Consolidation of Multiple Depth Maps

Irene Reisner-Kollmann*†
*Vienna University of Technology

Stefan Maierhofer†
†VRVis Research Center Vienna

Abstract

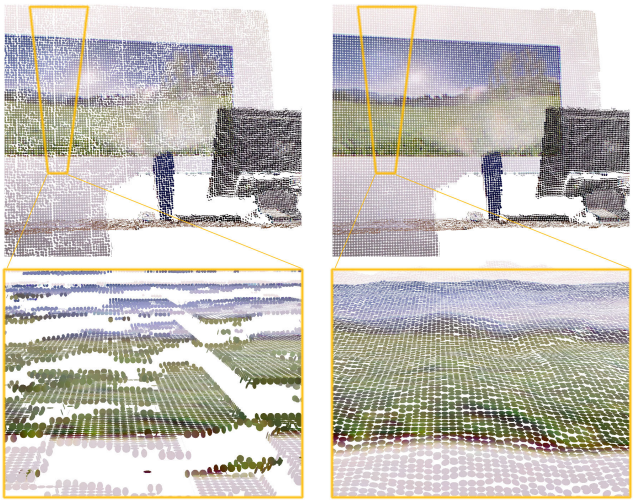
Consolidation of point clouds, including denoising, outlier removal and normal estimation, is an important preprocessing step for surface reconstruction techniques. We present a consolidation framework specialized on point clouds created by multiple frames of a depth camera. An adaptive view-dependent locally optimal projection *operator denoises multiple depth maps while keeping their structure in two-dimensional grids. Depth cameras produce a systematic variation of noise scales along the depth axis. Adapting to different noise scales allows to remove noise in the point cloud and preserve well-defined details at the same time. Our framework provides additional consolidation steps for depth maps like normal estimation and outlier removal. We show how knowledge about the distribution of noise in the input data can be effectively used for improving point clouds.*

1. Introduction

Consumer depth cameras provide a simple and cost-efficient way of generating 3D point clouds from real world scenes. A single frame provides a dense and colored point cloud as seen from one viewpoint. By combining multiple frames it is possible to capture large interior scenes [4]. The resulting point clouds provide a good impression of the captured scene, but generating precise surfaces is challenging due to noise, outliers and registration inaccuracies.

Depth precision of stereo cameras is generally decreasing quadratically with increasing distance to the camera. While the actual error is random noise and cannot be removed automatically, the noise scale can be traced back to camera design and analyzed systematically. If multiple frames of a depth camera are combined, the scene will contain reliable 3D points captured from near viewpoints as well as points with high noise from distant camera locations. While some parts may only contain points with a similar signal-to-noise ratio, there are also parts with a mixture of different noise scales.

Our goal is to improve surface reconstruction from multiple depth maps by including the information about the varying precision of points. In overlapping areas, points with a high confidence should have more impact on the final surface than noisy points. Well defined areas containing mainly confident points should keep all details and thus shouldn't undergo too much smoothing. On the other hand, areas without any confident points have to be smoothed in order to get rid of outliers and noise.



(a) Input point cloud (b) avdLOP

Figure 1. A depth map is cleaned from noise and outliers, while details in well-defined areas are preserved. The monitor in the foreground is hardly affected by the avdLOP operator, because it contains only a small amount of noise. However, the wall in the background is strongly improved with our algorithm as can be seen in the bottom images.

It is hard to achieve these goals with general point processing tools where all points are handled equally. Without any additional information it is not possible to distinguish between noise and real features in the data, especially when there are areas with noise larger than the scale of small, but well defined, details. Point processing techniques often depend on a global neighborhood size. A large value will remove important details in well defined areas, while a small value won't denoise areas with poor signal-to-noise ratio.

In this paper we introduce a variation of the locally optimal projection (LOP) operator [10], where we incorporate

additional information given by depth maps. We explicitly adapt the neighborhood size as well as the importance of points according to their estimated noise scale. Additionally, we modify the LOP operator such that the structure of individual depth maps is preserved.

