A Compact Stereoscopic Video Representation for 3D Video Generation and Coding

Zhebin Zhang[+,*], Ronggang Wang[+,#], Chen Zhou[+], Yizhou Wang[+,§], Wen Gao[+]

[+]*Nat'l Eng. Lab. for Video Tech., Peking University, Beijing, 100871,China*
[*]*Key Lab. of Intel. Info. Proc., Inst. of Comp. Tech., Chinese Academy of Sciences*
[#]*Peking University ShenZhen Graduate School, ShenZhen, 518055, China*
[§]*Key Lab. of Machine Perception (MoE), Sch'l of EECS, Peking University*
[*]*Graduate Sch'l, Chinese Academy of Sciences, Beijing, 100039, China*
zbzhang@jdl.ac.cn rgwang@szpku.edu.cn {zhouch, Yizhou.Wang, wgao}@pku.edu.cn

Abstract: We propose a novel compact representation for stereoscopic videos - a 2D video and its depth cues. Depth cues are derived from an interactive labeling process during 2D-to-3D video conversion, they are contour points of foreground objects and a background geometric model. By using such cues and image features of 2D video frames, depth maps of the frames can be recovered. Compared with traditional 3D video representation, the proposed one is more compact. We also design algorithms to encode and decode the depth cues. The representation benefits both 3D video generation and coding. Experimental results demonstrate that the bit rate can be saved about 10%-50% in coding 3D videos compared with multi-view video coding and 2D+depth methods. A system coupling 2D-to-3D video conversion and coding (CVCC) is proposed to verify advantages of the representation.

1. Introduction

Stereoscopic video technologies have been well acknowledged as the next major milestone in digital video industry nowadays. Two major ones are 3D video content generation and coding.

(1) 3D video content generation Besides directly capturing stereoscopic video, 2D-to-3D video conversion is another important means of content generation. This is because (i) the amount of directly captured programs is not large enough to satisfy the 3D video industry, especially for 3DTV broadcasting; (ii) Some classic 2D movies or TV programs can be rejuvenated through the conversion. High quality depth map is the key in 2D-to-3D video conversion. It is usually estimated using monocular cues in 2D video. However, automatic depth estimation from single view videos still remains a challenging problem in computer vision field. On the other hand, 3D video synthesis from poor depth map tends to resulting visual fatigue [1]. Hence, it is necessary to introduce human in the loop, so that reliable depth cues can be generated. As a by-product, such depth cues are expected to improve 3D video coding efficiency.

(2) 3D video coding There are two main categories of stereo video coding methods, MVC (multi-view video coding) [2] and 2D+Depth. MVC method encodes 3D videos by exploiting the inter-view redundancy. The bit rate of MVC is usually high due to the large amount of multi-view video data. 2D+Depth method encodes a 2D video and depth maps to provide a low bit rate and a better capability for 3D rendering, e.g. ISO/IEC 23002-3[3]. However, the two components are encoded independently in the traditional coding schemes, or only motion correlation between a 2D video and depth maps is

1068-0314/12 $26.00 © 2012 IEEE
DOI 10.1109/DCC.2012.26

considered [4]. The bit rate of a 3D video is still much higher than a 2D video. In fact, image features of a 2D video and its structure of depth maps are tightly correlated. Is there a way that can leverage such correlation to further reduce the bit rate?

In this paper, we propose a novel compact stereoscopic video representation to improve coding performance of the generated 3D video by exploiting labeled information during 2D-to-3D video conversion. The representation consists of a 2D video and its depth cues, where depth cues include the coordinates and depth values of foreground objects contour points and a background geometric model. We propose an algorithm, which selects the minimum number of object contour points so that an object shape can be losslessly recovered jointly with image appearance features. Using such a representation, depth maps can be much more compactly represented and reliably recovered from the cues, and from which a 3D video can be easily synthesized. Compared with traditional 3D video representations, the proposed scheme is much more compact. Experimental results show that the bit rate can be saved about 10%-50% in coding 3D videos using the proposed representation.

