Improving depth maps with limited user input

Patrick Vandewalle, René Klein Gunnewiek and Chris Varekamp

Philips Research Eindhoven, High Tech Campus, 5656AE Eindhoven, The Netherlands

ABSTRACT

A vastly growing number of productions from the entertainment industry are aiming at 3D movie theaters. These productions use a two-view format, primarily intended for eye-wear assisted viewing in a well defined environment. To get this 3D content into the home environment, where a large variety of 3D viewing conditions exists (e.g. different display sizes, display types, viewing distances), we need a flexible 3D format that can adjust the depth effect. This can be provided by the image plus depth format, in which a video frame is enriched with depth information for all pixels in the video frame. This format can be extended with additional layers, such as an occlusion layer or a transparency layer. The occlusion layer contains information on the data that is behind objects, and is also referred to as occluded video. The transparency layer, on the other hand, contains information on the opacity of the foreground layer. This allows rendering of semi-transparencies such as haze, smoke, windows, etc., as well as transitions from foreground to background. These additional layers are only beneficial if the quality of the depth information is high. High quality depth information can currently only be achieved with user assistance. In this paper, we discuss an interactive method for depth map enhancement that allows adjustments during the propagation over time. Furthermore, we will elaborate on the automatic generation of the transparency layer, using the depth maps generated with an interactive depth map generation tool.

Keywords: image plus depth, semi-automatic depth generation, matting

1. INTRODUCTION

To enable stereoscopic depth perception, several technologies have been suggested that allow different images to be provided to each eye. Most of these use special glasses, such as polarized or shutter glasses. Glasses-based solutions exist in for example the 3D Ready DLP display by Samsung and the 3D Xpol/Arisawa LCD display by Hyundai. Auto-stereoscopic displays, however, take a different approach: a different image is provided to each eye without the need of special eyewear. The auto-stereoscopic displays offer their different views through the use of a sheet of lenticular lenses or barriers in front of the screen. Examples of auto-stereoscopic displays are the Alioscopy[1] and Tridelity[2] lenticular displays, and the Newsight barrier displays.[3]

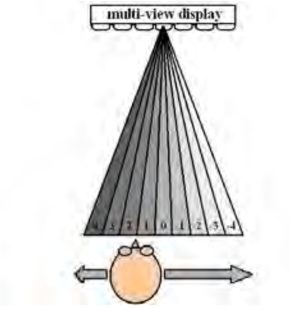


Figure 1. Spatial distribution of views on a multi-view display.

Multiview auto-stereoscopic displays can show a high number of different views (e.g. 9) at a given moment in time. This approach gives a viewer not only a stereoscopic effect, it also allows the viewer to experience horizontal motion parallax, by moving his head horizontally. Figure 1 shows how the different views are spatially distributed.

Current multiview displays strongly differ in their number of views and pixels and in their depth reproduction capabilities. Furthermore, the amount of depth perceived on stereoscopic displays also depends on the viewing distance and the display size.[4] Therefore, a display independent interface format that enables adjustment of the depth effect is preferred.

Stereoscopic Displays and Applications XXI, edited by Andrew J. Woods, Nicolas S. Holliman,
Neil A. Dodgson, Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 7524, 752414
© 2010 SPIE-IS&T · CCC code: 0277-786X/10/$18 · doi: 10.1117/12.840236

The preferred *display independent* input format for home 3D displays is the *image plus depth* format,[5–7] which has been standardised in MPEG-C part 3. Being independent of specific display properties, such as the number of views, view mapping on the pixel grid, etc., this interface format allows optimal multiview visualisation of content from many different sources, while maintaining interoperability between display types. To be more specific, the image plus depth format is an extension of the conventional video format. For each sample point in the input images, it specifies not only information about the color, but depth related data as well.

However, the majority of (legacy) video content is 2D: video captured from a single viewpoint. Furthermore, a growing number of new 3D productions from the entertainment industry are aiming at 3D movie theaters that use a stereo format, primarily intended for eye-wear assisted viewing. This content does not have associated depth related data. Thus, in order to be display independent and flexible, the content should be converted to the image plus depth format. In other words, each pixel should be annotated with a depth value.

