Hallucination-Free Multi-View Stereo

Michal Jancosek and Tomas Pajdla

Center for Machine Perception, Department of Cybernetics
Faculty of Elec. Eng., Czech Technical University in Prague
{jancom1,pajdla}@cmp.felk.cvut.cz

Abstract. We present a multi-view stereo method that avoids producing hallucinated surfaces which do not correspond to real surfaces. Our approach to 3D reconstruction is based on the minimal s-t cut of the graph derived from the Delaunay tetrahedralization of a dense 3D point cloud, which produces water-tight meshes. This is often a desirable property but it hallucinates surfaces in complicated scenes with multiple objects and free open space. For example, a sequence of images obtained from a moving vehicle often produces meshes where the sky is hallucinated because there are no images looking from the above to the ground plane. We present a method for detecting and removing such surfaces. The method is based on removing perturbation sensitive parts of the reconstruction using multiple reconstructions of perturbed input data. We demonstrate our method on several standard datasets often used to benchmark multi-view stereo and show that it outperforms the state-of-the-art techniques [1].

Keywords: multi-view stereo, stereo, 3D reconstruction.

1 Introduction

Promising approaches to Multi-View Stereo (MVS) reconstruction, which have appeared recently [1–7], get the degree of accuracy and completeness comparable to laser scans [8, 9]. Yet, producing complete reconstructions of outdoor and complicated scenes is still an open problem.

In this work we extend previous work by presenting a method for hallucination-free MVS reconstruction. By hallucinated surfaces we mean the surfaces which are present in the reconstruction but do not correspond to real surfaces.

We build on the global approach to 3D reconstruction based on the minimal s-t cut of the graph derived from the Delaunay tetrahedralization of a dense 3D point cloud [4, 10]. It solves the visibility task by accumulating energy in free space between the surface and cameras, which is later used to solve the s-t cut. The main advantages of this approach are robustness to noise in the 3D point cloud and producing water-tight reconstructions. This approach is very opportunistic and tends to produce complete meshes by using a strong visibility prior to explain missing data by surfaces.

[1] The authors were supported by SGS10/186/OHK3/2T/13, FP7-SPACE-241523 PRoViScout and MSM6840770038.

K.N. Kutulakos (Ed.): ECCV 2010 Workshops, Part II, LNCS 6554, pp. 184–196, 2012.

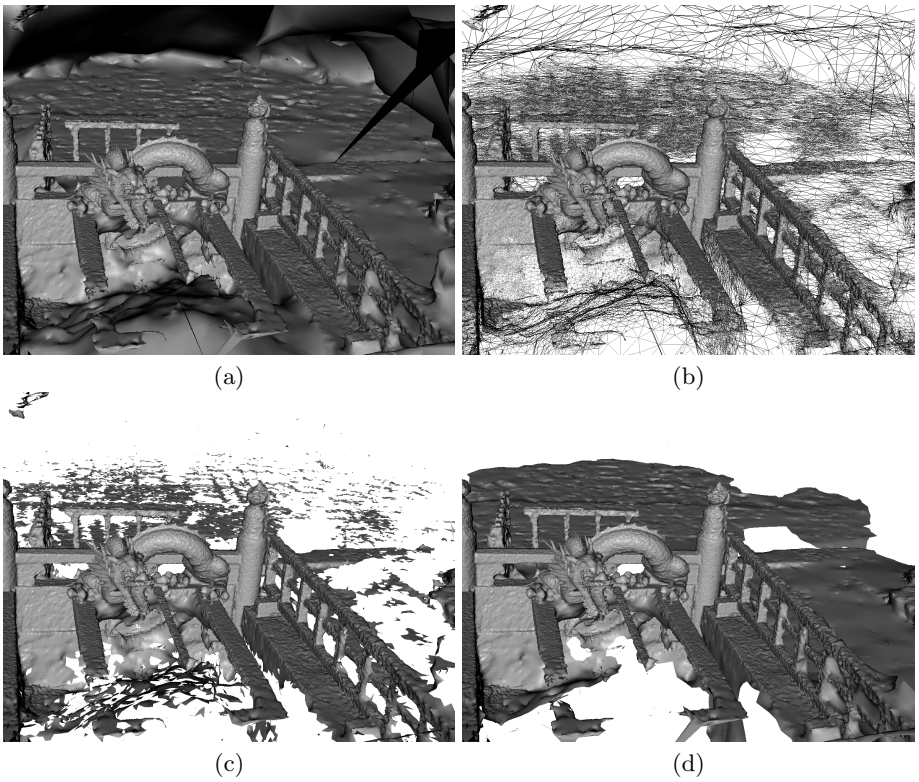


Fig. 1. Dragon-P114 data set: (a) 3D surface before removing hallucinations, (b) 3D surface wireframe before removing hallucinations, (c) 3D surface after removing hallucinations using approach proposed in [7], (d) 3D surface after removing hallucinations using our method.