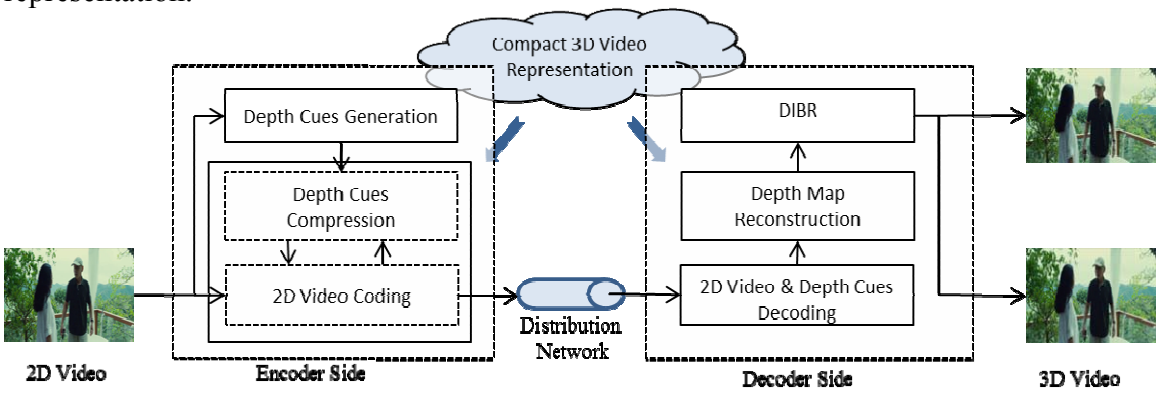


Figure 1: The architecture of the proposed CVCC

We design a system using the proposed representation to couple the 2D-to-3D video conversion and 3D video coding, namely CVCC. As shown in Fig. 1, at the encoder side, depth cues are generated from an interactive 2D-to-3D conversion module, and then a 2D video and its depth cues are compressed jointly. At the decoder side, a 2D video and depth cues are decoded from bit-stream, and then the depth cues are utilized to reconstruct depth map according to image features of 2D video. At last, the stereoscopic video is synthesized based on a DIBR method [9].

2. Depth Cues Generation for Representing Depth Map

A depth map is represented by two sets of cues, $\mathcal{S} = \mathcal{S}^{\mathrm{f}} \cup \mathcal{S}^{\mathrm{b}}$. $\mathcal{S}^{\mathrm{f}}$ is for foreground objects and $\mathcal{S}^{\mathrm{b}}$ is for the background. In this section, we describe in detail about how depth cues are generated from our interactive 2D-to-3D conversion module (Fig. 2).

A. Foreground Depth Map Modeling: $\mathcal{S}^{\mathrm{f}}$

As shown in Fig. 2, each labeled foreground object is represented by a plenary surface. Surface normal direction can be adjusted manually. The depth value $\mathrm{d}_{\mathrm{n}}^{\mathrm{f}}$ of a foreground pixel z_{n} is estimated using Eqn (1) according to the pixel position (x, y), surface normal$(\theta_{\mathrm{v}}, \theta_{\mathrm{h}})$ and the depth value of object's ground point$(\mathrm{x}_{\mathrm{g}}, \mathrm{y}_{\mathrm{g}})$. The ground point is the lowest point of object contour touching the ground.

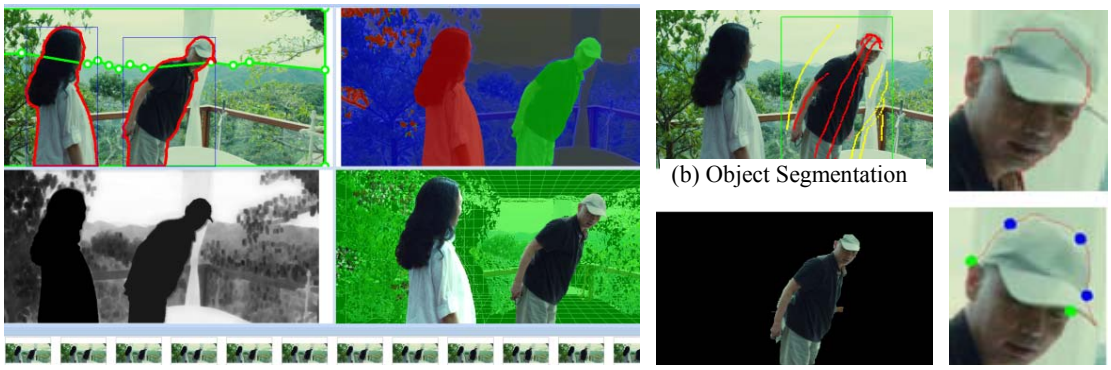


Figure 2: 2D-to-3D conversion module. Top-left of (a): segmented objects (red) and background model (green); Bottom-left of (a): depth map preview (darker object is nearer)

$$d_n^f(x,y) = d\left(x_g, y_g\right) + s_w\left(\frac{x - x_g}{w}\right) sin\theta_v + s_h\left(\frac{y - y_g}{h}\right) sin\theta_h \quad (1)$$

w and h are the object bounding box width and height in pixel, respectively. s_w and s_h denote the scale factors of relative size of an object to the scene defined by user. Consequently, the depth maps of foreground objects can be parameterized into

$$\mathcal{S}^f = \{\mathcal{S}_1^f, \mathcal{S}_2^f, \ldots \mathcal{S}_i^f \ \ldots \mathcal{S}_F^f\}$$
$$\mathcal{S}_i^f = \{C, d\left(x_g, y_g\right), x_g, y_g, w, h, s_w, s_h, \theta_v, \theta_h\} \quad (2)$$

C is the set of contour points of a foreground object $\mathcal{S}_i^f$. This parameterized model results in a compact representation for the depth of foreground regions.