2. Related work

Huhle et al. [7] presented a non-local denoising algorithm for depth maps, where color information can be optionally included in the filter. Their method works on single depth maps and does not take advantage of information from other viewpoints.

Many different techniques are available for reconstructing surfaces from point-sampled data. A common approach is to create an implicit surface definition from a point set. Well-known algorithms include the projection on tangent planes [5], radial basis functions [2], and Poisson reconstruction [8]. The moving least squares (MLS) approach [9, 1] creates a new point set surface by projecting points onto a local polynomial function and thus removing outliers and noise. These reconstruction techniques implicitly use a smoothness assumption, but variations for maintaining sharp edges are available (e.g. [3]).

2.1. Locally optimal projection (LOP)

The locally optimal projection (LOP) operator [10] creates a cleaned point set from a noisy input point set, possibly containing outliers and non-uniform sampling. In contrast to previously mentioned reconstruction techniques, LOP projects points onto the surface without using a local 2D parameterization. LOP is closely related to the multivariate L_1 median, which is defined as the point minimizing the sum of Euclidean distances to a data set P.

Given an unorganized point set $P = \{p_j\}_{j\in J} \subset \mathbb{R}^3$, LOP projects a set of points $X^0 = \{x_i\}_{i\in I} \subset \mathbb{R}^3$ onto the surface defined by P. Usually, X^0 is a downsampled version of the given point set P and is upsampled to the original density after the projection. Nevertheless, it is possible to project an arbitrary point set onto the surface. LOP is solved by the fixed point of an iterative solution. For each iteration $k = 1, 2, 3, ...,$

$$\alpha_j^i = \frac{\theta(\|\xi_{ij}^k\|)}{\|\xi_{ij}^k\|}, \qquad \beta_{i'}^i = \frac{\theta(\|\delta_{ii'}^k\|)\,|\eta'(\|\delta_{ii'}^k\|)|}{\|\delta_{ii'}^k\|} \tag{1}$$

are defined for each $i \in I$, where $\xi_{ij}^k = x_i^k - p_j$ and $\delta_{ii'}^k = x_i^k - x_{i'}^k$. Then, the projected points $X^{k+1} = \{x_i^{k+1}\}$ at iteration $k+1$ are defined as

$$\sum_{j\in J} p_j \frac{\alpha_j^i}{\sum_{j\in J}\alpha_j^i} + \mu \sum_{i'\in I\setminus\{i\}} (x_i - x_{i'}) \frac{\beta_{i'}^i}{\sum_{i'\in I\setminus\{i\}}\beta_{i'}^i}. \tag{2}$$

The first term attracts points to the object surface while the second term is a repulsive force to keep the points well distributed. The function $\theta(r) = e^{-r^2/(h/4)^2}$ is a rapidly decreasing weight function with a finite support radius of h. $\eta(r)$ is another decreasing function, defined as $1/(r^3)$, penalizing points which come too close to neighboring points.

Huang et al. [6] provide an extension named WLOP (weighted locally optimal projection) for achieving a uniform distribution although the input point cloud is highly non-uniform. Weighted local densities are computed for each point p_i in P and x_i in X^k which measure how densely points are distributed within the neighborhood size. The attraction term is weighted lower for dense areas, while the repulsion term is weighted higher. Additionally, the authors propose a new repulsion function $\eta(r) = -r$, which has a smoother slope and thus penalizes more at larger r.

3. Adaptive view-dependent LOP

The LOP operator is useful for consumer depth cameras, as interior scenes are very diverse and thus the reconstruction should be independent from a local parameterization. It is possible to apply similar adaptations, as described for LOP in this section, to other reconstruction techniques.

