Multiview Point Cloud Filtering for Spatiotemporal Consistency

Robert Skupin, Thilo Borgmann and Thomas Sikora

Communication Systems Group, Technische Universität Berlin, Berlin, Germany
skupin@mailbox.tu-berlin.de, {borgmann, sikora}@nue.tu-berlin.de

Keywords: Point Cloud Filtering, Multiview Resampling, Spatiotemporal Consistency.

Abstract: This work presents algorithms to resample and filter point cloud data reconstructed from multiple cameras and multiple time instants. In an initial resampling stage, a voxel or a surface mesh based approach resamples the point cloud data into a common sampling grid. Subsequently, the resampled data undergoes a filtering stage based on clustering to remove artifacts and achieve spatiotemporal consistency across cameras and time instants. The presented algorithms are evaluated in a view synthesis scenario. Results show that view synthesis with enhanced depth maps as produced by the algorithms leads to less artifacts than synthesis with the original source data.

1 INTRODUCTION

Stereoscopic video that allows for depth perception through two view points is already a mass market technology and means for acquisition, transport, storage and presentation are broadly available. Autostereoscopy is targeted towards a more immersive viewing experience with glasses-free display technology and an increased freedom of viewer position (Smolic et al., 2006). The necessary rich scene representations consist of camera or camera views from more than two view points. Out of this large data set, only two suitable camera views are visible to a spectator at a given time to achieve depth perception (Dodgson, 2005). Transmitting the necessary amount of camera views may challenge the available infrastructure, e.g. with respect to capacity. A solution may be the transmission of a limited set of camera views with their associated depth information, referred to as depth maps, to allow synthesis of additional camera views at the end device without sacrificing transmission capacity (Vetro et al., 2008).

However, the quality of synthesized views depends on the quality of the provided depth information, i.e. its accuracy and consistency, which may be degraded, e.g through compression or estimation errors (Merkle et al., 2009). In (Scharstein and Szeliski, 2002) and (Seitz et al., 2006), the authors give an extensive taxonomy and evaluation of techniques for multiview reconstruction algorithms that distinguishes global methods, e.g. graph-cuts based optimization algorithms as in (Vogiatzis et al., 2005) or (Starck and Hilton, 2005) and local methods such as depth map fusion (Merrell et al., 2007). Camera view point, image noise as well as occlusions, texture characteristics, motion or other influences challenge multiview reconstruction algorithms. As illustrated in Fig. 1 on the widely used ballet data set (Zitnick et al., 2004), estimated depth maps may suffer from object boundary artifacts across camera view points (a), movement artifacts at static areas over time (b) or artifacts at revealed occlusions from edge view points of the camera setup (c).

Another aspect that can make the representation and processing of multiview data cumbersome is the

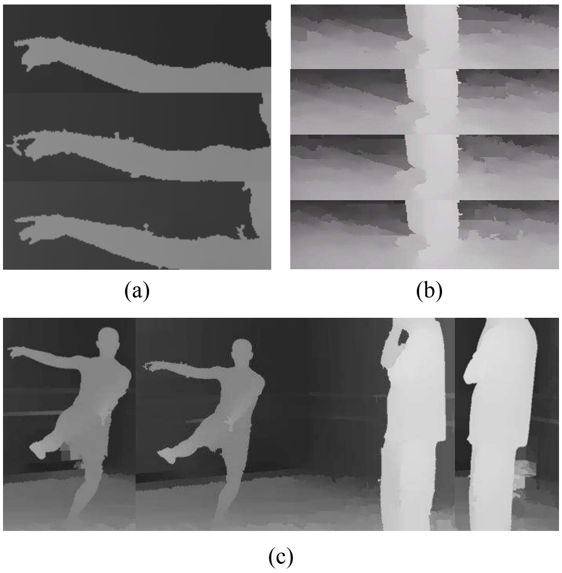

Figure 1: Various types of depth map artifacts: (a) across cameras, (b) over time, (c) caused by occlusions.