In the system, each image pixel z_n is associated with a label $\alpha_n \in \{0,1\}$, where α_n=0 denotes background, and α_n =1 denotes foreground. As shown in Fig.2 (b), the foreground/background labels are obtained by Graph Cut [6] through minimizing

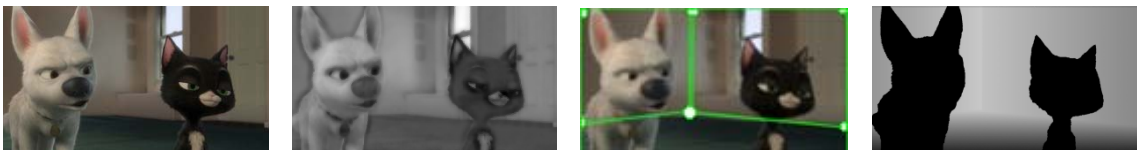
$$E(\boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{z}) = \sum_n E_d(\alpha_n, k_n, \boldsymbol{\theta}, z_n) \ + \gamma \sum_{(m,n)\in\mathbb{N}} [\alpha_n \neq \alpha_m] e^{-\beta \|z_m - z_n\|^2},$$
$$E_d(\alpha_n, k_n, \boldsymbol{\theta}, z_n) = \ -\log p(\alpha_n, k_n, \boldsymbol{\theta}, z_n) \quad (3)$$

$p(\alpha_n, k_n, \boldsymbol{\theta}, z_n)$ is the probability of a pixel labeled by α_n, whose distribution is modeled by a Gaussian Mixture Models (GMM) in RGB space. Model parameter $\boldsymbol{\theta}$ is learned from user-input foreground (red in Fig.2 (b)) and background (yellow) scribbles. The second term of the energy function penalizes label differences among neighboring pixels $\mathbb{N}$ with respect to their color difference.

User can further refine the segmentation by adjusting object contour. As shown in Fig.2 (d), an incorrect section of object contour is selected by appointing two anchor points (green) on the boundary of object. Several points (blue dots), defined as the "*Control Points*", can be further selected and dragged to align the curve with object boundary. The alignment process can be optimized by Intelligent Scissors [7].

B. Background Depth Map Modeling: $\mathcal{S}^b$

It is impractical to label each object in a scene for efficient 3D content production. According to the research result on visual perception [8], the stereoacuity is inversely proportional to the square of the viewing distance, which means that human beings generally cannot accurately estimate the depth of objects at far distance. Consequently, for far away objects, their partial order in distance is sufficient to descript their spatial



(a)Original frame (b)Auto-estimation depth map (c)Stage model labeling (d)Stage model depth map

Figure 3: An example of background modeling

relation. Hence, in the system, near objects are labeled in detail, but for background, simplified model or an automatic depth estimation (may not be so accurate) algorithm is adopted. We adopt the algorithm proposed in [9] to automatically estimate the depth of a background pixel z_n as follows:

$$d_n^b = w_1 \cdot d_n^a + \beta \cdot d_n^m, \tag{4}$$

where d_n^a and d_n^m are monocular depth cue of atmospheric radiance[10] and pseudo depth from motion [11], respectively. As shown in Fig. 3, if a user is not satisfied with the automatic depth estimation result, an alternative model [12], namely stage model (Fig. 3(c)), is provided. The stage model consists of several planner surfaces, whose geometry is defined by a set of control points (white dots). A user can add, delete, and drags the points to fit geometric structure of the background. Then depth values of the control points are set according to user's perception. The depth map of each surface is linearly interpolated from the associated control points, and then the whole depth map of background can be approximated as shown in Fig. 3(d). Consequently, a background depth map can be model with *M* control points and their depth values as

$$\mathcal{S}^b = \{M, \mathcal{p}_1 \dots \mathcal{p}_M, d_1 \dots d_M\}, \tag{5}$$

where $\mathcal{p}_i$ is control point at (x_i, y_i) and d_i is its depth value. $M = 0$ indicates the automatic estimation is adopted. Similar to $\mathcal{S}^f$, $\mathcal{S}^b$ also reduces the representation of the depth map of background regions.

