

Incremental Online Reconstruction of Locally Quadric Surfaces

Josua Bloeb¹ ^a and Dominik Henrich¹ ^b

¹University of Bayreuth

Chair for Applied Computer Science III

(Robotics and Embedded Systems)

Universitaetsstrasse 30, 95447 Bayreuth, Germany

{josua.bloess, dominik.henrich}@uni-bayreuth.de

Keywords: Reconstruction, Segmentation, Point cloud, Surface fitting, Quadrics, Online computation

Abstract: Representing surfaces of a digital 3D model using high level geometric information is key to lots of geometric processing and other use cases. In order to obtain these 3D models, various scanning methods have been proposed. We contribute a method for incremental reconstruction of surfaces from a series of point clouds. For this, we use a robust over-segmentation technique on a point cloud and build a memory efficient graph structure upon it. Over time, we expand this graph structure as global representation. We also propose a fast fitting algorithm for quadric surface patches to the graph structure. We validate the overall performance in our experiments.


1 INTRODUCTION


Digital reconstruction of real world objects from point clouds is an avid and ever evolving field of research. Some of its various use cases are grasp planning in robotics (Makhal et al., 2018), manufacturing (Vempati et al., 2018) or creation of models for entertainment (Beraldin et al., 2005). The process of digital reconstruction is a challenging task and requires special hardware and specifically trained personal. In order for the result to be useful in an industrial environment, it must be compatible with established CAD concepts and software. This in turn requires the geometry of the digital reconstruction to be represented by high level information like geometrical primitives or Splines in contrast to low abstraction representations like triangle meshes. The process of CAD reconstruction usually involves two steps: data acquisition and a computationally expensive analysis for high level geometric features (Buonamici et al., 2018). Due to this separation, it is important that the data acquisition is done thoroughly and flawlessly. Otherwise, it can take multiple iterations of data acquisition and analysis until a sufficient result is achieved. An alternative approach is an incremental reconstruction where the analysis of point clouds is done online during data acquisition, e.g. (Newcombe et al., 2011).

With incremental reconstruction, a user can react to the online computed intermediate result and adapt the data acquisition accordingly, e.g. by re-capturing poorly scanned areas of the object. However, existing approaches for online reconstruction usually produce results with low abstraction like triangle meshes or approximated signed distance functions. Few approaches have been proposed for incremental CAD reconstruction. These are limited in the range of detectable geometric shapes geometric primitives, e.g. (Denker et al., 2013).

For incremental CAD reconstruction, two important aspects have to be considered: 1. Depth images must be accumulated in an incrementally expandable data structure. 2. Geometric features must be extracted from this data structure online. Both aspects must enable a responsive overall system. We consider 1 Hz a sufficient frame rate as motivated by a system with similar premises (Sand, 2019). In this paper, we consider incremental CAD reconstruction of quadric surfaces and contribute to both of the aforementioned aspects: 1. A memory-efficient expandable segment graph of over-segmented point clouds. 2. A region-merging algorithm that allows fast quadric fitting on this graph.

To our best knowledge, this is the first approach to combine reconstruction of general quadric surface from a series of point clouds with a computational efficiency that allows an incremental online execution.

^a  <https://orcid.org/0000-0001-8636-3168>

^b  <https://orcid.org/0000-0003-0250-2728>

2 RELATED WORK

An extensive survey of techniques for reconstruction of shapes from point clouds is given in (Kaiser et al., 2019). According to this survey, *RANSAC* based approaches promise computational efficiency and robustness to outliers.