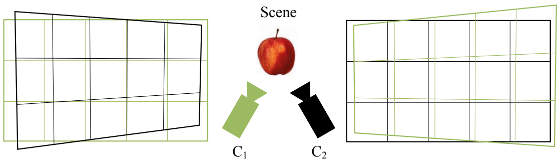

Figure 2: Illustration of the sampling grid of two cameras C_1 and C_2 capturing a scene and the perspective transformation of the sampling grids into each other.

distribution of samples across multiple camera sampling grids. A camera sensor constitutes a discrete, rectangular and uniform sampling grid. Using multiple cameras to capture a common scene as illustrated in Fig. 2, samples taken by a first camera C_1 can be transformed to the sampling grid of a second camera C_2 with a different camera view point and vice versa. The resulting spatial distribution of transformed samples on the target camera sampling grid is typically continuous and projectively distorted. Reconstructing a point cloud from camera sensor samples, the individual 3D points are structured through a 3D projection of the 2D sampling grid of the camera. This sampling grid in 3D space is thereby determined by the camera position, orientation and properties, i.e. the intrinsic and extrinsic camera parameters. Using multiple cameras for point cloud reconstruction, the respective sampling grid projections are superimposed and the spatial distribution of 3D points is neither rectangular nor uniform. A representation that conforms to a single sampling grid but preserves as much of the captured information as possible simplifies subsequent processing, e.g. filtering of the inconsistencies illustrated in Fig. 1.

We build our work on the algorithms presented in (Belaifa et al., 2012) that resample and filter point cloud data reconstructed by multiple cameras to create a consistent representation across cameras. In this work, we enhance the presented algorithms and extend them to resample and filter multiview video over time and across cameras. Furthermore, an objective evaluation scenario is presented in order to objectively evaluate the performance of the presented algorithms. Therefore, the source camera views together with the enhanced depth maps are tested in a view synthesis scenario against synthesis results of the original source material.

Section 2 presents the algorithms while the evaluation procedure and results are given in section 3 followed by a short summary and conclusion in section 4.

2 POINT CLOUD RESAMPLING AND FILTERING

The presented algorithms operate in two consecutive stages, i.e. resampling and filtering as illustrated in Fig. 3. First, in the initial resampling stage, samples of moving objects are removed from the source data for temporal filtering and a color matching scheme is applied. Two separate approaches for alignment of 3D points to a common 3D sampling grid are presented, i.e. either through sampling voxels in 3D space or reconstructed surface meshes. Second, the resampled data undergoes a filtering stage based on clustering to remove artifacts by a weighting threshold and achieve spatiotemporal consistency across cameras and time instants. From the resampled and filtered point clouds, enhanced depth maps can be produced through projection to the target camera image plane. The algorithms intend to preserve details captured only by single cameras while filtering artifacts of the depth estimation. The following subsections describe the algorithms in detail.

2.1 Voxel and Mesh based Resampling

In the initial resampling stage, each sample at 2D sampling point (u, v) taken from each camera C_i, with $i = 0 \dots (n - 1)$ in a n camera data set, is projected

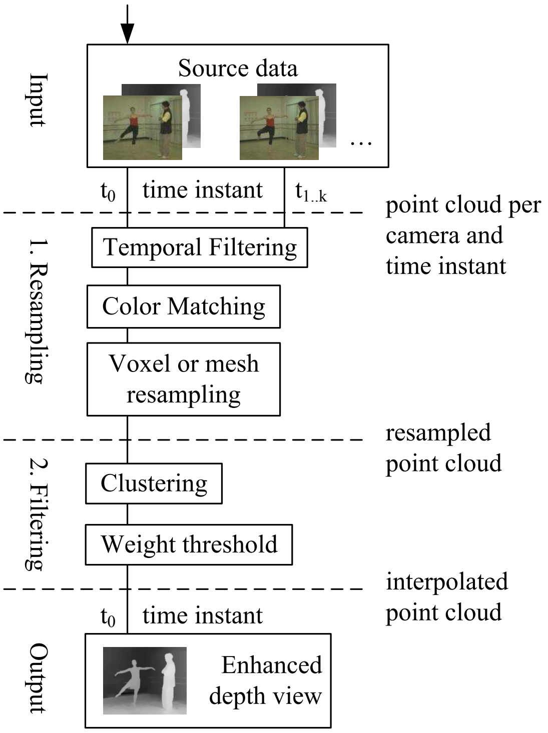

Figure 3: Algorithm flow chart.