In contrast to this, the alternative state-of-the art method [7] is focused on producing a noise free oriented point cloud to generate 3D mesh using [11]. This approach is rather conservative and tends to reconstruct surfaces which have strong support from the data. It leaves the space free where the support is not sufficient.

In many situations, the former method is preferable to the later one since it is difficult to fill in the holes in 3D reconstruction in later processing stages when the access to original images may not be available anymore. Recent work of Vu [1], which produces the initial mesh as in [10] and later refines it by using a variational method, demonstrates that this approach can be used to produce excellent results for closed objects and scenes sufficiently represented by images from all directions.

In this work, we focus on removing hallucinated surfaces which are often generated by the approach [10] when reconstructing complicated scenes with multiple objects and free open space. This happens, for instance, when reconstructing outdoor scenes where a hallucinated surface is generated in place of the sky to obtain a closed surface, Figures 1(a,b) and 2(a,b).

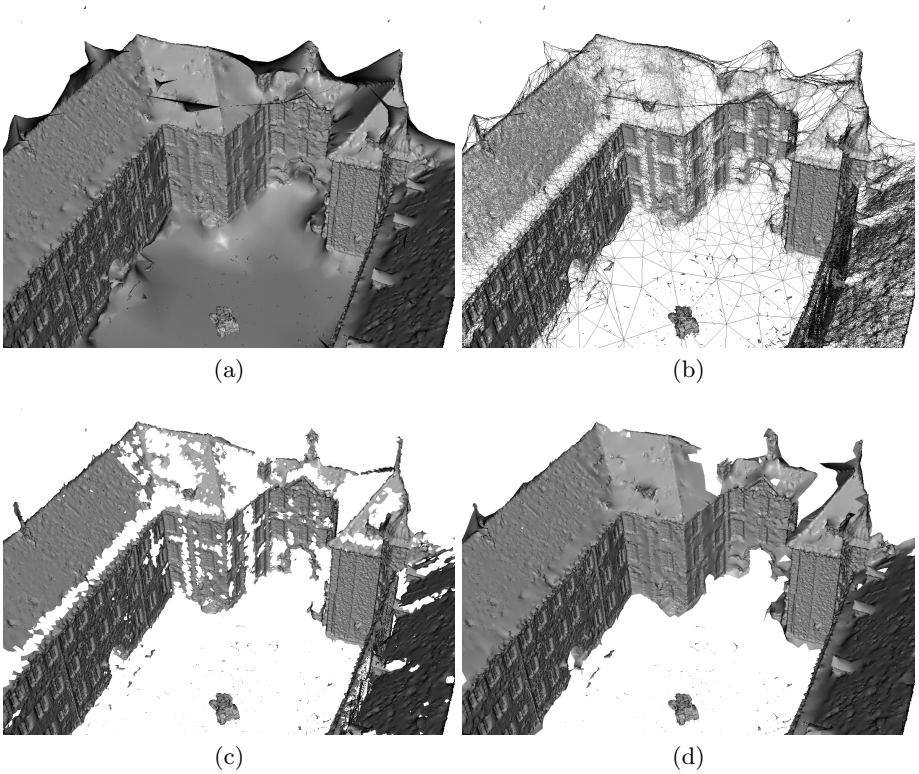


Fig. 2. Castle-P30 data set: (a) 3D surface before removing hallucinations, (b) 3D surface wireframe before removing hallucinations, (c) 3D surface after removing hallucinations using approach proposed in [7], (d) 3D surface after removing hallucinations using our method.

In [7] the problem is solved by removing large triangles, i.e. they discard the triangles whose average edge length is greater than six times the average edge length of the whole mesh. Here we show that there exist important situations when the large triangles are not hallucinations and are needed to produce complete meshes. Our approach can avoid discarding such triangles. On the other hand we show that sometimes we need to discard small triangles, too.

Our method can play an important role in a large-scale city modeling system. In such datasets top views are usually missing which, as we have mentioned above, often leads to hallucinations.

