MOVIE DIMENSIONALIZATION VIA SPARSE USER ANNOTATIONS

M. Becker, M. Baron, D. Kondermann

Heidelberg Collaboratory for Image Processing
University of Heidelberg, Germany

M. Bußler, V. Helzle

Animationsinstitut
Filmakademie Ludwigsburg, Germany

ABSTRACT

We present a workflow to semi-automatically create depth maps for monocular movie footage. Artists annotate relevant depth discontinuities in a single keyframe. Depth edges are then learned and predicted for the whole shot. We use structure from motion where possible for sparse depth cues, while the artist optionally provides scribbles to improve the intended visual effect. Finally, all three sources of information are combined via variational inpainting scheme.

As the outcome of our method is artistic and cannot be evaluated quantitively, we apply our method to a current movie production, showing good results on different scenes. We further evaluate the depth edge localization compared to the "ground truth" provided by artists. To enable experimentation with our approach, we offer our source code.

1. INTRODUCTION

Plausible reconstructed depths are indispensable for 2D-3D movie conversion. The creative industry imposes strong quality requirements on such depth maps which current fully automated pipelines do not produce. These typically fail in scenes containing occlusions, small moving particles, strong specular highlights or translucent objects. Therefore, user intervention is additionally needed to reliably convert 2D footage.

Finally, changes in depth map are often made manually to alter the depth perception to support a specific dramatic composition.

Costs and overall processing time influence the competitiveness of companies dealing with high quality conversions. Today, most companies rely on expensive work flows which require lots of man power (e.g. about 400 artists converted Titanic 3D[1]). Tools targeted at aiding the conversion process therefore should aim to reduce the overall processing time while producing results of comparable quality.

In this paper we propose a pipeline based on learning depth discontinuities, structure from motion and variational depth inpainting. Unlike more commonly used segmentation based methods, the advantage of our approach is that the outcome of our inpainting process directly provides smooth depth maps with sharp edges *only at artistically relevant* depth discontinuities. We do not have to take care of segment boundaries which lie at continuous depth junctions and lead to depth offsets require further user interactions.

1.1. Related Work

A vast body of literature exists to fully automatically address discontinuity in optical flow[16], stereo estimation[7], (video or super-pixel) segmentation[13, 3, 9], matting[6], and virtually any image processing or computer vision approach producing dense results. One of the predominant approaches is the use of total variation regularizers. In this paper, we introduce the idea of semi-automatic annotation of edges, which can be generalized to all of these methods to improve their accuracy.

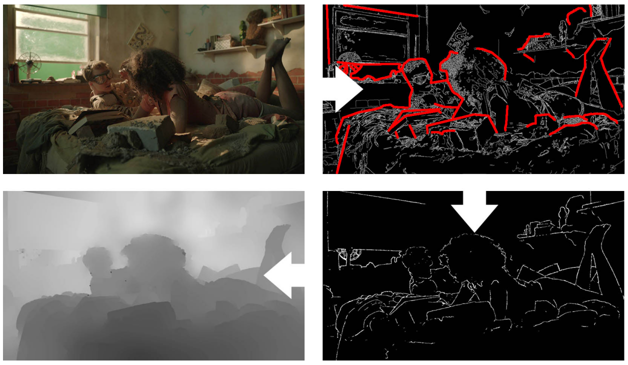

Figure 1. Depth discontinuities are labeled, learned by a random forest and afterwards predicted for several subsequent frames. We use those discontinuity edges as interpolation regularization. Thus, depth maps can be interpolated from sparse depth information. As result, our depth maps are smooth with sharp edges only at discontinuities.

More specific to the problem of movie dimensionalization, several related ideas are discussed in the following.

Tools such as VisualSFM[15] and Phototourism[11] compute structure from motion. Multiview stereo approaches are used to densify sparse motion reconstructions(cf. e.g. [2]). Saxena et al. [10] rely on learning to fully automatically assign depths to single still images. A similar goal was achieved for videos by Karsch et al. [8].

Yet, real movies show difficult conditions as e.g. independently moving objects with motion blur. Our experience with such image sequences gave us the insight that fully automatic approaches seldomly achieve results in a quality required by the creative industry. Exceptions are cases of highly constrained and carefully set up scenes as for example the *Trinity* scene in *The Matrix*[2](1999).