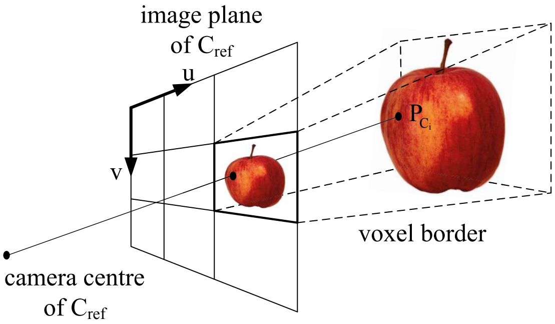

Figure 4: Schematic of the voxel-based approach with a sampling volume constituted by a C_{ref} sampling grid projection.

to point (X,Y,Z) in 3D space by means of central projection with the appropriate camera parameters. The third component Z, i.e. the depth component in 3D space, is determined by a non-linear mapping of the associated sample value in the corresponding depth map.

In the voxel-based resampling approach, the common 3D sampling grid is established by projecting the borders of the 2D sampling grid of a reference camera C_{ref} to 3D space as illustrated through dashed lines in Fig. 4. The volume enclosed by the scene depth range and the projected borders of a given 2D sampling point (u,v) of C_{ref} to 3D space constitutes a sampling voxel. 3D points reconstructed from a contributing camera C_i, exemplary annotated as P_{C_i} in the figure, that are located within the respective voxel constitute the sampling set $A(u,v)$ for each (u,v) of C_{ref}. This approach is referred to as *voxel resampling* in the following sections.

A more complex method to resample the point cloud data uses a generated 3D surface mesh for each individual camera C_i as illustrated as dashed lines in Fig. 5. The surface mesh is generated by projecting all samples of C_i to 3D space and constructing triangles across each pair of neighboring sample rows. The intersection of the surface mesh with the projection of each sampling position (u,v) of C_{ref} is used as basis to linearly interpolate the depth value at the intersection and constitute the sampling set $A(u,v)$. This approach is referred to as *mesh resampling* in the following sections.

2.2 Color Discontinuities

When reconstructing a point cloud from multiple cameras, depth discontinuities at object boundaries tend to vary notably between camera view points. Figure 6 provides an example of this observation, with (a) depicting a point cloud reconstruction of a foreground detail from C_{ref} source data only and (b) showing a point cloud reconstructed from all cameras of the data set with a C_{ref} camera view overlay and noticeably dilated object boundaries. Apart from the additive depth estimation error noise of all cameras, the imprecision of estimated camera parameters leads to minor translation and rotation of the reconstructed point cloud per cameras with respect to each other.

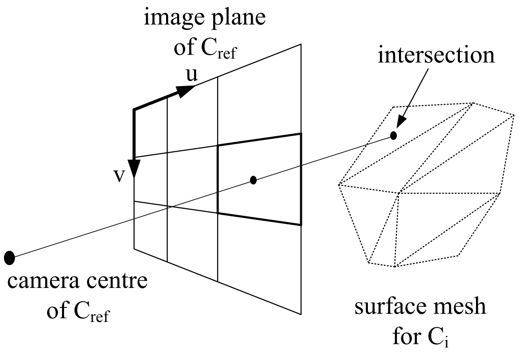

Figure 5: Schematic of the mesh-based approach. Sampling points are generated from the intersection of the projected reference sampling point and the 3D surface mesh of each camera C_i.

With the basic assumption, that depth discontinuities at object boundaries tend to coincide with color discontinuities as in (Tao and Sawhney, 2000), a threshold can be established to filter samples that do not match color discontinuities of the C_{ref} camera view. Therefore, both resampling approaches use a threshold m_c on the L_1 distance between the RGB vector of the current C_i sample and the corresponding C_{ref} sample to ensure correspondence of the two. The effect of color matching is depicted in (c) of Fig. 6 showing the same reconstruction as (b) with active color threshold m_c.