Our algorithm modifies the LOP operator in two ways. First, we use confidence values to consider different noise scales in a range scan. Second, we want to keep depth maps and thus adapt points only in their viewing direction. Opposite to the original LOP, it is not possible to distribute arbitrary input points on the surface with our adapted algorithm. Instead it is assumed that the input point cloud P, or a downsampled version of it, is used as starting point X^0 for the iterative LOP algorithm.

3.1. Adapting to different noise scales

The precision of point clouds generated by depth cameras strongly varies with the distance from the sensor. While nearby surfaces are captured quite accurately, more distant objects are represented with a high level of noise. For stereo cameras, the precision approximately decreases quadratically with the distance to the camera. Including this information into the LOP operator can improve the reconstruction quality of both, single depth maps and multiple registered depth maps captured from different viewpoints.

A larger influence neighborhood is required for noisy points as they might be located far from the actual surface. Another problem, especially valid for single depth maps, is that there might not even exist any confident points in a larger neighborhood. In this case we accept that the surface is smoothed by the enlarged neighborhood size, but the noise is effectively removed.

In the beginning, the neighborhood size is adapted for each point x_i corresponding to its noise scale, which we assume to increase quadratically for increasing depth values. A global neighborhood size h is defined for a camera distance of 1 meter and is adjusted for each projected point x_i

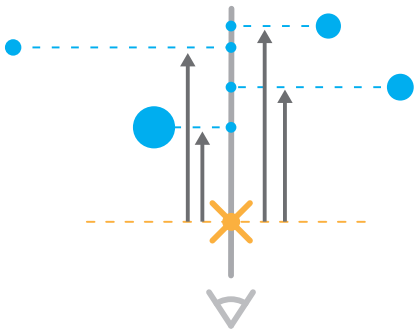


Figure 2. View-dependent projection: The point at the orange cross (x_i) is projected onto the unknown surface defined by the blue points (p_j). The point can only move along its viewing direction in order to maintain the depth map. The point sizes visualize the weighting function $\theta(r)$ based on Euclidean distances.

to $h_i = hz_i^2$ where z_i denotes the distance of point x_i to its camera plane.

When a point is projected near to confident points during the iterative algorithm, the support radius has to be adapted to the increased confidence value. A new neighborhood size h_i^{k+1} is computed in each iteration (Eq. 3), based on the depth values within the local neighborhood.

$$h_i^{k+1} = h \left(\frac{\sum_{j \in J} z_j \, \theta(\left\| \xi_{ij}^k \right\|, h_i^k)}{\sum_{j \in J} \theta(\left\| \xi_{ij}^k \right\|, h_i^k)} \right)^2 \tag{3}$$

Additionally, points should be attracted more by surface points with high precision than by noisy points. Therefore, we weight surface points p_j with the function $\rho(d) = 1/d$ applied on the points' depth values z_j.

Equation 4 shows the modified computation of α_j^i for adapting to different noise scales. The modified weight function $\theta(r, d)$ has the adapted support radius as additional parameter.

$$\alpha_j^i = \frac{\theta(\left\| \xi_{ij}^k \right\|, h_i) \, \rho(z_j)}{\left\| \xi_{ij}^k \right\|} \tag{4}$$

3.2. View-dependent projection

There are several reasons for keeping point-sampled data organized in a set of depth maps. The alignment of points in a two-dimensional grid can be favorable for fast computations of surface properties such as normal estimation. In case of consumer depth cameras, some applications may benefit from the coupling between depth maps and color images, or the temporal order of capturing is needed.

In order to keep depth maps as seen from the original viewpoint, translations along the viewing directions are the only possible adjustments of the point cloud. Instead of summing over surface points, we first project the points onto the viewing ray of the currently projected point and sum over displacements relative to the original point along this ray (see Figure 2). The weighting function $\theta(r)$ is still applied to Euclidean distances between x_i and p_j. This ensures that points located further away from the viewing ray are weighted lower than nearby points.