Directly related to our approach is the work of Guttmann et al. [4], who are among the first authors to address dimensionalization. They utilize user scribbles for semi-automatic stereo extraction. A more recent tool is *Depth Director* by Ward et al. [14]. The authors propose a segmentation based conversion approach, also utilizing user scribbles to process single objects. As a next step, sparse depth cues are assigned to these segments. Our

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

[2] http://www.matrixeyewear.com/blog/breaking-down-the-special-effects-of-the-matrix

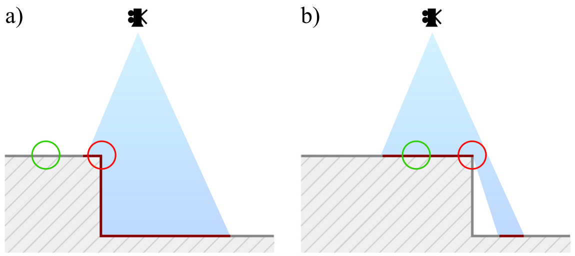

Figure 2. Depth edges highlighted by red circles. While the kink in Figure a) is a C^0 discontinuity, we basically utilize C^{-1} edges such as in b) which forces a gap between depth values. Green circles induce locations for putative texture only edges.

approach differs in that we focus on depth edge annotation and subsequent variational depth interpolation based on *both* SfM and user scribbles.

2. OUR CONTRIBUTION

We choose a learning based approach to predict depth edges. Then, depth cues are calculated from motion. Additional user scribbles can be assigned, if depth cues do not supply sufficient information (e.g. scenes without motion or unstructured objects). Finally, depth cues and depth edges are used as input to an interpolation process to obtain dense and smooth depth maps.

As mentioned earlier, a tool should enable artists in creating results with as little interaction as possible in a small amount of time. We believe that supplying rough user scribbles to mainly annotate depth edges is easier and can be done faster than painting depth maps by hand. In the following we describe the individual models of our pipeline.

2.1. Depth Edges

We aim to use depth edges as boundary conditions for the the following inpainting process in section 2.3. In contrast to texture edges at which object depths do not change in general, depth edges depend on the objects shape and position in a scene. Figure 2 illustrates two different types of depth edges marked by red circles, possible texture-only edges could be located at the green circles. While the left C^0 edge appears as kink for the according view, the right C^{-1} edge induces a depth gap in its according projection. Our approach can treat both of these types. For our application we basically rely on this second C^{-1} edge type.

2.1.1. Annotating Depth Edges

Depth edges are labeled using our interactive framework. User annotations may be imprecise. In order to facilitate quick and easy annotations, we chose the canny edge detector to provide edge suggestions and take the intersection of canny edges and user labels to refine the result. The parameters of the edge detector were chosen such that a maximum amount of edges is found. Once all relevant depth edges are annotated in a single key frame, we propagate those labels to the remaining frames. Propagated edges have to be precise. It is not reasonable to use optical flow for this purpose since flow at discontinuities cannot be estimated in a quality needed for our application. Thus, we rely on a learning approach explained in the following section.

Edge Cue Descriptions	**Num**
Color cues	**32**
C1. Colors channels	6
C2. Neighbor colors next to putative edge	6
C3. Color Histograms	12
C4. Mean Color of neighbor segments	6
Edge cues	**79**
E1. Color gradient	4
E2. Histogram of gradients	5
E3. Eigen values	4
E4. Histogram of eigen values	20
E5. Ratio between eigen values	4
E6. Eigen values and ratios of neighbor pixels	32
E7. Hessian	3
E8. Diffusion tensor	3
E9. Bilateral filter	3
E10. Gradient of optical flow	1

Table 1. Features assigned to random forest. Most features are calculated on HSV channels, some for RGB aswell.

2.1.2. Learning Depth Edges

To learn depth edges we chose the random forest framework with sample stratification in each tree to account for class imbalance in the learning set. Beside depth edges we classify pixels as texture edges or as pixels not lying on an edge.