3. 3D Video Coding

A. Coding Foreground Object Shapes

A segmented foreground object contour is represented by a series of adjacent points (as introduced in Sect.1.A). In this section, a lossless 2D shape coding algorithm is proposed to select a minimum number of control points from a dense object contour so that an object shape can still be recovered. Specifically, we adopt the Intelligent Scissors (IS) algorithm [7] to skip redundant contour points using image features as shape recovery cues. In other words, the selected control points are those with which the object contour can be faithfully recovered using image features as constraints. The algorithm is described in Alg.1. In IS, an image lattice is represented by a graph $G(V, E)$. The weight on a graph edge is defined as

Algorithm 1: Lossless Compression
Input: Texture image *I*
One Object contour $C(p_1, .. p_i, . p_n, p_1)$
Output: Contour control points Cp
1. Initialize Cp with one element p_1;
2. i=1;
3. ***while*** i<n
4. ***for*** j=i+1 to j<=n
4.1 Set a rectangle region R_{ij} on *I*, taking p_i and p_j as top-left and bottom-right points, and *vice versa*.
4.2 Get a shortest path e_{ij} using the IS algorithm on R_{ij} between p_i and p_j
4.3 ***if*** $e_{ij}=(p_i, \dots p_j)$
4.4 continue;
else
4.5 add p_{j-1} into Cp;
4.6 i = j-1;
4.7 break;
4.8 ***end*** // for
end // while

$$l(p,q) = w_z f_z(q) + w_G f_G(q) + w_D f_D(p,q) \quad (6)$$

where w_z, w_G and w_D are constant weights that balance three terms f_z, f_G, f_D. The three terms are functions of image features such as Laplacian zero-crossing, image intensity gradient magnitude and gradient direction, respectively. A smaller edge weight indicates that a pair of neighboring pixels p and q tends to be on an object contour. Given any two nodes on the graph, p_{k_j} and $p_{k_{j+1}}$, a Dijkstra-like algorithm [7] is used to search the shortest path $\Im^*_{k_j,k_{j+1}}$ between them by minimizing the cost

$$\mathcal{L}\left(\Im_{k_j,k_{j+1}}\right) = \sum_{x \in \Im_{k_j,k_{j+1}}} l(p_x, p_{x+1}), \quad (7)$$

which is the sum of graph edge weight on each searching path $\Im_{k_j,k_{j+1}}$.

Alg.1 selects control points in a greedy manner, it starts from a random contour point as the first control point, then set the next one as a candidate control point on the contour, the algorithm uses IS to recover a segment of boundary between the candidate and its preceding control point. If the boundary is exactly the same as the original contour segment between the two points, the candidate is not accepted. Otherwise, its previous contour point is accepted as a control point. The algorithm continuously tests each contour point successively till the whole contour is traversed. Since the algorithm guarantees the contour can be exactly recovered from the control points, Alg.1 is a lossless contour compression algorithm. We argue that the Proposition 1 is true for Alg.1:

Proposition 1: under the condition of lossless compression, the number of control points selected by Alg.1 is approximately optimal, which equals the optimal solution or at most exceeds it by 1. (The proof is given in Appendix A)

Using the Alg. 1, for example, an object contour with approximately 2000 contour points can be compressed to 200-300 control points. The compression ratio changes with the local image feature and generally increases with the gradient magnitude of object edges. To further improve the compression ratio, the constraint of 4.3 in Alg.1 is relaxed as

$$\mathfrak{D}(e_{ij}, (p_i, \dots p_j)) \le T_c$$

$$\mathfrak{D}(X,Y) = \max\{\sup_{x \in X} \inf_{y \in Y} d(x,y), \sup_{y \in Y} \inf_{x \in X} d(x,y)\} \quad (8)$$

where $\mathfrak{D}$ is the Hausdorff Distance[13] between two sections of contour points set, T_c is a threshold for contour compression error. In this paper, with a threshold $T_c = 1$, a contour with 2000 points can be compressed up to 30~40 control points.

B. Coding Foreground Depth Parameters, Background Model and 2D Video

The foreground depth parameters $\{d(x_g, y_g), x_g, y_g, w, h, s_w, s_h, \theta_v, \theta_h\}$ are encoded by fixed length coding. When $M \neq 0$, the background is represented by the stage model (as shown in Fig.3(c)), and the stage model control points and depth $\{\mathcal{p}_1 \dots \mathcal{p}_M, \mathcal{d}_1 \dots \mathcal{d}_M\}$ are also encoded by fixed length coding. The coding length for one frame are within 2 Kb; when $M = 0$, only $\mathcal{S}^b = \{0\}$ is encoded.

