MULTILAYER SEMITRANSPARENT-EDGE PROCESSING FOR DEPTH-IMAGE-BASED RENDERING

Mikhail Erofeev, Dmitriy Vatolin

Lomonosov Moscow State University
Moscow, Russia

ABSTRACT

Owing to the movie industry's wide use of 2D-to-3D conversion techniques, the problem of synthesizing stereoscopic views using depth-image-based rendering (DIBR) is extremely important. A major challenge for DIBR is processing semitransparent edges near depth-map discontinuities. Existing approaches either can only deal with simple cases in which the background behind the object does not change significantly, or they segment the input image into two layers. Unfortunately, such segmentation is impossible to carry out correctly for depth discontinuities formed by multiple objects, leading to visual artifacts in the generated views. Our proposed method of multilayer semitransparent-edge processing avoids these limitations and outperforms competitors in a subjective visual-quality comparison of the synthesized views.

Index Terms— Depth map, matting, DIBR.

1. INTRODUCTION

Despite recent progress in stereoscopic-camera manufacturing, techniques for converting 2D to stereoscopic 3D (S3D) continue to see wide use. For example, 24 of 38 stereoscopic films released in 2015 employed 2D-to-S3D conversion [3]. This fact owes to the technical issues associated with both native S3D capture and postprocessing of natively captured content [17].

The most widely used 2D-to-S3D conversion technique in the film industry is depth-image-based rendering (DIBR). The primary challenges that arise when using this technique are the following:

1. Depth-map estimation (i.e., estimating the map that defines the distance from each image pixel to the viewer; see Fig. 2(b). This problem has a variety of solutions:
   (a) drawing a depth map by hand for each video frame (the most expensive method, but it ensures the highest-quality result),
   (b) interpolation of hand-drawn depth maps for key frames in the rest of the video [12, 16],
   (c) use of fully automatic depth-map estimation techniques [5, 7] (unfortunately, such methods can deliver high-quality results only in certain cases).
2. Occlusion completion (i.e., filling frame areas that are covered by foreground objects but will be visible in the generated views; see Fig. 1(a). A variety of tools and methods can help solve this problem [18, 14, 2]. Also, occlusion inpainting by stretching edge pixels [4] remains a widely used technique in the industry.
3. Stereoscopic-view generation by shifting pixels from the original image in accordance with a depth map and combining the resulting image with the output of the previous step. Applying such a shift to most image pixels is trivial, except for some along semitransparent object edges. These edge pixels do not correspond to one particular object in the frame, since their color is a mix of colors from multiple objects (or areas) located at various depths. Simply assuming that such pixels correspond to only one object leads to visual artifacts in the generated S3D views (see Fig. 1(b).

(a) Example of occlusion areas (b) Example of incorrectly processed semitransparent edges

Fig. 1. Examples of two major challenges to stereoscopic-view generation: (a) occlusion completion (occlusions are highlighted with a checkered pattern)—i.e., filling areas that are hidden in the original view but visible in the new views; (b) semitransparent-edge processing (incorrect treatment of semitransparent pixels can cause visual artifacts, such as edges that are too sharp or too blurry).

In this paper we study the problem of semitransparent-edge processing to aid in generating stereoscopic views. This problem is particularly important for images that depict objects with numerous semitransparent pixels (e.g., hair, fur,

This study was partially supported by RFBR, research project no. 15-01-08632 a.

978-1-5090-5743-6/16/$31.00 ©2016 IEEE

(a) Source image (b) Disparity map (c) Areas selected for further processing

Fig. 2. Example of input data for the proposed method: (a) source image and (b) disparity map. The first step involves estimating the mask (c) of pixels along the disparity-map edges (the example shows an estimated mask, in red, over the original disparity map).

motion blur, and lens blur) and for depth maps containing inaccuracies along object edges (such inaccuracies regularly arise in practice; for example, the edge in a depth map might be several pixels away from the edge in the corresponding image).

To simplify notation in this paper, we use disparity maps instead of depth maps (the disparity map is inversely proportional to the depth map and is directly proportional to the shifts that should be applied to pixels when generating S3D views).