Our approach utilizes the fact that an image sequence of the same shot contains very similar frames. Therefore, instead of learning a model applicable to all kinds of scenes, we learn a model *per shot* based on the user annotations in a *single frame*. We choose features which can be categorized as color features or as edge features. Beside that, most features are determined for the color channels H, S and V separately. Also, most features are calculated for three different scale spaces[3]. Table 1 gives an overview of our feature selection.

Color features $C1, C2, C3, C4$ are the most specific features of our set since colors may change between scenes. While color feature $C1$ indicates possible color areas, the other three features $C1$, $C2$ and $C3$ contain information about color neighborhoods. We chose those neighbor pixel coordinates (for features $C2$, $C4$, $E6$) with respect to eigen vector directions (determined from structure tensor). Thus, neighbors of edge pixels should be located in and perpendicular to the edge directions.

The combination of color features and edge features enables reasonable predictions of different edge types. Color and brightness often changes at depth boundaries and so, color differences at edges can be used to classify them.

2.2. Depth Cues

We use structure from motion using *VisualSFM* [15] where possible to retrieve sparse depth information. To get uniformly distributed depths we replace SIFT features by optical flow correspondences. If footage is unusable for SfM purposes due to missing camera motion or not trackable rigid objects, depth has to be assigned by hand. User scribbles can be attached to key frames propagated to subsequent frames using optical flow.

[3] For different scale spaces we blur with $sigma = 1, 3, 9$

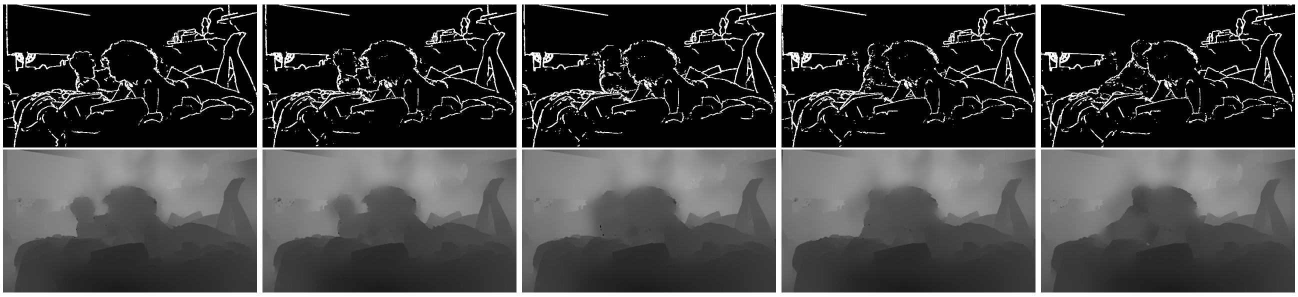

Figure 3. Converted sequence by depth label annotations for frame 1 and 25. The first row shows the predicted depth edges, the second row the corresponding depth maps. Sparse depth cues were taken from structure from motion. User scribbles were used to correct the depth of the moving person.

2.3. Depth Interpolation

In order to perform interpolation of the sparse depths $d(\vec{x}_1), ..., d(\vec{x}_K)$, respectively obtained at the support positions $\vec{x}_1, ..., \vec{x}_K$, we use a variational approach. That is, we minimize an energy functional globally over the whole image range Ω w.r.t. the sought dense depth map $\hat{d}(\vec{x})$. This energy functional consists of a data term

$$E_d = \sum_{k=1}^{K} p_k \cdot \Phi\left(\left(\hat{d}(\vec{x}_k) - d(\vec{x}_k)\right)^2\right), \qquad (1)$$

ensuring matching of sparse depths against the dense depth map and a λ-weighted prior term

$$E_p = \int_{\Omega} \Phi\left(\|\vec{\nabla}\hat{d}(\vec{x})\|_2^2\right)\, d\vec{x}, \qquad (2)$$

imposing a smoothness constraint on the resulting depth map. Here, we chose $\lambda = 400$, whereas Φ denotes a suitable chosen penalty function. To obtain sharp borders and smooth areas, we use the Charbonnier penalty function $\Phi(\Delta^2) = \sqrt{\Delta^2 + \varepsilon^2}$, parameterized with $\varepsilon = 0.01$, which represents a differentiable approximation of the l_1 norm [1].