RANSAC (Random Sample Consensus) is a technique where a parameterized model of a shape is fitted to a point cloud. A small number of points is randomly drawn from the point cloud so that all model parameters can be determined. This is repeated until a shape is found that fits the point cloud with a high inlier ratio (Fischler and Bolles, 1981). The *RANSAC* concept has the advantage of being robust towards outliers and also being time efficient for the task of finding the best fitting shape for a given point cloud. However, if the points for each shape have to be determined first, *RANSAC* becomes more computationally expensive (Gotardo et al., 2003). Hierarchical sampling and lazy cost evaluation can be used to speed up the process significantly, still the computation time exceeds multiple seconds (Schnabel et al., 2007). For the planar case, the algorithm was adapted to parallelization on GPUs, which speeds up the computation to a frame rate of 7 Hz (Alehdaghi et al., 2015). In *RANSAC*, prior defined models limit the reconstruction to shapes of these specific models (Oesau et al., 2016). Some works have extended *RANSAC* for general quadrics (Frahm and Pollefeys, 2006). Point normals have been used as additional constraint to lower the number of required points and thus increasing the computational performance (Birdal et al., 2019).

Other techniques find the parameters of a shape model using fitting methods from linear algebra. A key requirement to this approach is that each point can be embedded into a space where the model parameters have linear influence on a cost function. Then solving an Eigenvalue problem of the scatter matrix of the points, after said embedding, yields an algebraic solution (Taubin, 1991), (Lukács et al., 1998). This embedding is possible for quadrics (Andrews and Séquin, 2013). We explain this in more detail in Section 3.3.1. These techniques can be enhanced with knowledge about prior probabilities in order to improve the fitting result when only small point clouds are available (Beale et al., 2016). The normals of each point can be used to increase the robustness of the fit (Tasdizen et al., 1999). Strict type constraints can be applied to the problem to force the result to be some desired type of quadric (Andrews and Séquin, 2013).

Segmentation of point cloud is a key component of surface reconstruction. Segmentation is the procedure of partitioning the points from a point cloud

into segments with some criterion of homogeneity. This criterion can be based on some prior knowledge about object shapes so that each segment corresponds to a real-world object (Kim et al., 2012). For general scene reconstruction, this prior information is not available, and techniques that are based on point feature similarity are more suitable for segmentation. A set of point features is proposed with the point feature histogram (Rusu et al., 2009). Point feature based segmentation techniques produce a locally correct segmentation since point features are computed from the local neighbourhood of a point. Smoothness can be used as local homogeneity criterion to compute an over-segmentation of a point cloud (Papon et al., 2013). In order to obtain a final segmentation from this over-segmentation, a homogeneity criterion between segments is required. Principal curvatures and normal similarity of segments can be used to merge them (Arbeiter et al., 2014). Another possibility is a global criterion using a priori knowledge about shapes occurring in the scene. For many approaches this a priori knowledge is the assumption that a scene consists of a small set of parameterizable primitives (Vanco and Brunnett, 2002). We take a similar approach, but we assume general quadrics and use an according model to impose our homogeneity criterion.

The typical process for reconstruction from point clouds consists of a phase of data acquisition followed by segmentation and model extraction (Buonamici et al., 2018). This comes with a computational load of processing all the acquired data at once, which makes these systems unsuitable for online applications. An alternative approach is to use volumetric representations where point clouds refine a signed distance function (SDF). The SDF is defined across the whole volume of the scene and implicitly defines a surface at its zero-level. It can be represented by a voxel-grid of function values (Newcombe et al., 2011). Hence, memory consumption limits the volume of reconstruction for these techniques. Using GPU-accelerated computations, these techniques reach high frame rates of more than 40 Hz. More sophisticated representations of the SDF, like octrees, have been used to lower memory consumption (Whelan et al., 2012).

For limited sets of primitives, online, incremental reconstruction concepts have been proposed. Planar boundary representations can be built incrementally from the data of a hand-held device with 2 Hz (Sand, 2019). A spatial partitioning and aggregation of point features can be used to reconstruct planes, cylinder and spheres (Denker et al., 2013).