Our proposed method naturally exploits the multilayer structure of natural images. For a given input image and disparity map, it outputs an S3D image. The method employs image-matting techniques to decompose input data into multiple layers such that each layer corresponds to a single disparity value. Thus, we encode pixels located at semitransparent object edges as a linear blend of multiple pixels located at various depths. This transformation eliminates shift-selection ambiguity for mixed pixels, since the proposed multilayer representation can simply be warped to a new S3D view by applying an individual shift to each layer.

In the next section we discuss prior approaches to semitransparent-edge processing for stereoscopic-view generation. Then, in Section 3, we provide a detailed description of our proposed method. Finally, in Section 4, we discuss the results of a subjective comparison of our method with existing approaches.

2. RELATED WORK

Most existing work on the problem of semitransparent-edge processing for stereoscopic-view generation has been devoted to interpolating additional views from an input multiview image. In these efforts, how to generate new views (a topic we consider in this paper) is a subset of the interpolation problem.

An initial method for generating new views that takes into account pixel transparency is described in [20]. It takes as input a multiview video sequence, captured using a multicamera rig, and produces additional views captured by virtual cameras. The authors apply Bayesian matting to four-pixel neighborhoods around the estimated disparity-map edges and thereby decompose the input image into two layers (foreground and background) along those edges. As we show in Section 4, correctly performing such a decomposition is impossible for regions that have more than two layers (e.g., where three objects located at various depths intersect). The authors of [10] employ a similar technique: segment the input image into foreground and background layers near the depth-map discontinuities and then apply a guided filter [9] to estimate the transparency map.

The method proposed in [13] takes as input an image, its depth map and a background image (i.e., the image with the foreground objects removed) and generates stereoscopic views. To decompose the input image into foreground and background layers, the authors employ a simple matting method based on the assumption that the background image perfectly matches the input image (such an assumption holds only for images generated by computer graphics and for images with uniform backgrounds). They demonstrate results for two artificial images.

Fukushima et al. [8] propose an alternative solution to handling semitransparent edges. Instead of trying to decompose the input image into layers by means of matting, the authors perform a blur transfer. The first step is to dilate the depth map toward the background and generate new views without considering pixel transparency. The method then blurs the generated views along the depth-map edges to hide sharp seams. It can handle intersections of multiple objects at different depths, but the authors report that it fails to handle significant background changes.

A multilayer depth-map representation similar to what we use in this paper is proposed in [15]. But the representation in [15] does not take into account pixel transparency. Furthermore, the authors propose a method of multilayer depth-map capture that only works in a controlled lab environment.

As we show in the next section, our proposed method is designed to overcome two major problems with existing approaches: handling significant background changes and dealing with the intersection of multiple objects.

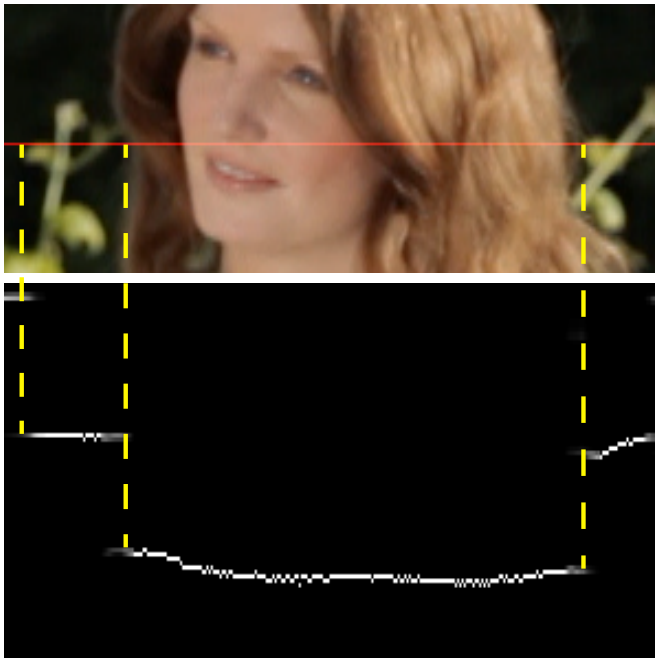

Fig. 3. Multilayer disparity map for a single input-image line (highlighted with red at the top of the figure). Each line from the bottom image depicts a layer in the multilayer disparity map (the bottommost line corresponds to the layer closest to the viewer). The pixel intensity encodes the transparency (white pixels are solid, black pixels are fully transparent). Noticeable correspondences appear between the top and bottom images (e.g., plants, the actress's hair and the background).