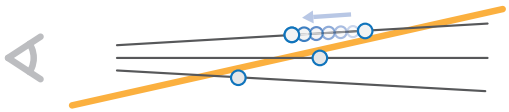


Figure 3. On surfaces with a steep angle to the viewing direction points might get attracted to point clusters away from the real surface. The orange line represents the real surface, the blue points are sample points slightly off the surface. The point on the top view ray would be projected away from the surface if no repulsion term is applied.

The first term in Equation 2 now becomes

$$x_i^k + v_i^k \sum_{j \in J} (\xi_{ij}^k \cdot v_i^k) \frac{\alpha_j^i}{\sum_{j \in J} \alpha_j^i}, \tag{5}$$

where v_i^k is the normalized view vector from the camera center to point x_i^k.

3.3. Repulsion term

The second term in Equation 2 strives for equally distributing points on the surface by putting a penalty on points located very near to each other. In our case, it is only important that points within one frame are well distributed, as the data is organized in individual depth maps rather than in a global, evenly spaced, point set.

A good distribution is implicitly given for points on surfaces oriented parallel to the camera plane, as points can only move along the viewing direction. Applying the repulsion term on such surfaces would even have a negative effect on the convergence. As the points cannot move within the plane, the only way for moving nearby points apart would point away from the actual surface.

On the other hand, a regularization term is needed for surfaces which are nearly parallel to the viewing direction. Figure 3 shows how points might get attracted to local clusters and a fair point distribution can improve the reconstruction result.

These observations lead to the following modifications of the repulsion term:

$$\mu(1 - |v_i \cdot n_i|) \sum_{i' \in I(i) \setminus \{i\}} -(\delta_{ii'}^k \cdot v_i) \frac{\beta_{i'}^i}{\sum_{i' \in I(i) \setminus \{i\}} \beta_{i'}^i} \tag{6}$$

$$\beta_{i'}^i = \frac{\theta(\left\| \delta_{ii'}^k \right\|, hz_i) \eta'(\left\| \delta_{ii'}^k \right\|)}{\left\| \delta_{ii'}^k \right\|} \tag{7}$$

The set $I(i)$ contains all points captured from the same viewpoint as point x_i. The normalized view direction v_i

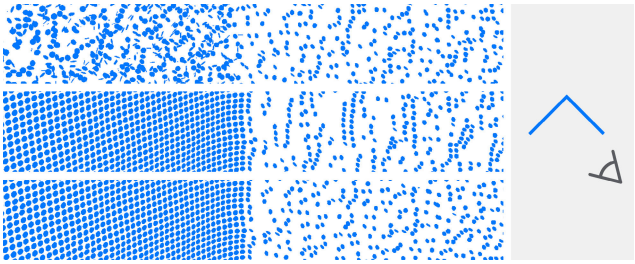


Figure 4. From top to bottom: original point set, projection without repulsion, projection with repulsion $\mu = 0.45$. The left part of the scene is parallel to the camera plane whereas the right part is orthogonal to the camera plane. Including the repulsion term improves the projection at the right part of the scene, while there is hardly any effect on the left part.

points from the camera center to point x_i, n_i denotes the normal at point x_i. The weight function θ uses again a variable support radius. The neighborhood size is here adjusted to the depth of a point to account for varying distances between neighboring pixels due to perspective projection. The effect of our new repulsion term is shown in Figure 4.