A 2D video can be encoded by any traditional 2D video coding algorithm. In our system, H.264/AVC High profile is adopted. Since there is strong dependency between a 2D video and its depth cues, a 2D video is encoded by considering the coding distortion impact on depth cues reconstruction. For macroblocks covering foreground object contours, a new rate distortion cost is defined as $\mathcal{J} = D + D_{cue} + \lambda R$, where D, R, and λ denote the decoded 2D video distortion, encoded bits count and Lagrangian multiplier

respectively. D_{cue} symbolizes the distortion of recovered object contours. Depth cues are encoded as the "user-custom SEI" in H.264/AVC bit-stream.

To summarize, a converted stereoscopic video is encoded as follows:

(1) A depth map is represented by depth cues $(\mathcal{S}^f, \mathcal{S}^b)$.
   a) Each object contour C is further compressed into a compact set of "control points" using Alg.1.
   b) The other foreground depth cues and background model $\mathcal{S}^b$ are coded by fixed length coding;

(2) The 2D video can be compressed using any traditional video coding algorithm.

Algorithm 2:
Input: A decoded 2D frame I ;
Contour $C(p_1, p_2, \dots p_N)$ recovered by IS
Output: A smoothed object contour C'
1. Set temperature $= T_0$;
   Set sampling count W = 0;
2. ***while*** $T > T_c$
3. Randomly sample one p_{k_i} from C
4. Compute $\mathcal{L}(\Im_{k_i})$
5. Set $\mathcal{L}_{min} = MAX$ and $p_{min} = NULL$
6. ***for*** each location p_{ni} in the $K_s \times K_s$ neighborhood Nb_{K_s} of p_{k_i}
6.1. Compute $\mathcal{L}(\Im_{n_i})$
6.2. **if** $\mathcal{L}_{min} < \mathcal{L}(\Im_{n_i})$
6.3. $\mathcal{L}_{min} = \mathcal{L}(\Im_{n_i})$
6.4. $p_{min} = p_{ni}$
end //for
7. ***if*** $\mathcal{L}_{min} < \mathcal{L}(\Im_{k_i})$ $p_{k_i} = p_{min}$
else
8. Randomly sample new p'_{k_i} in Nb_{K_s}
9. Compute $\mathcal{L}(\Im_{k'_i})$
10. ***if*** $\exp\left\{\frac{\mathcal{L}(\Im_{k'_i}) - \mathcal{L}(\Im_{k_i})}{T}\right\} > Rand[0,1)$
11. $p_{k_i} = p'_{k_i}$;
12. W = W + 1;
13. ***if*** W $> \varepsilon \times K_s \times K_s$
Break;
else
T $= 0.8 \times$ T
end //while

4. 3D Video Decoding

A 2D video and its depth cues are first decoded from bit-stream by H.264/AVC decoding module. For background, if $M = 0$ in $\mathcal{S}^b$, background depth map is estimated by the automatic depth estimation algorithm [9] from its corresponding 2D video frame; otherwise, the decoded stage model is adopted. For foreground, object contours are recovered by applying Intelligent Scissors on a 2D frame between each pair of neighboring control points Cp. The depth map of objects is computed using the depth parameters according to Eqn.(2). However, when Alg.1 is relaxed to lossy compression, the reconstructed contour at decoder side tends to appear zigzag artifacts on the boundaries with weak gradient intensity. To address this problem, Alg.2 is proposed to enforce a smoothness constraint to the path cost function $\mathcal{L}$(see Eqn.(7)) in the original Intelligent Scissors. The new cost function of a path $\Im_{k_{j-1},k_j,k_{j+1}}$ is defined as

$$\mathcal{L}'(\Im_{k_j}) = \mathcal{L}_{\text{IS}}(\Im_{k_j,k_{j+1}}) + \mathcal{L}_{\text{IS}}(\Im_{k_{j-1},k_j}) + \lambda \mathcal{L}_{\text{Smooth}}(\Im_{k_{j-1},k_j,k_{j+1}}) \qquad (9)$$

$\mathcal{L}_{\text{IS}}$ is IS contour reconstruction cost of a section $(p_{k_{j-1}}, \dots, p_{k_j}, \dots p_{k_{j+1}})$ of an object contour $C(p_1, p_2, \dots p_N)$, defined in Eqn.(7). λ balances the smoothness constraint $\mathcal{L}_{\text{Smooth}}$. $\mathcal{L}_{\text{Smooth}}$ penalizes the difference between two successive curve section directions,

$$\mathcal{L}_{\text{Smooth}} = \frac{1}{2}\sum_{i=0}^{m}\left(1 - \frac{\boldsymbol{r}_i \cdot \boldsymbol{r}_{i+1}}{\|\boldsymbol{r}_i\|\|\boldsymbol{r}_{i+1}\|}\right) \qquad \boldsymbol{r}_i = \begin{cases} (p_{i+K_s}, p_i), & i + K_s \le N \\ (p_N, \quad p_i), & i + K_s > N \end{cases} \qquad (10)$$

m is the total number of "key points", which are sampled from C with sampling step K_s(K_s=5 in this paper). Alg.2 then is performed based on the key points.