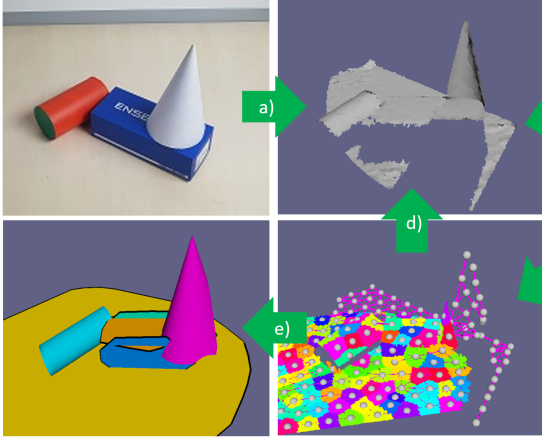


Figure 1: A schematic overview of our approach showing control flow in green arrows and intermediate results.

3 CONCEPT

In this section, we present our approach towards incremental reconstruction of quadric surfaces.

Given is a series of point clouds (P^1, P^2, \dots) . Each P^t is a set of points. For a point p , its position is denoted as $\text{POS}(p) \in \mathbb{R}^3$. We assume that the extrinsic parameters of the depth sensor are known, e.g. by using a robot manipulator. Our goal is to find a set of quadrics $\{q_1, q_2, \dots\}$ that reconstructs the surface captured by (P^1, P^2, \dots) with a low geometrical error. A *quadric* q is a quadratic polynomial

$$q(x, y, z) := C_q^T \theta(x, y, z). \quad (1)$$

with

$$C_q = (c_{xx}, \dots, c_1)^T \in \mathbb{R}^{10} \quad (2)$$

$$\theta(x, y, z) := (x^2, y^2, z^2, xy, xz, yz, x, y, z, 1) \quad (3)$$

Each quadric q defines a surface

$$\mathcal{S}_q := \{(x, y, z) \in \mathbb{R}^3 \mid q(x, y, z) = 0\} \quad (4)$$

An overview of our approach is given in Figure 1. These are the individual steps, following sections explain them in more detail:

- a) a point cloud is recorded
- b) build local segment graph from over-segmentation
- c) expand a global segment graph using the local segment graph
- d) incrementally accumulate point clouds in global segment graph
- e) region-merging fits quadric surfaces

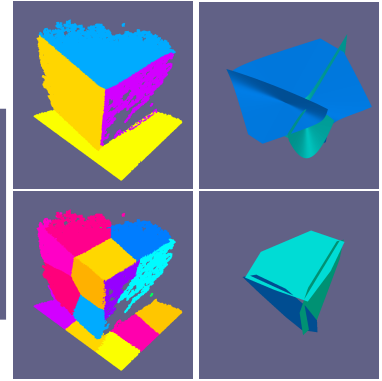


Figure 2: Segmen- Figure 3: Fitting tation (top) and an without (top) and over-segmentation with (bottom) normal constraints.

3.1 Over-Segmentation

Given a point cloud P , let $S^* = \{P_0^*, P_1^*, \dots, P_i^* \subseteq P\}$ be the segmentation of P where each segment P_i^* can be fitted by one quadric surface. The direct computation of S^* is a hard problem. In this work, we reduce the complexity of this problem by first computing an over-segmentation of S^* . An *over-segmentation* $S = \{P_0, P_1, \dots\}$ of S^* is a segmentation of P where

$$\forall P_i \in S \exists P_j^* \in S^* : P_i \subseteq P_j^*. \quad (5)$$

An example of over-segmentation is visualized in Figure 2.

We use *supervoxel clustering* (Papon et al., 2013) to compute S . The algorithm selects seed points in P using a uniform grid (parameter s_{size} governs the cell size of the grid). Clusters are formed by assigning each point to the respectively nearest seed using a metric that considers spatial distance and similarity of the estimated surface normal in a point. Afterwards, the seed of each cluster is updated to the mean position and normal of all points in the cluster. These steps are repeated until the clusters converge, five iterations are suggested (Papon et al., 2013).