2.3 Extension to the Temporal Domain

In (Belaifa et al., 2012) the authors consider samples from multiple camera view points while the used data set actually consists of sequences of video frames and

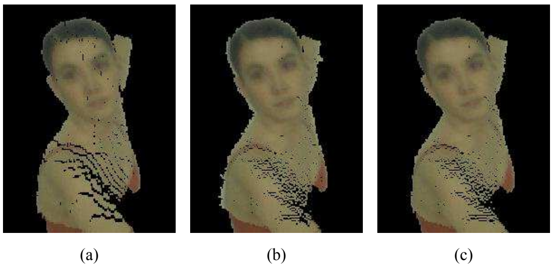

Figure 6: Effect of color threshold on depth discontinuities: (a) point cloud reconstruction of C_{ref}, (b) resampled point cloud from all C_i and overlay of the C_{ref} camera view without color threshold and (c) with color threshold.

thus contains additional information for point cloud reconstruction and subsequent processing in the temporal domain. Considering this domain allows filtering of movement artifacts on static objects such as the floor as shown in Fig. 1 (b). For this purpose, creation of the sampling sets $A(u,v)$ at the time instant t_0 in the resampling stage relies on source data from time instants $t_0 ... t_k$ where k as the temporal filtering depth. Samples belonging to objects that move in the course of the additional k frames should not be added to $A(u,v)$. Therefore, a binary mask view M_j for source data of each camera C_i of time instants $t_j > t_0$ is created and used in the resampling stage according to the following procedure.

- Initialize M_j to zero and apply Gaussian filter to the camera view luma components L of C_i at time instants t_{j-1} and t_j.
- Set M_j to one at sample positions where the difference between L_{j-1} and L_j exceeds a threshold. If $j > 1$, additionally set M_j to one at sample positions where the difference between L_0 and L_j exceeds a threshold.
- Compute a dense optical flow map between L_j and L_{j-1} based on polynomial expansion according to (Farnebäck, 2003). Set M_j to one at sample positions with motion vectors that exceed a threshold.
- Dilate M_j with a rectangular kernel.
- Samples of C_i source data for which the corresponding sample in M_j is set to one are not considered in the subsequent resampling.

An example for the source data masked by M_j and a temporal filtering depth of $k = 3$ is given in Fig. 7. A detail of the original camera view from a single camera C_i for time instants t_0 to t_3 is given from left to right in the top row (a). Row (b) shows samples of corresponding depth map of C_i that subsequently contribute to the resampling stage after M_j is applied, where the white areas at t_1 to t_3 correspond to active areas of M_j. The bottom row (c) shows the reconstructed point cloud of all cameras at t_0 to t_3 which jointly contribute to the resampling stage for t_0. In order to compensate removed samples of moving objects at t_1 to t_3 in the following filtering stage, the weight of samples from each camera C_i at t_0 for which the corresponding sample in M_j is set to one is increased accordingly.

2.4 Filtering Stage

In the subsequent filtering stage, the resampled source data from multiple cameras C_i and time instants t_i is filtered. While merging the samples of the contributing cameras within $A(u,v)$, the aim is to filter artifacts and preserve details. Our algorithm is based on hierarchical group-average linkage clustering as described in (Hastie et al., 2001) and regards all samples in $A(u,v)$ as clusters with weight equal to 1 along the depth coordinate axis. Iteratively, the two closest samples within $A(u,v)$ are merged by linearly interpolating their depth values to the new depth Z_{new} and agglomerating their weight. The position of the merged sample along the horizontal and vertical dimension in 3D space correspond to the projection of the C_{ref} sampling grid position to Z_{new} in 3D space.

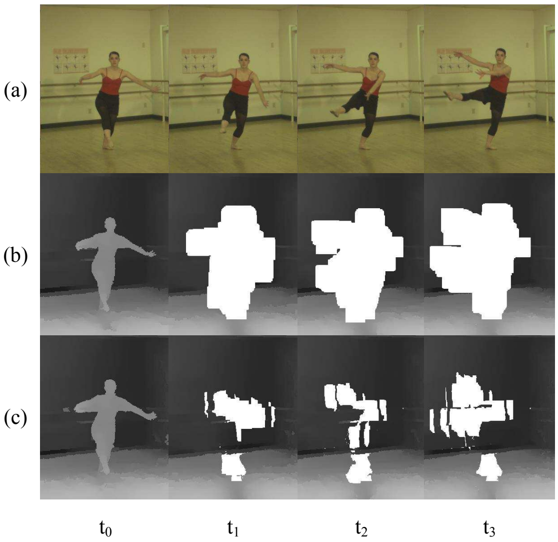

Figure 7: Moving object removal for temporal filtering: (a) camera view, (b) corresponding depth map with overlay of mask M_j in white, (c) reconstructed point cloud of all cameras C_i.

