DEEPSTEREOBRUSH: INTERACTIVE DEPTH MAP CREATION

Sebastian Knorr[1,3,*], *Matis Hudon*[1,*], *Julian Cabrera*[2,*], *Thomas Sikora*[3], *Aljosa Smolic*[1 †]

[1]Trinity College Dublin, [2]Universidad Politécnica de Madrid, [3]Technische Universität Berlin

ABSTRACT

In this paper, we introduce a novel interactive depth map creation approach for image sequences which uses depth scribbles as input at user-defined keyframes. These scribbled depth values are then propagated within these keyframes and across the entire sequence using a 3-dimensional *geodesic distance transform* (3D-GDT). In order to further improve the depth estimation of the intermediate frames, we make use of a convolutional neural network (CNN) in an unconventional manner. Our process is based on online learning which allows us to specifically train a disposable network for each sequence individually using the user generated depth at keyframes along with corresponding RGB images as training pairs. Thus, we actually take advantage of one of the most common issues in deep learning: over-fitting. Furthermore, we integrated this approach into a professional interactive depth map creation application and compared our results against the state of the art in interactive depth map creation.

Index Terms— depth estimation, 2D-to-3D conversion, deep learning, CNN, interactive depth map creation, geodesic distance transform

1. INTRODUCTION

Depth estimation from monocular images and video sequences is an ongoing and highly active research area with many computer vision applications like e.g. robot navigation, free-viewpoint-video, 2D to stereo 3D conversion, urban reconstruction, object segmentation and detection, etc.. Despite being a classical problem in computer vision, inferring the geometrical distribution of a generic scene from a single view is still an open research topic since it is an ill-posed problem. Approaches for depth map creation can be divided into three categories: manual approaches as currently used in high-quality cinematic 2D-to-3D conversion workflows [20, 23], interactive (semi-automatic) approaches which are a combination of user-input and computer vision algorithms, e.g. [25, 7, 16, 19, 15], and automatic approaches entirely based on traditional computer vision algorithms for depth estimation [29, 10] or, more recently, on convolutional neural networks, e.g. [3, 18, 13, 5, 27, 8].

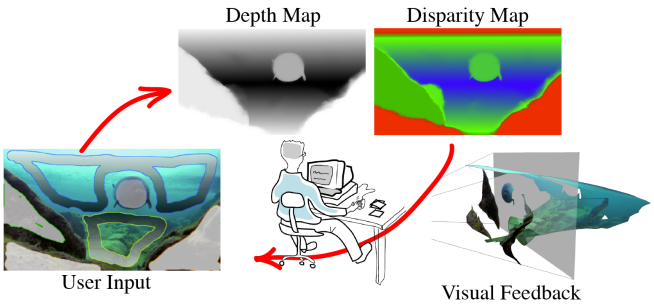

Fig. 1: Interactive realtime 2D-to-3D conversion workflow.

Fully automatic approaches aim at exploiting different visual cues such as camera movement, object movement, illumination, etc. to estimate the depth of the scene. Results provided by these approaches are generally characterized by a limited accuracy and temporal inconsistency. In this sense, recent works based on deep learning techniques have shown promising results, but they do not generalize well and can only be applied on images with similar scene content. Furthermore, these approaches also suffer from the lack of quality of the underlying training data. They are often generated from stereo datasets using disparity estimation or are captured with e.g additional time-of-flight cameras [6]. Thus, they are error-prone themselves, i.e. these errors will be learned by the network. Therefore, fully automatic approaches are not able to generate depth maps of sufficient quality for several application domains such as 3D-TV or 3D cinema. Moreover, automatic approaches do not allow any creative input which is often essential for cinematic film productions.

On the other hand, manual approaches are time- and labor-intensive [20, 23]. First, an accurate object segmentation of the scene has to be performed using specific rotoscoping or keying tools, and then a manual depth assignment for each object has to be carried out. This is usually a costly process and only affordable for cinematic film productions [23].

The main motivation of this paper is to speed-up the process of 2D-to-3D conversion of video sequences while improving accuracy and consistency of the resulting depth maps in an interactive manner as depicted in Fig. 1. Therefore, we

*These authors contributed equally to this work.

†This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under the Grant Number 15/RP/2776. This work has also been partially supported by the Ministerio de Economía, Industria y Competitividad (AEI/FEDER) of the Spanish Government under project TEC2016-75981 (IVME). The Titan X Pascal used for this research was donated by the NVIDIA Corporation.

introduce a novel interactive depth map creation framework for video sequences which is integrated in the professional interactive 2D-to-3D conversion software *imcube 3D Daily*[1]. A user scribbles depth values into objects or areas of selected keyframes. These scribbled depth values are then propagated within these keyframes and across the entire sequence using a 3-dimensional *geodesic distance transform* (3D-GDT), which is the first main contribution of this work. We will show that our GDT-based approach outperforms the StereoBrush introduced in [25].

As a second main contribution, we replace the 3D-GDT for depth propagation between keyframes and introduce a novel approach for intermediate depth map estimation by making use of a convolutional neural network (CNN) in an unconventional manner. In our approach we do not target to solve the general problem of estimating the depth map of any given image. Instead, we consider the problem of estimating the depth maps of a limited set of images like can be the case of frames within a shot or a scene in a video sequence. Our process is based on online learning which allows us to train a disposable network for each sequence individually using the user generated depth at keyframes along with corresponding RGB images as training pairs. Thus, we actually take advantage of one of the most common issues in deep learning: the lack of generalization more popularly known as over-fitting.