3.2 Segment Graph

We define a *segment graph* $G = (V, E)$. Vertices V are segments from an over-segmentation of point clouds and edges E indicate whether two segments are spatially neighbored.

For each point cloud P^t in a series (P^1, P^2, \dots) , we build a *local* segment graph G_t^l using an over-segmentation of P^t . We incrementally build a *global* segment graph G_g^t by merging G_g^{t-1} and G_t^l .

We start with $G_g^0 = (\emptyset, \emptyset)$. An over-segmentation S of P^t is computed according to Section 3.1, G_t^l is

computed from S . For $t = 1$, we set $G_g^1 := G_l^1$. For $t > 1$, over-segmentation of P^t is modified:

- the nearest point in P^t to the center of each segment from G_g^{t-1} is selected as seed. Points selected by the uniform grid are only used as seeds, if there is not already another seed withing distance s_{size}
- only centers of clusters not initialized by the uniform grid are updated. Centers of clusters initialized by segments from G_g^{t-1} are static.

Effects of these modifications are shown in Figure 4.

The incremental update is done by merging $G_g^{t-1} = (V_g^{t-1}, E_g^{t-1})$ and $G_l^t = (V_l^t, E_l^t)$ to $G_g^t = (V_g^t, E_g^t)$:

We denote V_U the set of all segments in G_g^{t-1} that initialized a cluster in the over-segmentation of P^t . For all $P_i \in V_U$, we denote $\alpha(P_i) \in V_l^t$ the segment whose seed center was used to select the seed of P_i . First, segments in V_U are updated with the respective points from V_l^t :

$$V_U^* := \{P_i \cup \alpha(P_i) | P_i \in V_U\} \quad (6)$$

Segments of the next global segment graph are formed by

$$V_g^t := V_U^* \cup (V_g^{t-1} \setminus V_U) \cup (V_l^t \setminus \alpha(V_U)) \quad (7)$$

Edges E_g^t are the union of E_l^t and E_g^{t-1} considering segments P_i and $\alpha(P_i)$ as equivalent.

3.3 Fitting

A region-merging algorithm is used on the global segment graph G_g^t with a homogeneity criterion based on the Taubin Fit (Taubin, 1991).

3.3.1 Taubin Fit

Given a set of points P , the Taubin Fit (Taubin, 1991) can be used to compute a quadric q_P with minimal average approximated squared distance of the points in P to S_{q_P} . The fitting quadric q_P minimizes

$$d(q, P) := \frac{\sum_{p \in P} q(\text{POS}(p))^2}{\sum_{p \in P} \|\nabla q(\text{POS}(p))\|^2} = \frac{C_q^T M_P C_q}{C_q^T N_P C_q} \quad (8)$$

for q with

$$M_P := \sum_{p \in P} \theta(\text{POS}(p)) \theta(\text{POS}(p))^T \quad (9)$$

$$N_P := \sum_{p \in P} \nabla \theta(\text{POS}(p)) \nabla \theta(\text{POS}(p))^T \quad (10)$$

Finding q_P can then be stated as the generalized Eigenvalue problem

$$q_P = \arg \min_q \frac{C_q^T M_P C_q}{C_q^T N_P C_q} \quad (11)$$

From Equation 8, for two point sets P_a, P_b , the Taubin Fit of $P_a \cup P_b$ can be stated as

$$d(q, P_a \cup P_b) = \frac{C_q^T (M_{P_a} + M_{P_b}) C_q}{C_q^T (N_{P_a} + N_{P_b}) C_q} \quad (12)$$

This means that it is sufficient to store M_P and N_P for points P in a segment of the segment graph. No explicit representation of P is required. In this representation, the union of two point sets $P_a \cup P_b$ is computed as:

$$\begin{aligned} M_{P_a \cup P_b} &= M_{P_a} + M_{P_b} \\ N_{P_a \cup P_b} &= N_{P_a} + N_{P_b} \end{aligned} \quad (13)$$