2 Motivation

The approach by using triangle (or related tetrahedron) property like average edge length [7], maximal edge length, triangle area, radius of circumscribed

sphere of the related tetrahedron, and so on, can not be used to remove hallucinated surfaces well in general. The reason is that it is possible (and we demonstrate it in our experiments) that the reconstruction pipeline will produce the mesh where one can often find two subsets of triangles of the same average edge length of the mesh such that the first subset corresponds to a hallucination but the second one corresponds to a real surface. Therefore, it is in general not possible to find a threshold on these dimensional parameters which would separate hallucinated triangles, see Figure 5 .

The main idea behind our approach is motivated by the following observation. The surfaces which have strong support from the data are mostly not affected by small perturbations. By the strong support we mean that there are many reconstructed points near the real surface. On the other hand the surfaces, which are originally created due to false positive points, are usually strongly affected by perturbations because false positives are generated randomly and usually sparsely distributed far from true surfaces.

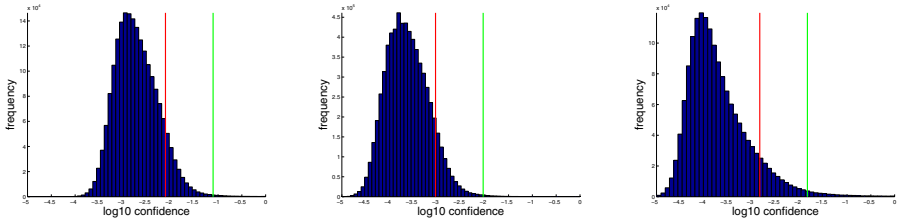
We assign a confidence value to each triangle based on the sensitivity to perturbations (see Section 6). We build the s-t graph [12] from the triangulation and the confidences and solve it by the implementation [13]. If a triangle is labelled as sink we deem the triangle as hallucinated.

One can argue that large triangles will be mostly created from false positive 3D points. That is often true but large triangles are also created in textureless parts of the scene (see Figure 2) and in parts which have very oblique viewing angles (see Figure 1). Such triangles are important because they make the reconstruction complete. The main contribution of our method is to keep such triangles in the reconstruction while removing the triangles that are hallucinated.

We have to point out that our method may keep unseen parts of the surface when they are a part of the visual hull of the scene (see Figure 6 with the roof region behind dormers is filled). We do not consider such parts as hallucinations.

3 Reconstruction Pipeline

Our MVS pipeline is similar to the pipeline proposed in [1]. First, we compute feasible camera pairs based on the epipolar geometry as in [14]. Next, we detect and match SIFT features [15] in the feasible camera pairs using [16]. We triangulate matches and create seeds. A seed is a 3D point with a set of cameras it was triangulated from. For each camera we compute the minimal and the maximal depth based on the related seeds. Then, we perform the plane-sweeping and filtering (see Section 4) at several scales. To remove hallucinated surfaces, we run a mesh computation k times from differently perturbed data. In each iteration we perturb (see Section 5) the point cloud generated by plane-sweeping and use it as the input to our implementation of the method proposed in [10]. We do not perform mesh refinement as in [1] but do mesh smoothing as in [10]. This gives us k meshes. We remove hallucinated surfaces from the first mesh using other meshes (see Section 6 and 7). This gives us the resulting mesh.



(a) (b) (c)

Fig. 3. Histogram of log_{10} of confidences for (a) Castle-P30 dataset, (b) Fountain-P11 dataset and (c) Dragon-P114 dataset with s (red) and t (green) values. (see text)

4 Plane-Sweeping and Filtering

Our plane-sweeping is slightly different from the state-of-the art [17–19]. We do plane sweeping for each reference camera with respect to the set of α nearest feasible target cameras (we use $\alpha = 4$ in all of our experiments). For each pixel p and for each depth d (corresponding to a plane parallel to the reference image plane) we compute the photo-consistency $f(p, d)$ of the depth as follows. For each pair of the reference r and target t cameras we compute photo-consistency value $c(p, d, r, t)$ as the NCC between 5×5 window centred in the pixel on the reference image and its projection to the target image. The projection is generated by the homography inducted by the plane and consistent with the epipolar geometry between the reference and target images. The NCC value is in the range $\langle -1, 1 \rangle$, where value -1 represents the worst photo-consistency and value 1 the best one. The photo-consistency $f(p, d)$ equals the maximum of $c(p, d, r, t)$ over all target cameras t and fixed r. The reconstructed depth $\gamma(p)$ of the pixel p is chosen as the depth d for which $f(p, d)$ is maximal and $f(p, d) > \delta$ (we use $\delta = 0.8$ in all of our experiments). If there does not exist such depth, then we set $\gamma(p)$ as unknown. This plane-sweeping strategy produces a lot of true positive 3D points, but a lot of false positive ones. Therefore, we perform a simple but fast and effective filtering after plane-sweeping. For each 3D point we search for other 3D points in its small neighbourhood. If there are at least β (we use $\beta = 2$ in Fountain dataset and $\beta = 3$ in all others) 3D points from β different cameras, then we accept the point. We consider all depths which were filtered out as unknown. We choose this approach because we have experimentally verified that this approach produces more true positives than for example [18, 19]. On the other hand it still (even after filtering) produces some false positives. But this is not critical because we are later using a strong tool [10] which can effectively deal with noise.