3. PROPOSED METHOD

The inputs for our method are the image I, the disparity map D (with integer elements between 0 and 255), the width w of the area for edge processing, the position z of the screen plane, and the strength p of the 3D effect. The method outputs left and right stereoscopic views.

The main steps of our proposed method are the following:

1. Estimation of areas for further processing. We select pixels along disparity-map edges, since these pixels are likely to be a mix of colors from multiple objects at various depths. As we discussed in Section 1, a single shift is insufficient to warp such pixels to a new S3D view, and they require nontrivial processing.

2. Estimation of multilayer disparity map. We introduce a novel multilayer disparity-map representation such that the ith layer encodes the transparency of pixels with disparity i. Thus, the number of layers in this representation matches the number of unique values in the input disparity map.

3. Computation of layer colors. Because the pixels that we process are a mix of multiple object colors, we must reconstruct the original colors.

4. Occlusion inpainting. We need to fill areas that are occluded by foreground objects but become visible in the new views. In this paper, we consider two ways to recover missing information: linear interpolation and use of a background image provided as an additional input.

5. Layer shifting. Each layer in the multilayer disparity map is associated with a single disparity value. Thus, we apply a global shift to each layer, warping it to a new view.

6. Generation of S3D views by merging the layers.

We repeat steps 5 and 6 for each stereoscopic view. Below, we discuss each step in more detail.

3.1. Estimation of areas for further processing

To estimate a mask for pixels that may be a mix of objects with various depths, we use an approach similar to what [10] describes. We compute D^{max} by dilating the input disparity map w times (w is a user-specified parameter) using a 3×3 kernel. Similarly, we use erosion to compute D^{min}. The final mask contains pixels for which $D^{max} - D^{min} > 10$. Thus, we highlight areas of radius w along disparity-map edges. Fig. 2(c) shows an example of an estimated mask.

Because later steps in our method impose no requirements on the disparity-edge mask, a variety of approaches to estimating this mask will work in lieu of the one described above. Also, the user can manually edit the generated mask (e.g., by highlighting strands of hair that were not selected automatically).

3.2. Estimation of multilayer disparity map

The goal of the next two steps is to convert the input image and disparity map to a multilayer representation such that each layer corresponds to only one disparity value. This representation enables us to solve the problem of shifting mixed pixels by decomposing them into multiple layers and applying the shift to each layer individually. We can express it as follows:

$$I = \sum_{i=1}^{n} \alpha^i F^i, \tag{1}$$

where α^i is a transparency map of the layer with disparity i, F^i is the image from layer i, and n is the maximum value in the disparity map (255 in our implementation). Performing this decomposition is similar to natural-image matting (which decomposes a given image to a foreground, a background, and a transparency map) using a given trimap (a map that labels each pixel as foreground, background, or unknown).

We used learning-based matting [19] for our multilayer case. For each integer i in the range 0 to n, we perform the following steps:

1. Generate a trimap such that pixels with disparity i are marked as foreground, pixels belonging to the edge map are marked as unknown and the rest are marked as background.

2. Compute a transparency map $\alpha^{'i}$ by applying learning-based matting to the input image and by using the trimap from the previous step.

To compute the final multilayer disparity map, we normalize the transparency values in each pixel:

$$\alpha^i = \frac{\alpha^{'i}}{\sum_{j=0}^{n} \alpha^{'j}}. \tag{2}$$

In Fig. 3, we show an example of an estimated multilayer disparity map. A naïve implementation of the above approach will have extremely high computational complexity. For each layer, learning-based matting computes the transparency map $\alpha^{'i}$ by solving the following system of linear equations:

$$H\alpha^{'i} = b_i. \tag{3}$$

Since we apply learning-based matting to the same input image and use a trimap with the same unknown area, the system we solve in each step has exactly same matrix H. This observation enables us to reduce the computational complexity by computing the Cholesky decomposition of H and reusing it in each step.

