empirically. This process is a reassignment of the depth values for each pixel by scanning the frame from
left to right. For every pixel, the $\Gamma$ right neighbors are used to determine the value of that pixel. If neighboring pixel has a
value greater than the threshold, that pixel value is assigned to the leftmost pixel and it is updated incase there is more than
one pixel greater than the threshold. If none of the pixels are greater than the threshold, we keep the pixel value as zero.
We note that $A_{2T}$ is also calculated in a manner similar to $A_{2M}$. These steps correspond to the first stage of four main
stages. In the second stage, we scan the frame from right to left such that the value of each pixel is determined as follows:

$$S_2 [m, n] = \begin{cases} 
S_1 [m, n + k], & S_1 [m, n + k] > \epsilon \\
S_1 [m, n], & \text{otherwise}
\end{cases} \quad (9)$$

This process is repeated by performing the scanning from top to bottom producing $S_3$ with $S_2$ as an input. Finally, the
scan is done from bottom to top with $S_3$ as an input producing the final depth estimation, $Z_{TM}$. It should be noted that in
the first stage the contribution of motion and texture are adjusted using the scaling factor $s$. However, in other stages, the
reassignment of depth values is performed over the depth values obtained directly from previous stages.
Table 1. 3VQM and PSNR values for six different videos.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>3VQM</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>M-T</td>
<td>Y</td>
</tr>
<tr>
<td>Balloons</td>
<td>4.24</td>
<td>2.16</td>
</tr>
<tr>
<td>Cafe</td>
<td>4.69</td>
<td>3.03</td>
</tr>
<tr>
<td>Champ. Tower</td>
<td>4.07</td>
<td>0.41</td>
</tr>
<tr>
<td>Kendo</td>
<td>4.21</td>
<td>2.90</td>
</tr>
<tr>
<td>Love Birds</td>
<td>4.84</td>
<td>3.98</td>
</tr>
<tr>
<td>Pantomime</td>
<td>4.44</td>
<td>0.17</td>
</tr>
</tbody>
</table>

Figure 3. Pantomime depth maps and synthesized views.

3. SIMULATION RESULTS AND ANALYSIS

In this section, we introduce the results and analysis of the proposed depth reconstruction methodology. Since this work targets DIBR applications, estimated depth maps were evaluated through quality assessment of the rendered videos using the obtained depth maps. The synthesis of these videos was done based on DIBR with Hierarchical Hole-Filing (HHF) [9]. The Assessment of the obtained videos was then performed using PSNR and the 3D vision-based quality measure (3VQM) developed in [10].

We tested the proposed approach on different video sequences from the 3D MOBILE project. Table 1 shows the distortion metrics for these sequences. For every sequence, the table shows the 3VQM and PSNR values. Furthermore, we report the same metrics for the rendered sequence using the ground truth depth map sequence as a baseline for comparison. This baseline is labeled “Ground truth” in Table 1. From this work and others, it was shown that the PSNR does not coincide with subjective quality scores of 3D videos. We still report the PSNR values for comprehensiveness.

At first, we can analyze the results of intensity based depth extraction. From the 3VQM values in Table 1, at least one of the three channels produces a 3VQM value close to the baseline. In general, the 3VQM value obtained by the video generated using the depth estimate from luminance is high for Love Birds, Cafe and Kendo and it is close to the one obtained by the videos synthesized using the ground truth depth. However, the 3VQM values for the depth estimate from luminance are very low for the Pantomime and Champagne Tower sequences. Due to having a black background, the depth estimate from luminance obtained for these two sequences has a large number of unknown pixels. These can be observed in the background of the estimated depth map in Figure 3(b). The 3VQM values for the chrominance estimate of the two sequences are higher due to having a simple color structure. We notice that a sequence with rich colors, such as the Cafe sequence, performs better over C_R channel as compared to a sequence with high variations in colors, such as the Love Birds sequence. Overall, the estimates from luminance and chrominance result in high quality synthesized videos. The 3VQM values are consistent with our subjective evaluation of individual video sequences. The PSNR values
do not reflect a uniform pattern and our subjective evaluation of the sequences also confirmed that there are contradictions between the fidelity of the resulting 3D video sequences and the PSNR values.

Figure 3 shows examples of the ground truth and estimated depth maps along with their corresponding synthesized views from the Pantomime sequence. In terms of the overall subjective quality, the synthesis from the Y-based depth outperforms the other two despite the low 3VQM values, which was clarified earlier in this section. This can be observed by looking at Figure 3 (f)-(h). However, the Y-based depth fails to accurately capture non-uniformly textured area, such as the circled areas on the right in Figure 3 (f)-(h). These areas are best represented in the C_B-based synthesis (Figure 3 (h)). Furthermore, the edges are highly spatially distorted in the synthesized views from the C_R-based and C_B-based depths (Figure 3 (g) and (h)). These distortions around the edges vary between the two views depending on the color content. For example, heavily red areas are more distorted in C_R-based synthesized view, as can be observed in the circled area in the middle in Figure 3 (g). On the other hand, the circled areas on the left are more distorted in the C_B-based synthesized view (Figure 3 (h)). These spatial distortions are absent in the Y-based depth synthesized view, as seen in Figure 3 (f). Overall, the rendered views from the reconstructed depth maps are comparable to the rendered view from the ground truth depth map.