5 The Principle of the Removal of Hallucinated Surfaces

Method [10] tends to produce closed meshes and hence it generates false (hallucinated) surfaces in places which are not well captured by any camera. The hallucinated surfaces are often related to missing cameras which would otherwise

lead to cleaning the space between the cameras and the real surface of the scene. When cameras are not present, method [10] hallucinates surfaces from sparsely distributed false positive points which are present in the point cloud. On the other hand, surfaces, which are strongly supported by the data, i.e. where the point cloud is dense near the real surface, or which are strongly supported by a visibility prior, and hence do not have to be strongly supported by the data, are not affected by the sparsely distributed false positive points. We build our approach on this observation. We introduce a small amount of false positive points into the original point cloud several times and filter out unstable hallucinated surfaces. We implement it by adding a small amount of noise into the depth maps constructed by plane-sweeping in each iteration. Consider a depth map and all the pixels with unknown depth in that map. We randomly choose γ (we use $\gamma = 0.1$ in all of our experiments) percent of the pixels with unknown depth and assign them depths randomly. The new depth is chosen randomly from the four times the depth range of the camera. The values were selected experimentally and were sufficient in all of our experiments.

6 Perturbation Based Triangle Confidence Computation

We assign a confidence value to each triangle of each of the k meshes. Let's assume the $i-th$ mesh and the $j-th$ triangle t. For each $k-th$ mesh $k \neq i$ we find the nearest triangle t_k to the triangle t. We measure the distance of triangles by the distance of the triangle centers (c_t, c_{t_k}). Now, for each pair (t, t_k) of triangles we compute $d(t, t_k) = \min\{d(c_t, t_k), d(c_{t_k}, t)\}$ over all pairs t_k, t (with fixed t) where $d(c_t, t_k)$ is the distance of the point c_t to the plane defined by triangle t_k. To compute the confidence of the triangle $\delta(t)$ we use Gaussian kernel voting to cluster values $d(t, t_k)$.

7 Graph-Cut Based Hallucinations Removing

To remove triangles with high confidence, we formulate a minimum s-t cut problem [12]. We create a graph from the mesh such that the nodes correspond to triangles. If two triangles are neighbouring, then we create the edge between the corresponding nodes. We compute $90th$ percentile s of all triangle confidences to find the threshold on the triangle confidence. The threshold at the $90th$ percentile is very conservative because majority of confidences are from small triangles which have usually similar confidences near zero, see Figure 3. We introduce value $t = 10\,s$. Value s should correspond to triangles which should be definitely in the final mesh, value t to triangles which should definitely be removed. We assign $(s - \delta(t))^2$ value to each s-edge and $(t - \delta(t))^2$ value to each t-edge. To each edge between nodes we assign value $s + (t - s)/4$. This value is established experimentally to remove isolated triangles. We use this value in all of our experiments. To solve the s-t cut problem we use the implementation described in [13]. The final mesh consists of the triangles represented by the s-nodes.