The estimation of depth data from existing video content has been researched extensively. Various algorithms have been presented that generate depth data for existing video material either from multiview/stereoscopic content[8,9] or from 2D video content.[10,11] Most of these algorithms are based on automatic generation of depth maps. There are, however, also many tools and/or plug-ins available that are user assisted and generate high quality depth maps.[12–14]

The image plus depth format can be extended with additional layers to improve the quality of the rendered views. The image plus depth format with its optional additional layers is also referred to as the Layered Depth Video (LDV) format. Such additional layers could be an occlusion and/or a transparency layer. An occlusion layer resolves pixels that are not visible in the center view but will become visible from a different view point. Such an occlusion layer is typically also enriched with depth information. Methods for estimating an occlusion layer have been described by Barenbrug[15] and Klein Gunnewiek et al.[16] A transparency layer or alpha map[17] on the other hand, contains information on opacity. This enables rendering of semi-transparencies and transitions at object boundaries. An implementation of such a transparency layer/alpha map in the image plus depth framework is discussed by Barenbrug.[18] In this paper, we discuss the automatic generation of a transparency layer.

The described additional layers are only beneficial if the quality of the depth maps is sufficiently high. High quality depth information can currently only be obtained with user interaction. Therefore, we also present a method for depth map enhancement with limited user input.

2. RELATED WORK

Automated depth/disparity estimation from mono, stereo or multiview input video has been widely investigated. If the temporal axis is not exploited, automatic depth estimation from mono input video can only be based on heuristic cues and/or statistical models. Such heuristic cues could for example be slant, gravity (objects at the bottom of the image are typically in front), focus/texture (assuming the objects in front are most in focus), etc. These initial algorithms can be extended with camera viewpoint estimation, 3D scene geometry estimation and local object detection. Model based approaches incorporating these extension are presented by Hoiem et al.[19] and Saxena et al.[20] For monoscopic video, the temporal axis will also be used to extract depth information from object and camera motion. This helps as well in improving temporal stability.

Having an input video captured from more than one view point will boost the performance of the depth/disparity estimation process. In the past decade, the quality of these depth/disparity estimation algorithms for stereo and multiview input video has been strongly improved. The strongest improvements have been achieved by incorporating multiple information sources. An overview of the quality of the various depth estimators for stereo input is presented by Scharstein and Szeliski[8] and their associated website[*]. This website hosts a benchmark on multiview disparity estimation algorithms (using more than two cameras) as well.

However, the accuracy of such automatic algorithms is still not sufficiently high for a large variety of video data. The main cause for the lesser performance of depth/disparity estimators is the lack of sufficient discriminative means in homogeneous areas and between objects with similar colors. In homogeneous areas, a feature can be matched with many similar features within the same neighborhood. In this way, a wide range of disparity values would give a good match. Reflections could become the dominant discriminative feature, which will result in possibly erroneous depth values.

[*] http://vision.middlebury.edu

Similarly, if two objects at different depths, but with similar color, overlap in the video, it is difficult to estimate the correct object boundaries. At these depth transitions depth/disparity estimation will become error prone. Furthermore, leakage between the foreground and background depth values will typically appear at object boundaries, also referred to as blending. This leakage will make precise estimation and rendering of such a depth transition difficult.

User interaction can be applied to steer the depth/disparity estimation process in order to make the next quality step in disparity estimation. User input would typically be applied to the areas where issues with the disparity estimation result in poorly rendered views. Typically, there is a limited number of problem areas, such that user input and therefore also the additional effort will be limited. This process is referred to as supervised depth/disparity estimation. Srivastava et al. have developed a supervised depth estimation method for monoscopic still images by extending the learning-based algorithm by Saxena et al.[20] to include interactivity in 3D estimation.[21] After an automatic 3D reconstruction, a user can first roughly indicate the foreground object(s) if present. Next, he can place scribbles to indicate regions of the background that belong to the same plane. These constraints are then integrated in a Markov Random Field approach. Russell and Torralba have developed LabelMe3D: a learning-based approach to infer 3D models from rough contours and semantic labels given by a user.[22]

VideoTrace is a system that generates 3D models from a single video using structure from motion analysis.[23] The shape of an object can be traced by the user to produce polygons. VideoTrace does not require pixel-accurate line input since it fits the input curves to local strong super-pixel boundaries of a segmentation that is computed in advance. Efficient methods for semi-automated depth estimation from monoscopic input use manually annotated key-frames and propagate depth between those key-frames.[24] In this paper, we target view-interpolation using monoscopic video as input.