3.3.2 Region Merging

Since a segment graph $G = (V, E)$ represents an over-segmentation, segments have to be merged to use the Taubin Fit for quadric fitting. We define a homogeneity criterion between sets of segments, i.e. *regions*, $R_a \subseteq V$ and $R_b \subseteq V$:

$$\begin{aligned} &\text{HOM}(R_a, R_b) \\ &:= \max \left\{ d(q_{\tilde{P}_a \cup \tilde{P}_b}, \tilde{P}_a), d(q_{\tilde{P}_a \cup \tilde{P}_b}, \tilde{P}_b) \right\} \\ &\quad \text{with} \end{aligned} \quad (14)$$

$$\tilde{P}_a := \bigcup_{P_i \in R_a} P_i \text{ and } \tilde{P}_b := \bigcup_{P_i \in R_b} P_i.$$

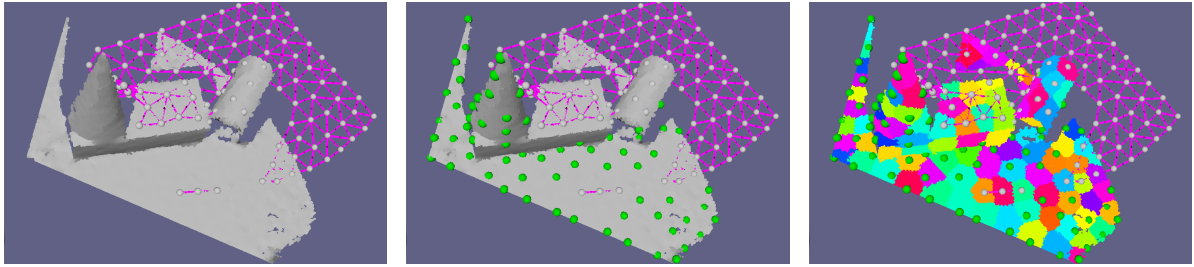
Region merging proceeds as follows: We initialize a set of 1-segment regions by $R \leftarrow \{\{P_i\} | P_i \in V\}$. Then, we iteratively unify the pair of neighboring regions R_a and R_b that minimizes $\text{HOM}(R_a, R_b)$ as long as this value does not exceed a user-defined threshold t_{err} .

Finally, we use the Taubin Fit to find the best fitting general quadric for each region $R_i \in R$. We also use constrained Taubin Fit (Andrews and Séquin, 2013) to find the best fitting plane for R_i . As long as the Taubin error for the plane is below t_{err} , we prefer the plane fit over the general fit.

Even though, quadrics in general position have a smooth surface, an edge of e.g. a cube can be approximated with arbitrary precision using a quadric. Our algorithm often fits a common quadric to two sides of an edge. We put an additional constraint on the edges E . We only keep edges between vertices whose average point normals are roughly the same. We use a threshold of 0.7 for the dot product of the normals of two segments. The effect is depicted in Figure 3.

4 Experiments

We evaluate our concept in terms of accuracy and computational efficiency. For this, we use public and



(a) G_g^{-1} and P^l before segmentation (b) seeds for over-segmentation (c) result of over-segmentation

Figure 4: Segments from seeds selected by vertices of G_g^{-1} are centered around these vertices. Segments initialized by additional seeds (green) move during supervoxel clustering.

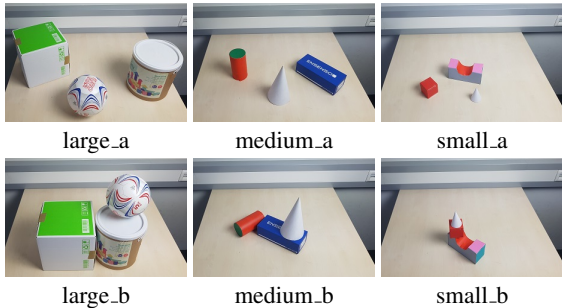


Figure 5: Images of the scenes used for our data set

custom-made data sets (Section 4.1) and compare our results to existing approaches (Section 4.2). We also analyze the influence of the parameters of our approach.