For each sampling grid position (u,v) of C_{ref}, the clustering process stops when all samples in $A(u,v)$ have a distance to each other greater or equal than a minimum cluster distance threshold m_d. This end condition ensures back- and foreground separation of the given scene. The mapping between the sample value of a depth map and the depth coordinate in 3D space is not necessarily linear, e.g. to capture more depth detail of objects closer to the camera. To ensure equal preservation of details regardless of the object depth in the scene, m_d is given in terms of depth levels rather than Euclidean distance in 3D space.

After the clustering process, a minimum cluster weight threshold m_w is applied to remove samples with a weight less than m_w from $A(u,v)$, i.e. outlier samples that are spatially far from any other samples in $A(u,v)$ and are thus not merged. As the size of $A(u,v)$ considerably varies over the sampling grid depending on camera setup and the resampling ap-

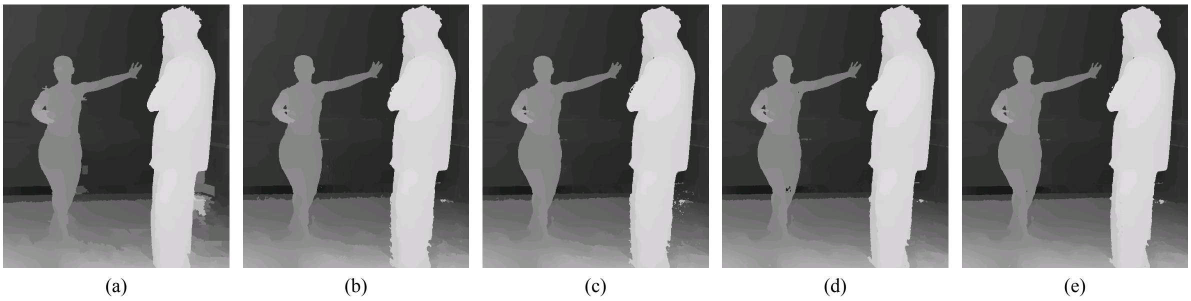

Figure 8: Exemplary results of the enhanced depth maps: (a) original depth map,(b) enhanced depth map using voxel resampling, (c) mesh resampling, (d) and (e) with temporal filtering depth $k = 2$ using voxel resampling and mesh resampling, respectively.

proach, m_w is given as relative to sum of weights within $A(u,v)$. If none of the samples in $A(u,v)$ satisfy m_w, only the sample with the largest weight is kept.

3 EVALUATION

The presented algorithms are evaluated in a view synthesis scenario on the 100 frames ballet and breakdancers data set (Zitnick et al., 2004) based on the assumption that depth maps with less inconsistencies across cameras and over time lead to a higher quality of synthesis results. As base line for the evaluation, the original texture and depth maps of all available cameras C_i are used to synthesize camera views at all camera view points.

Synthesis is carried out through projection of all relevant samples of the data set into 3D space and back to the target image plane of interest. At each sampling position (u,v) of the target image plane, a blending procedure of samples ensures decent image quality while preserving artifacts that originate from the depth maps. The quality of synthesis results compared to the corresponding original camera views is measured frame-wise in terms of PSNR and MSSIM (Wang et al., 2004) with an 8×8 rectangular window.

The same synthesis procedure is followed using the original camera views but depth maps enhanced by the algorithms presented in this work. The synthesis results with enhanced depth maps are compared to the respective original camera views likewise.