High quality depth maps create the possibility to generate useful transparency and occlusion data. The generation of high quality occlusion data is out of the scope of this paper and has been presented on earlier occasions.[15,16] In this paper, we will discuss automatic generation of transparencies. One of the first commercial products capable of successfully extracting an alpha map from a natural image was Knockout. Around that time, a paper was presented by Ruzon and Tomasi,[25] in which techniques are presented that were used by later approaches as well. In particular, it introduced the concept of a trimap and the idea of using statistical modeling to estimate alpha values. A trimap divides the image into three regions: a part that is definitely foreground, a part that is definitely background, and an uncertain part in which the transition is located. Local trimaps can easily be deduced from high quality depth maps. Bayesian matting methods provide in general good results and are still highly regarded and frequently used today. Popular algorithms are the work described by Zitnick et al.[26] and improvements thereon by Sindeyev et al.[27]

Trimaps can also be derived from a segmentation map. Graph cut approaches can segment an image into foreground and background objects using just a few hand picked samples of the image. GrabCut[28] uses an iterative graph cut scheme to segment the image. Using this segmentation, it is trivial to construct a trimap onto which we can apply other matting algorithms. Hasinoff[17] presented an algorithm that is based on GrabCut but is capable of directly extracting the trimap without relying on an intermediate segmentation. Likewise, Levin et al.[29] presented an algorithm that implicitly performs segmentation while trying to determine alpha mattes based on spectral matting. These algorithms are relatively complex and should therefore be performed on the acquisition side. Since we can obtain high quality depth maps by means of user interaction, we can easily construct a trimap needed for the transparency estimation which makes the transparency map estimation easier. Therefore we present a low complexity transparency estimation algorithm that can in principle be performed at the display side. We will benchmark our algorithm to high complexity transparency estimation algorithms. Before going into details of the transparency estimation, we will first give a description of the semi-automatic depth map improvement algorithm.

3. ALGORITHMS

3.1 Supervised depth map improvement

Let us assume we have (part of) a video containing a single camera shot, for which we want to estimate depth maps. First, we compute an initial depth map for the entire sequence using a state-of-the-art depth/disparity estimator. A few such algorithms were discussed in the previous section. We will not go into the details of such an estimator here, but concentrate on the improvement made using manual interaction.

Even when using a state-of-the-art algorithm to create depth maps for a particular shot, there will typically still be areas where problems occur. We apply user interaction to address these problems. As indicated in the previous section, one

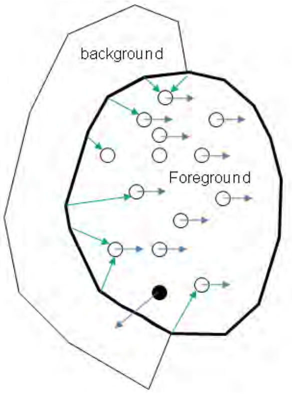


Figure 2. Two filter regions are specified: the local foreground object and the local background. Multiple feature points are placed on the foreground object (close to the occluding boundary). The operator can deactivate (closed circles) an indicated (active) feature point (open circles) if it contains an erroneous motion vector. The estimated motion vectors are indicated with blue arrows. A correspondence is established between each contour point and the nearest reliable feature point (some of these are indicated in the figure using green arrows).

of the common types of errors occur at depth transitions in low contrast areas. Due to the low contrast, the object edge over which the depth changes cannot be precisely located, such that the transition is typically smeared out (or wrongly positioned). We therefore allow a user to indicate the boundaries of an object that was badly segmented. We also give the opportunity to indicate an adjacent region outside the segmented object (local background). The algorithm will then smoothen the depth values in both regions, without smoothing over the indicated object edge. This creates a sharp object transition at the correct location, which is needed for good transparency estimation and high quality rendering.

Typically, we would want to indicate boundaries only in a key-frame of a video shot. The user can therefore make corrections in a single frame, which are then propagated throughout the sequence. This is advantageous, both to minimize the amount of user input required, and to optimize temporal stability. If a user would manually indicate the object contour in each frame, small variations in the selection would result in a disturbing flickering effect in the rendered views.