(a) Before smoothing (b) After smoothing

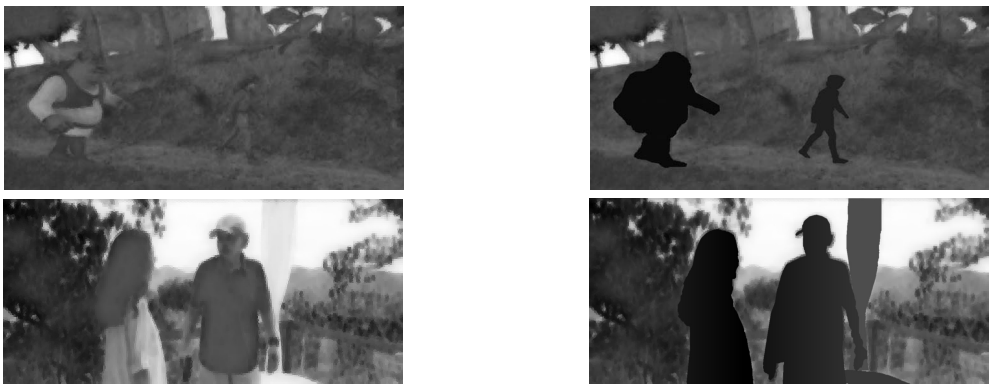
Figure 4: Foreground object contour smoothing for lossy compression. (Red circles denote "control points" obtained in Alg.1)

Using the definition of cost in Eqn.(10), the weight of a graph edge in G(V, E) is no longer a fixed value. So a greedy searching scheme like Dijkstra algorithm is not able to obtain an optimal path any more. Dynamic Programming can be used, but it is too computationally expensive. Hence, we adopt the Simulated Annealing algorithm to find the optimal smooth object boundary. The algorithm implementation details are shown in Alg.2. In the simulated annealing process, the change of total energy cost can be efficiently computed locally. Furthermore, as the contour sections generated by Intelligent Scissors are close to the contour with global minimal cost, the initial temperature T_0 can be relatively low. Hence, a faster decreasing rate can be adopted, which means the total cost is quickly converged to minimum. In our implementation, T_0 is set to 100 to balance the performance and speed, the temperature decreasing factor is set to 0.8, and the threshold of termination is set to 1. Fig.4 gives an example which is improved by Alg. 2.

Once the depth map is reconstructed, stereo video is synthesized by DIBR method [9].

5. Experiment results

In this section, experimental results shows the advantage of the proposed compact representation and the CVCC system. Two 2D movies are selected as the test data, one is a 1080p cartoon movie "Shrek3", and the other is a 720p regular movie of "If you are the one". The CVCC is compared with two representative methods of MVC and 2D+depth.



(a)Depth map estimated by an automatic method[9] (b) Depth map reconstructed by the CVCC

Figure 5: Depth map comparison between an automatic method [9] and the CVCC

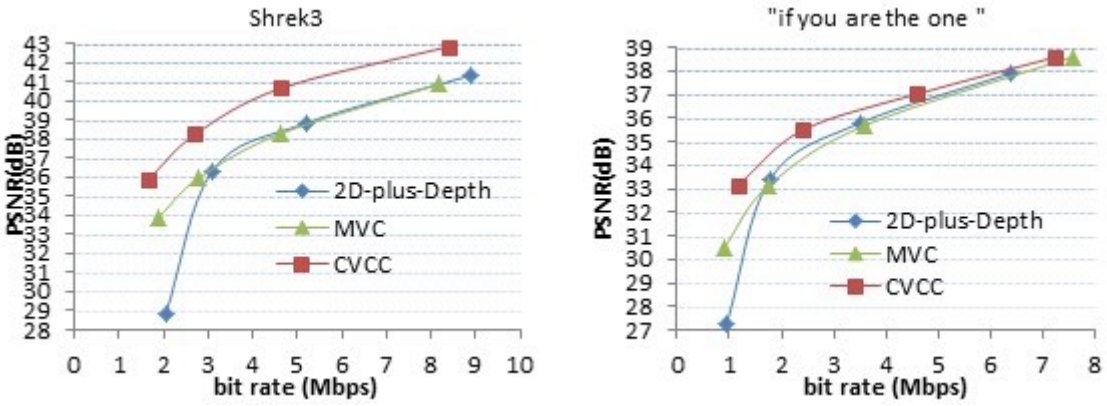


Figure 6: Rate distrotion comparision among the three 3D video coding methods

Depth cues generated by the CVCC improve the quality of 2D-to-3D converted videos. Fig.5 gives a comparison between depth maps generated by using an automatic conversion method in [9] and the maps reconstructed by the CVCC. It is obvious that the accuracy of depth map is greatly improved by CVCC, especially in the regions of foreground objects.