Enhanced depth maps of all cameras were produced for all frames of the data sets with a temporal filtering depth of $k = \{0,\ldots,2\}$. This range gives an outlook on the effects of the proposed technique on the view synthesis and limits the amount of additional data that has to be processed in the resampling and subsequent filtering stage of the algorithms. A color threshold of $m_c = 0.18$ relative to the combined maximum sample value of the three color channels and a minimum cluster distance of $m_d = 0.03$ relative to the maximum depth map sample value is evaluated. The minimum cluster weight m_w is chosen as 0.13 of the sum of weights in $A(u,v)$ without temporal filtering. This choice is motivated by the amount of n cameras in the data set. A threshold m_w slightly higher than $1/n$ is not satisfied by outliers that stem from single cameras and remain disjoint during clustering. As the number of samples to process in the filtering stage grows with the temporal filtering depth, so does the weight of outlier clusters after filtering. Therefore, the threshold of removal of outliers is increased to $m_w = 0.26$ and $m_w = 0.34$ for $k = 1$ and $k = 2$, respectively.

Across all experiments, the quality of the synthesis results is above 32dB PSNR and an MSSIM of 0.86 on average. Figure 8 shows a detail of an original depth map of the ballet sequence of the right-most camera (a) and the corresponding enhanced depth map as produced by the voxel resampling (b) and the mesh resampling approach (c) without filtering in the temporal domain. Exemplary results for a temporal filtering depth of $k = 2$ are given for voxel resampling in (d) of Fig. 8 and mesh resampling in (e). It can be seen from the figure that the algorithms suc-

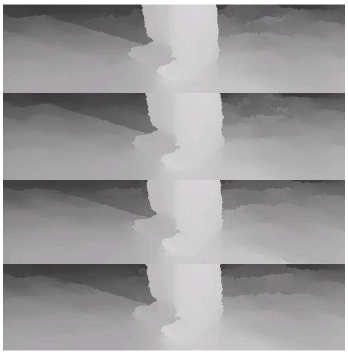

Figure 9: Exemplary results for filtering in the temporal domain.

cessfully reduce artifacts at object boundaries and revealed occlusions. Figure 9 shows a detail of the same four consecutive frames of the ballet sequence as in Fig.1 (b) after applying the algorithm with voxel resampling and a temporal filtering depth of $k = 2$ with noticeably smoothed artifacts of the static areas.

The top plot in Fig. 10 reports the ΔPSNR of synthesis results of the two algorithms with both data sets compared to the synthesis results of the original source data over the temporal filtering depth k and averaged over all frames and all cameras while the bottom plot reports the results in terms of ΔMSSIM. Positive values report a quality improvement for the algorithms.

It can be seen that the synthesis results based on the enhanced depth maps achieve a ΔPSNR above 0.45dB for the ballet data set and a minimal positive tendency with an increasing temporal filtering depth. For the breakdancers data set that contains more motion only a smaller quality improvement is reported and the results show a lower quality increase with increasing temporal filtering depth. Although overall MSSIM gains are minimal, they show a similar behavior and tendency as the PSNR measurements.

Figure 11 reports the results of the experiments over camera index i without performing temporal filtering and averaged over all frames. It can be noticed that the best performing synthesis achieves well above 1dB PSNR gain for the ballet data set and MSSIM measurements show a tendency similar to that of the PSNR measurements. A noticeable drift of results can be observed from the left-most to the right-most camera for both tested data sets, which may be caused by scene geometry and synthesis setup, i.e. depth map artifacts may impair synthesis results to a varying extent.

While in general the synthesis results with enhanced depth maps of the ballet data set (little motion with strong back-/foreground separation) show notable objective and visual improvements, the results of the breakdancers data set (much motion with weak back-/foreground separation) do so only to a small degree.

Figure 12 gives a comparison of the original camera view (a), the synthesis result with source data (b), the synthesis results of voxel resampling (c) and mesh