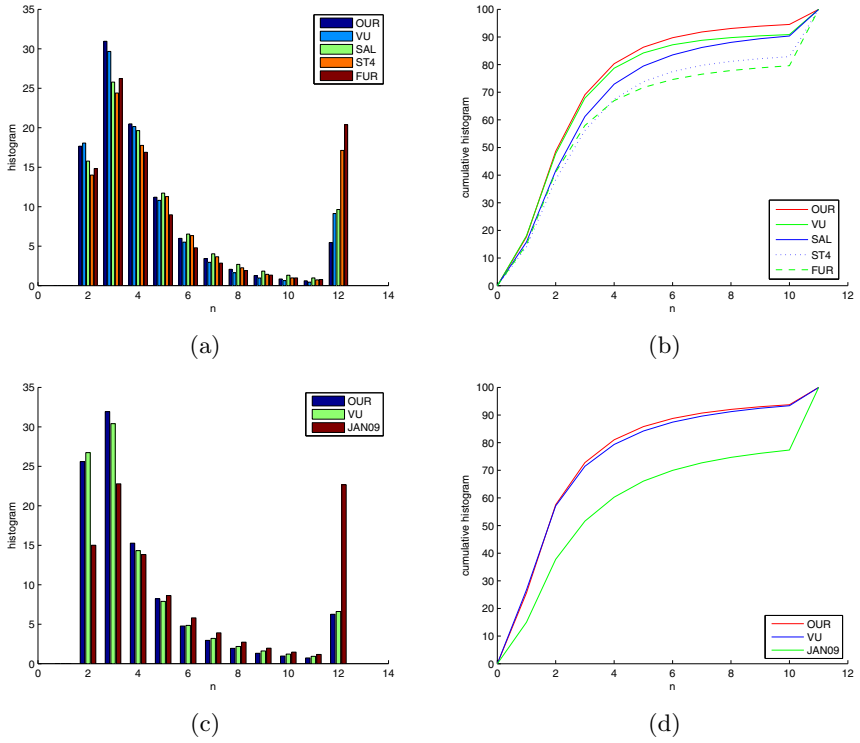
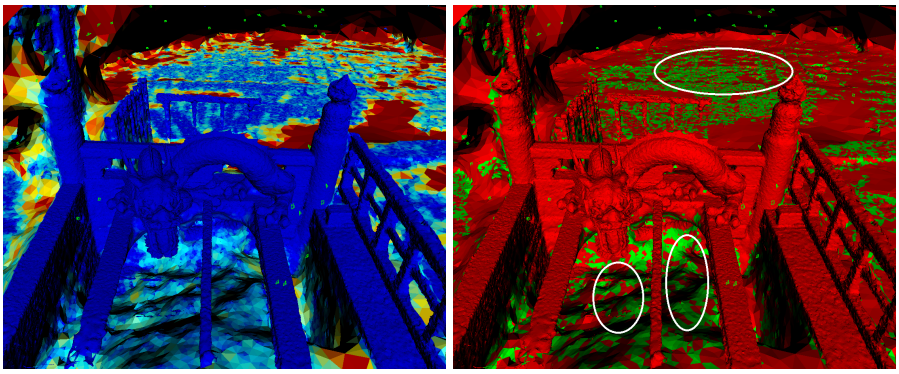


Fig. 4. Strecha's evaluation [9] for the Fountain-P11 (a,b) and Castle-P30 (c,d) data sets. OUR - this paper, VU [1], SAL [20], ST4 [21], FUR [7], JAN09 [14]. (a,c) histograms of the relative error with respect to $n\sigma$ for all views. The σ is determined from reference data by simulating the process of measurement and can vary across the surface and views. (b,d) relative error cumulated histograms.

8 Performance Discussion

In this section we provide the time performance discussion of our pipeline. We discuss how to make significant speedup, too.

The plane-sweeping was performed on the original scale and on two, three and four times sub-sampled images. The plane-sweeping is implemented on GPU. The computation time of one depth map varies from a few minutes on 3072×2048 resolution to a second on the smallest scale (768×512). The time complexity is cubic, because the number of depths scales with the image resolution. The computation time of the filtering step is approximately tens of seconds. We have to point out that the plane-sweeping and filtering is performed only once for each camera at each scale. The computation time of one perturbation iteration using our implementation of [10] is approximately tens of minutes. The time depends on the number of input points. In our experiments the number of points varies



(a) (b)

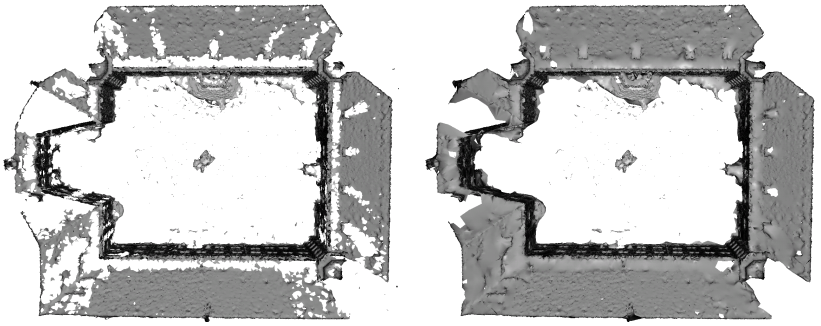
Fig. 5. Dragon-P114 data set: (a) 3D surface before removing hallucinations colored by average edge length of the triangle, blue - smallest, red - 10-times average edge length of the whole mesh and more (using JET color model), (b) red - triangles with average edge length smaller than 3-times or larger than 6-times average edge length of the whole mesh, green - triangles with average edge length between 3-times and 6-times average edge length of the whole mesh, top ellipse - real surface, bottom ellipses - hallucination. Conclusion: average edge length does not separate real surface triangles from the hallucinated ones.

in the range of one to six millions. The computation time of consistencies and graph-cut based hallucinations removal is approximately minutes.

