EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

A stereoscopic video conversion scheme based on spatio-temporal analysis of MPEG videos

Guo-Shiang Lin[1], Hsiang-Yun Huang[2], Wei-Chih Chen[2], Cheng-Ying Yeh[2], Kai-Che Liu[3] and Wen-Nung Lie[2,4*]

Abstract

In this article, an automatic stereoscopic video conversion scheme which accepts MPEG-encoded videos as input is proposed. Our scheme is depth-based, relying on spatio-temporal analysis of the decoded video data to yield depth perception cues, such as temporal motion and spatial contrast, which reflect the relative depths between the foreground and the background areas. Our scheme is shot-adaptive, demanding that shot change detection and shot classification be performed for tuning of algorithm or parameters that are used for depth cue combination. The above-mentioned depth estimation is initially block-based, followed by a locally adaptive joint trilateral upsampling algorithm to reduce the computing load significantly. A recursive temporal filter is used to reduce the possible depth fluctuations (and also artifacts in the synthesized images) resulting from wrong depth estimations. The traditional Depth-Image-Based-Rendering algorithm is used to synthesize the left- and right-view frames for 3D display. Subjective tests show that videos converted by our scheme provide comparable perceived depth and visual quality with those converted from the depth data calculated by stereo vision techniques. Also, our scheme is shown to outperform the well-known TriDef software in terms of human's perceived 3D depth. Based on the implementation by using "OpenMP" parallel programming model, our scheme is capable of executing in real-time on a multi-core CPU platform.

Keywords: Stereoscopic video conversion, Depth estimation, Depth cue, 3D perception, DIBR

1. Introduction

Recently, 3D (more accurately, stereo 3D) images/videos, which surely move our home audio-visual entertainment towards a greater perceptual realism, are attracting more attention in applications, such as multimedia, games, TV broadcasting, and augmented reality. With the advances in the technologies of 3D content capturing (e.g., dual-eye cameras or time of flight depth camera) and stereoscopic display, the influence of 3D videos on human beings' daily life are getting more important. Though many LCD-TV manufacturers are promoting their 3DTV products to the market from year 2010, the popularity is however limited by the availability of 3D video content. Though digital 3DTV broadcasting via Digital Broadcasting Satellite in Japan, 3D Digital Multimedia Broadcast system in Korea, and Advanced Three-dimensional Television System Technologies, and FP7 framework program [1] in Europe are currently in operation or under development, the sources of 3D video content are still not diverse enough. Since professional 3D video capturing devices are not so popular and normally expensive, the lack of sufficient amount of 3D video content motivates researchers to convert existing 2D videos into their stereoscopic versions [2]. With a rapid provision of abundant 3D video content, a quick progress in consumer electronics industry can thus be ensured.

The technique of converting a 2D video into a 3D stereoscopic version is called stereoscopic video conversion (SVC) or 2D-to-3D video conversion [3,4]. Recently, some researchers and companies, such as Dynamic Digital Depth (DDD), HDlogix, Sony Image Works, and Victor Company of Japan, have paid much attention on this technique [4]. One kind of SVC methods [5-7] tries to create stereo effect (for a two-view display) without estimating the depth map. This kind of depth-free methods relies on its ability in analyzing motion information and then directly synthesizing the left- and right-views

* Correspondence: ieewnl@ccu.edu.tw
[2]Department of Electrical Engineering, National Chung Cheng University, 168, University Rd., Ming-Hsiung, Chia-Yi 621, Taiwan
[4]Advanced Institute of Manufacturing with High-tech Innovations (AIM-HI), National Chung Cheng University, Chia-Yi, Taiwan
Full list of author information is available at the end of the article

from the original image sequence. The basic concept of this kind of methods is similar to structure from motion [4]. For example, Okino et al. [6] proposed a Modified Time Difference (MTD) method, by which binocular images are generated by selecting two frames with a time delay determined according to the magnitudes of estimated motion vectors (MVs). The problem to be solved is how to choose an appropriate matching frame for a given base frame. Another problem is how to find a suitable mapping between the disparity and the magnitude of an MV. The MTD method is however only suitable for image sequences with horizontal object motion. The work of Wang et al. [5] presents a similar strategy, but is restricted to image sequences without object motions. Kim et al.'s study [7] accepts MPEG-4 video as input, extracts the background Video Object Plane (VOP) and the primary (foreground) VOP, classifies the background motion type (left motion, right motion, or static) according to its MV field, and finally assigns disparities for the foreground and the background VOPs individually.

The other kind estimates a depth map for each 2D color frame and then synthesizes a pair of left- and right-view (i.e., binocular) images for stereo display [8-11]. Compared with the depth-free methods, this 2D-plus-depth format is more advantageous from the viewpoint of applications. For example, multi-view video for autostereoscopic displays can be generated based on a popular Depth-Image-Based-Rendering (DIBR) technique and the "perceived depth" can be adjustable under varying viewing conditions. In addition, the overhead for compressing the depth map is only about 20% in the bit rate [12], whereas that for compressing the secondary view (or, the right-view) might be over 40% [13].

Though depth plays such an important role in 3DTV applications, its derivation from a mono-view image or video is really challenging. In [8], depth maps were built by using only MVs extracted from the compressed video. Obviously, estimating depths solely from the motion cue cannot be suitable for various types of videos. In [9], image depths were calculated by measuring and combining other cues such as contrast, sharpness, and chrominance of the input image. In [14], an object-based SVC method was proposed, where depth ordinal (based on occlusion reasoning) and depth consistency between detected objects are analyzed for depth estimation.

Other studies [9,15], on the other hand, emphasize on separating foreground from background, estimating depths individually, and then combining them into a single depth image. To provide an impressive stereo effect, the estimation of a background depth profile is necessary. For example, Angot et al. [16] define six profiles of background depths and select a proper one according to image features. Another popular method is to establish the depth geometry of backgrounds by detecting vanishing points/lines [17] in the image. However, line features necessary for vanishing point/line detection might not be apparent or even do not exist in a video sequence.

A well-known software TriDef, developed by DDD, adopts an off-line machine learning algorithm [10,18], where image features, such as color components and 2D coordinates, are used to construct a relation between an image pixel and its associated depth by using a classifier (e.g., neural network). Since it is straightforward and fast to extract these image features and make depth estimation (using neural classifier as an estimator), the conversion can be achieved easily in real time. However, there is a severe drawback that pixels locating at the lower and central parts of an image are likely to be assigned with nearer depths, resulting in a smoothly tilted depth pattern for most input images. That is, TriDef presents unlayered depths in light of any image content.

Most of the SVC methods [3,9,14,19] estimate depths solely from cues in a single frame. The drawback is that depth fluctuation may occur in temporal domain (since depth cue estimation is an ill-posed problem and thus unstable) and hence make viewers uncomfortable. It thus demands that depths be estimated by referring to information from more previous frames. However, a bulk of buffers to store information propagated from previous frames should be avoided for practical consideration.

From the viewpoint of human aid, SVC methods can be categorized into three classes: manual, semi-automatic, and fully automatic [4]. Though manual and semi-automatic SVC methods can provide high-quality depth, their drawback is heavy time consumption. On the other hand, fully automatic SVC methods can convert existing 2D videos to 3D content in a more efficient manner. Therefore, in this article, we aim at developing a depth-based automatic SVC scheme to generate stereoscopic videos with the popular MPEG videos as the input. Our method is capable of automatically detecting shot change, classifying the following video shot, and accordingly performing proper algorithms and tuned parameters to estimate initial depth maps based on depth cues from spatio-temporal analysis. Subtitles play an important part in most of the commercial videos. Its depth arrangement in 3D videos will certainly affects viewers' comfort. The detection of subtitle regions and their corresponding depth assignment are developed in this article. Furthermore, interpolation and recursive temporal filtering of the initially estimated depth maps are performed to make depth edges conform to color edges and avoid depth fluctuation, respectively. Since the temporal filtering is done recursively, extra buffering of information from previous frames is kept a minimum. To make real-time conversion, a reality solely by

software, parallel programming on a multi-core CPU platform is also implemented.

The remainder of this article is organized as follows. Section 2 describes the design concept of our scheme. Sections 3, 4, and 5 elaborate the pre-processing, depth estimation, and post-processing steps of our scheme, respectively. In Section 6, experiment results are given and finally Section 7 draws some conclusions and future work.

2. An overview of system design

As addressed before, it is difficult to estimate satisfactory depths from a single or a couple of frames via a universal algorithm. This is partly due to the insufficiency of image cues absolutely relating to depths and partly due to the fact that image cues often vary unstably along the whole sequence. Hence, it demands to have a shot-based SVC scheme, of which the depth scenario/geometry and associated image features are assumed similar within a shot. In addition, our goal is to develop a real-time SVC software which accepts a compressed video bit stream as the input and operates on popular multi-core CPU platforms. To this end, several design key points are proposed.