4. Further consolidation steps

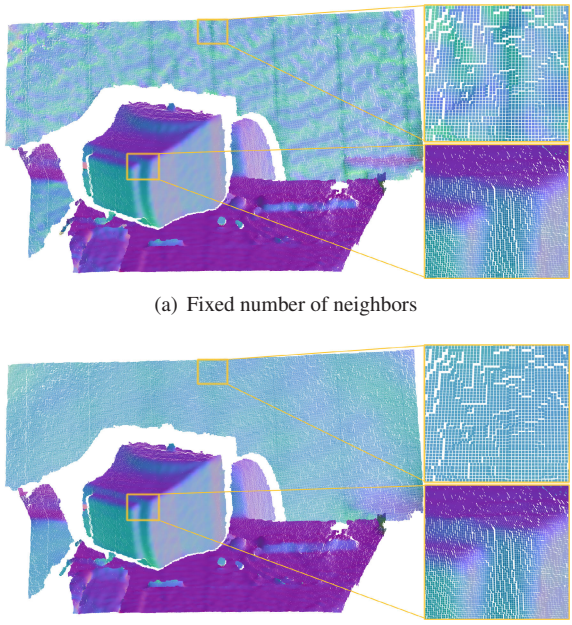
Consolidating a point set, or a set of depth maps, is an important preprocessing step for other actions such as rendering or further reconstruction steps. In this section we describe additional tasks besides the projection operator, including normal estimation, outlier removal, and resampling.

4.1. Normal estimation

Reliable surface normals are needed for the modified repulsion term in our projection operator (see Section 3.3). Further, they are an important property for many other point processing tasks.

We compute a surface normal n_i by fitting a plane with a least-squares approach to the neighborhood of point x_i within radius r[11]. The neighborhood size r is a crucial parameter, as a high value will smooth edges too much while a low value will lead to unreliable normals. Similar to our adaptive projection operator we use the knowledge about the noise distribution in depth maps. Each point uses an individual neighborhood size $r_i = rz_i^2$, where z_i is the point's depth value and r defines the neighborhood size for a depth of one meter. This is the same modification as for the adaptive support radius in Section 3.1 and again reflects the quadratic attenuation of precision with increasing depth. During the avdLOP algorithm, the neighborhood size for estimating normals is recomputed in every iteration according to Eq. 3 in order to take increasing confidence values into account.

Consistently oriented normals are easily created by ensuring that normals point to the side of the tangent plane where the camera is located. An example of our normal estimation can be seen in Figure 5.



(a) Fixed number of neighbors

(b) Varying neighborhood size

Figure 5. Normal estimation (a) with 200 nearest neighbors and (b) with a varying neighborhood size (15 mm at one meter distance). Foreground objects, such as the monitor, get well defined normals with both methods. The normals for the flat wall in the background are much stronger affected by noise when a fixed number of neighbors is used.

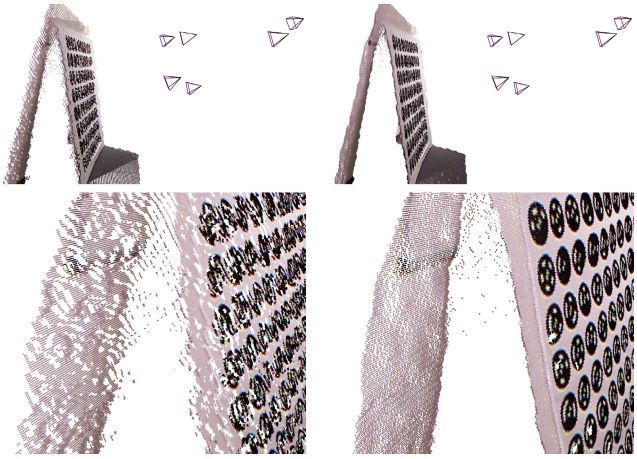
4.2. Depth map resampling

Applying the LOP operator on all depth values might be too costly for large datasets. In many cases it is sufficient to apply the projection on a subset of points and resample the depth maps in between. We interpolate between neighboring points in the image space, but weight them according to their Euclidean distances in 3D to ensure that depth values are not interpolated across large depth discontinuities.

4.3. Outlier removal

The view-dependent projection operator adjusts 3D points only along their viewing direction and doesn't affect points far from any other points. While noise is effectively removed along the viewing direction, outliers and deviations parallel to the camera plane might still be present.