We have tested our method with $k = 10$. We think that $k = 3$ should be enough, too. The code can be later optimized with respect to that only small amount of 3D points are changing in each iteration. Therefore we would build the triangulation from original points and remember it. Later we would add perturbed points (0.1% of original points in all of our experiments) and update the weights to the triangulation in each iteration. This optimization would cause that the computation time of all iterations should be similar to the computation time of one iteration.

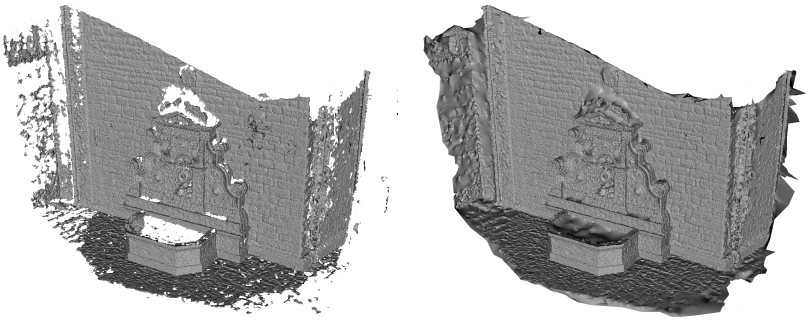
9 Results

Based on the experiments we observed that using the criterial function based on the triangle (or related tetrahedron) property like average edge length, maximal edge length, triangle area and so on, can not be used to remove hallucinated surfaces with sufficient quality. Figure 5 shows the input mesh (including hallucinations) colored by the average edge lengths using JET color model. Red color represents the triangles whose average edge lengths are greater or equal to 10-times average edge length of the whole mesh. Blue color represents the triangles which average edge lengths goes to zero. Areas marked by the ellipses show two different parts of the mesh. Both of them (see the distribution of colors) contain the triangles with the same average edge lengths. But, the top set



(a) (b)

Fig. 6. Castle-P30 data set (top view): (a) 3D surface after removing hallucinations using approach proposed in [7], (b) 3D surface after removing hallucinations using our method.



(a) (b)

Fig. 7. Fountain-P11 data set: (a) 3D surface removing hallucinations using approach proposed in [7], (b) 3D surface after removing hallucinations using our method

of triangles represents real object part and the bottom ones are hallucinated. This example demonstrates that it is impossible to find a threshold which would separate the hallucinated part from the real one in general. We carried out several experiments on this dataset using maximal edge length, triangle area and maximal radius of circumscribed sphere of the related tetrahedron, and all of them produced similar results.

To evaluate the quality of our reconstructions, we present results on data sets from the standard Strecha's [9] evaluation database. We show the result for three different outdoor datasets: Fountain-P11, Castle-P30, Dragon-P114. The first two datasets are Strecha's datasets [9] and the last one is the data set of a dragon's sculpture in Kyoto. Strecha's Fountain-P11 data set contains 11 3072×2048 images. Strecha's Castle-P30 data set contains 30 3072×2048 images. Dragon data set contains 114 1936×1296 images.