(1) shot classification is necessary to enable shot-adaptive depth estimation;
(2) motion information (e.g., MVs) is extracted directly from the compressed bit stream, but not re-estimated from the reconstructed pixels, for speed consideration;
(3) there should have a less number of references to previous frames in depth estimation to prevent the requirement of large memory buffers and a long time delay;
(4) 3D perception artifacts in the temporal domain are much more observable than those in the spatial domain and should be eliminated with priority; and
(5) parallel processing on prevailing multi-core platforms should be optimized for speedup.

Figure 1 illustrates the block diagram of our proposed SVC scheme, which is composed of pre-processing, depth estimation, post-processing, and view synthesis. Pre-processing includes MPEG decoding, shot change detection, MV refinement, camera motion estimation/compensation, and MV interpolation. After detecting shot change, the information following the shot boundary frame is sent for classification of next coming shot, whose result determines the manner that a shot should be processed. Refinement of MVs is necessary since MPEG MVs based on the criterion of least residue energy may not be suitable for motion analysis. Re-estimation of MVs by other methods (e.g., optical flow) is however not suggested for real-time consideration. Camera motion estimation/compensation is demanded for SVC schemes that consider true object motion as an important cue for depth estimation. On the other hand, MV interpolation is capable of increasing the spatial resolution of the estimated depth image for better 3D perception quality.

As for depth estimation, we adopt a shot-adaptive strategy, by which features including inter-frame difference, frame complexity, and camera motion parameters at the shot-change boundary frame are analyzed for shot classification. Four shot categories are designed for analysis. To be consistent with MPEG videos, all computations of depth cues in spatial domain are block-based. To make depths smooth within an object and sharp near object boundaries, a depth-based foreground segmentation algorithm is developed for further depth refinement.

As pointed out in [20], spatial misalignment of edges in the depth and color images will degrade the stereo visual quality. This means that spatial blockiness and jerkiness in the depth map should be avoided. On the other hand, most of the current SVC works place less emphasis on solving perception artifacts resulting from temporal depth inconsistency. Hence, our post-processing stage is to scale up the block-based initial depth map, align it to color edges, and simultaneously make it smooth in the temporal domain. To achieve this, the Joint Bilateral Upsampling (JBU) algorithm [21] is first modified to interpolate the estimated depth map and then recursive temporal filtering is performed to eliminate possible 3D perception artifacts (e.g., depth fluctuation) resulting from wrong depth estimations.

As for view synthesis, the popular image warping technique, DIBR [22], is applied to construct the left- and

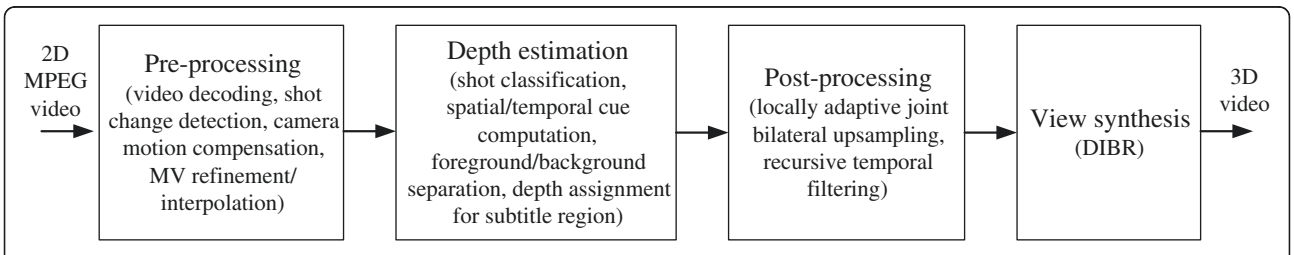

Figure 1 Block diagram of the proposed SVC scheme.

right-views for 3D display. We elaborate each part of the proposed SVC scheme in the following sections.

3. Pre-processing

The procedures of pre-processing are detailed in Figure 2.

3.1. Shot change detection

Histogram difference (*HD*) is usually adopted as a feature to detect shot changes due to its simplicity and less sensitivity to camera and object motions [23]. A frame is first divided into 16 regions and *HD* is calculated for each region as

$$HD(t,u,v) = \sum_{j=1}^{N_b} |H(t,u,v,j) - H(t-1,u,v,j)|, \quad 0 \le u, v \le 3 \tag{1}$$

where $H(t,u,v,j)$ and $H(t-1,u,v,j)$ represent the jth bins of the histograms of the (u,v)th region in the tth and $(t-1)$th frames, respectively, $|\cdot|$ denotes the absolute operator, and N_b is the number of bins (here $N_b = 256$). After computing $HD(t,u,v)$, the number of regions whose content significantly differs from k previous frames is computed as

$$SHD(t) = \sum_{u=0,v=0}^{3,3} U\left(HD(t,u,v) - \min\left(T^S(u,v), \frac{\alpha}{k} \cdot \sum_{j=t-k}^{t-1} HD(j,u,v) \right) \right), \tag{2}$$

where α is a pre-defined constant, $T^S(u,v)$ is a region-dependent threshold, $\min(\cdot)$ represents the minimum operator, and $U(\cdot)$ denotes the unit step function $U(x) = 1$ for $x > 0$ and $U(x) = 0$ for $x \le 0$. The average *HD* value over the k past frames is calculated and used as an alternative threshold to adapt to various scenic changes. Since humans often pay more attention to the central zones of a frame, their corresponding $T^S(u,v)$ are set higher than others. The event of shot change can then be detected by thresholding $SHD(t)$.

3.2. MV refinement and compensation

According to [24], motion parallax is the dominant depth cue for human beings at a viewing distance of less than 10 m. It can be revealed by image MVs which are often inversely proportional to the depth of a moving object (i.e., a larger MV possibly corresponds to a nearer object). However, MVs retrieved from MPEG videos are encoding-oriented and might be incorrect from the viewpoint of motion analysis. They should be refined before being further analyzed or processed.

The procedure of refining MVs is similar to that proposed in [25], but much easier and faster for implementation. First, four bins (corresponding to the four quadrants in the Euclidean plane) are prepared for direction histogramming of the 3×3 MVs around a considered MacroBlock (MB, 16×16 pixels). The dominant one is found if the cluster (bin) size is above a threshold (e.g., 4). MV of the current MB is replaced with the mean MV of the dominant cluster if it does not belong to that, but remains unchanged otherwise. The result of this sub-procedure is denoted as (MV_H^C, MV_V^C).

In addition, MVs should be compensated with the amount of camera motion, which often occurs in home and movie videos, to find true object motions. Techniques of estimating camera motion (also called global motion) parameters can be seen in [26] and will not be discussed in detail here. After estimating the camera motion parameters (e.g., pan, tilt, zoom, rotation), the above refined MV can be further compensated as

$$MV_H^S = MV_H^C - \Delta H \quad \text{and} \quad MV_V^S = MV_V^C - \Delta V \tag{3}$$

where ΔH and ΔV stand for the compensation amounts calculated according to the camera motion parameters.

3.3. MV interpolation

As is well known, MVs retrieved from most of the MPEG videos are MB-based, except those encoded with the 4MV

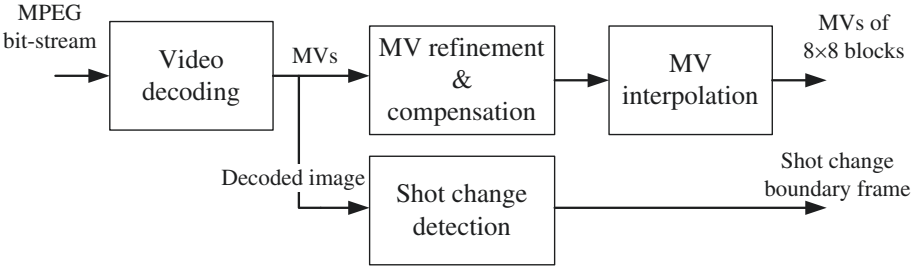

Figure 2 Block diagram of pre-processing.

mode. Our MV interpolation for each block of 8 × 8 pixels is based on an affine motion model [27], which can be expressed as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \times \begin{bmatrix} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 \end{bmatrix}^T, \quad (4)$$

where (x, y) and (x', y') represent the coordinates of corresponding point pair between two consecutive frames, p_1–p_6 are transform parameters, and the superscript symbol T denotes the transpose operator. The point pairs established by at least 3 MVs of an MB and its 8-neighboring ones could be used to derive a system of equations based on Equation (4) for solving p_1–p_6 by using the least square error method.