Given that all parts of a scene are captured by multiple frames, outliers do not have any close points in other frames. Within their own frame, outliers appear at a boundary, i.e. neighboring points are occluded or a large depth discontinuity arises next to the point. This leads to a simple approach for outlier removal, where iteratively boundary points, that do not appear in other frames, are deleted.



(a) Input point set (b) avdLOP ($h = 0.03$, $\mu = 0$)

Figure 6. A planar surface, captured by six frames, is reconstructed with avdLOP after five iterations. The bottom row shows close-ups of the most distant frame which contains the largest noise scales.

Table 1. CPU runtime for applying the avdLOP operator. F-No: number of frames; P-No: number of original points; X-No: number of projected points; T0: time for $\mu = 0$; TR: time for $\mu = 0.2$. Times are given for all iterations in seconds. All examples were run on an Intel Core i7, 2.67 GHz CPU with 12 GB RAM.

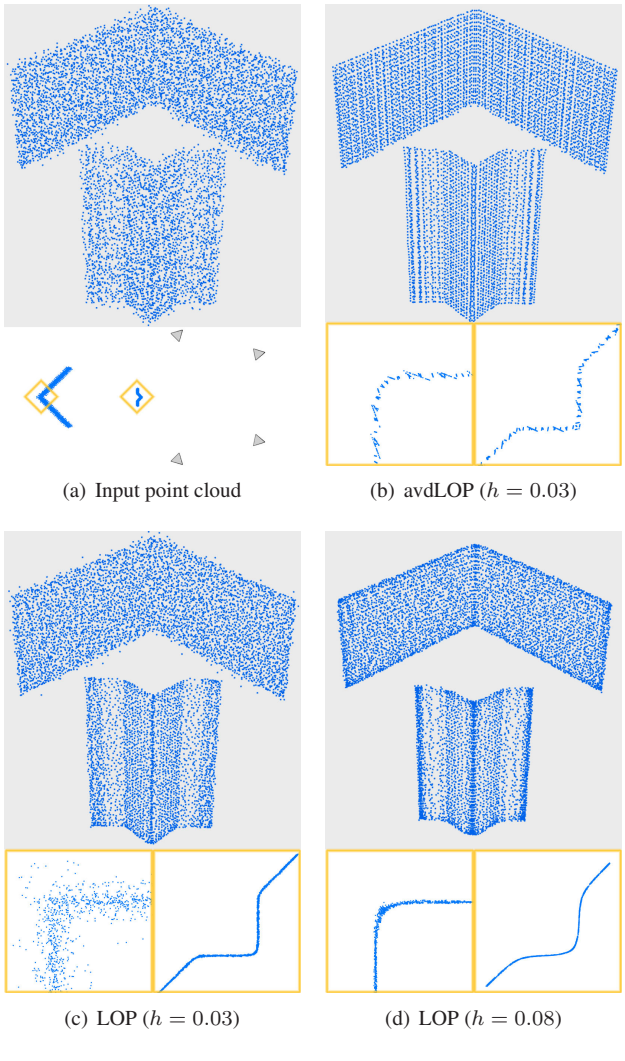
	F-No	P-No	X-No	T0	TR
Figure 1, 8	11	2849989	178182	488	-
Figure 6	6	390624	97590	32	103
Figure 7	4	126804	7919	16	84
Figure 9	3	164444	41125	16	128

5. Results

Figures 1, 6, 8 and 9 show the results of applying avdLOP on depth maps captured with Microsoft's Kinect. For each data set, multiple frames from the freely moved camera have been registered with a combination of SIFT-features and ICP. The resulting point clouds contain noise, slight misalignments, deviations due to missing calibration and outliers due to wrong depth calculations. All distances are given in meters.

Figure 7 shows a synthetic example where a simple scene has been captured by four virtual depth cameras. The synthetically generated depth maps do not contain any artifacts besides the quadratically increasing noise.