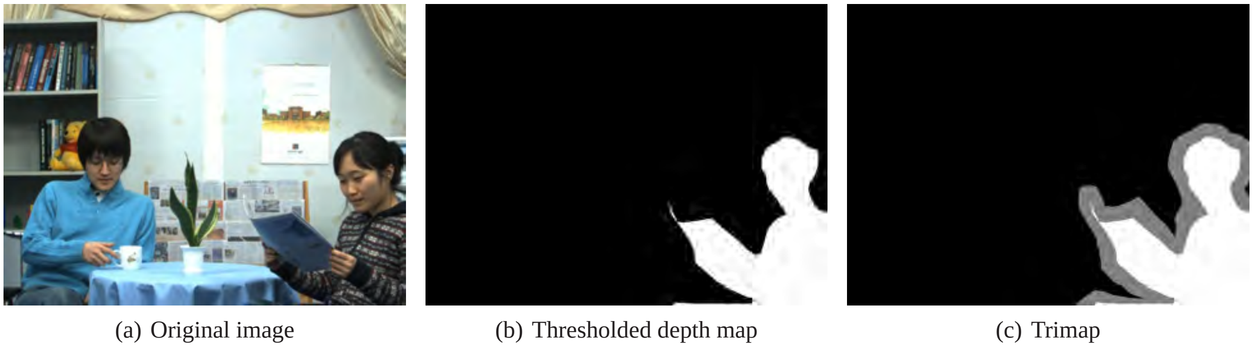
In order to achieve this, we need to propagate the segmentation made by the user through the shot. However, as already indicated, this contour is typically drawn across regions with low contrast, and is therefore difficult to track. As it indicates the border between a foreground object and the background, there is also a potential risk of the contour following the motion of the background instead of the foreground.

Existing video post-production tools (e.g. from Adobe) allow an operator to interactively create time-consistent tracks of feature points. An object contour can then be linked to these tracking data such that it follows the object outline through the video. This technique is generally referred to as rotoscoping.[30] We propose a slightly different method: depth correction using interactive feature tracks that steer the position of the contours. At a key-frame location, the operator indicates the foreground and background using two contours as described above. Multiple feature points are placed on the foreground object as close as possible to the occluding boundary (see Figure 2). All feature points are tracked over several frames, and contour points are linked to the closest feature point. They get the same motion as the linked feature point. In each frame the operator can deactivate a given feature point (closed circle in Figure 2), which is then ignored for further motion compensation of the two regions. Correspondences between contour points and the closest feature points are automatically re-established in each frame and depend on the features that are still active. By specifying many feature points on the foreground object, non-rigid motion can also be (partially) taken into account. Different parts of the contour are tracked differently, depending on which feature point is closest, allowing for some deformation of the contour. This method allows for robust tracking of the object throughout the video sequence. In each frame, the smoothing operation is automatically applied to the depth map as indicated above.

In summary, we require a user to segment an object with incorrect depth values in a key-frame of a video sequence. Next, we ask for a few salient feature points on the foreground object. These are used to track the object in the entire sequence. Depth values are then smoothened within the object and outside, creating a crisp transition at the object edge. As a result, we get accurate depth maps, for which transparency maps can be estimated to make the transitions look more natural in the rendered views.

3.2 Alpha map estimation

We will now describe an algorithm to estimate the alpha map based on a cross-bilateral filter approach.[31,32] First, a local *trimap* is generated from the depth map. This local trimap is generally defined only near large depth transitions. It divides the image in three regions: a background layer ($\alpha = 0.0$), a foreground layer ($\alpha = 1.0$), and an undefined layer (α unknown). An example of a trimap and its associated depth and texture image is given in Figure 3. The undefined layer is located around a depth transition that can be either manually specified or automatically determined. The width of the undefined area is typically predetermined. If the camera parameters are present, it could be derived from these parameters. The undefined layer is obtained through morphological operations (erosion and dilation) around the depth transition.


(a) Original image

(b) Thresholded depth map

(c) Trimap

Figure 3. Frame from the Newspaper sequence (a) with thresholded depth map (b) and the resulting trimap (c). An uncertain region (gray) is created between the foreground (white) and background (black) regions using morphological operations on the thresholded depth map.

The goal of the cross-bilateral filter in our approach is to estimate the precise alpha values in the undefined region, using the knowledge about the foreground and background layer.

A cross-bilateral filter weighs the (known) samples of the initial alpha map $A^{(0)}$ (derived from the trimap) using spatial distance and color difference (distance in color space) at a corresponding location in the corresponding texture image. In this way, contributions are only accepted from nearby pixels with similar color as the corresponding pixel in the texture image. Since different objects typically have different colors, the only samples that will be considered are those that belong to the same object and thus have related alpha values.