Our CNN is based on the classic encoder-decoder architecture that has proven to be efficiently applied to a wide variety of computer vision related problems. In this work we show that it can also: (i) learn how to compute accurate depth maps for the frames within the considered set, and (ii) perform as an interactive tool since its training dataset consists of just the keyframes selected by the user. To the best of our knowledge, the combination of annotation (here the scribbling of depth) and online deep learning for 2D-to-3D conversion, which we refer to as *DeepStereoBrush* (DSB), has never been introduced in the literature before. Finally, We will show that the depth map interpolation based on deep learning outperforms the 3D-GDT and thus yields better results for the entire image sequence.

2. RELATED WORK

Interactive and efficient tools for object segmentation, natural image matting and video colorization have been introduced in the literature for more than a decade, e.g. in [11, 28, 12, 1, 24]. A first attempt to convert video footage with interactive user input in the form of scribbles was proposed by Guttmann et al. [9]. Their approach combines a diffusion scheme, which takes into account the local saliency and the local motion over time, coupled with a classification scheme that assigns depth to image patches. In [7], the authors introduced a simple stroke-based depth estimation framework for images. Since humans are not efficient in estimating absolute depth values, they also introduced *normal* strokes, which enable the user to place surface orientation constraints.

Liao et al. [16] proposed a semiautomatic system that converts conventional videos into stereoscopic videos by combining motion analysis, such as structure-from-motion (SfM) and optical flow estimation, with user interaction. They compared their approach with subjective experiments against [9] and showed that it is less time consuming while improving the subjective quality of the stereo results. However, the authors only provide resulting anaglyph pictures of the evaluated scenes but no depth maps which makes it difficult to assess their depth estimation approach. In [19], the authors introduced a method to obtain a depth map from a single image of a scene by exploiting both image content and user interaction. Compared to previous works, their system allows different kinds of user inputs such as perspective and equality constraints in certain areas of the scene. More recently, Liao et al. [14] presented a novel pipeline to generate a depth map from a single image which supports a variety of user controls, such as a non-linear depth mapping, a steering mechanism for their diffusion method (e.g., directionality, emphasis, or reduction of the influence of image cues), and, similar to [19], relative depth indications. While such additional constraints improve the quality of the depth maps, the user intuition decreases due to more manual input. A fact that these two approaches are more suitable for higher quality but single image conversion. Inspired by sculpturing, Lin et al. [17] addressed the depth map generation of 2D paintings as an iterative stroking-and-viewing process. Instead of time-consuming optimization of cost functions, they formulated the problem as a filter-based scheme to achieve reasonable interactive response time.

Finally, Wang et al. [25] introduced the so called *StereoBrush*, which integrates interactive depth estimation using sparse user scribbles and stereo rendering into a single optimization framework to convert 2D images and videos directly into stereo 3D. Their method assumes a piecewise continuous depth representation, preserving visual continuity in most areas, while creating sharp depth discontinuities at important object boundaries. A similar framework was proposed in [26]. While [25] is based on the minimization of a cost function, [26] employs a concept called *geodesic distance transform* (GDT), which we have integrated into our framework in a more user-friendly and GPU-optimized manner (see Section 3 for details).

3. INTERACTIVE FRAMEWORK

In this section, we introduce our approach of a professional interactive 2D-to-3D conversion workflow (see Fig. 1). A user scribbles depth values into objects or areas of selected keyframes and the software estimates a dense depth map by propagating the depth values between all pixels within this frame using the *geodesic distance transform* (GDT). The user can further improve the results by modifying the depth scribbles or adding new depth scribbles. While the user is scrib-

[1] https://imcube.jimdo.com/products/2d-to-3d/3d-daily/

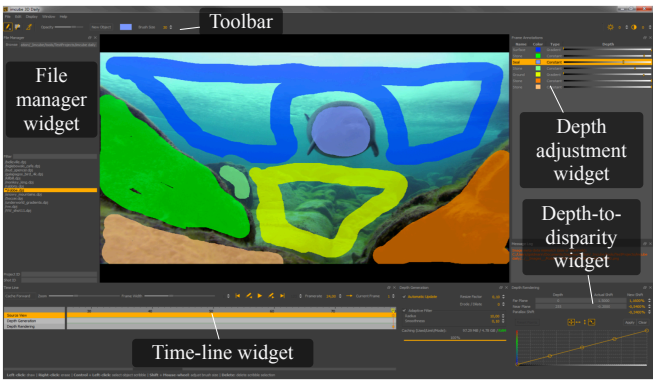

Fig. 2: Screenshot of the GUI of *imcube 3D Daily* with annotated (colored) scribbles. The depth of each scribble can be assigned and adjusted with the widget on the top right.

bling depth values into the image, he gets immediate visual feedback either in the form of depth maps, stereo left and right views or as a 3D model of the scene. A screenshot of the graphical user interface (GUI) is depicted in Fig. 2. The GUI consists of a couple of widgets which allow easy manipulation of e.g. depth settings (depth adjustment widget for depth fine-tuning) and rendering settings (depth-to-disparity widget for linear or non-linear mapping of depth values to disparity).