Based on the computed p_1–p_6, the corresponding point (x', y') in the reference frame for a given block (8 × 8 pixels) can be obtained by substituting the (x, y) coordinates of its left-upper corner point into the right-hand side of Equation (4). The interpolated MV for that block can then be derived as $(x' - x, y' - y)$. In case of insufficient point-pairs for solving p_1–p_6, the MV of an inter-coded MB is copied to its four descendants, or a maximum value (usually the value of search range for motion estimation) is assigned for intra-coded MBs.

4. Depth estimation

Figure 3 illustrates the block diagram of the depth estimation and post-processing procedures. In this section, we introduce the procedures of depth estimation.

4.1. Shot classification

The performance of a non-adaptive SVC scheme is not satisfactory in dealing with different kinds of video content. In the proposed SVC scheme, each segmented shot would be classified into four categories: (C1) neither object nor camera motion exists, (C2) no object motion but camera motion exists, (C3) object motion exists and frame complexity is low, and (C4) object motion exists and frame complexity is high. To determine the category of a video shot, features in terms of inter-frame difference, frame complexity, and camera motion parameters are calculated. Note that due to real-time requirement, shot classification is based on features calculated merely from the shot change boundary frame (instead of frames of the whole shot). That is, the result of shot classification (C1–C4) will endure until next shot change boundary frame is detected and re-classified.

Inter-frame difference between two adjacent frames is computed as

$$FHD(t) = \sum_{u,v=0}^{3} U\left(T^F(u,v) - HD(t,u,v)\right), \quad (5)$$

where $T^F(u,v)$ is a threshold smaller than $T^S(u,v)$ in Equation (2). As for the measure of frame complexity, the variance of pixel values in a frame is concerned. It is intended that the larger the variance is, the higher the frame complexity is.

If $FHD(t)$ (t represents the frame index of the detected shot change boundary) is smaller than a threshold, the corresponding shot thereafter is classified as the C1 category. Otherwise, if correlation of the MVs is high (i.e., a large portion of MVs are consistent in terms of direction and magnitude), C2 is then identified. When C1 or C2 is not identified and the frame complexity is low, C3 will be identified. Otherwise, the shot is assigned with C4. The shot classification procedure is summarized as in Figure 4. For different categories of shots, different depth estimation algorithms or parameters will be adopted.

4.2. Initial depth estimation

Human visual system perceives depth by combining multiple cues [28] from all domains to estimate distances of objects or relative displacements between them. Popular monoscopic depth perception cues known to the human beings include motion parallax, texture gradient, brightness, atmospheric perspective, linear perspective, and so on [4,29]. Therefore, one issue of SVC is to compute monocular cues and suitably fuse them to obtain stereoscopic information. In our system, the frame next to the shot change boundary is used for initial depth estimation.

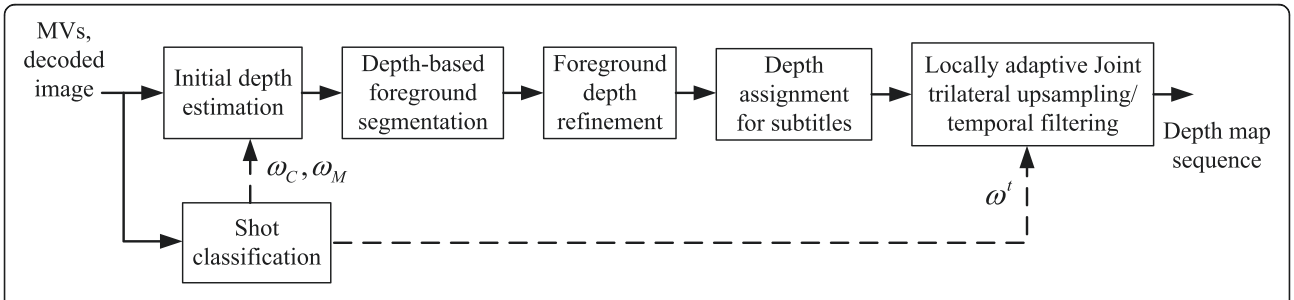

Figure 3 Block diagram of depth estimation and post-processing.

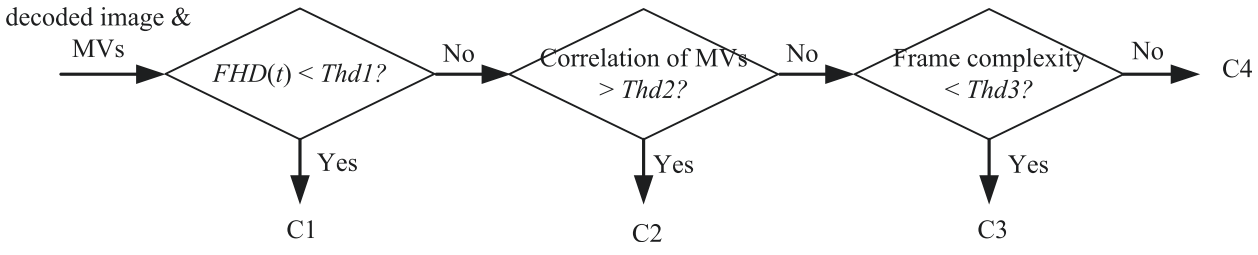

Figure 4 Shot classification procedure.

The motion parallax [4,29] describes the relative motion of objects against the background. It is the dominant depth cue for human beings at a viewing distance of less than 10 m [24] and has popularly been adopted for depth estimation [11,19]. Atmospheric perspective [4,29], also called aerial perspective, explains the impact of space or atmosphere between an object and an observer on the appearance of an object. Atmospheric perspective induces a phenomenon that a far object looks hazy or is of low contrast [4,29]. Here, we devise an algorithm of depth estimation based on motion parallax and atmospheric perspective.

4.2.1. Motion parallax cue

Similar to the studies of [11,19], the magnitude of MV is used to estimate the distance of a viewed object. The cue f^M based on motion parallax is defined as follows:

$$f^M(t,u,v) = \sqrt{MV_H^2(t,u,v) + MV_V^2(t,u,v)} \tag{6}$$

where MV_H and MV_V are the horizontal and vertical components, respectively, after MV interpolation, and (u,v) denotes the (8×8) block index. Note that at the shot change boundary frame, the motion cue is unreliable and should be ignored. On the other hand, the MV field of the previous frame is retained if the current frame is intra-coded (i.e., I frame) and not at a shot change boundary.

4.2.2. Atmospheric perspective cue

In [15,30], high-order statistics are measured as depth cues. Their disadvantage is the sensitivity to noise. For real-time and robustness considerations, we define a contrast term f^C based on Michelson contrast measure [31] to reflect the atmospheric perspective cue of each block as below

$$\bar{I}(t,u,v) = \frac{1}{64}\sum_{i,j=0}^{7} I(t,u+i,v+j), \tag{7}$$

$$I^H(t,u,v) = \frac{\sum_{i,j=0}^{7} I(t,u+i,v+j) \times U(I(t,u+i,v+j) - \bar{I}(t,u,v))}{\sum_{i,j=0}^{7} U(I(t,u+i,v+j) - \bar{I}(t,u,v))}, \tag{8}$$

$$I^L(t,u,v) = \frac{\sum_{i,j=0}^{7} I(t,u+i,v+j) \times U(\bar{I}(t,u,v) - I(t,u+i,v+j))}{\sum_{i,j=0}^{7} U(\bar{I}(t,u,v) - I(t,u+i,v+j))}, \tag{9}$$

and

$$f^C(t,u,v) = \frac{I^H(t,u,v) - I^L(t,u,v)}{I^H(t,u,v) + I^L(t,u,v)}, \tag{10}$$

where $I(t)$ is the tth luminance image of original resolution, $\bar{I}(t, u, v)$ denotes the mean value of the (u,v)th block, and $I^H(t,u,v)$ and $I^L(t,u,v)$ represent the average values of pixels above and below $\bar{I}(t, u, v)$, respectively. According to Equation (10), we can observe that the smaller the f^C is, the lower the contrast is. In addition, since $I^H(t,u,v)$ and $I^L(t,u,v)$ are the average values, it is expected that f^C is more robust than that computed traditionally.