Thus, the cross bilateral filter matting operation for a pixel $p = (x, y)$ of a preliminary alpha map $A^{(0)}$ created from a color image C is given by:

$$A_p = \frac{1}{N} \sum_{p' \in \Omega} f(\|p - p'\|_2) g(\|C_p - C_{p'}\|_2) A^{(0)}_{p'} \ , \tag{1}$$

with

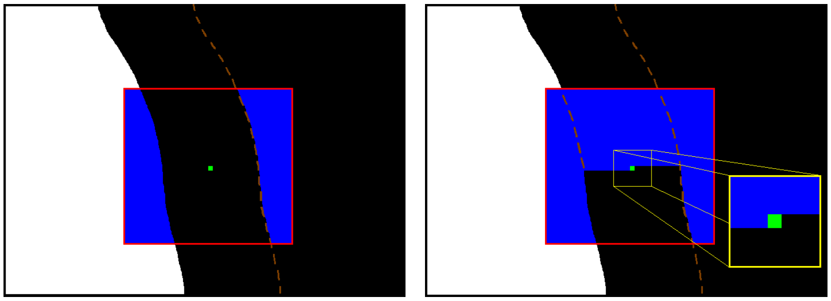
$$N = \sum_{p' \in \Omega} f(\|p - p'\|_2) g(\|C_p - C_{p'}\|_2) \ . \tag{2}$$

The functions f and g are both Gaussian kernels G_σ of the form:

$$G_\sigma(z) = \frac{1}{\sqrt{2\pi}\sigma} \, e^{-\frac{z^2}{2\sigma^2}} \ , \tag{3}$$

where σ is the standard deviation of the Gaussian distribution and Ω the window around the pixel to be filtered. The window should be sufficiently large that it easily overspans the unknown area.

There are two possible ways to create a filter window for a particular pixel. The first alternative is to limit the filter window only to foreground and background pixels in the neighborhood that were present from the beginning. The second alternative is to adaptively include the border pixels that have already been filtered (see Figure 4).



(a) Only take initial foreground and background pixels into account.

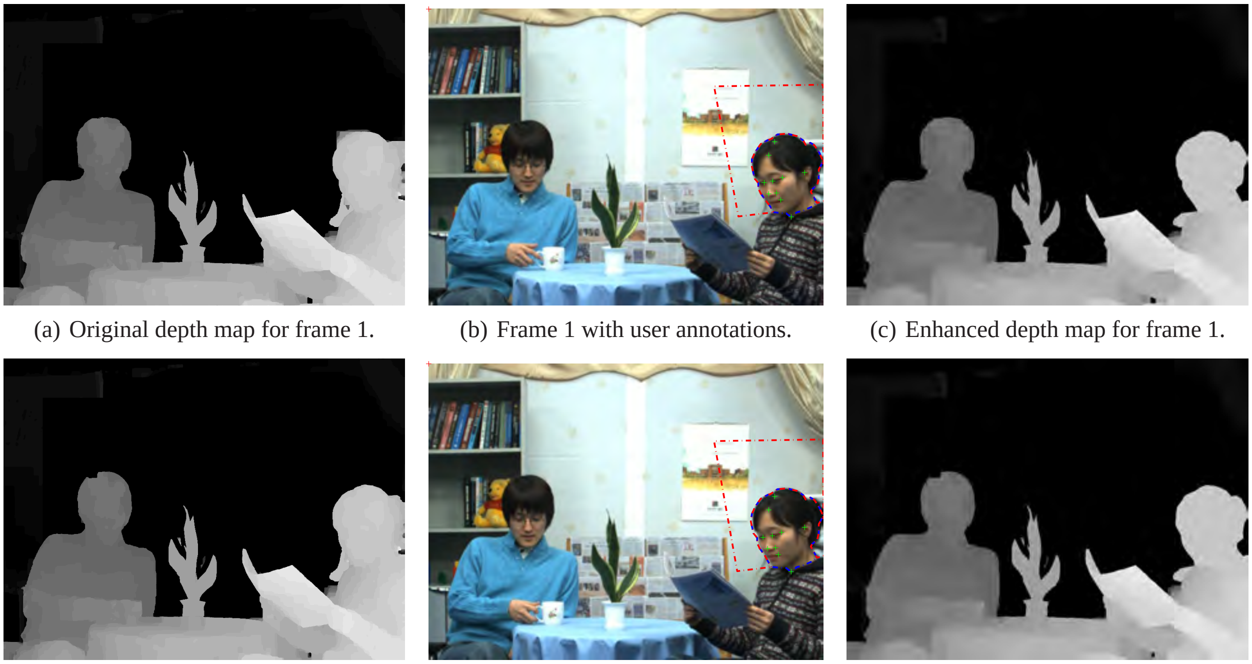
(b) Include also pixels that have already been filtered.