3.1. Geodesic distance transform

The *geodesic distance transform*, which is a specialization of the general distance [22] takes a binary mask image M as input, which contains foreground pixels with value 1 and background (or seed) pixels with value 0. According to Fabbri et al. [4] the distance transform of M is a map, in which each pixel contains the shortest distance to any of the seed pixels using a distance function $d(\mathbf{p}, \mathbf{q})$:

$$D(\mathbf{p}) = min\{d(\mathbf{p}, \mathbf{q}) \mid M(\mathbf{q}) = 0\}, \tag{1}$$

where D is the *distance map* or *distance transform* of M, and $\mathbf{p}$ and $\mathbf{q}$ are pixel locations. Yatziv and Sapiro [28] defined the distance function as follows:

$$d(\mathbf{p}, \mathbf{q}) = \min_{\Gamma \in \mathcal{P}_{p,q}} \int_0^1 \mid \nabla I(s) \cdot \Gamma'(s) \mid \; ds, \tag{2}$$

where $\mathcal{P}_{p,q}$ denotes the set of all possible paths between the points $\mathbf{p}$ and $\mathbf{q}$. $\Gamma(s) : \mathbb{R} \rightarrow \mathbb{R}^2$ is one of those paths parametrized by its arclength $s \in [0, 1]$ and $\Gamma'(s)$ is its spatial derivitive, which is a unit vector along the direction of the path, i.e. this distance represents the shortest possible path between pixel $\mathbf{p}$ and $\mathbf{q}$, whereas only color or intensity differences between pixels in image I are considered and no spatial information.

According to Criminisi et al. [2], the integral in Equation (2) can be approximated with the following sum:

$$\sum_{k=1}^{n} \parallel I(x_k) - I(x_{k-1}) \parallel, \tag{3}$$

where $\parallel \cdot \parallel$ denotes the L2-norm (i.e. the Euclidean norm). The possible paths between $\mathbf{p}$ and $\mathbf{q}$ are assumed to be chains of neighboring pixels $(x_0 = \mathbf{p}, x_1, ..., x_{n-1}, x_n = \mathbf{q})$, considering an eight-neighborhood structure on the pixel grid.

An example for the GDT applied on an image is shown in Fig. 3. We refer to [26] for a detailed description of the implementation of the GDT.

3.2. Blending

In order to propagate scribbled depth values using the GDT, the image colorization approach of Yatziv and Sapiro [28] is employed. The main idea is to compute the GDT for each distinct scribble value and then blend these values for each pixel independently by weighting the values based on their corresponding distance at this pixel. First, a distance map D_s has to be computed for each distinct scribble value s using Equations (1) and (2). Therefore, the masks M_s have to be computed first by assigning a value of 0 to scribbled pixel locations and 1 anywhere else.

With the obtained distance maps, Yatziv and Sapiro [28] compute the final color value by a weighted sum of the scribbled colors for each pixel, what they refer to as color blending. Replacing the color by depth, this is expressed by:

$$depth(\mathbf{p}) = \frac{\sum_{\forall s} W\left[D_s(\mathbf{p})\right] \cdot s}{\sum_{\forall s} W\left[D_s(\mathbf{p})\right]}, \tag{4}$$

where $W[\cdot]$ is a function that transforms small distances into high values and vice versa and can be defined as:

$$W(d) = d^{-r}, \; r > 0, \; d > 0, \tag{5}$$

where r is a blending parameter and set to $r = 3$ in our implementation. Note that d may have values equal to zero after applying the GDT, e.g. at scribbled pixels. Thus, the distance d has to be thresholded before applying this function. Further note that, using such a blend function, the final depth value at each pixel is actually a mixture of all existing scribble values. However, the blend function ensures that the scribble value with a small distance gets a high weight and hence a high contribution to the final depth value.

The whole propagation procedure is summarized in the flowchart in Fig. 3.

3.3. Depth propagation in image sequences

In order to extend the 2-dimensional GDT for single frames to image sequences, temporal neighbors have to be included into the neighborhood definition when computing the GDT. This can be seen as stacking all the images of the sequence on top of each other to form a 3-dimensional image cube. Such representation had also been introduced in [1] but for segmentation and matting in videos. In our GDT-based depth map creation approach, we increase the eight-neighborhood structure on the pixel grid as described in Section 3.1 in the temporal

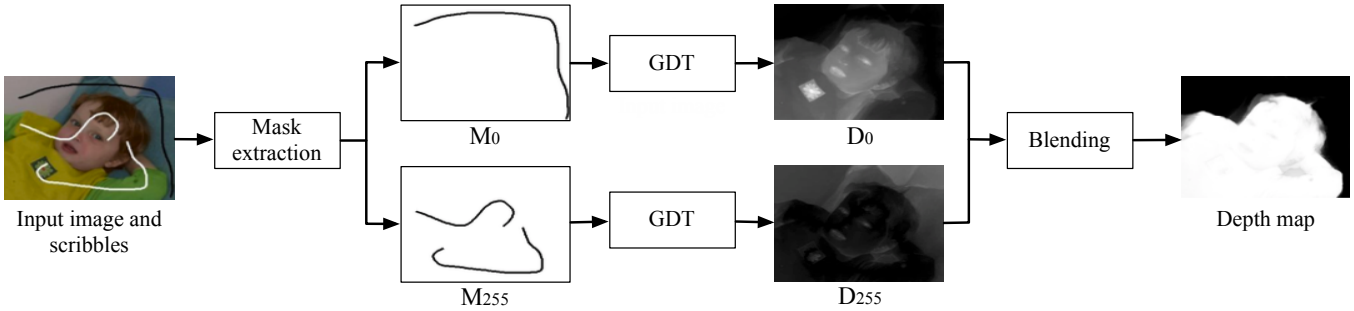

Fig. 3: This flowchart illustrates the scribble propagation procedure for a sample image with black (gray value 0) and white(gray value 255) scribbles. Note that disconnected strokes are treated as a single scribble if they have the same value [26].

domain by adding two neighboring pixels at the same location of the source pixel in the consecutive frames. We refer to this spatio-temporal GDT as 3D-GDT.