In the case of motion-texture based depth extraction, the 3VQM values obtained for the synthesized videos using the reconstructed depth maps are based on a value of $s = 0.7$ for Balloons, Champagne Tower, Kendo and Pantomime sequences. However, the 3VQM values for the Cafe and Love Birds sequences are obtained by setting $s = 1.0$, where the motion-based depth becomes more reliable due to the highly textured and random background. We developed and implemented an algorithm that jointly maps motion and texture to depth levels. From our experiments, the combination of motion and texture outperforms the separate mapping. Figure 4 shows results of three depth estimates. Motion-based depth estimation detects changes in the scene corresponding to the motion of foreground objects. For example, the hands and the arms are well represented in the motion-based depth map (Figure 4 (a)). However, texture-based depth estimation fails to account for mild-textured areas of foreground objects, such as the regions around the hand in Figure 4(b). When we combine texture and motion, individual estimates complement each other and we get a more convincing depth map as shown in Figure 4(c). Interestingly, the ground truth depth map does not differentiate between the wall or the background and the floor as shown in Figure 4(d). It is important to notice that the ground truth is not the perfect depth map and thus it might not be error-free as in this case. For this particular case, our combined depth estimate from texture and motion differentiate between the background and the floor as illustrated in Figure 4(c).

Depth map reconstruction methods and HHF algorithm complement each other in such a way that even we have clear artifacts in the extracted depth maps, we can obtain a high quality rendered view. In order to illustrate how our method works for various video sequences with different characteristics, extracted depth maps and rendered views are provided for all of the sequences that we have used for objective quality assessment in Figure [5-9].

Depth maps have the same number of pixels as the reference image. However, they do not contain separate channels for different color spaces and each pixel can be represented by 8 bits. The proposed method eliminates the necessity for transmitting the depth map and avoids the errors and degradations that can result from transmission. In order to illustrate the bitrate savings, size of the transmission packages that correspond to 2D plus depth map and 2D plus depth cues are tabulated in Table 2 for a single texture and depth frame. Table 3 illustrates bandwidth savings when we use our method of sending the color reference video with side information as depth cues as compared to sending reference video and depth map. All video streams (color and depth) are compressed using H.264 and the side information is not compressed. From the table, we have a saving between 14% to 40% while the quality is not compromised as in Table 1. Notably, in all our experiments we used the Hierarchical Hole Filling [9] (HHF) algorithm to fill all the gaps in frames synthesized from received or reconstructed depth maps.
### Table 2. Transmission package size and saving ratio for six different videos.

<table>
<thead>
<tr>
<th>Image Sequence</th>
<th>Compressed(2D + Depth map)(Bytes)</th>
<th>Compressed(2D) + Depth cues(Bytes)</th>
<th>Savings ratio(”%”)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balloons</td>
<td>3,153,920</td>
<td>2,363,394</td>
<td>74.94</td>
</tr>
<tr>
<td>Cafe</td>
<td>8,298,496</td>
<td>6,221,826</td>
<td>74.98</td>
</tr>
<tr>
<td>Champ. Tower</td>
<td>4,923,392</td>
<td>3,690,498</td>
<td>74.96</td>
</tr>
<tr>
<td>Kendo</td>
<td>3,153,920</td>
<td>2,363,394</td>
<td>74.94</td>
</tr>
<tr>
<td>Love Birds</td>
<td>3,153,920</td>
<td>2,363,394</td>
<td>74.94</td>
</tr>
<tr>
<td>Pantomime</td>
<td>4,923,392</td>
<td>3,690,498</td>
<td>74.96</td>
</tr>
</tbody>
</table>

### Table 3. Transmission package sizes and savings ratio for five different videos.

<table>
<thead>
<tr>
<th>Image Sequence</th>
<th>Compressed(2D + Depth map)(Bytes)</th>
<th>Compressed(2D) + Depth cues(Bytes)</th>
<th>Savings ratio(”%”)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balloons</td>
<td>254,648</td>
<td>155,686</td>
<td>38.86</td>
</tr>
<tr>
<td>Champ. Tower</td>
<td>240,189</td>
<td>161,478</td>
<td>32.77</td>
</tr>
<tr>
<td>Kendo</td>
<td>356,271</td>
<td>216,846</td>
<td>39.13</td>
</tr>
<tr>
<td>Love Birds</td>
<td>156,410</td>
<td>129,143</td>
<td>17.43</td>
</tr>
<tr>
<td>Pantomime</td>
<td>630,024</td>
<td>536,622</td>
<td>14.83</td>
</tr>
</tbody>
</table>

### 4. CONCLUSION

In this paper, we proposed a novel approach to extract the depth map using depth monocular cues from texture, motion and intensity. At the sender, we extract a set of depth cues from a given depth map. These cues were used at the receiver to estimate the depth map given the colored reference video by exploring texture, motion and intensity information in the sequence. We tested the estimated depth maps by assessing the quality of DIBR-based synthesized videos. In all cases, our approach produces 3VQM values and subjective quality close to the baseline. However, in order to guarantee a permanent high-quality standard, we have to perform a robust linear combination of these individual cues. Interactions between monocular cues should be taken into account to perform the linear combination adaptively. This paper will be the basis of our intended work to combine monocular cues for a high-quality standard at the receiver. We have worked on the reconstruction of depth map from the combination of monocular cues as it is explained in [11] and we will work on the assignment of reliability metrics so that adaptive linear combination would be possible.
Figure 6. Cafe depth maps and synthesized views.

Figure 7. Champagne Tower depth maps and synthesized views.

Figure 8. Kendo depth maps and synthesized views.

Figure 9. Love Birds depth maps and synthesized views.
References