4.2.3. Combination of depth cues

According to [28], the overall depth can be estimated as a weighted combination of different depth cues. Since each shot is classified into four categories, as addressed in Section 4.1, the initial depth d^E for each frame therein is calculated adaptively with different parameters below:

C1: $d^E(t,u,v) = d^E(t-1,u,v)$

C2: $d^E(t,\ u,\ v) = \hat{f}^M(t,\ u,\ v)$

C3: $d^E(t,u,v) = \omega_M \cdot \hat{f}^M(t,u,v) + \omega_C \cdot \hat{f}^C(t,u,v)$,
where $\omega_M = 0.6$, $\omega_C = 0.4$;

C4: $d^E(t,u,v) = \omega_M \cdot \hat{f}^M(t,u,v) + \omega_C \cdot \hat{f}^C(t,u,v)$,
where $\omega_M = 0.8$, $\omega_C = 0.2$;

where ω_M and ω_C are pre-determined weighting parameters of the motion parallax and atmospheric perspective cues, respectively, and $\hat{f}^M$ and $\hat{f}^C$ are normalized versions (between 0 and 255) of f^M and f^C, respectively. Values of ω_M and ω_C are determined experimentally. Note that normalization is performed before weighted combination. Larger $\hat{f}^M$'s and larger $\hat{f}^C$'s lead to larger d^E's, which stand for nearer distances. Though more cues can be collected for more accurate depth estimation, only two cues (one from temporal domain and the other from spatial domain) are chosen for speed consideration.

4.3. Depth-based foreground segmentation and foreground depth refinement

By binary thresholding on initial depth map d^E, the foreground area (represented by an object mask Ω_t^F) can be segmented out.

It is unavoidable that some holes exist within the object mask. Hence, morphological operations [32] are first applied to Ω_t^F to bridge the gaps of contours and then a hole-filling process is performed thereto. Also, object areas of small size are removed from Ω_t^F. We denote the object mask after the afore-mentioned shape refinement process as $\tilde{\Omega}_t^F$. Accordingly, depths corresponding to the hole-filled foreground regions should be subject to refinement (denoted as $\tilde{d}^E$) to make the depth map spatially smooth. A scan-line-based linear interpolation algorithm can work well. After foreground depth refinement, the low-resolution depth map can be expressed as follows:

$$d^L(t,u,v) = \begin{cases} \tilde{d}^E(t,u,v) & \text{if}(t,u,v) \in \tilde{\Omega}_t^F \\ d^E(t,u,v) & \text{if}(t,u,v) \notin \tilde{\Omega}_t^F \end{cases}. \quad (11)$$

Note that all the above processes in Sections 4.2 and 4.3 are based on the blocks of 8×8 pixels, hence speeding the processing.

4.4. Depth assignment for subtitles

In addition to the video content, the impact of the subtitle's depth on 3D visual quality should be also concerned. Though some commercial software [33] provides the capability of manually adding external subtitles to video and assigning depths for them simultaneously, it is however our focuses to automatically detect subtitles embedded in frames and assign the depths for them (to the best of the authors' knowledge, few works discussed this issue). Flickering or depth fluctuation of the subtitles will certainly lower down the perceived visual quality and make viewers uncomfortable, which motivates us to assign a constant depth for the detected subtitles along the whole sequence. However, we still face a problem of maintaining a constant depth for the whole subtitle area or for individual characters in subtitles. Since the later alternative necessitates precise and stable character segmentation for the same subtitle that endures several frames, the former one is selected in our system in considering stable quality.

A subtitle is usually placed at the lower part of a frame, which makes its localization easy. For the detection of a subtitle area, features of contrast, edge, and color are used. Only blocks of high contrast (previously calculated as f^C in Equation (10), strong edge strength (calculated via Sobel operator), near subtitle color (here, pre-defined as white), and proper position are identified. Figure 5 shows an example, where Figure 5a is the decoded frame and Figure 5b,c demonstrates the detected subtitle region. Since the four boundaries of a subtitle region are block-aligned, a stable depth for subtitles occurring in consecutive frames can easily be achieved.

5. Post-processing

The depth map estimated in Section 4 (i.e., $d^L(t,u,v)$ in Equation 11) is block-based for speed consideration. It should be scaled up to the pixel level and aligned to conform to the color edges for better 3D perception. We describe the post-processing in detail here

5.1. Locally adaptive joint trilateral upsampling (LA-JTU)

Notice that edge misalignment between color and depth images after depth upsampling may cause visual artifacts when synthesizing the left- and right-view images. To enlarge $d^L(t,u,v)$ and spatially smooth it with the depth edges being registered to the corresponding color edges, a JBU algorithm has ever been proposed [21]. Its rationale is to interpolate and smooth a depth map, while preserving the edge information, by computing a weighted average for each pixel (x, y) in the high-resolution depth image d^H. Within a local window $\Omega_{(x,y)}$ centered at (x, y), each pixel (x', y') is associated with a weight which is a function of the Euclidean distance and color difference with respect to the central one. However, JBU cannot function well when (x, y) and (x', y') (1) have similar colors but different depths, or (2) have different colors but similar depths. In these two cases, wrong depth interpolations for (x, y) will cause the ghost and flickering artifacts.

Our modifications to JBU are twofolds: (1) adding an extra depth-weight term to form the so-called trilateral filter, and (2) the weighting is locally adaptive. First, we detect depth discontinuity for each window $\Omega_{(x,y)}$ by computing $d_{\max}$ and $d_{\min}$ as below

$$d_{\max} = \max\left\{ d^L\left(t, \left\lfloor \frac{x'}{8} \right\rfloor, \left\lfloor \frac{y'}{8} \right\rfloor\right), \text{where } (x',y') \in \Omega_{(x,y)} \right\} \quad (12)$$

$$d_{\min} = \min\left\{ d^L\left(t, \left\lfloor \frac{x'}{8} \right\rfloor, \left\lfloor \frac{y'}{8} \right\rfloor\right), \text{where } (x',y') \in \Omega_{(x,y)} \right\}, \quad (13)$$

where max{} denotes the maximum operator and $\lfloor \cdot \rfloor$ is the floor operator. When the difference between $d_{\max}$

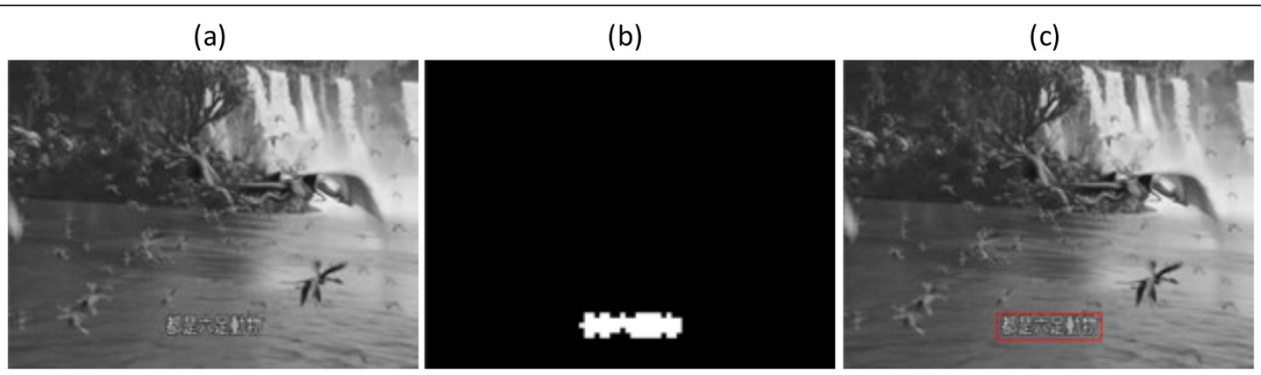

Figure 5 An example of subtitle detection: (a) decoded image, (b) detected subtitle blocks, and (c) detected subtitle region.

and $d_{\min}$ is small (i.e., a smooth depth area), the depth of (x, y) is interpolated according to the depth-weighting term only (the color-weighting is ignored). Otherwise (i.e., a significant depth discontinuity exists), both the color and depth weights are considered. Defining

$$(u,v)=\left(\left\lfloor\frac{x}{8}\right\rfloor,\left\lfloor\frac{y}{8}\right\rfloor\right)\text{and}(u',v')=\left(\left\lfloor\frac{x'}{8}\right\rfloor,\left\lfloor\frac{y'}{8}\right\rfloor\right),\quad(14)$$

which stand for the corresponding block index in the low-resolution depth map for the pixel pair (x, y) and (x', y'), our LA-JTU algorithm to derive high-resolution depth map $d^H(t, x, y)$ is expressed as

$$d^H(t,x,y)=\frac{\begin{array}{l}\sum_{(x',y')\in\Omega_{(x,y)}} d^L(t,u',v')\cdot f_1'(|I(t,x,y)\\ \qquad -I(t,x',y')|)\cdot f_2(|d^L(t,u,v)\\ \qquad -d^L(t,u',v')|)\cdot f_3(x',y'))\end{array}}{\begin{array}{l}\sum_{(x',y')\in\Omega_{(x,y)}} f_1'(|I(t,x,y)\\ \qquad -I(t,x',y')|)\cdot f_2(|d^L(t,u,v)\\ \qquad -d^L(t,u',v')|)\cdot f_3(x',y'))\end{array}}\quad(15)$$

$$f'_1(|I(t,x,y)-I(t,x',y')|)=\begin{cases}1, & \text{if } d_{\max}-d_{\min}\geq T^d\\ f_1(|I(t,x,y)-I(t,x',y')|), & \text{if } d_{\max}-d_{\min}<T^d\end{cases}\quad(16)$$

$$f_1(|I(t,x,y)-I(t,x',y')|)=2^{\sum_{q\in\{r,g,b\}}-|I_q(t,x,y)-I_q(t,x',y')|/0.125},\quad(17)$$

$$f_2(|d^L(t,u,v)-d^L(t,u',v')|)=2^{-|d^L(t,u,v)-d^L(t,u',v')|/0.125},\quad(18)$$

$$f_3(x',y')=\begin{cases}1, & \text{if } (x',y')\in\Omega_{(x,y)}\\ 0, & \text{if } (x',y')\notin\Omega_{(x,y)}\end{cases}\quad(19)$$

where T^d is a threshold and I_q, $q = r, g, b$, represent the three color components.