Please note that in case of fast motion of objects, which had been scribbled in one keyframe, an object in two successive frames may not overlap anymore, i.e. the color may change for the two temporal neighbor pixels resulting in a high geodesic distance [26]. In order to deal with fast moving objects, additional keyframes with input scribbles need to be selected.

4. DEEP LEARNING APPROACH FOR DEPTH MAP INTERPOLATION

Deep learning solutions have been succesfully proposed for different computer vision related problems, including depth estimation from a single view. Nevertheless, the quality of the depth maps provided by these fully automatic approaches based on CNNs is still far from the results obtained using semi-automatic tools. In this work, we do not address the problem of automatic estimation of depth maps for generic scenes. Instead, we focus on the more constrained problem of estimating depth maps for the intermediate frames of the sequence. For that, we propose a CNN that is only trained with the scribbled depth maps of the keyframes.

Our proposed CNN follows an encoder-decoder architecture, since this type of network has proven to produce state of the art results for several computer vision problems including latest research on depth estimation [5][8]. More specifically, the encoder consists of seven convolution steps that half the dimensions of the image in each step. The convolution blocks consist of a triplet of a 2D convolution, a batch normalization and a ReLu layer. The decoder consists of seven deconvolution blocks which follow the upsampling-deconvolution scheme proposed in [21] in order to minimize the appearance of checkerboard pattern artifacts. Additionally, to preserve the high frequency details of the images, skip connections with the features computed at the encoder are used. Finally, dropout layers are used for the first two deconvolution blocks to prevent the network from an excessive over-fitting. A summary with the main characteristics of the DeepStereoBrush CNN is presented in Table 1.

	Convolution block	Output dimensions
1	5×5 2D-conv, stride 2, F filters	$H/2 \times W/2 \times F$
2	Batch normalization	$H/2 \times W/2 \times F$
3	ReLu	$H/2 \times W/2 \times F$
	Deconvolution Block	**Output dimensions**
1	Image upsampling $\times 2$	$H \times 2 \times W \times 2 \times F$
2	5×5 2D-conv, stride 1, F filters	$H \times 2 \times W \times 2 \times F$
3	Batch normalization	$H \times 2 \times W \times 2 \times F$
4	ReLu	$H \times 2 \times W \times 2 \times F$
	DeepStereoBrush CNN	**Output**
1	5×5 2D-conv, stride 2, 64 filters	$H/2 \times W/2 \times 64$
2	ReLu	$H/2 \times W/2 \times 64$
3	Conv block, 128 filters	$H/4 \times W/4 \times 128$
4	Conv block, 256 filters	$H/8 \times W/8 \times 256$
5	Conv block, 512 filters	$H/16 \times W/16 \times 512$
6	Conv block, 512 filters	$H/32 \times W/32 \times 512$
7	Conv block, 512 filters	$H/64 \times W/64 \times 512$
8	Conv block, 512 filters	$H/128 \times W/128 \times 512$
9	Deconv block, 512 filters	$H/64 \times W/64 \times 512$
	Dropout 50 %	$H/64 \times W/64 \times 512$
	Concat with layer 7	$H/64 \times W/64 \times 1024$
10	Deconv block, 512 filters	$H/32 \times W/32 \times 512$
	Dropout 50 %	$H/32 \times W/32 \times 512$
	Concat with layer 6	$H/32 \times W/32 \times 1024$
11	Deconv block, 512 filters	$H/16 \times W/16 \times 512$
	Concat with layer 5	$H/16 \times W/16 \times 1024$
12	Deconv block, 256 filters	$H/8 \times W/8 \times 256$
	Concat with layer 4	$H/8 \times W/8 \times 512$
13	Deconv block, 128 filters	$H/4 \times W/4 \times 128$
	Concat with layer 3	$H/4 \times W/4 \times 256$
14	Deconv block, 64 filters	$H/2 \times W/2 \times 64$
	Concat with layer 2	$H/2 \times W/2 \times 128$
13	Deconv block*, 64 filters	$H \times W \times 64$
	*No batch normalization, No ReLu	
14	Tanh	$H \times W \times 1$

Table 1: Summary of the end-to-end DeepStereoBrush CNN for estimating depth maps of intermediate frames.

The model is trained end-to-end from a random initialization in a supervised manner using as ground truth the depth data from the scribbled key frames and the GDT algorithm. The loss function selected is the L1-norm between the estimated disparity and its correspondent ground truth value.