3.3. Computation of layers colors

Using the multilayer disparity map we estimated in the previous step, we compute the image for each layer F^i (see Equation (1)). A similar solution for the two-layer case (background and foreground) appears in [11]. To generalize this approach to the multilayer case, we introduce the following cost function:

$$\begin{aligned} C(I,\alpha,F) = &\sum_{p\in\Omega} \left\| I_p - \sum_{i=1}^{n} \alpha_p^i F_p^i \right\|^2 + \\ &+ \sum_{i=1}^{n} \sum_{p\in\Omega} \left| \frac{\delta \alpha_p^i}{\delta x} \right| \left\| \frac{\delta F_p^i}{\delta x} \right\|^2 + \\ &+ \sum_{i=1}^{n} \sum_{p\in\Omega} \left| \frac{\delta \alpha_p^i}{\delta y} \right| \left\| \frac{\delta F_p^i}{\delta y} \right\|^2 + \\ &+ \sum_{i=1}^{n} \sum_{p\in\Omega} \alpha_p^i \left\| \frac{\delta F_p^i}{\delta z} \right\|^2, \end{aligned} \tag{4}$$

where Ω is the image area and $\frac{\delta}{\delta x}$, $\frac{\delta}{\delta y}$ and $\frac{\delta}{\delta z}$ are derivatives computed along the x, y, and z axes, respectively (we use the z axis to index disparity layers). The first term ensures that the composition of the multilayer representation is equal to the input image (see Equation (1)). The second and the third terms enforce smooth color transitions in the spatial domain. These three terms are a trivial generalization of the cost function from [11]. The fourth term enforces smooth color transitions between neighboring layers with nonzero transparency. In our experiments, this term enabled us to significantly increase the numerical stability of our approach.

(a) Multilayer disparity map before shift

(b) Multilayer disparity map after shift

Fig. 4. Example of cracking caused by a discrete representation of the disparity map: (a) shows one line from the multilayer disparity map depicting a solid object with nonconstant disparity; (b) shows a crack (i.e., one semitransparent pixel) that appears inside the solid object after the shift.

We compute the layer images F by minimizing the cost function:

$$F = \arg\min_{F} C(I,\alpha,F). \tag{5}$$

To obtain this minimum, we can solve a system of linear equations. Notably for a typical disparity map with 1920×1080 resolution, this system of equations will have tens of millions of variables. Because our method fails to reliably compute the color of pixels with zero transparency, we eliminate variables corresponding to invisible pixels that lack visible neighbors in a circle of radius w. According to our observations, this transformation reduces the number of variables by a factor of 10.

3.4. Occlusion inpainting

The multilayer input-image representation we computed in the previous steps (see Equation (1)) is unsuitable for the occlusion-inpainting and layer-shifting steps, because it fails to take into account the order of the layers. Thus, we transform it to a representation that models the composition of layers in order by distance from the viewer:

$$\begin{aligned} &I^0 = \tilde{\alpha}^0 F^0, \beta^0 = \tilde{\alpha}^0 \\ &I^i = (1-\beta^0)\tilde{\alpha}^i F^i + I^{i-1}, \beta^i = (1-\beta^0)\tilde{\alpha}^i + \beta^{i-1} \\ &I = I^n. \end{aligned} \tag{6}$$

This representation matches the one that some image- and photo-editing software packages (e.g., Adobe Photoshop) employ. We can convert representation (1) to representation (6)