4.1 Data Sets and Evaluation Methods

Our custom-made data set comprises six scenes (see Figure 5). For each scene, we captured 15 point clouds from different poses using an Ensenso N10 depth sensor mounted on a robotic manipulator for extrinsic calibration. Each point cloud consists of $752 \times 480 = 360,960$ points. We also evaluate our approach for single view reconstruction on the ITODD data set (Drost et al., 2017) as well as a subset of the redwood data set (Choi et al., 2016). We build a ground truth set of quadrics for each data set by manual segmentation and Taubin Fit. We refer to (Andrews and Séquin, 2013) for an analysis of the accuracy of Taubin Fit itself. In order to compare a set of ground truth quadrics $Q^* = \{q_1^*, q_2^*, \dots\}$ to a set of reconstructed quadrics $Q = \{q_1, q_2, \dots\}$ we define a similarity relation $\sim_{\text{QUAD}}: Q^* \times Q \rightarrow \{\text{TRUE}, \text{FALSE}\}$. For the evaluation of $q^* \sim_{\text{QUAD}} q$ we test the type, shape and pose of q and q^* for similarity. We obtain the shape of a quadric q as the three Eigenvalues of the upper left 3×3 matrix of the matrix representation of q . For evaluation we define the following sets:

$$\text{true positives} = \{q \in Q | \exists q^* \in Q^* : q^* \sim_{\text{QUAD}} q\}$$

	ours						RANSAC		
	t_{err}	s_{size}	v_{size}	tp	fp	fn	tp	fp	fn
large_a	$1 \cdot 10^{-5}$	0.08	0.008	6	0	0	3	2	3
medium_a	$5 \cdot 10^{-6}$	0.04	0.004	6	1	1	2	1	5
small_a	$5 \cdot 10^{-6}$	0.02	0.002	8	3	1	3	3	6
large_b	$1 \cdot 10^{-5}$	0.08	0.008	5	0	2	1	0	6
medium_b	$1 \cdot 10^{-5}$	0.04	0.004	6	0	0	1	4	5
small_b	$1.5 \cdot 10^{-6}$	0.02	0.002	6	2	3	4	5	5

Table 1: Results of RANSAC (Schnabel et al., 2007) and ours with true positives (tp), false positives (fp) and false negatives (fn) and hand optimized parameter values.

$$\text{false positives} = \{q \in Q | \nexists q^* \in Q^* : q^* \sim_{\text{QUAD}} q\}$$

$$\text{false negatives} = \{q^* \in Q^* | \nexists q \in Q : q^* \sim_{\text{QUAD}} q\}$$

4.2 Results

We investigate the influence of following parameters:

- t_{err} : error threshold for quadric fit in Section 3.3
- s_{size} : grid size for seed selection in Section 3.1
- v_{size} : size of a voxel grid used by (Papon et al., 2013) to downsample a point cloud

Table 1 shows hand optimized for these parameters for our data set. According to this, s_{size} depends on the size of the objects in the scene and v_{size} can be set to $\frac{s_{\text{size}}}{v_{\text{size}}} = 10$. Figure 6 shows the reason for false positives with small objects: Segments like 409 mistakenly cross the border between 400 and 398, which causes the two planar surfaces to be merged into one quadric. The errors in the medium and large data sets can mostly be attributed to partially scanned objects. Small connected components in the segment graph can often be fitted by a plane, even if the underlying surface is curved, see Figure 7a. Table 1 shows the results for primitive fitting using RANSAC (Schnabel et al., 2007). RANSAC fails to respect surface boundaries or fits large spheres to planar surfaces, see Figure 7b.