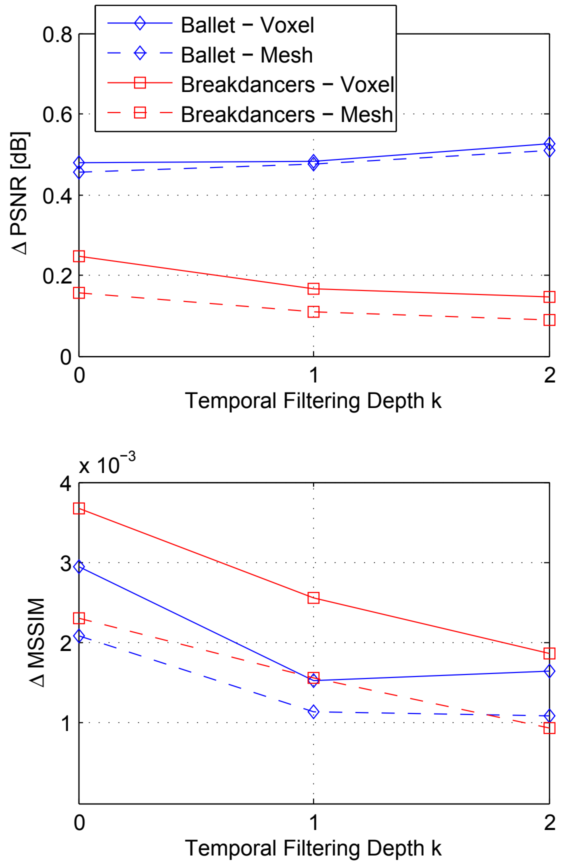

Figure 10: Synthesis quality improvements for both data sets over temporal filtering depth with presented algorithms in PSNR and SSIM averaged over all cameras and frames.

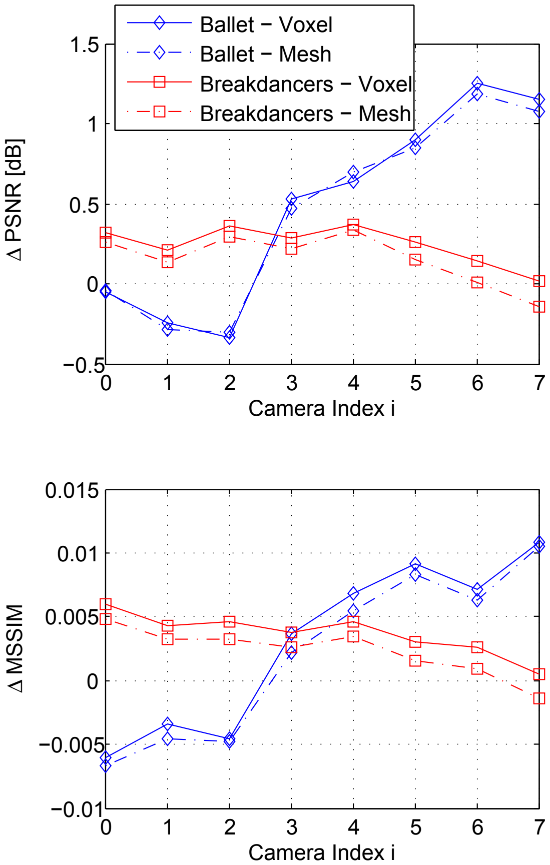

Figure 11: Synthesis quality improvements for both data sets over camera index i with presented algorithms in PSNR and MSSIM averaged over all frames.

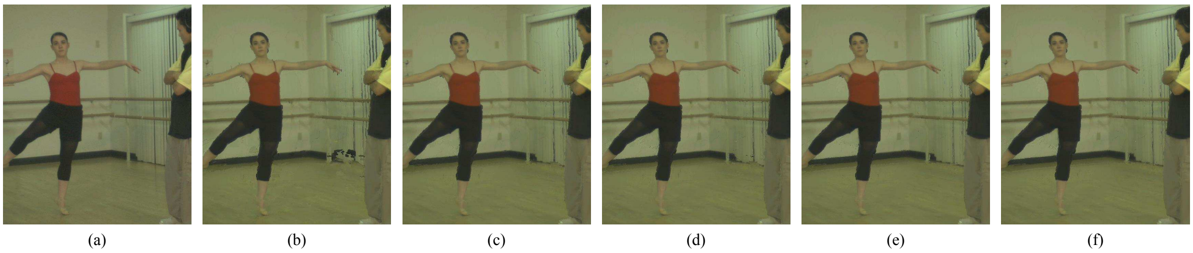

Figure 12: Exemplary results of the view synthesis: (a) original camera view, (b) synthesis with source data, (c) voxel resampling, (d) mesh resampling, (d) and (e) temporal filtering depth of $k = 2$ for voxel resampling and mesh resampling, respectively.

resampling (d) without temporal filtering, the respective synthesis results with a temporal filtering depth $k = 2$ in (e) and (f). While effects in the temporal domain are hard to notice from single frames, synthesis artifacts related to depth map artifacts on object boundaries and revealed occlusions can be noticed in (b) which do not occur in (c) to (f). Overall, the quality of synthesis results with enhanced depth maps does not vary significantly with a negligible positive margin for the voxel based approach and temporal filtering.