The algorithm usually converges after a few iterations. Figure 10 illustrates the convergence behavior for two data sets. Computation time for one iteration depends on the number of points and the parameter h. Applying the projection with $\mu = 0$ reduces the computation time, since estimating normals is only necessary for computing the repulsion term. Timings for the example data sets are provided in Table 1.



(a) Input point cloud (b) avdLOP ($h = 0.03$)

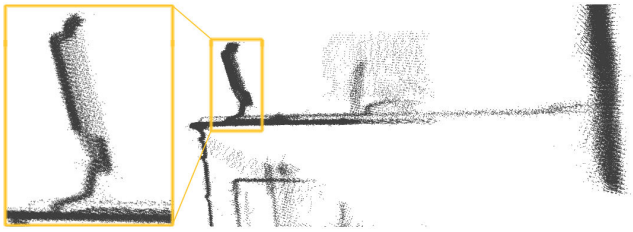
(c) LOP ($h = 0.03$) (d) LOP ($h = 0.08$)

(e) Front part (f) Rear part

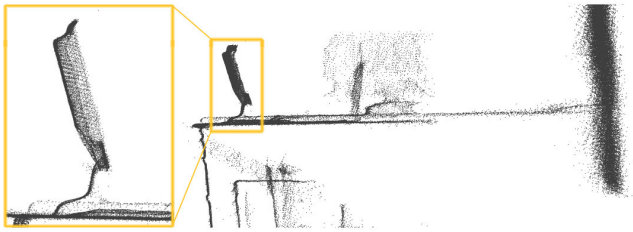
Figure 7. avdLOP vs. LOP on virtually generated depth maps, after 10 iterations with $\mu = 0.2$. (b) avdLOP removes the noise on the surface in the background and keeps the edge in the foreground. General LOP is either not able to denoise distant surfaces (c) or overly smooths details in near surfaces (d). This can also be seen in charts (e) and (f) which illustrate the accuracy, i.e. the average distance between filtered sample points and the original scene rectangles. Only avdLOP is able to provide good results for the front as well as the rear part of the scene.

6. Conclusions

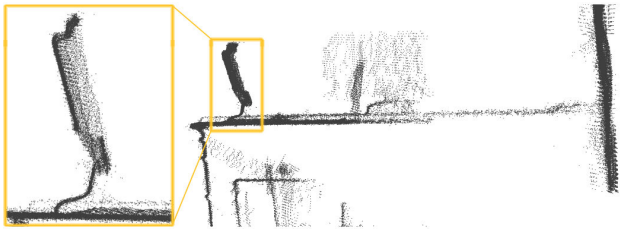
We have presented a modified version of the locally optimal projection (LOP) operator for specifically addressing



(a) Input point set



(b) LOP (5 iterations, $h = 0.06$, $\mu = 0.4$)



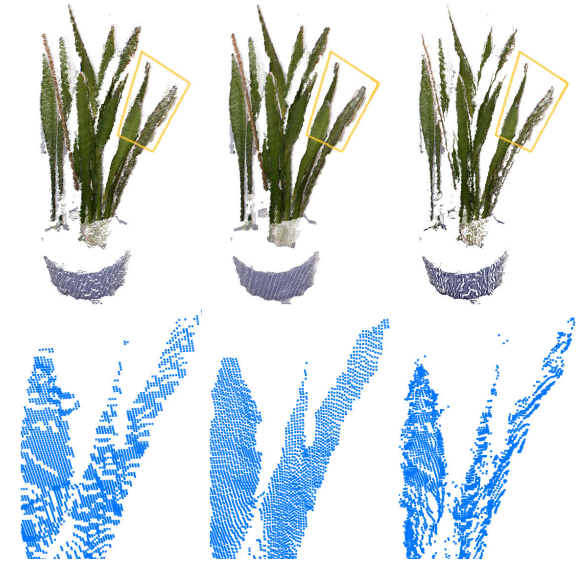
(c) avdLOP (5 iterations, $h = 0.015$, $\mu = 0$)