5.2. Temporal filtering

As noted, the depth map $d^H(t, x, y)$ is created by referring only two consecutive frames (i.e., t and $t - 1$). This has the disadvantage of unstable depths frame-by-frame, especially when the motion or contrast cue fluctuates due to varying lighting. According to experiences, depth artifacts in the temporal domain are much more harmful to human's 3D perception quality than those in the spatial domain. Also note that large depth variations will result in a discontinuity or bending of the object contours (especially lines in the vertical direction) after image rendering (discussed later). In fact, the distortion of object contours and horizontal/vertical lines in an image is an important factor in measuring video quality [34]. This motivates us to apply a temporal filtering to $d^H(t, x, y)$'s before they can be used for image rendering.

Ideses et al. [17] adopted local 3D-DCT (optional) and median filtering to process the noisy depth maps. The amount of neighborhoods for temporal filtering was seven frames. Obviously, a large memory for buffering the depth maps is necessary. Here, we exploit a memory-efficient recursive temporal filter to achieve the same purpose (temporal median filtering with a window size less than five frames

has ever been tested in this study, but is seen to have an inferior performance). The recursive temporal filtering is expressed as

$$\tilde{d}^H(t,x,y) = \omega^t \cdot \tilde{d}^H(t-1,x,y) + (1-\omega^t)\cdot d^H(t,x,y), \quad (20)$$

where ω^t, $0 < \omega^t < 1.0$, is the weighting factor to determine the temporal smoothness of the depth map; $\tilde{d}^H(t,x,y)$ is the filtered result. Obviously, the prior estimated depth $d^H(t-k,x,y)$ has an exponentially weighted contribution, $(\omega^t)^k \cdot (1-\omega^t) \cdot d^H(t-k,x,y)$, to $\tilde{d}^H(t,x,y)$. Taking $\omega^t = 0.75$ for example, only 8% of contribution is left after four frames of decay, i.e., $0.08 \cdot d^H(t-4,x,y)$. This kind of exponentially weighted running average filter requires no window size definition and only one buffer is needed to store the past $\tilde{d}^H(t-1,x,y)$. Note that at the shot change boundary frame, ω^t is set to 0 to prevent incorrect depth propagation (see Figure 3).

6. Experiment results

To evaluate the performance of our proposed SVC scheme, several image sequences, e.g., "Breakdancer", "Flamenco", "Akko&kayo", "Ballet", "Close to you", "True legend", "2012", "New moon", and one music video, whose frame sizes are all 640×480 pixels, are used for testing. Among them, depths calculated by using stereo vision techniques are provided for Breakdancer and Ballet (thanks to Microsoft Co. [35]) and considered as ground truths. All the video clips are MPEG-encoded (in an encoding structure of "IPPP. . .I") at a frame rate of 30 Hz. The parameter settings in post processing are (1) a window size (i.e., $\Omega_{(x,y)}$) of 17×17 pixels for LA-JTU and (2) $\omega^t = 0.75$. The converted 3D videos are played on an Acer 3D notebook (Aspire 5738DG) with a 3D display (odd-even interleaved scan lines, viewed with polarizing glasses).

To evaluate the goodness of the produced 3D content, both objective and subjective tests [31,34] are conducted. We measure not only perceived depth (i.e., 3D effect) and visual quality of synthesized frames, but also temporal smoothness of estimated depth maps. Here, a metric φ is adopted to measure the depth inconsistency/variation between adjacent depth maps:

$$\varphi(t) = \frac{100}{N_H N_W} \sum_{\substack{0 \le x < N_H \\ 0 \le y < N_W}} U\left(\left| \tilde{d}^H(t,x,y) - \tilde{d}^H(t-1,x,y) \right| > T^{\varphi} \right), \quad (21)$$

where T^{φ} is a pre-defined threshold and N_H and N_W are the height and width of a frame, respectively.

The better the three indices (perceived depth, visual quality, and temporal smoothness) are, the better the performance of an SVC method is. For subjective tests, five grades, similar to mean opinion score (MOS) described in [31], are adopted: 5 (excellent), 4 (good), 3 (fair), 2 (poor), and 1 (bad).

6.1. Depth estimation and view synthesis

Figure 6 shows the results of three test shots in "2012", "Close to you", and "New moon", which are classified as C2, C3, and C4, respectively. Figure 6b–e demonstrates the results of motion cue, contrast cue, estimated depth map, and synthesized image, respectively. Since no contrast cue ($\omega_C = 0$) is used in C2, the third image of the first column is left blank.

It is observed from Row 4 of Figure 6 that combination of motion and contrast cues according to the shot-classification result is effective in identifying the foreground objects (especially for Figure 6d2, d3). The result demonstrates that block-based initial depth estimation, enhanced with LA-JTU and temporal filtering, is sufficient to provide satisfactory depth maps for stereo conversion, while keeping the conversion time limited for real-time applications (see Section 6.5 later). As for the detection of the subtitle region, the result in Figure 6d2, where the constant depth is set to 255, is also satisfactory.

6.2. Evaluation of depth assignment for subtitles

We experiment with four types of depth assignment for subtitles: (1) $\tilde{d}^H(t,x,y)$, (2) constant 255, (3) constant 128, and (4) constant depth to all segmented characters in subtitles. The subjective test results in terms of MOS are 3.11, 4.21, 4.061, and 3.063, respectively. This result matches the analysis in [2]. Obviously, a bad visual comfort (e.g., flickering artifact) will be perceived for human beings when discontinuity of depths in subtitle region occurs in either the temporal (type 1) or spatial (type 4) domain. In addition, the difference in human perception is not so significant when the constant depth value is changed (types 2 and 3).

6.3. Effectiveness of recursive temporal filtering

Figure 7 demonstrates the synthesized left view of "Ballet" with/without recursive temporal filtering in our scheme. It is found that due to temporal variations in $d^H(t,x,y)$, zigzag artifacts around the vertical edges (marked with red circles) are resulted. This artifact is eliminated after temporal filtering is enabled (the second row in Figure 7). Other artifacts in the form of background pop-up due to changing lighting are also reduced. Figure 8 illustrates another example for the effectiveness of temporal filtering.

Evaluation of depth inconsistency based on Equation (21) for "Breakdancer" and "Ballet" is shown in Figure 9, where T^{φ} is set to 10. Obviously, the solid curves (with temporal filtering) are leveled down with respect to the