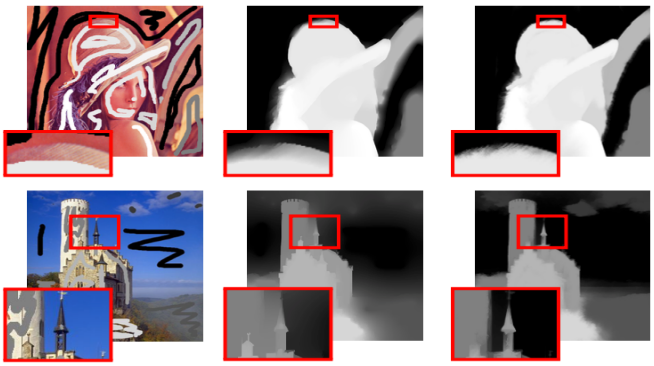

Fig. 4: Comparison between the StereoBrush (optimization-based) [25] (middle column) and our GDT-based approach [26] (right column) using the same input scribbles.

5. RESULTS

5.1. Comparison of GDT with StereoBrush

We compared our GDT-based approach against a re-implementation of the StereoBrush (i.e., an optimization-based approach) introduced by Disney Research in [25]. Fig. 4 shows exemplary results of the optimization-based and GDT-based approach for two source images with user scribbles. While the optimization-based approach creates smoother depth maps, the GDT approach delivers sharper depth maps along edges, which is usually preferred in order to reduce the so-called *rubber matte effect*[2] at depth jumps when converting the image to stereo 3D. The *smoothing* of the depth maps for the optimization-based approach may even result in erroneous depth between objects as can be noticed in the middle of the bottom row in Fig. 4 (see close-up). A main disadvantage of the optimization-based approach, however, is its computational complexity which makes the interaction approximately 5 to 10 times slower compared to the GDT-based approach.

In Fig. 5 we compare our GDT approach against the StereoBrush on the picture set used in [25]. Note that we used the depth maps provided by Disney Research Zurich and did not run our re-implementation of the StereoBrush on this data. Again, the GDT approach delivers sharper edges at depth jumps which is preferred. Furthermore, the StereoBrush seemed to cause errors in certain regions. For instance, the depth propagation at the top right corner of the *cave* picture (first column) is incorrect. Such incorrect depth propagation is even more noticeable in the picture with the drawing of the rail tracks. The depth at the end of the tunnel does not match with the scribbled depth.

A more comprehensive evaluation of both approaches can be found in [26]. The results show that the GDT-based approach outperforms the optimization-based approach of the StereoBrush for both single images and image sequences.

[2]https://www.fxguide.com/featured/art-of-stereo-conversion-2d-to-3d-2012/

5.2. Performance evaluation of our online deep learning based interpolation method

In order to create a depth map sequence from a RGB sequence, we first scribbled keyframes within an image sequence and computed the depth within each keyframe using our GDT-based approach. Then, we applied two different interpolation techniques, the 3D-GDT described in section 3.3 and the DSB-CNN based on the online trained CNN described in section 4, to eventually fill the gaps between the scribbled keyframe depth maps. Finally, we compare quantitatively and qualitatively the results of the two interpolation techniques versus ground truth for a set of 9 image sequences containing different scenes: static, dynamic, outdoor, indoor and even an underwater shot (sequence 9). The scribbled keyframe depth maps are used for training the network specifically for each sequence. The working image resolution is 480x270 pixels and the number of keyframes for each sequence is included in Table 2. In our experiments, we found that training such a network is relatively fast and takes about 10 to 30 min per shot with an Nvidia Titan Xp (see section 6 for a further discussion). The ground truth depth maps had been obtained from a manual cinematic 3D conversion workflow, and are only used as reference for the depth scribble process and for the comparison of both interpolation approaches.

Table 2 shows the mean absolute error (MAE) of interpolated frames (scribbled keyframes excluded) for both techniques versus ground truth depth. For eight over nine sequences the DSB interpolation outperforms the 3D-GDT. Furthermore, in the only sequence were the 3D-GDT performs better (sequence 8) the two average mean absolute difference errors are very close. We also show per-frame mean absolute error in Fig. 6 for eight of our nine sequences. These curves show that the DSB-CNN produces in general more stable results than the 3D-GDT when interpolating between scribbled keyframe depth maps (lower average standard deviation of the mean absolute error). Note that this is also the case for sequence 8, despite the slightly higher average error of the DSB-CNN, the curve seems to indicate that the result is more consistent temporally. Fig. 7 shows a visual comparison for several frames of the estimated and the ground truth depth maps from sequence 9. In this case the interpolated depth maps using DSB-CNN are also visually better than the ones interpolated using 3D-GDT. DSB-CNN produces more consistent and also more accurate depth maps when comparing to ground truth. At this point we need to mention that we scribbled depth values into objects using the ground truth as reference in order to eventually compare our results on the same basis. In a real production, ground truth is not needed.

Finally, Fig. 8 shows different renderings of frame 75 of sequence 3: the anaglyph stereo images, color-coded disparity maps and 3D scene views. The DSB-CNN leads to renderings closer to ground truth and is therefore more realistic with a much higher quality and depth range than the 3D-GDT.

Fig. 5: Comparison between the StereoBrush (optimization-based) [25] (middle row) and our GDT-based approach [26] (bottom row) using similar but not identical scribbles.