Figure 4. Sketch of different approaches to determine the filter window Ω. The pixel currently subject to filtering is colored in green and all pixels in the neighborhood (red rectangle) that are part of the filter window are colored in blue. In Figure (a) all border pixels are excluded from the filter window, while in Figure (b) the border pixels that have already been filtered are also included.

4. DISCUSSION & RESULTS

4.1 Depth map enhancement

In Figure 5, we give an example of the depth map enhancement algorithm. The contour of the woman's head is drawn in a key-frame, along with an outside local background region (see Figure 5(b)). A number of salient features on the head are then indicated to permit precise tracking. The enhancements to the depth map are shown in Figure 5(c). These corrections are then propagated through the sequence, and the result in frame 10 can be seen in Figure 5(d)-(f). The contour has been tracked quite accurately and in a temporally stable manner. Some inaccuracies to the left of the head are still visible in frame 10 due to small tracking errors.



(a) Original depth map for frame 1.

(b) Frame 1 with user annotations.

(c) Enhanced depth map for frame 1.

(d) Original depth map for frame 10.

(e) Frame 10 with propagated annotations.

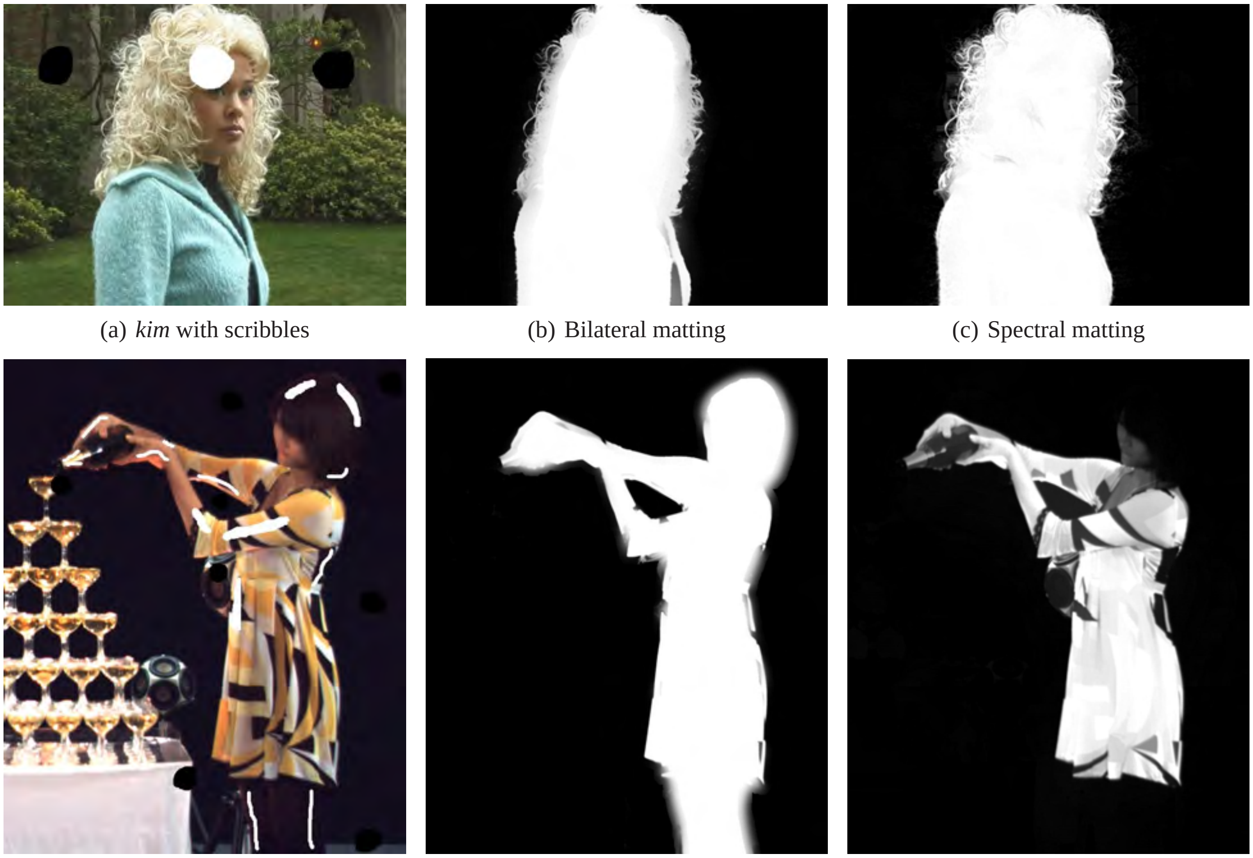
(f) Enhanced depth map for frame 10.