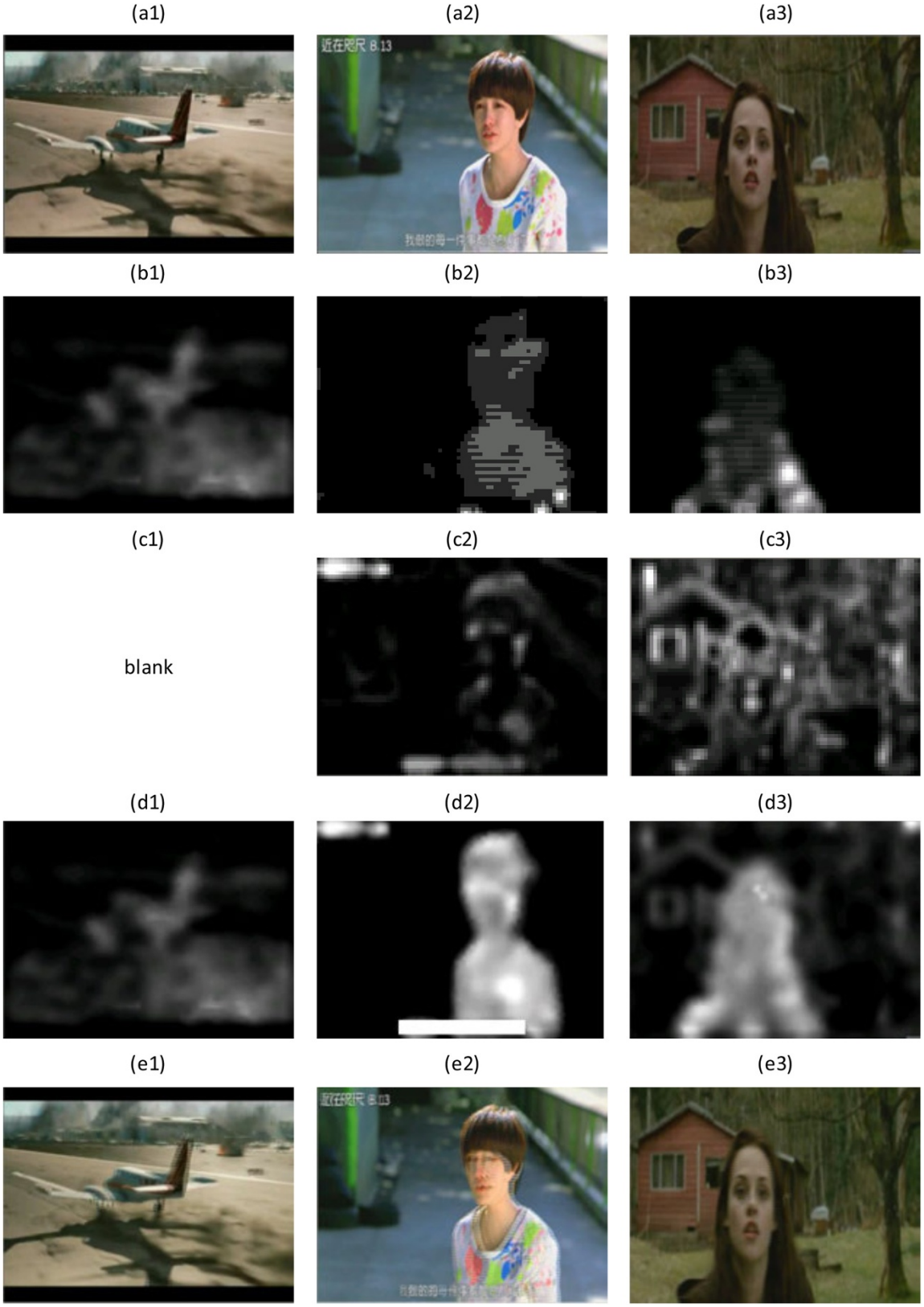

Figure 6 Results of depth estimation and view synthesis. Row 1, decoded image; Row 2, motion cue; Row 3, contrast cue; Row 4, estimated depth map (high resolution); Row 5, synthesized interlaced image; Column 1, "2012"; Column 2, "Close to you"; Column 3, "New moon".

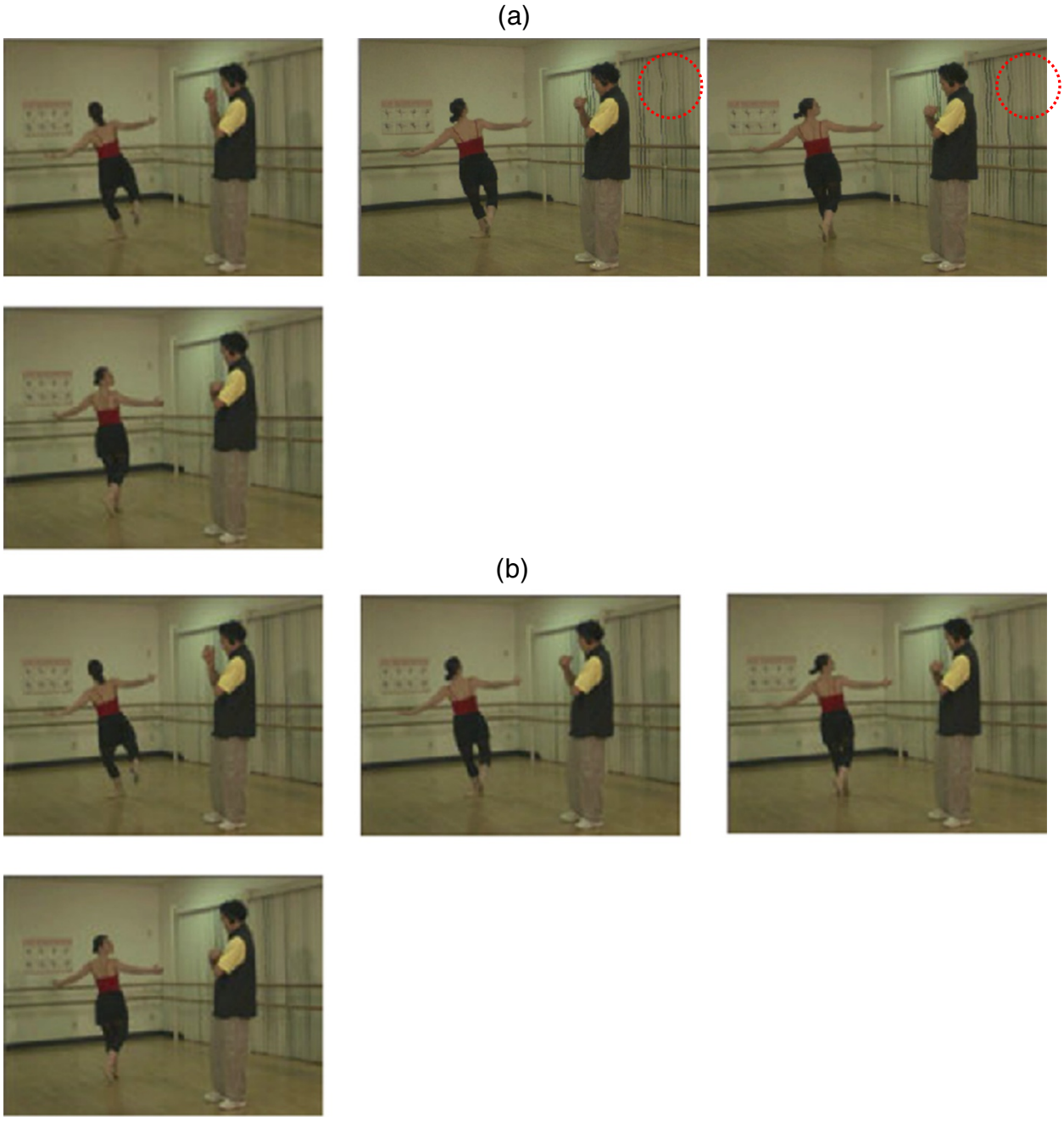

Figure 7 Synthesized left views for "Ballet": (a) without recursive temporal filtering, (b) with recursive temporal filtering. The sequence starts from left to right.

dashed curves (without temporal filtering). In addition, the standard deviations of φ for "Breakdancer" and "Ballet" are reduced from (6.34, 12.5) to (3.91, 6.41) when enabling recursive temporal filter. Also note that depth inconsistency for "Breakdancer" is larger than that for "Ballet", reflecting the faster movement or larger depth variation for Breakdancer.

6.4. Subjective comparison with TriDef

TriDef 3D software [4,18] is implemented with DDD's unique SVC scheme to make existing 2D photos and movies viewable in 3D perception. Here, subjective tests are conducted on seven video clips to show the human perception difference between the 3D videos converted by using our SVC scheme and by using TriDef.