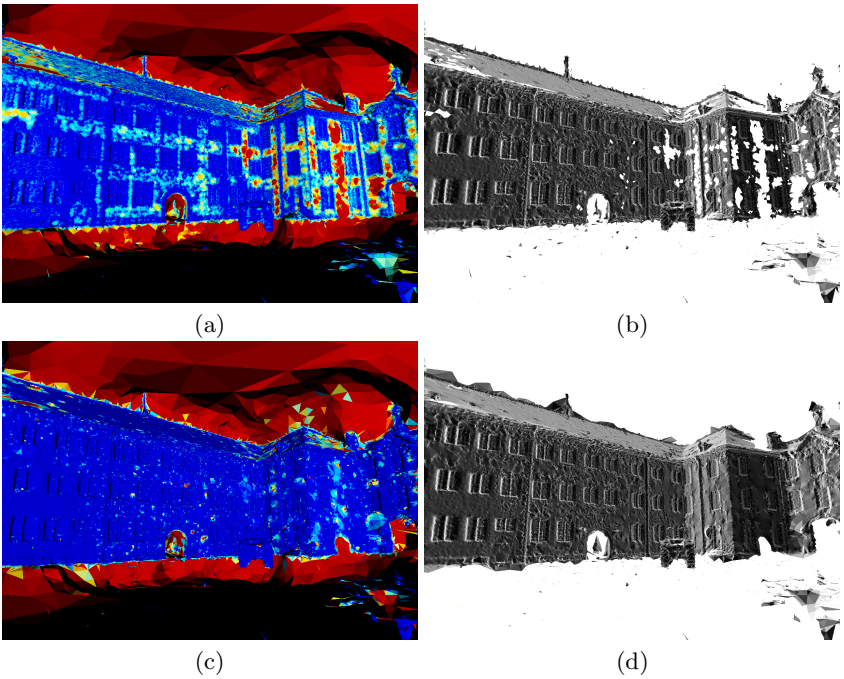


Fig. 8. Castle-P30 data set: (a) 3D surface before removing hallucinations colored by average edge length of the triangle, blue - smallest, red - 10-times average edge length of the whole mesh and more (using JET color model), (b) 3D surface after removing hallucinations using approach proposed in [7], (c) 3D surface before removing hallucinations colored by the perturbation based confidence (blue - smallest confidence, red - largest confidence), (d) 3D surface after removing hallucinations using our method.

Figure 4 shows the evaluation on the Strecha's Fountain-P11 as well as Castle-P30 data sets. See the Strecha's evaluation page for JAN10 results and their comparison. The histograms shows that our reconstructions are more or less on the same level at $2\,\sigma$ and $3\,\sigma$ as the method [1] which uses an additional mesh refinement step. The cumulative histograms shows that our method outperforms all other methods in completeness. In Figure 4 (a) and (b) we are comparing our results with four best methods [1, 7, 20, 21]. For complete results, we refer the reader to the challenge website [22]. This experiment demonstrates that methods based on the opportunistic approach [10] produces complete and accurate results and outperforms the other state-of-the-art methods, and it is therefore important to deal with its negatives which was the goal of this paper.

We made several experiments to demonstrate that our method produces better results than the state-of-the art approach proposed in [7], which solves this problem by removing large triangles, i.e. they discard the triangles which average edge length is greater than six times the average edge length of the whole mesh.

Figures 1 and 5 show the comparison of the results computed using the approach proposed in [7] with the results computed using our method on Dragon-P114 data set. Figures 6 (a) and (c) show that our method can better deal with larger triangles which are on the ground and which cover surfaces captured at oblique angles. Figures 6, 8 and 2 show the comparison of the results computed using the approach proposed in [7] with the results computed using our method on Castle-P30 data set. Figures 6 (a) and (c) show that our method can better preserve large triangles which are in between the windows where is a lack of the texture but which are not hallucinated. Figure 7 shows the comparison of the results computed using the approach proposed in [7] with the results computed using our method on Fountain-P11 data set. It shows that our method can avoid discarding important low-textured parts of the scene.

Figure 9 shows the detailed view of the dragon's head before and after removing hallucinations using our method (untextured and textured). We have used the nearest camera to texture each triangle without texture color unification.