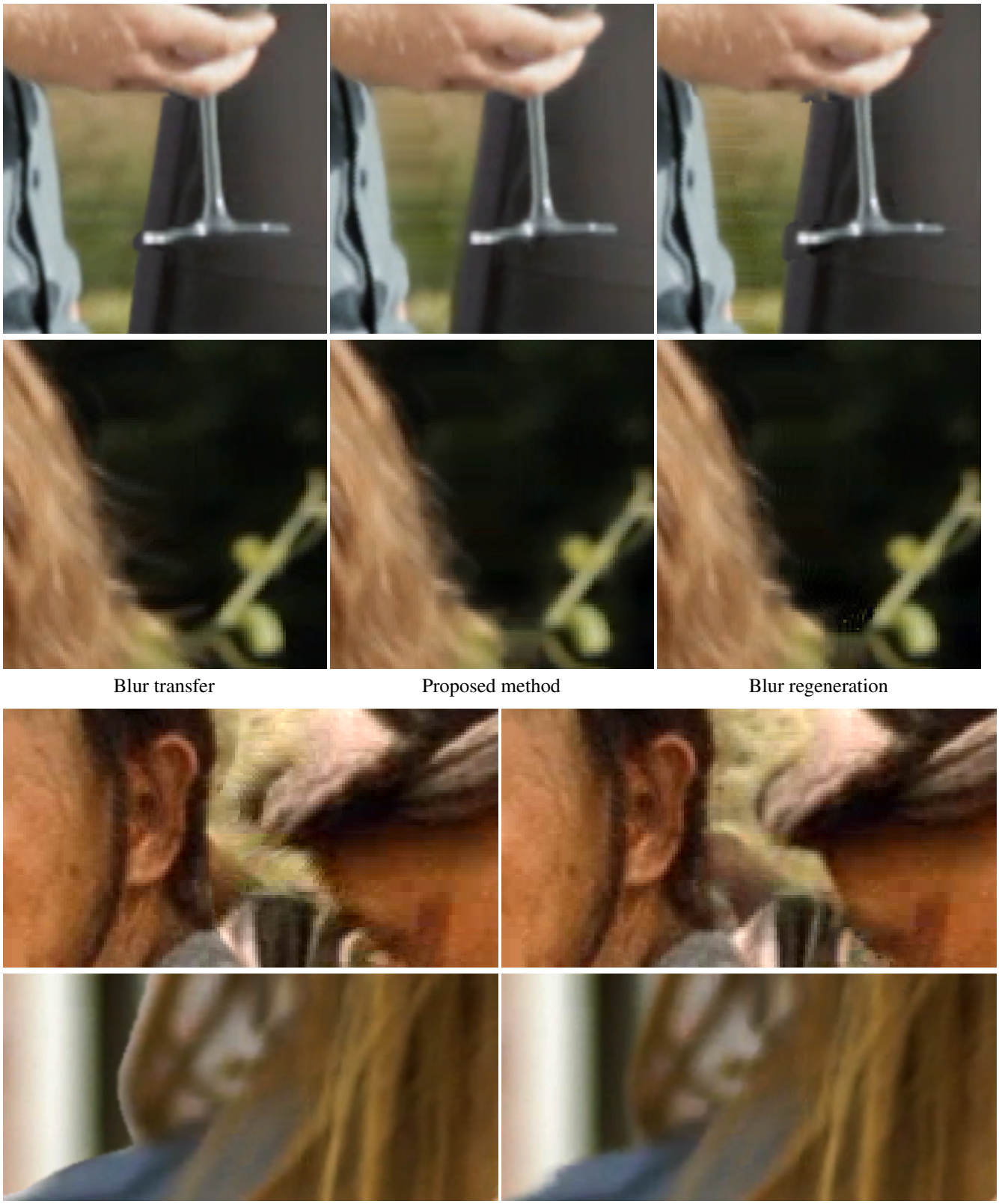

Fig. 5. Example views generated using our proposed method, blur transfer, and blur regeneration. As the figure shows, a blur transfer often produces artifacts that appear as parts of the background stuck to a foreground object. The blur-regeneration method creates artifacts at intersections of three or more objects, since two-layer segmentation is impossible for such areas.