To compare the coding performance of CVCC with MVC and 2D+depth, we first generate a raw "left view" video and a "right view" video by DIBR from the raw 2D video and depth map derived from our interactive 2D-to-3D video conversion module. For the MVC method, we use "Stereo High Profile" implemented in the newest H.264/AVC reference software JM18.0 to code the generated raw "left view" video and the "right view" video. For the 2D+depth method, we encode the raw 2D video and the depth map (generated by CVCC) by H.264/AVC High Profile respectively, and synthesize the "left view" video and "right view" video based on the decoded 2D video and depth map by DIBR method. For the proposed CVCC, we encode the raw 2D video and depth cues generated by our system, reconstruct the depth map based on the decoded depth cues, and synthesize the "left view" video and "right view" video based on the decoded 2D video by DIBR method. We calculate the average distortion of reconstructed "left view" video and "right view" video at decoder side to the raw "left view" and "right view" video synthesized at encode side by the three methods respectively, and compare rate distortion performance of them.

90 frames from "shrek3" and "if you are the one" are tested respectively. The frame rate is set as 30 fps, GOP structure of "IPPP…" and 5 reference frames are used, search range is set as 32. Four QPs of 28, 32, 36 and 40 are used for MVC and 2D+depth methods to generate different bit rates coded videos and depth maps. Since the bit rate of depth cues in our system is only between 100kbps-200kbps, we reallocate the saved bits to the 2D video. QPs of 24, 28, 32 and 36 are used to encode the 2D video in CVCC. The rate distortion performances of the three methods are showed in Fig. 6 and Table 1. Compared with MVC, CVCC improves the decoded 3D video quality by 3.43 dB and 0.99dB at the same bit rate or saves 55.23% and 14.24% bit rate at the same decoded 3D video quality on "shrek3" and "if you are the one" respectively. Compared with 2D+depth, CVCC improves the decoded 3D video quality by 3.99 dB and 1.19dB at the same bit rate or saves 55.48% and 19.25% bit rate at the same decoded 3D video quality on "shrek3" and "if you are the one" respectively.

Table 1: Coding performance improvement by CVCC

	CVCC vs. MVC		CVCC vs. 2D+Depth	
Test sequences	Bit rate saved	Video qualtiy improved	Bit rate saved	Video qualtiy improved
Shrek3	55.23%	3.43dB	55.48%	3.99dB
“if you…”	14.24%	0.99dB	19.25%	1.19dB

Table 2: Experiment results in detail for the three 3D video coding methods

		MVC		2D+Depth		CVCC		
Test sequence	QP	Total bitrate（kbps）	Average distortion (dB)	Total bitrate（kbps）	Average distortion (dB)	Bitrate of depth cues(kbps)	Total bitrate（kbps）	Average distortion (dB)
Shrek3	28	8190.29	40.9	8889.66	41.34	120.33	8392.93	42.85
	32	4640.23	38.3	5190.53	38.84	118.79	4635.82	40.67
	36	2783.38	35.95	3094.64	36.34	110.21	2707.14	38.25
	40	1875.81	33.86	2046.97	28.82	120.89	1645.82	35.85
“if you are the one”	28	7560.23	38.6	6389.06	37.91	60.56	7240.78	38.59
	32	3575.14	35.68	3495.97	35.79	59.23	4598.36	37.03
	36	1731.74	33.1	1774.36	33.405	62.21	2394.65	35.48
	40	907.77	30.529	949.45	27.27	61.82	1168.94	33.17

6. Conclusion

We proposed a novel compact stereoscopic video representation. A system called CVCC is designed for both 3D video generation and coding. Using the representation, the coding performance of the converted stereoscopic video is greatly increased and the quality of depth map is also improved. The new representation can also be extended to other related fields, such as object-based video coding, ROI-based coding, and intelligent video content understanding.

Acknowledgement

This work was supported by the grant of National Basic Research Program of China (973 Program) 2009CB320904 and the grant of Key Projects in the National Science & Technology Pillar Program 2011BAH08B03.

Reference

[1] M. Lambooij, W. Ijsselsteijn, M.Fortuin, I. Heynderickx, “Visual discomfort and visual fatigue of stereoscopic displays: A review.” J. of Imaging Science and Technology, 2009.