In order to cope with occluding and uncovering regions of the depth map and obtain temporal consistency, we use an additional p_k as a weighting factor within the data term. As this factor expresses consistency over time, it depends on the old dense depth map $\hat{d}_{t-1}$ and the current sparse depths d_t. It is given by a relative probability measure

$$p_k = \exp\left(-\frac{\left(d_{t+1}(\vec{x}_k) - \hat{d}_t(\vec{x}_k)\right)^2}{2\sigma^2}\right), \qquad (3)$$

parameterized with the standard deviation σ, and motivated by the fact that covering and uncovering of regions implies depth changes, in general, whereas the depth change of a single region can be assumed as smooth over time. In our experiments, we chose the standard deviation $\sigma = 1000$. Proper implementation of the prior term denoted by eq. (2) depends on proper discretization of the partial derivatives of the ∇ operator. We discretize these partial derivatives by taking all pairwise differences between a currently considered central position $\vec{x}$ and all horizontal and vertical neighbor positions $\mathcal{N}(\vec{x})$. We handle image and depth boundaries the same way by excluding them from this neighborhood.

Optimization of the energy functional

$$E = E_d + \lambda \cdot E_p \qquad (4)$$

is being performed straightforward by first setting up the Euler Lagrange equations and then solving for the sought dense depth map.

3. EXPERIMENTS AND RESULTS

3.1. Depth Maps

Depth maps were created by labeling depth edges in frames $\{1, 25\}$ of a sequence with 25 frames. We make use of the camera motion. Thus, depth cues were calculated using *VisualSFM* [15] for camera tracking and correspondences were converted from state-of-the-art optical flow [12]. Since the body in the image center moved around, no reliable depth from motion can be retrieved for this area. Additional depth cues were assigned manually to overhaul this drawback.

3.2. Comparison to other depth contour annotations

We compare our depth maps to results achieved by the interpolation process using alternate regularization masks which are edges from superpixels and canny edges. All four depth maps created for this purpose use the same sparse depth cues. Figure 4 shows the results for those masks. Red squares highlight areas, where depth maps falsely show depth gaps induced by inappropriate edges. Blue squares show incomplete edges at depth boundaries. Results from superpixels, which could be compared to results by segmentation due to the consistent segment boundaries do not have to deal with color bleeding but smooth areas are only possible by merging segments which will lead to the loss of details. Arbitrary edges from edge detectors as the canny edges induce both shortcomings. Color bleeding will be received at incomplete edges and separated depth artifacts occur at closed edge loops. Our approach also shows bleeding in few regions but performs very good on smooth edges while widely retains the depth discontinuities.

4. CONCLUSION

We proposed a semi-automatic approach to learning depth edges via random forests. The concept can be used in many low-level vision algorithms such as rotoscoping, optical flow or stereo estimation.

The outcome of our workflow are detailed smooth and dense depth maps with sharp edges at discontinuities. The absence of stepping artifacts caused by other methods based e.g. on segmentation or optical flow creates a pleasant viewing experience while simultaneously empowering the artist to achieve the intended emotional response of the viewer via visual effects.