Sequence	1	2	3	4	5	6	7	8	9
# of frames (# of keyframes)	94 (5)	275 (12)	393 (17)	325 (8)	300 (12)	581 (14)	33 (4)	127 (6)	25 (6)
3D-GDT	16.15	5.44	12.13	24.23	9.55	8.84	19.64	**10.63**	16.01
DSB-CNN	**11.42**	**4.04**	**9.04**	**22.15**	**9.35**	**7.95**	**18.86**	10.86	**13.22**

Table 2: Mean absolute error comparison of 3D-GDT versus DSB-CNN. Depth values are coded between 0 and 255.

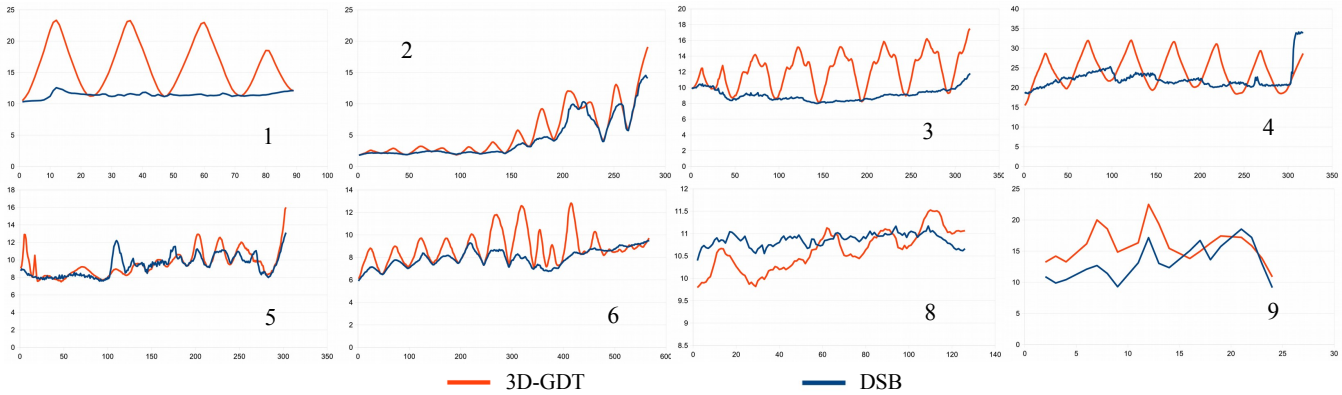

Fig. 6: Per frame mean absolute error (sequences 1-2-3-4-5-6-8-9) of 3D-GDT (red) versus DSB-CNN (blue).

This difference in subjective quality can also be observed in the objective results in Fig 6. As sequence 3 is a dynamic shot with tracking camera, foreground objects in intermediate frames may not overlay with the original scribbles at the keyframes any longer as mentioned in section 3.3. Instead, they may cross the position of scribbles which belong to the background, i.e. foreground objects will move closer to the background as can be seen on the left and right side of the desk as well as at the candles in the middle column of Fig. 8.

6. CONCLUSION

In this paper, we introduced a novel interactive depth map creation framework for image sequences which uses depth-scribbles as input at user-defined keyframes. These scribbled depth values were then propagated within these keyframes and across the entire sequence using a 3-dimensional *geodesic distance transform* (3D-GDT). In order to further improve the depth estimation of the intermediate frames, we applied online learning of a CNN for each sequence individually using

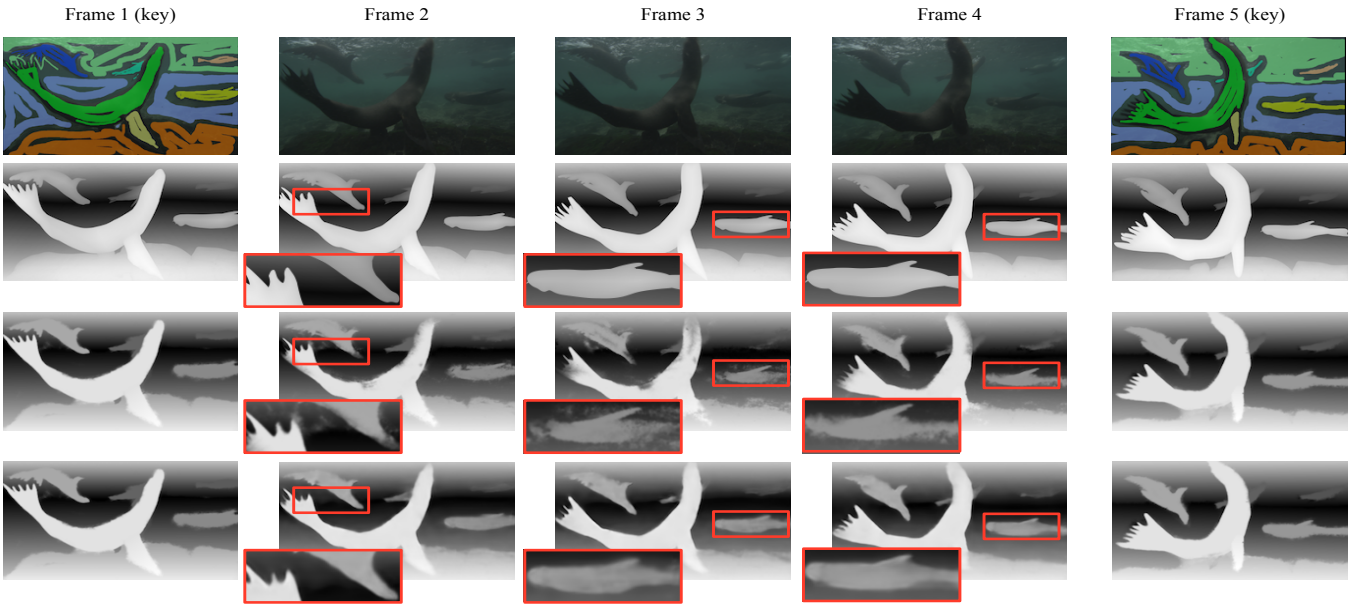

Fig. 7: Sample depth maps of sequence 9: source images with scribbles on keyframes (1st row), high quality ground truth depth maps from manual cinematic 3D conversion workflow (2nd row), 3D-GDT (3rd row), DSB-CNN (4th row).