4 SUMMARY AND CONCLUSIONS

This work presents algorithms to resample and filter point cloud data reconstructed from multiple cameras and multiple time instants. In an initial resampling stage a voxel and a surface mesh based approach are presented to resample the point cloud data into a common sampling grid. Subsequently, the resampled data undergoes a filtering stage based on clustering to remove artifacts of depth estimation and achieve spatiotemporal consistency. The presented algorithms are evaluated in a view synthesis scenario. Results show that view synthesis with enhanced depth maps as produced by the algorithms leads to less artifacts than synthesis with the original source data. The difference in achieved quality between the voxel and the surface mesh based approach is negligible and with regard to the computational complexity of the surface mesh reconstruction, the voxel based approach is the desirable solution for resampling.

Filtering in the temporal domain shows slight synthesis quality improvements when moving objects are confined to a limited region of the scene as in the ballet data set. For data sets in which moving objects cover larger areas such as in the breakdancer data set, temporal filtering does not improve synthesis results compared to filtering across cameras. The presented motion masking excludes samples within a relatively wide image area with respect to the actual moving object. Therefore, depth map artifacts in the corresponding areas are not interpolated in the filtering stage and thus affect synthesis.

REFERENCES

Belaifa, O., Skupin, R., Kurutepe, E., and Sikora, T. (2012). Resampling of Multiple Camera Point Cloud Data. In *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*, pages 15–19. IEEE.

Dodgson, N. A. (2005). Autostereoscopic 3D Displays. *Computer*, 38(8):31–36.

Farnebäck, G. (2003). Two-Frame Motion Estimation Based On Polynomial Expansion. In *Image Analysis*, pages 363–370. Springer.

Hastie, T., Tibshirani, R., and Friedman, J. J. H. (2001). *The Elements of Statistical Learning*, volume 1. Springer New York.

Merkle, P., Morvan, Y., Smolic, A., et al. (2009). The Effects of Multiview Depth Video Compression On Multiview Rendering. *Signal Processing: Image Communication*, 24(1-2):73–88.

Merrell, P., Akbarzadeh, A., Wang, L., Mordohai, P., Frahm, J.-M., Yang, R., Nister, D., and Pollefeys, M. (2007). Real-Time Visibility-Based Fusion of Depth Maps. *Computer Vision, IEEE International Conference on*, 0:1–8.

Scharstein, D. and Szeliski, R. (2002). A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International journal of computer vision*, 47(1):7–42.

Seitz, S., Curless, B., Diebel, J., Scharstein, D., and Szeliski, R. (2006). A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms.

Smolic, A., Mueller, K., Merkle, P., Fehn, C., Kauff, P., Eisert, P., and Wiegand, T. (2006). 3d Video and Free Viewpoint Video-Technologies, Applications and Mpeg Standards. In *Multimedia and Expo, 2006 IEEE International Conference on*, pages 2161–2164. IEEE.

Starck, J. and Hilton, A. (2005). Virtual View Synthesis of People From Multiple View Video Sequences. *Graphical Models*, 67(6):600–620.

Tao, H. and Sawhney, H. S. (2000). Global Matching Criterion and Color Segmentation Based Stereo. In *Applications of Computer Vision, 2000, Fifth IEEE Workshop on.*, pages 246–253. IEEE.

Vetro, A., Yea, S., and Smolic, A. (2008). Toward a 3D Video Format for Auto-Stereoscopic Displays. In *Optical Engineering+ Applications*, pages 70730F–70730F. International Society for Optics and Photonics.

Vogiatzis, G., Torr, P. H., and Cipolla, R. (2005). Multi-View Stereo Via Volumetric Graph-Cuts. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 391–398. IEEE.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *Image Processing, IEEE Transactions on*, 13(4):600–612.

Zitnick, C., Kang, S., Uyttendaele, M., Winder, S., and Szeliski, R. (2004). High-Quality Video View Interpolation Using a Layered Representation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 600–608. ACM.