[2] ISO/IEC JTC1/SC29/WG11, “Study Text of ISO/IEC 14496-10:2008/FPDAM 1 Multiview Video Coding”, Doc. N9760, Archamps, France, May 2008.

[3] ISO/IEC JTC1/SC29/WG11, “Text of ISO/IEC FDIS 23002-3 Representation of Auxiliary Video and Supplemental Information”, Doc. N8768, January 2007.

[4] M. Tanimoto, "Overview of free viewpoint television," Signal Process. Image Communication, 2006.
[5] C. Rother, V. Kolmogorov, A. Blake. "GrabCut: Interactive Foreground Extraction using Iterated Graph Cuts, " ACM Transactions on Graphics (SIGGRAPH'04), 2004
[6] Y. Boykov and M.-P. Jolly, 2001. "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images". ICCV, 2001.
[7] Eric N. Mortensen and William A. Barrett, "Intelligent scissors for image composition, " In Proc. ACM SIGGRAPH '95, pp.191–198.1995.
[8] I. P. Howard, B.J. Rogers, "Seeing in Depth". Oxford University Press, 2002.
[9] Zhebin Zhang, Yizhou Wang, Tingting Jiang, Wen Gao. "Visual Pertinent 2D-to-3D Video Conversion By Multi-cue Fusion," ICIP, IEEE, 2011.
[10] K. He, J. Sun, X. Tang, Single Image Haze Removal Using Dark Channel Prior, CVPR2009.
[11] Xuming He, Alan Yuille, Occlusion Boundary Detection using Pseudo-Depth, ECCV2010.
[12] V.Nedović, A.W.M.Smeulders, A.Redert, J.-M.Geusebroek, "Stages as Models of Scene Geometry," T-PAMI, IEEE, 2010.
[13] R. Tyrrell Rockafellar, Roger J-B Wets, Variational Analysis, Springer-Verlag, 2005, pg.117

Appendix A

Proof of Proposition 1: Suppose $Cp^*(p_{k_1^*}, p_{k_2^*}, .. p_{k_l^*}, .. p_{k_{N^*}^*})$ is the optimal set with minimal control point number, where k_l^* is the index of a contour point, and the control point set found by Alg.1 is denoted as $Cp(p_{k_1}, p_{k_2}, .. p_{k_j}, .. p_{k_N})$ (k_1=1). If $p_{k_1^*} \neq p_{k_1}$, p_{k_1} is added to Cp^* and a new list $Cp^{\#}(p_{k_1'}, p_{k_2'}, .. p_{k_l'}, .. p_{k_{N'}'})$ is formed such that $p_{k_1'} = p_{k_1}$; otherwise $Cp^{\#} = Cp^*$.

(1) **Axiom 1:** a sub-path of a shortest path is also a shortest path

(2) Now we use induction to prove $k_i \geq k_i'$, which **intuitively** means the selected point by Alg. 1 can't appear before the corresponding points in the optimal solution.

   a) First, as $p_{k_1'} = p_{k_1}$ according to our construction of $Cp^{\#}$, $k_1 \geq k_1'$ holds.

   b) Now we prove that if $k_{i-1} \geq k_{i-1}'$, then $k_i \geq k_i'$. If not, then $k_i + 1 \leq k_i'$. Denote the shortest path between k_i, k_j as $\mathfrak{I}_{k_i,k_j}^*$, then $\mathfrak{I}_{k_{i-1},k_i+1}^*$ forms a sub-path of $\mathfrak{I}_{k_{i-1}',k_i'}^*$, and according to 4.2~4.6 of Alg.1, $\mathfrak{I}_{k_{i-1},k_i+1}^*$ is different from the original contour segment between $p_{k_{i-1}}$ and p_{k_i+1}. By substituting $\mathfrak{I}_{k_{i-1},k_i+1}^*$ into $\mathfrak{I}_{k_{i-1}',k_i'}^*$, the shortest path $\mathfrak{I}_{k_{i-1}',k_i'}^*$ is different from the original contour; whereas, $p_{k_{i-1}'}$ and $p_{k_i'}$ belong to $Cp^{\#}$, hence $\mathfrak{I}_{k_{i-1}',k_i'}^*$ should be the same as the original contour, which results in a contradiction.

(3) The **process** of b) can be repeated till i=N'(N' is the total number of points in $Cp^{\#}$), and from $p_{k_{N'}}$ to p_{k_1}, no additional control points is needed since the shortest path $\mathfrak{I}_{k_{N'},k_1}^*$ is a sub-path of $\mathfrak{I}_{k_{N'}',k_1'}^*$. So the number of points found by Alg.1, N, equals N', which exceeds the point number N^* of optimal solution at most by 1