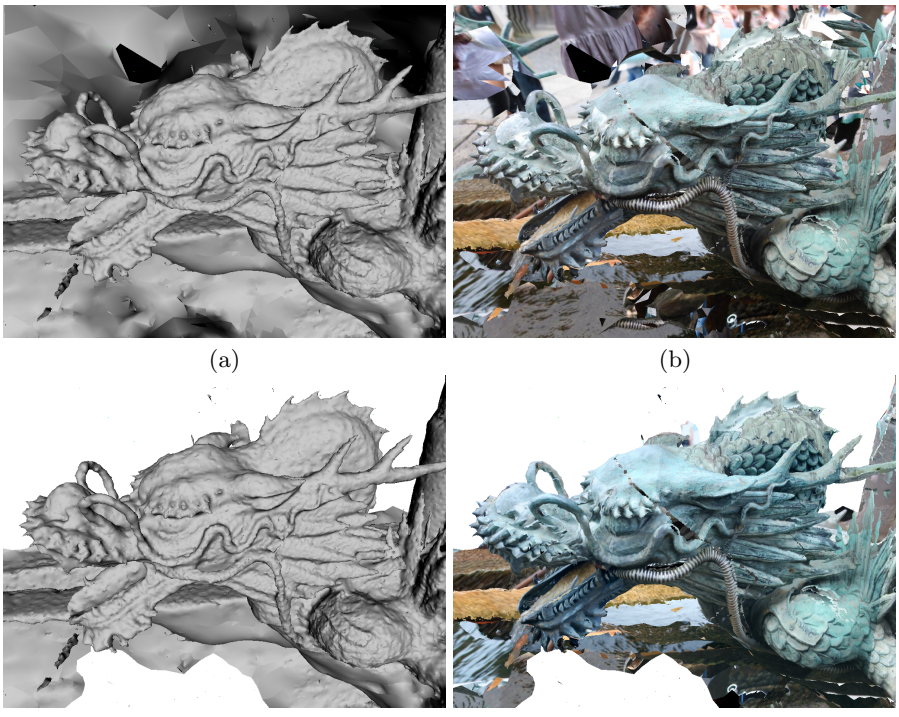


Fig. 9. Detailed view of the dragon's head. (a) 3D surface before removing hallucinations untextured, (b) 3D surface before removing hallucinations textured, (c) 3D surface after removing hallucinations using our method untextured, (d) 3D surface after removing hallucinations using our method textured.

10 Conclusion

In this work we proposed a hallucination-free multi-view stereo method. We demonstrated that the quality of our reconstructions is comparable to the best state-of-the art methods on several benchmark datasets. We have shown experimentally that our method produces more complete reconstructions while removing falsely generated surfaces.

References

1. Vu, H., Keriven, R., Labatut, P., Pons, J.P.: Towards high-resolution large-scale multi-view stereo. In: CVPR (2009)
2. Bradley, D., Boubekeur, T., Heidrich, W.: Accurate multi-view reconstruction using robust binocular stereo and surface meshing. In: CVPR, pp. 1–8 (2008)
3. Zaharescu, A., Boyer, E., Horaud, R.: TransforMesh: A Topology-Adaptive Mesh-Based Approach to Surface Evolution. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007, Part II. LNCS, vol. 4844, pp. 166–175. Springer, Heidelberg (2007)
4. Labatut, P., Pons, J., Keriven, R.: Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In: ICCV, pp. 1–8 (2007)
5. Campbell, N.D.F., Vogiatzis, G., Hernández, C., Cipolla, R.: Using Multiple Hypotheses to Improve Depth-Maps for Multi-View Stereo. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 766–779. Springer, Heidelberg (2008)
6. Goesele, M., Snavely, N., Curless, B., Hoppe, H., Seitz, S.: Multi-view stereo for community photo collections. In: ICCV (2007)
7. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multi-view stereopsis. In: CVPR, pp. 1–8 (2007)
8. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In: CVPR, pp. 519–528 (2006)
9. Strecha, C., von Hansen, W., Van Gool, L., Fua, P., Thoennessen, U.: On benchmarking camera calibration and multi-view stereo for high resolution imagery. In: CVPR, pp. 1–8 (2008)
10. Labatut, P., Pons, J., Keriven, R.: Robust and efficient surface reconstruction from range data. In: Computer Graphics Forum (2009)
11. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: SGP 2006, pp. 61–70 (2006)
12. Ford, L.R., Fulkerson, D.R.: Flows in Networks. Princeton University Press (1962)
13. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Transactions on Pattern Analysis and Machine Intelligence 26, 1124–1137 (2004)
14. Jancosek, M., Shekhovtsov, A., Pajdla, T.: Scalable multi-view stereo. In: 3DIM (2009)
15. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV 60, 91–110 (2004)
16. Wu, C.: SiftGPU, `http://cs.unc.edu/~ccwu/siftgpu`

17. Gallup, D., Frahm, J., Mordohai, P., Yang, Q., Pollefeys, M.: Real-time plane-sweeping stereo with multiple sweeping directions. In: CVPR, pp. 1–8 (2007)
18. Yang, R., Pollefeys, M.: Multi-resolution real-time stereo on commodity graphics hardware. In: CVPR, pp. I: 211–I: 217 (2003)
19. Collins, R.: A space-sweep approach to true multi-image matching. Technical Report UM-CS-1995-101 (1995)
20. Salman, N., Yvinec, M.: Surface reconstruction from multi-view stereo. In: Modeling-3D (2009)
21. Strecha, C., Fransens, R., Van Gool, L.: Wide-baseline stereo from multiple views: a probabilistic account. In: CVPR (2004)
22. Strecha, C.: Evaluation page, `http://cvlab.epfl.ch/~strecha/multiview/denseMVS.html`