Figure 5. Depth map enhancement using a small amount of user interaction. The user has drawn the contour of the head in frame 1 as well as some feature points. These are automatically propagated and depth maps are filtered accordingly.

4.2 Alpha map estimation

A comparison between the spectral matting algorithm by Levin et al.[29] and the bilateral filter matting is depicted in Figure 6. The spectral matting algorithm performs the segmentation and matting in one processing step. This segmentation step is steered by scribbles, which helps the segmentation for these sequences significantly. There are quite some parameter settings for the spectral matting algorithm. Empirically, we have determined that two values are influencing the results significantly. The presented results have been obtained with 200 eigenvectors ($eigs\_num = 200$) and 10 clusters ($nclust = 10$) for the Kim image[33] and 50 eigenvectors ($eigs\_num = 50$) and 14 clusters ($nclust = 14$) for the Champagne Tower frame. The bilateral matting does not use the indicated scribbles, but starts from a trimap derived from an available high quality depth map. The results using spectral matting are somewhat better than the results obtained with the bilateral approach. However, when processing multiple frames of a sequence, the temporal stability is sometimes an issue. For the *Champagne Tower* sequence, the quality of spectral matting does not outperform the bilateral approach. The spectral matting algorithm has difficulties with the low textured areas of the dress of the girl.

Furthermore, the processing time required for the spectral matting is orders of magnitude larger than for the bilateral approach. While a computationally intensive method like spectral matting would have to be applied at the content creation side, bilateral matting can be applied either at the content creation side or at the receiver side. A bilateral matting method could be implemented for example in a TV set, such that no transparency map has to be transmitted.



(a) *kim* with scribbles (b) Bilateral matting (c) Spectral matting

(d) *Champagne Tower* with scribbles (e) Bilateral matting (f) Spectral matting

Figure 6. Comparison between spectral matting and matting using bilateral filter. The scribbles are only used for the spectral matting case. The bilateral filter uses the input depth map. Top row: Kim image. For the spectral matting the parameters were set to $eigs\_num = 200$, $nclust = 10$. Bottom row: Selection of frame from Champagne Tower sequence. For the spectral matting, the parameters were set to $eigs\_num = 50$, $nclust = 14$.

5. CONCLUSIONS

In this paper, we have presented a depth map enhancement method for creating depth maps with a higher quality using some manual user input. Quite some algorithms exist that can perform depth estimation. However, most of the time, the quality is not sufficient, typically in a limited number of areas. With the proposed supervised correction process, we can improve the quality of the depth maps significantly with a typically limited input from the operator. High quality depth maps allow for an automatic transparency map generation at the display side. The bilateral method for estimating a transparency layer is an efficient and cost-effective method. The quality of the transparency layer using a bilateral filter comes close to the quality of the transparency maps of state-of-the-art matting algorithms, at a fraction of the computational cost.

6. ACKNOWLEDGEMENTS

We would like to thank Markus Franke (CAU, Kiel) for his work during his internship at Philips Research. He implemented the alpha estimation algorithm and performed the comparison to spectral matting. We are also grateful to Prof. Yung-Yu Chuang at National Taiwan University for sharing the Kim image with us, and to Prof. Tanimoto at Tanimoto Lab, Nagoya University and Prof. Ho at Gwangju Institute of Science and Technology for the Champagne Tower and Newspaper sequences, respectively. These two sequences are provided as MPEG test sequences. This work is partially supported by the European Commission in the 3D4YOU project, ICT 215075.