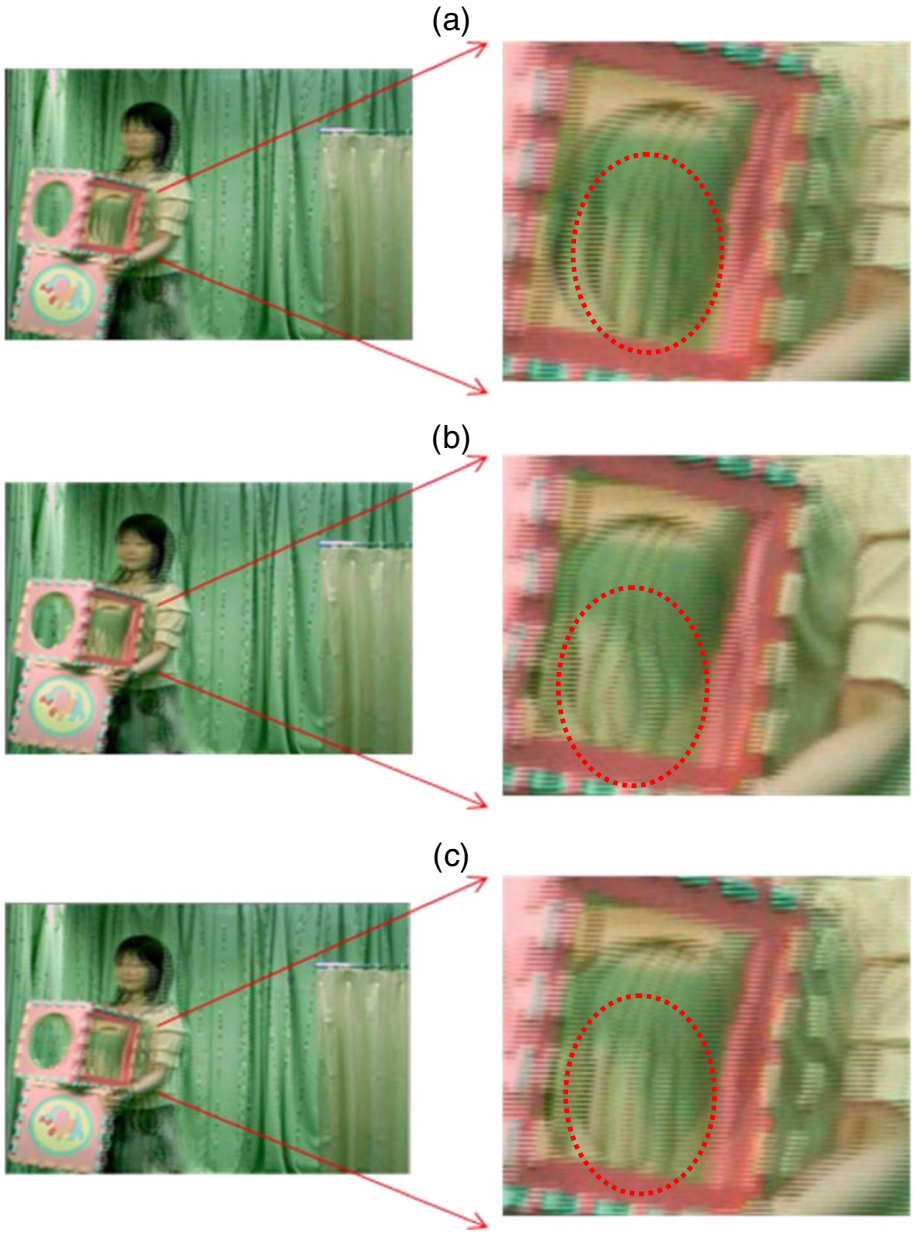

Figure 8 Synthesized interlaced views for "Akko&kayo": (a) previous frame, (b) current frame without temporal filtering, (c) current frame with temporal filtering.

The video clips are randomly displayed to 12 testees in the subjective test. The testees are requested to score on the "perceived depth" and "visual quality". Since depths of "Ballet" and "Breakdancer" are provided with ground truths, 3D video converted from them are also compared. As shown in Table 1, the perceived depths for 3D videos converted from our estimated depths and from ground truths are considerable; both outperform TriDef's. On the contrary, TriDef provides better visual quality for the synthesized left- and right-view images. Table 2 showing the perceived depths for the other test video reveals similar performances.

According to Tables 1, 2, and Figures 7, 8, and 9, the proposed SVC scheme can function well in terms of perceived depth, visual quality, and temporal smoothness for several kinds of videos.

6.5. Parallel programming

The proposed SVC scheme is implemented, based on OpenMP parallel programming model, in a personal

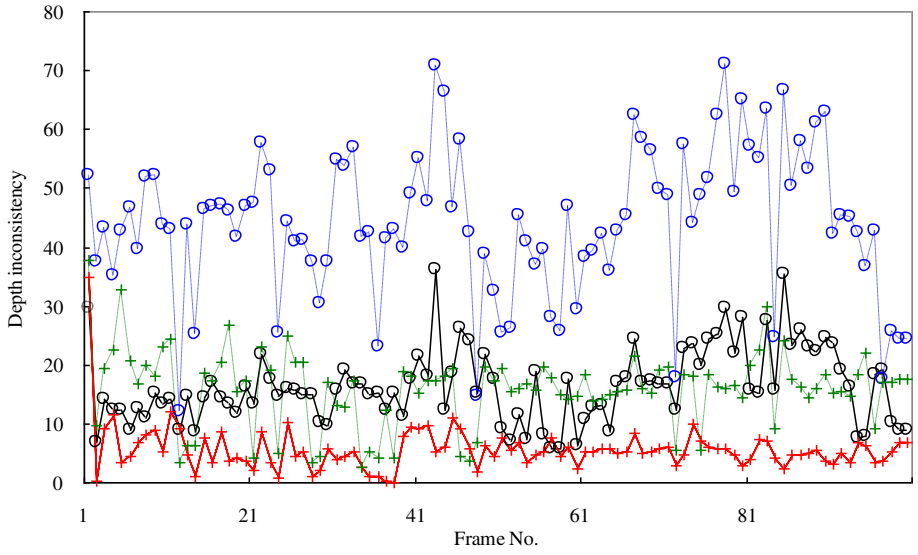

Figure 9 Evaluation of depth inconsistency; "o": Breakdancer, "+": Ballet; solid curve: with temporal filtering, and dashed curve: without temporal filtering.

computer with multi-core CPU platform to meet the real-time requirement. In multi-core platform, a program is composed of several threads that can possibly be executed in parallelism by multiple cores.

Our SVC software is tested on two multi-core platforms: Intel Core 2 Quad CPU at 2.4 GHz and Intel I7 CPU at 2.67 GHz, both operate with 3 GB RAM. The average speed performance is shown in Table 3. It is found that the process of MPEG decoding, depth estimation, and view synthesis can be completed in real-time (30–40 Hz). The speed performance can even be better after code optimization. Note however that for further commercial use, the synthesized left- and right-view frames should be better scaled up to fit the display size before frame interleaving.

Table 1 Comparisons on perceived depth and visual quality

Test sequence	Depth source	Perceived depth	Visual quality
Ballet	Microsoft Co.	4.38	3.07
	TriDef	3.38	3.93
	Proposed	4.19	3.00
Breakdancer	Microsoft Co.	4.38	3.36
	TriDef	3.63	3.79
	Proposed	4.32	3.57
Average	Microsoft Co.	4.38	3.21
	TriDef	3.50	3.86
	Proposed	4.25	3.29

7. Conclusion and future work

In this article, an automatic SVC scheme accepting MPEG videos as input is proposed. Our depth estimation is based on spatio-temporal analysis of video data, including estimations of motion and contrast cues which are capable of reflecting the relative depths between the foreground and the background areas. This study is specifically featured of (1) a shot-adaptive (categories C1–C4) algorithm adapting to diverse video content, (2) use of initially down-sampled depth estimation and following LA-JTU algorithm to reduce the computing load significantly, (3) use of a recursive temporal filter (Equation 20) to reduce possible depth fluctuations resulting from wrong depth

Table 2 Comparisons on perceived depth

Test sequence	Depth source	Perceived depth
Akko&kayo	TriDef	3.38
	Proposed	4.07
Flamenco	TriDef	3.19
	Proposed	4.07
Music video	TriDef	2.94
	Proposed	4.13
True legend	TriDef	3.19
	Proposed	3.53
2012	TriDef	3.00
	Proposed	4.00
Average	TriDef	3.14
	Proposed	3.96

Table 3 Speed performance of our SVC scheme for 640 × 480 pixels at 30 Hz format video

	With OpenMP Intel Core 2 Quad (2.4 GHz) (sec @frame)	With OpenMP Intel I7 (2.67 GHz) (sec @ frame)
MPEG decoding (640 × 480 pixels at 30 Hz)	0.0075	0.0063
Depth estimation	0.0170	0.0155
View synthesis/interleaving (output: 640 × 480 pixels)	0.0077	0.0034
Total	0.0322 (31.05 Hz)	0.0252 (39.68 Hz)

estimation, and (4) a processing architecture suitable for real-time implementation on multi-core platform.

Several kinds of videos are tested to evaluate the performance of the proposed SVC scheme. Our scheme is now capable of converting videos of 640 × 480 pixels resolution in real-time (above 30 Hz) on commercial multi-core CPU platforms. Some results show that our processing scheme does lessen the impact of depth fluctuation on perceived 3D quality. Subjective tests show that videos converted by our SVC scheme provide comparable perceived depth and visual quality with those converted from the depth data calculated by stereo vision techniques. Also, our SVC scheme is shown to outperform the well-known TriDef software in terms of human's perceived 3D depth.