We sample values for each parameter around the optimal value for *medium_b*, see Figure 9. False positives, false negatives and true positives show that $t_{\text{err}} = 5 \cdot 10^6$ is a good value. For lower values,

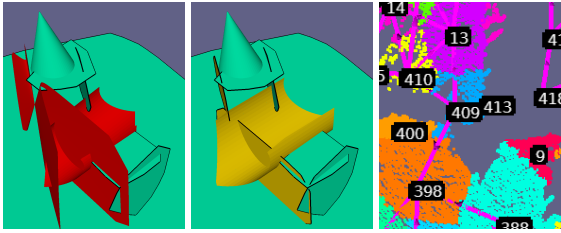
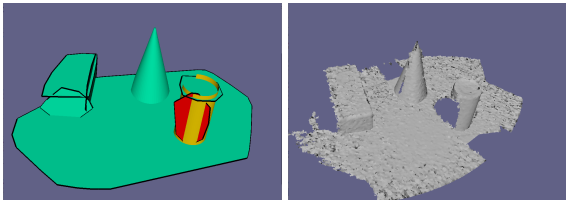
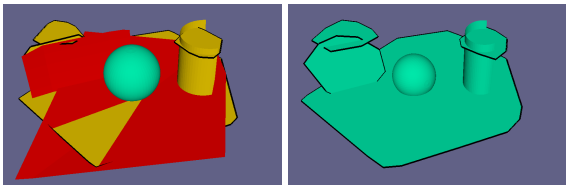


Figure 6: Results for *small_b* (left and middle) with false negative quadrics (orange), false positive quadrics (red) and true positive quadrics (green) and the local over-segmentation of the points causing the false positives (right).



(a) Results for *medium_a* (left) and input data (right)



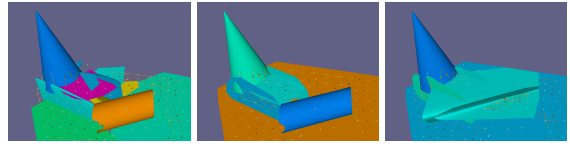
(b) RANSAC (left) and ours (right) on *large_a*

Figure 7: false negative quadrics (orange), false positive quadrics (red) and true positive quadrics (green)

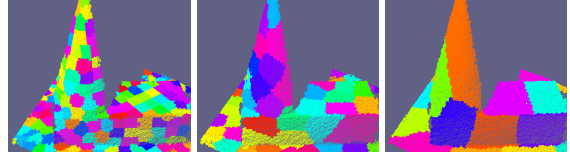
the algorithm many small surfaces. For higher values, surfaces are merged mistakenly, see Figure 8a. All metrics are optimal for $s_{\text{size}} = 0.04$. Computation time decreases from $s_{\text{size}} = 0.012$ to $s_{\text{size}} = 0.04$, because computation time of fitting depends on the number of segments (see Section 3.3). For higher values, the computation time increases again. This is caused by the implementation details of (Papon et al., 2013). Figure 8b shows that with $s_{\text{size}} = 0.08$ over-segmentation cannot preserve the border between cone and side of the cuboid. For $s_{\text{size}} = 0.02$, each segment is supported by too few points, some segments lie on the cuboid’s edges. Figure 9 shows that low values for v_{size} cause quantization errors.

We compare computation time for a series of point clouds using our approach and RANSAC (Schnabel et al., 2007). We incrementally reconstruct the data set *medium_b* using our approach and measure the time for processing of each point cloud. We compare this to RANSAC operating on increasing numbers of merged point clouds of this data set, see Figure 10 for results.