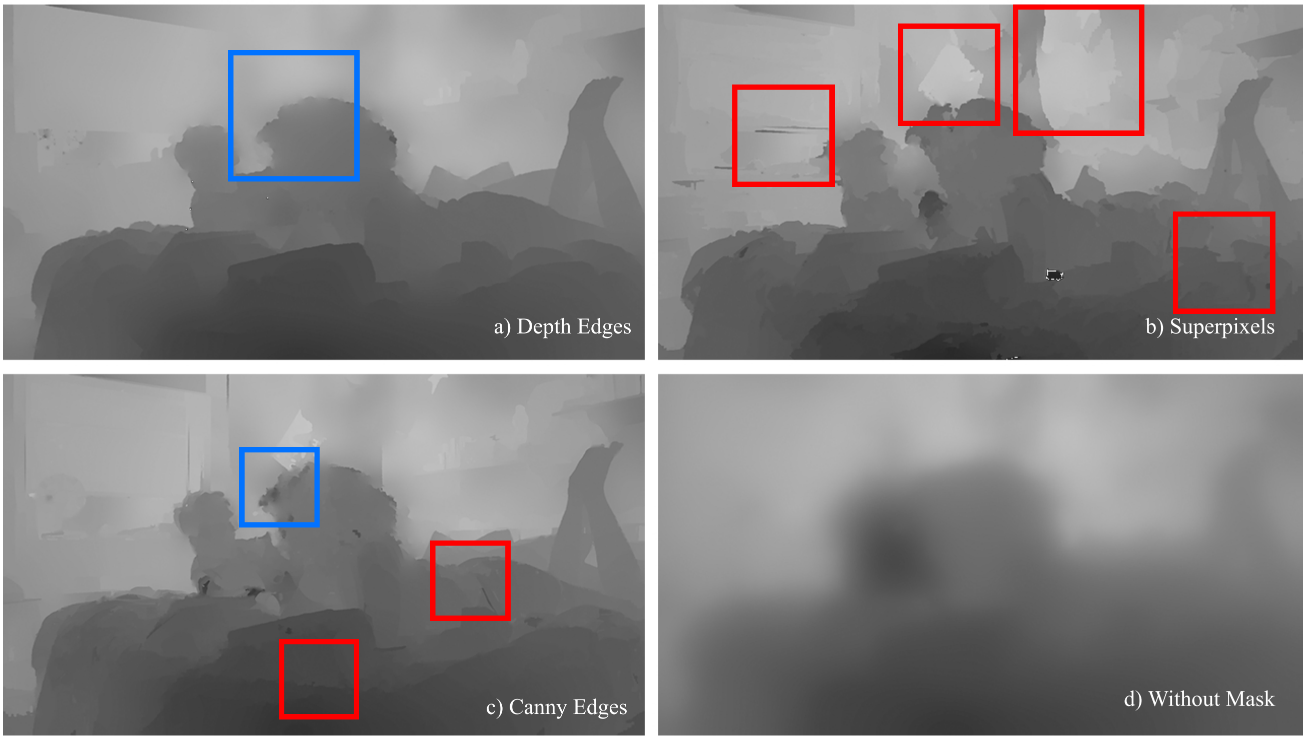

Figure 4. Comparison between different regularization masks. a) Our approach has to deal with partly incomplete edges which leads to color bleeding. b) Results obtained by edges taken from superpixels or segmentation have clean edges but show depth artifacts for single segments. c) Results by canny edges show both drawbacks.

5. REFERENCES

[1] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *ICIP (2)*, pages 168–172, 1994.

[2] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.

[3] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2141–2148. IEEE, 2010.

[4] M. Guttmann, L. Wolf, and D. Cohen-Or. Semi-automatic stereo extraction from video footage. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 136–142, 2009.

[5] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[6] K. He, J. Sun, and X. Tang. Fast matting using large kernel matting laplacian matrices. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2165–2172. IEEE, 2010.

[7] H. Hirschmuller. Accurate and efficient stereo processing by semi-global matching and mutual information. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 807–814. IEEE, 2005.

[8] K. Karsch, C. Liu, and S. B. Kang. Depth extraction from video using non-parametric sampling. In *Computer Vision–ECCV 2012*, pages 775–788. Springer, 2012.

[9] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 309–314. ACM, 2004.

[10] A. Saxena, M. Sun, and A. Ng. Make3d: Learning 3d scene structure from a single still image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):824–840, 2009.

[11] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH Conference Proceedings*, pages 835–846, New York, NY, USA, 2006. ACM Press.

[12] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles, 2010.

[13] A. Vazquez-Reina, S. Avidan, H. Pfister, and E. Miller. Multiple hypothesis video segmentation from superpixel flows. In *Computer Vision–ECCV 2010*, pages 268–281. Springer, 2010.

[14] B. Ward, S. B. Kang, and E. P. Bennett. Depth director: A system for adding depth to movies. *IEEE Computer Graphics and Applications*, 31(1):36–48, 2011.

[15] C. Wu. Visualsfm: A visual structure from motion system, 2011.

[16] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(9):1744–1757, 2012.