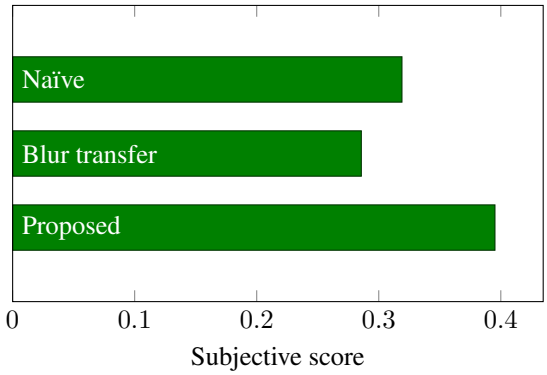

Fig. 6. Subjective scores computed using the Bradley-Terry model for pairwise comparison. We compared our proposed method with blur transfer and naïve approaches by asking 56 participants to evaluate 12 test images.

by using the following equation:

$$\begin{aligned} \tilde{\alpha}^0 &= \alpha^0 \\ \tilde{\alpha}^i &= \frac{\alpha^i}{1 - \sum_{j=0}^{i-1} \alpha^j}. \end{aligned} \tag{7}$$

This equation is undefined for any pixel underneath a stack of pixels with a total transparency equal to 1.0 (i.e., the pixel is completely occluded by pixels in upper layers). We generate the multilayer mask U containing these undefined pixels, then dilate the mask U by two pixels in the horizontal direction to the location of pixels with a transparency less than 0.1. Thus, we tend to eliminate pixels with poorly defined color.

We propose two approaches to filling occluded pixels (i.e., pixels in U): linear interpolation and incorporating information from the background image, which is provided as an additional input.

To fill occluded pixels by linear interpolation, we consider each unknown pixel u in layer i and find its closest left (l) and right (r) neighbors not in U. We search for l and r only in that layer and only in the horizontal line containing u. We compute the pixel's color F_u^i and transparency $\tilde{\alpha}_u^i$ as a linear combination of its neighbors:

$$\begin{aligned} \tilde{\alpha}_u^i &= \frac{d(r,u)}{d(r,u)+d(l,u)}\tilde{\alpha}_l^i + \frac{d(l,u)}{d(r,u)+d(l,u)}\tilde{\alpha}_r^i, \\ F_u^i &= \frac{d(r,u)}{d(r,u)+d(l,u)}F_l^i + \frac{d(l,u)}{d(r,u)+d(l,u)}F_r^i, \end{aligned} \tag{8}$$

where $d(\cdot,\cdot)$ denotes the Euclidean distance. If the pixel has only a left or right neighbor, we set its color and transparency equal to those of its neighbor. Finally, we set the color and transparency of any unknown pixel u in layer i equal to the color and transparency of pixel u in layer $i-1$ if that pixel is not in U. This step enables us to inpaint cracks (i.e., small occlusions of width less than one pixel caused by the discrete representation of the disparity map; Fig. 4 shows an example of such a crack).

If the background image and background disparity maps are provided as additional inputs, we transform them into a multilayer representation using the same method. Then we inpaint any information missing from the background image using the linear interpolation procedure described above. Finally, we replace unknown pixels in U with pixels from the background multilayer representation.

3.5. Layer shifting and generation of S3D views

Having a multilayer representation with all unknown pixels filled, we can warp it to the new S3D view by applying an individual shift to each layer. To generate the left view we shift layer i by $pi - z$ pixels ($-pi - z$ for the right view), where p and z are user-specified parameters denoting the strength of the 3D effect and the position of the screen plane, respectively. Finally, to generate the view image, we merge the shifted layers according to Equation (6).

4. RESULTS

We performed a visual comparison of the stereo views generated using our proposed approach with the results of two alternative approaches:

1. Blur transfer (we used our own implementation of the method described in [8])

2. Blur regeneration by two-layer matting (we used our own implementation similar to the one described in [10]).

All compared approaches used linear interpolation for occlusion inpainting, since no background image was available. Fragments of the views generated using these methods appear in Fig. 5. As the figure shows, the blur transfer method often produces artifacts that look like parts of the background stuck to a foreground object. The blur-regeneration method creates artifacts at intersections of three or more objects, since a two-layer segmentation is impossible for such areas.

Additionally, we performed a subjective pairwise visual-quality comparison of the generated views through the Subjectify.us web service [1]. We applied our proposed method, the blur-transfer method and the naïve view-generation approach to 12 test images with available depth maps (background image was not provided to methods). The web application presented 56 participants with the generated views in 2D mode. These participants were shown a sequence of view pairs generated by different methods; for each pair, we asked them to choose the one with better visual quality or indicate that they are approximately equal. Participants received $0.05 to compare 12 such pairs. Using the Bradley-Terry model [6], we transformed the pairwise data into subjective scores (see Fig. 6). Our proposed method offers the best visual quality according to the subjective-comparison results.