REFERENCES

1. Alioscopy, “Lenticular displays.” http://www.alioscopy.com/.
2. Tridelity, “Lenticular displays.” http://www.tridelity.de/.
3. Newsight, “Barrier displays.” http://www.newsight.com/.
4. R. Klein Gunnewiek and P. Vandewalle, “How to display 3d content realistically,” in *Proc. Fifth International Workshop on Video Processing and Quality Metrics for Consumer Electronics*, January 2010.
5. C. Fehn, “Depth-image-based rendering (DIBR) compression and transmission for a new approach on 3D-TV ,” in *Proc. Stereoscopic Displays and Applications*, 2004.
6. B. Barenbrug, “3D throughout the video chain,” in *Proceedings of Int. Congress of Imaging Science*, pp. 366–369, 2006.
7. W. H. A. Bruls, C. Varekamp, R. Klein Gunnewiek, B. Barenbrug, and A. Bourge, “Enabling introduction of stereoscopic 3D video: compression standards, displays and content generation,” in *Proceedings of Int. Conference on Image Processing*, pp. 89–92, 2007.
8. D. Scharstein and R. Szeliski, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” *International Journal on Computer Vision* **47**, pp. 7–42, April-June 2002.
9. Q. Yang, L. Wang, R. Yang, H. Stewénius, and D. Nistér, “Stereo matching with color-weighted correlation, hierarchical belief propagation and occlusion handling,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, **2**, pp. 2347–2354, 2006.
10. A. Torralba and A. Oliva, “Depth estimation from image structure,” *IEEE Transactions on pattern analysis and machine intelligence* **24**, pp. 1226–1238, 2002.
11. M. Han and T. Kanade, “Multiple motion scene reconstruction with uncalibrated cameras,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**, pp. 884–894, 2003.
12. Autodesk, “Autodesk maya.” http://usa.autodesk.com/.
13. The Foundry, “NukeX.” http://www.thefoundry.co.uk/.
14. Adobe, “Adobe Photoshop CS4.” http://www.adobe.com/.
15. B. Barenbrug, R.-P. M. Berretty, and R. Klein Gunnewiek, “Robust image, depth, and occlusion generation from uncalibrated stereo,” *Proc. Stereoscopic displays and applications* **6803**, 2008.
16. R. Klein Gunnewiek, R.-P. M. Berretty, B. Barenbrug, and J. P. Magalhães, “Coherent spatial and temporal occlusion generation,” in *Proceedings of SPIE*, **7237**, January 2009.
17. S. Hasinoff, S. Kang, and R. Szeliski, “Boundary matting for view synthesis,” in *Computer Vision and Image Understanding*, **103 (1)**, pp. 22–32, 2006.
18. B. Barenbrug, “Multi-layer image-and-depth with transparency made practical,” *Proc. Stereoscopic Displays and Applications* , 2009.

19. D. Hoiem, A. Efros, and M. Hebert, "Putting objects in perspective," *International Journal of Computer Vision* **80**(1), 2008.
20. A. Saxena, M. Sung, and A. Ng, "Make3d: Learning 3D scene structure from a single still image," *IEEE Transactions on pattern analysis and machine intelligence* **30**, pp. 824–840, 2009.
21. S. Srivastava, A. Saxena, C. Theobalt, S. Thrun and A.Y. Ng, "i23 - rapid interactive 3D reconstruction from a single image," *Proceedings of Vision, Modeling and Visualization* , 2009.
22. B. C. Russell and A. Torralba, "Building a database of 3d scenes from user annotations," *Proceedings IEEE Conference on Computer Vision and Pattern Recognition* , pp. 2711–2718, 2009.
23. A. v. d. Hengel, A. Dick, T. Thormahlen, B. Ward, and P. H. Torr, "Videotrace: Rapid interactive scene modelling from video," *ACM Transactions on Graphics* **26**, 2007.
24. C. Varekamp and B. Barenbrug, "Improved depth propagation for 2D to 3D video conversion using key-frames," *4th European Conference on Visual Media Production (IETCVMP)* **23**, pp. 1–7, 2007.
25. M. A. Ruzon and C. Tomasi, "Alpha estimation in natural images," *IEEE Computer Vision and Pattern Recognition* , pp. 18–25, 2000.
26. C. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Transactions on Graphics* **23**, pp. 600–608, 2004.
27. M. Sindeyev, V. Konushin, and V. Vezhnevets, "Improvements of bayesian matting," *Graphicon* , 2007.
28. C. Rother, V. Kolmogorov, and A. Blake, ""GrabCut": interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics* **23**, pp. 309–314, 2004.
29. A. Levin, A. Rav-Acha, and D. Lischinski, "Spectral matting," *Proc. IEEE Transactions on Pattern Analysis & Machine Intelligence* **30**, pp. 1699–1712, 2008. http://www.vision.huji.ac.il/SpectralMatting/.
30. A. Agarwala, A. Hertzmann, D. Salesin, and S. Seitz, "Keyframe-based tracking for rotoscoping and animation," *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004)* , 2004.
31. C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. Sixth International Conference on Computer Vision (ICCV)*, p. 839, 1998.
32. S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," *International Journal of Computer Vision* **81**(1), pp. 24–52, 2009.
33. Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, and R. Szeliski, "Video matting of complex scenes," *ACM Transactions on Graphics (Special Issue of the SIGGRAPH 2002 Proceedings)* **21**(3), pp. 243–248, 2002.