Theoretically, our algorithm can be applicable to most kinds of video shots, except those containing dim or low-contrast scenarios, which make our depth cue estimation ineffective. A direction for future work is to explore and combine other monoscopic depth perception cues for more accurate depth estimation under a given processing-time limitation. In addition, a more sophisticated technique based on human visual perception to prevent human's perception uncomfortableness caused by substantial depth estimation errors or fluctuations is needed. In the future, videos of HD format, or higher resolutions, will be more popular in our daily life, which necessitates the use of graphical processing unit (GPU) for real-time conversion. Fortunately, our proposed algorithm is advantageous of local, regular, and repeated operations, which makes its implementation on GPU easier.

Competing interests
The authors declare that they have no competing interests.

Acknowledgement
This research was supported by the National Science Council, Taiwan, under the grant of NSC 99-2221-E-194-003-MY3 and NSC 100-2628-E-212-001.

Author details
[1]Department of Computer Science and Information Engineering, Da-Yeh University, 168, University Rd., Dacun, Chang-Hua 515, Taiwan. [2]Department of Electrical Engineering, National Chung Cheng University, 168, University Rd., Ming-Hsiung, Chia-Yi 621, Taiwan. [3]Asian Institute of TeleSurgery, Chang Bing Show Chwan Memorial Hospital, Changhua, Taiwan. [4]Advanced Institute of Manufacturing with High-tech Innovations (AIM-HI), National Chung Cheng University, Chia-Yi, Taiwan.

Received: 9 March 2012 Accepted: 18 October 2012
Published: 12 November 2012

References
1. CM Cheng, SJ Lin, SH Lai, Spatio-temporally consistent novel view synthesis algorithm from video-plus depth sequences for autostereoscopic displays. IEEE Trans. Broadcast. **57**(2), 523–532 (2011)
2. HT Quan, M Barkowsky, PL Callet, The importance of visual attention in improving the 3D-TV viewing experience: overview and new perspectives. IEEE Trans. Broadcast. **57**(2), 421–431 (2011)
3. GS Lin, CY Yeh, WC Chen, WN Lie, A 2D to 3D conversion scheme based on depth cues analysis for MPEG videos, in *Proceedings of the IEEE International Conference on Multimedia and Expo*, ed. by (Singapore, 2010), pp. 1141–1145
4. L Zhang, C Vazquez, A Knorr, 3D-TV content creation: automatic 2D-to-3D video conversion. IEEE Trans. Broadcast. **57**(2), 372–383 (2011)
5. HM Wang, YH Chen, JF Yang, A novel matching frame selection method for stereoscopic video generation, in *Proceedings of the IEEE Int'l Conf. on Multimedia and Expo*, ed. by (New York, USA, 2009), pp. 1174–1177
6. T Okino, H Murata, K Taima, T Iinuma, K Oketani, New television with 2D/3D image conversion technologies. Proc. SPIE **2653**, 96–103 (1996)
7. M Kim, A Park, Y Cho, Object-based stereoscopic conversion of MPEG-4 encoded data, in *Proceedings of the IEEE Pacific Rim Conference on Multimedia*, ed. by (Tokyo, Japan, 2004), pp. 491–498
8. H Murata, Y Mori, *A Real-Time 2D to 3D Image Conversion Technique Using Image Depth* (SID, DIGEST, 1998), pp. 919–922
9. D Kim, D Min, K Sohn, A stereoscopic video generation method using stereoscopic display characterization and motion analysis. IEEE Trans. Broadcast. **54**(2), 188–197 (2008)
10. P Harman, J Flack, S Fox, M Dowley, Rapid 2D to 3D conversion. Proc. SPIE **6696**, 78–86 (2002)
11. M Pourazad, P Nasiopoulos, R Ward, Generating the depth map from the motion information of H.264-encoded 2D video sequence. EURASIP J. Image Video Process **2010**, 1–13 (2010)
12. AM Tekalp, E Kurutepe, MR Civanlar, 3DTV over IP: end-to-end streaming of multiview video. IEEE Signal Process. Mag. **24**, 77–87 (2007)
13. JC Chiang, WC Chen, LM Liu, KF Hsu, WN Lie, A fast H.264/AVC-based stereo video encoding algorithm based on hierarchical two-stage neural classification. IEEE J. Sel. Topics. Signal Process. **5**(2), 309–320 (2011)
14. Y Feng, JC Ren, JM Jiang, Object-based 2D-to-3D video conversion for effective stereoscopic content generation in 3D-TV applications. IEEE Trans. Broadcast. **57**(2), 500–509 (2011)
15. G Guo, N Zhang, L Huo, W Gao, 2D to 3D conversion based on edge defocus and segmentation, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, ed. by (2008), pp. 2181–2184
16. LJ Angot, WJ Huang, KC Liu, A 2D to 3D video and image conversion technique based on a bilateral filter, in *Proceedings of SPIE-IS&T Electronic Imaging, vol. 7526*, ed. by San Jose, USA, 2010)
17. I Ideses, L Yaroslavsky, B Fishbain, Depth map manipulation for 3D visualization, in *Proceedings of the 3DTV Conference: The True Vision—Capture, Transmission and Display of 3D Video* (Istanbul, Turkey, 2008), pp. 337–340
18. *TriDef 3D display software* http://www.tridef.com/3d-experience/
19. I Ideses, LP Yaroslavsky, B Fishbain, Real-time 2D to 3D video conversion. J. Real-Time Image Process. **2**, 3–9 (2007)
20. KJ Oh, S Yea, A Vetro, YS Ho, Depth reconstruction filter and down/up sampling for depth coding in 3-D video. IEEE Signal Process. Lett. **16**(9), 747–750 (2009)
21. J Kopf, MF Cohen, D Lischinski, M Uyttendaele, Joint bilateral upsampling. ACM Trans. Graph. **26**(3), 96-1–96-5 (2007)
22. C Fehn, Depth-image-based (DIBR), compression and transmission for a new approach on 3D-TV. Proc. SPIE **5291**, 93–104 (2004)
23. GS Lin, MK Chang, ST Chiu, A feature-based Scheme for detecting and classifying video-shot transitions based on spatio-temporal analysis and fuzzy classification. Int. J. Pattern Recognit. Artif. Intell. **23**(6), 1179–1200 (2009)
24. C Fehn, P Kauff, M Op de Beeck, F Ernst, W Ijsselsteijn, M Pollefeys, L Vangool, E Ofek, I Sexton, An evolutionary and optimised approach on

3D-TV. Proceedings of the International Broadcast Convention, 357–365 (2002)
25. WN Lie, CM Lai, News video summarization based on spatial and motion feature analysis, in *Proceedings of the Pacific-Rim Conference on Multimedia*, ed. by (Tokyo, Japan, 2004), pp. 246–255
26. YP Tan, DD Saur, SR Kulkarni, Rapid estimation of camera motion from compressed video with application to video annotation. IEEE Trans. Circuits Syst. Video Technol. **10**(1), 133–146 (2000)
27. MC Lee, WG Chen, CLB Lin, C Gu, T Markoc, SI Zabinsky, R Szeliski, A layered video object coding system using sprite and affine motion model. IEEE Trans. Circuits Syst. Video Technol. **7**(1), 130–145 (1997)
28. MJ Young, MS Landy, LT Maloney, A perturbation analysis of depth perception from combination of texture and motion cues. Vis. Res. **33**(18), 2685–2696 (1993)
29. B Mendiburu, *3D Movie Making—Stereoscopic Digital Cinema From Script to Screen* (Focal Press, 2009)
30. J Ko, M Kim, C Kim, 2D-to-3D stereoscopic conversion: depth-map estimation in a 2D single-view image. Proc. SPIE **6696**, 66962A.1–66962A.9 (2007)
31. HR Wu, KR Rao, *Digital Video Image Quality and Perceptual Coding* (CRC Press, Taylor & Francis Group, 2006)
32. RC Gonzalez, RE Woods, *Digital Image Processing*, 3rd edn. (Prentice-Hall, 2008)
33. *Z depth* http://www.sonycreativesoftware.com/zdepth
34. MH Pinson, S Wolf, A new standardized method for objectively measuring video quality. IEEE Trans. Broadcast. **50**(3), 312–322 (2004)
35. CL Zitnick, SB Kang, M Uyttendaele, S Winder, R Szeliski, High-quality video view interpolation using a layered representation. ACM SIGGRAPH ACM Trans. Graph. **23**(3), 600–608 (2004)

doi:10.1186/1687-6180-2012-237
Cite this article as: Lin *et al.*: A stereoscopic video conversion scheme based on spatio-temporal analysis of MPEG videos. *EURASIP Journal on Advances in Signal Processing* 2012 **2012**:237.