Figure 8. avdLOP vs. LOP on real data set. The original LOP operator (b) provides good results for the monitor in the foreground, but the same configuration fails to remove the noise on the background wall, seen at the right of the image. In contrast, the adaptive LOP (c) provides good results over the whole scene. A higher support radius h for the original LOP would overly smooth foreground objects, since the effect in these areas is already stronger than with avdLOP. Details of this data set can be seen in Figure 1.

point-sampled data from depth cameras. We exploit the fact that noise is quadratically increasing with the distance to the camera. Adapting to different noise scales allows to keep details in well-defined areas, while areas with large noise values have to be smoothed. Furthermore, points with poor precision can be projected onto accurate points available from other depth maps. Additionally, we keep the structure of input depth maps by restricting the projection onto viewing directions.

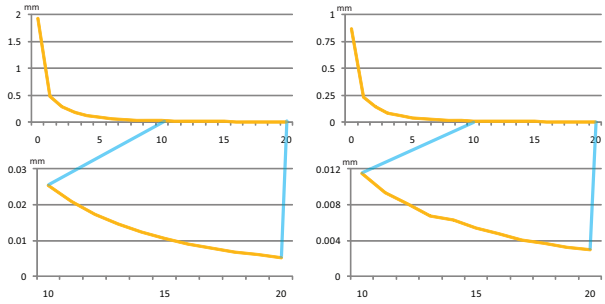
In the future, the algorithm might be further improved by incorporating additional knowledge from an extensive precision analysis of depth cameras. This might reveal additional parameters for the distribution of noise scale besides depth.

Our consolidation framework for depth maps provides effective improvements on the input data and thus is a valuable pre-processing tool for further reconstruction steps. It



(a) Original (b) avdLOP (c) LOP

Figure 9. Comparing avdLOP and LOP on thin and non-planar structures, captured by three frames. The right images show a close-up containing only the points from the most distant frames. The avdLOP-operator ($h = 0.03$, $\mu = 0$, 10 iterations) has a much lower shrinkage effect than the original LOP ($h = 0.03$, $\mu = 0.45$, 10 iterations) by keeping points in their viewing ray.



(a) For dataset in Fig. 6 (b) For dataset in Fig. 9

Figure 10. avdLOP has a smooth convergence behavior, which is visualized by plots of distances $\left\|X^{k+1} - X^k\right\| / |X|$. The bottom images show details for a subset of iterations.

allows to include also distant and noisy points into a reconstruction system and hence the necessary amount of frames can be reduced.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, pages 3–15, 2003.

[2] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction

and representation of 3d objects with radial basis functions. In *Proc. of ACM SIGGRAPH*, pages 67–76, 2001.

[3] S. Fleishman, D. Cohen-Or, and C. T. Silva. Robust moving least-squares fitting with sharp features. In *Proc. of ACM SIGGRAPH*, pages 544–552, 2005.

[4] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *International Symposium on Experimental Robotics*, 2010.

[5] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proc. of ACM SIGGRAPH*, pages 71–78, 1992.

[6] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. In *Proc. of ACM SIGGRAPH Asia*, 2009.

[7] B. Huhle, T. Schairer, P. Jenke, and W. Strasser. Robust non-local denoising of colored depth data. In *IEEE CVPR Workshop on Time of Flight Camera based Computer Vision*, pages 1–7, june 2008.

[8] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, 2006.

[9] D. Levin. Mesh-independent surface interpolation. *Geometric Modeling for Scientific Visualization*, pages 37–49, 2003.

[10] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer. Parameterization-free projection for geometry reconstruction. In *Proc. of ACM SIGGRAPH*, pages 1–6, 2007.

[11] M. Pauly, M. Gross, and L. P. Kobbelt. Efficient simplification of point-sampled surfaces. In *IEEE Visualization 2002*, pages 163–170, 2002.