the user generated depth at keyframes along with corresponding RGB images as training pairs. Furthermore, we integrated this approach into a professional interactive depth map creation application and compared our results against the state of the art in interactive depth map creation.

We first demonstrated that our GDT-based approach outperforms the optimization-based approach introduced by Disney Research for pictures. In a second experiment, we evaluated and compared our 3D-GDT and DSB-CNN approach on intermediate frames, i.e. non-scribbled frames. Our results showed that the DSB-CNN outperforms the 3D-GDT for most of the sequences in terms of MAE and for all sequences in terms of stability and consistency over time.

The online training time required by DSB-CNN is significantly lower than the training time required by other CNN-based approaches for depth estimation, since our training dataset just consists of the user-scribbled keyframes. Nevertheless, this required time may prevent its use in an interactive realtime workflow. To circumvent this circumstance, the 3D-GDT can be used during the real time operation of the scribbling and visual feedback process as depicted in Fig. 1 along with a fast depth-image-based renderer (DIBR) which includes a simple extrapolation technique to fill occlusions when rendering stereo pairs [23]. The final rendering of high-end depth maps and stereo image pairs can be then performed offline in a render farm where the 3D-GDT and the renderer are replaced by the DSB-CNN and a high-quality offline renderer with more advanced inpainting solutions, respectively. Such workflows are currently integrated in many professional vfx pipelines. For confidential reasons, we cannot disclose further details of the underlying rendering processes.

Finally, we will provide the used datasets including ground truth depth maps for future research in this area.

7. ACKNOWLEDGMENT

The authors want to thank Benjamin Welle for his preliminary work on the implementation of the geodesic distance transform for interactive depth map creation.

8. REFERENCES

[1] X. Bai and G. Sapiro. Geodesic matting: A framework for fast interactive image and video segmentation and matting. *International Journal of Computer Vision*, 82(2):113–132, 2009.

[2] A. Criminisi, T. Sharp, C. Rother, and P. P'erez. Geodesic image and video editing. *ACM Trans. Graph.*, 29(5):134:1–134:15, Nov. 2010.

[3] D. Eigen, C. Puhrsch, and R. Fergus. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In *Advances of Neural Information Processing Systems (NIPS)*, pages 2366–2374, 2014.

[4] R. Fabbri, L. D. F. Costa, J. C. Torelli, and O. M. Bruno. 2d euclidean distance transform algorithms: A comparative survey. *ACM Comput. Surv.*, 40(1):2:1–2:44, Feb. 2008.

[5] R. Garg, B. G. Vijay Kumar, G. Carneiro, and I. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. *European Conference on Computer Vision*, 9912 LNCS:740–756, 2016.

[6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[7] M. Gerrits, B. De Decker, C. Ancuti, T. Haber, C. Ancuti, T. Mertens, and P. Bekaert. Stroke-based creation of depth maps. In *Proceedings - IEEE International Conference on Multimedia and Expo*, 2011.

[8] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised Monocular Depth Estimation with Left-Right Consistency. In *IEEE Confer-*