5. CONCLUSION

We have proposed a method for generating new S3D views from a given input image and its disparity/depth map. The main feature of our approach is correct processing of semi-transparent object edges. The proposed multilayer representation of input data naturally encodes the multilayer structure of natural images, enabling us to handle intersections of more than two objects located at various depths. On the basis of a subjective comparison, our method outperforms the alternative methods. Nevertheless, the proposed method has extremely high computational complexity and memory requirements (e.g. the size of multilayer representation for Full HD input is 2 GB). Furthermore, the optimization scheme discussed in Section 3.3 tends to be numerically unstable for some input images. These drawbacks require further research and limit practical applications of the proposed method.

6. REFERENCES

[1] `http://subjectify.us/`.

[2] Furnace. `https://www.thefoundry.co.uk/products/plugins/furnace/`. : 2016-05-23.

[3] Real 3D or Fake 3D. `http://realorfake3d.com/`. : 2016-05-23.

[4] Yangkeun Ahn and Jiman Hong. Application of DIBR algorithm in real-time image. In *Proceedings of the 2012 ACM Research in Applied Computation Symposium*, pages 169–171. ACM, 2012.

[5] Dmitry Akimov, Dmitry Vatolin, and Maxim Smirnov. Single-image depth map estimation using blur information. In *Proceedings of the 21st International Conference on Computer Graphics and Vision GraphiCon'2011*, GraphiCon, pages 12–15. Moscow, Russia, 2011.

[6] Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[7] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems*, pages 2366–2374, 2014.

[8] N. Fukushima, N. Kodera, Y. Ishibashi, and M. Tanimoto. Comparison between blur transfer and blur regeneration in depth image based rendering. In *2014 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, pages 1–4, July 2014.

[9] Kaiming He, Jian Sun, and Xiaoou Tang. Guided image filtering. In *Computer Vision–ECCV 2010*, pages 1–14. Springer, 2010.

[10] Naoki Kodera, Norishige Fukushima, and Yutaka Ishibashi. Filter based alpha matting for depth image based rendering. In *Visual Communications and Image Processing (VCIP), 2013*, pages 1–6. IEEE, 2013.

[11] Anat Levin, Dani Lischinski, and Yair Weiss. A closed-form solution to natural image matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (TPAMI), 30(2):228–242, 2008.

[12] W. N. Lie, C. Y. Chen, and W. C. Chen. 2D to 3D video conversion with key-frame depth propagation and trilateral filtering. *Electronics Letters*, 47(5):319–321, March 2011.

[13] Wen-Nung Lie, Chun-Cheng Yeh, and Guo-Shiang Lin. Improving DIBR technique to resolve foreground color/depth edge misalignment. In *Multimedia & Expo Workshops (ICMEW), 2015 IEEE International Conference on*, pages 1–5. IEEE, 2015.

[14] Alasdair Newson, Andrés Almansa, Matthieu Fradet, Yann Gousseau, and Patrick Pérez. Video inpainting of complex scenes. *SIAM Journal on Imaging Sciences*, 7(4):1993–2019, 2014.

[15] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 231–242, New York, NY, USA, 1998. ACM.

[16] Ekaterina Tolstaya, Petr Pohl, and Michael Rychagov. Depth propagation for semi-automatic 2D to 3D conversion. volume 9393, pages 939303–939303–8, 2015.

[17] Dmitriy Vatolin, Alexander Bokov, and Alexey Fedorov. The trends in technical quality of stereoscopic movies - 5 years after "Avatar". *World of Technique of Cinema*, 37(3):17–28, 2015.

[18] A. Zachesov, M. Erofeev, and D. Vatolin. Depth map aided background reconstruction. In *New infromation technologies in autonomous systems*. HSE Moscow Institute of Electronics and Mathematics, 2015.

[19] Yuanjie Zheng and C. Kambhamettu. Learning based digital matting. In *International Conference on Computer Vision* (ICCV), pages 889–896, 2009.

[20] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 600–608. ACM, 2004.