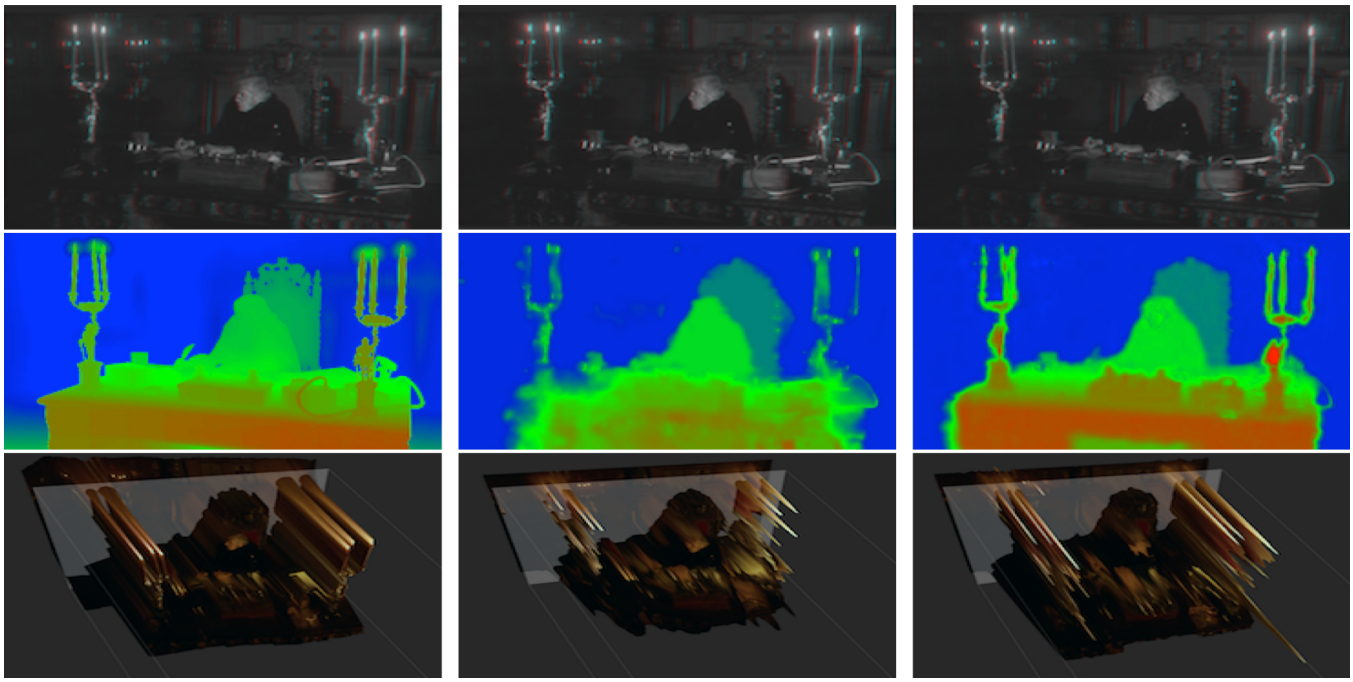

Fig. 8: Sample renderings of frame 75 of sequence 3. From left to right: ground truth, 3D-GDT, DSB-CNN. From top to bottom: anaglyph stereo image, disparity map (red: negative disparity, blue: positive disparity), 3D scene view.

ence on Computer Vision and Pattern Recognition, pages 270–279, 2017.

[9] M. Guttmann, L. Wolf, and D. Cohen-Or. Semi-automatic stereo extraction from video footage. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 136–142, 2009.

[10] J. L. Herrera, C. R. del Blanco, and N. Garca. Automatic depth extraction from 2d images using a cluster-based learning framework. *IEEE Transactions on Image Processing*, 27(7):3288–3299, July 2018.

[11] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *ACM SIGGRAPH*, page 689, 2004.

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

[13] B. Li, C. Shen, Y. Dai, A. Van Den Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 07-12-June, pages 1119–1127, 2015.

[14] J. Liao, S. Shen, and E. Eisemann. Depth annotations: Designing depth of a single image for depth-based effects. *Computers & Graphics*, 2017.

[15] J. Liao, S. Shen, and E. Eisemann. Depth Map Design and Depth-based Effects With a Single Image. In *Graphics Interface Conference*, pages 57–63, 2017.

[16] M. Liao, J. Gao, R. Yang, and M. Gong. Video stereolization: Combining motion analysis with user interaction. *IEEE Transactions on Visualization and Computer Graphics*, 18(7):1079–1088, 2012.

[17] Y. H. Lin, M. H. Tsai, and J. L. Wu. Depth sculpturing for 2D paintings: A progressive depth map completion framework. *Journal of Visual Communication and Image Representation*, 25(4):670–678, 2014.

[18] F. Liu, C. Shen, and G. Lin. Deep Convolutional Neural Fields for Depth Estimation from a Single Image. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5162–5170, 2015.

[19] A. Lopez, E. Garces, and D. Gutierrez. Depth from a single image through user interaction. *Spanish Computer Graphics Conference*, pages 1–10, 2014.

[20] B. Mendiburu. *3D Movie Making: Stereoscopic Digital Cinema from Script to Screen*. Focal Press/Elsevier, 2009.

[21] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[22] A. Rosenfeld and J. L. Pfaltz. Sequential operations in digital picture processing. *J. ACM*, 13(4):471–494, Oct. 1966.

[23] A. Smolic, P. Kauff, S. Knorr, A. Hornung, M. Kunter, M. Muller, and M. Lang. Three-dimensional video postproduction and processing. *Proceedings of the IEEE*, 99(4):607–625, April 2011.

[24] D. Sýkora, J. Dingliana, and S. Collins. LazyBrush: Flexible painting tool for hand-drawn cartoons. *Computer Graphics Forum*, 28(2):599–608, 2009.

[25] O. Wang, M. Lang, M. Frei, A. Hornung, A. Smolic, and M. Gross. StereoBrush : Interactive 2D to 3D Conversion Using Discontinuous Warps. In *Symposium on Sketch-Based Interfaces and Modeling*, 2011.

[26] B. Welle. Scribble-based depth map creation. Master's thesis, supervised by S. Knorr and T. Sikora, Technische Universität Berlin, Nov. 2012.

[27] J. Xie, R. Girshick, and A. Farhadi. Deep3D: Fully automatic 2D-to-3D video conversion with deep convolutional neural networks. In *European Conference on Computer Vision*, volume 9908 LNCS, pages 842–857, 2016.

[28] L. Yatziv and G. Sapiro. Fast image and video colorization using chrominance blending. *IEEE Transactions on Image Processing*, 15(5):1120–1129, May 2006.

[29] L. Zhang, C. Vazquez, and S. Knorr. 3d-tv content creation: Automatic 2d-to-3d video conversion. *IEEE Transactions on Broadcasting*, 57(2):372–383, June 2011.