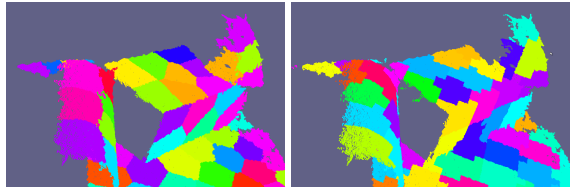
Figure 11 shows the numbers of correctly detected



(a) Results for $t_{\text{err}} = 5 \cdot 10^{-7}, 5 \cdot 10^{-6}, 2 \cdot 10^{-5}$



(b) Over-segmentation with $s_{\text{size}} = 0.02, 0.04, 0.08$



(c) Over-segmentation with $v_{\text{size}} = 0.004, 0.008$

Figure 8: Influence of different parameters on data set *medium_b*, values respectively from left to right

cylinders by RANSAC and our approach for each of the 15 point clouds from ITODD data set. The point clouds contain increasing numbers of cylinders. Our approach performs well for small numbers of shapes (≤ 4) and slightly outperforms RANSAC in this range. For > 9 cylinders, small connected components in the segment graph cannot be merged by region merging, see Figure 12.

We compare our approach to 4P-RANSAC (Birdal et al., 2019). We use the following data from redwood data set (Choi et al., 2016): *Rugby Ball* (a), *Pilates Ball* (b), *Big Globe* (c), *Apple* (d), *Orange Ball* (e). Figure 13 compares our results to those stated in (Birdal et al., 2019). 4P-RANSAC outperforms our system for most data sets. Our algorithm is less suited to deal with clutter in the scene. Figure 13 shows the false negative (orange) for the *Apple* data set and the segment graph (magenta edges and black vertices). The segment graph connects points of the hand holding an apple to the points of the apple, which causes our algorithm to fail finding the apple as a quadric.

5 Discussion and Prospects

In this paper, we present a novel technique for incremental reconstruction of a piecewise quadric surface from a series of point clouds. Our experiments show that our memory representation of segments in a segment graph allows for fast fitting of quadric surfaces to these segments. We show that our approach can

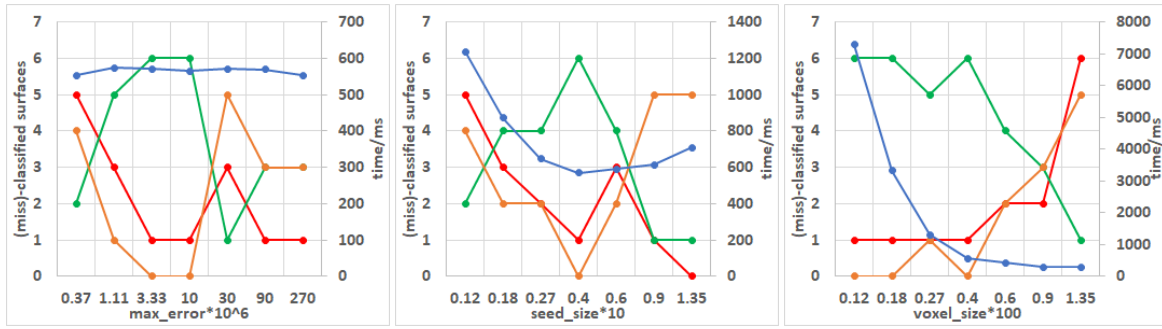


Figure 9: Average runtime per point cloud (blue), true positives (green), false positives (red) and false negatives (orange) for different parameter values on the data set *medium_b*.

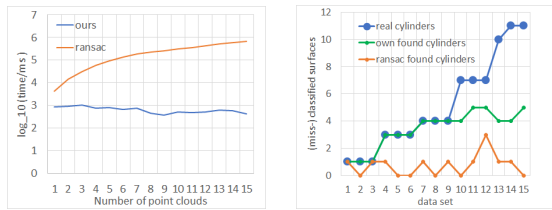


Figure 10: Runtime of RANSAC and ours for increasing number of points. Figure 11: Per scene metrics for ours and (Schnabel et al., 2007) on ITODD.

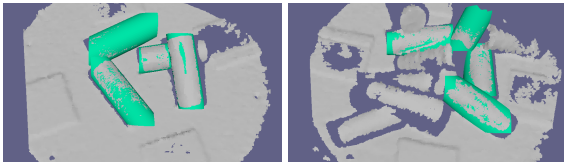


Figure 12: Cylinder detected by our approach in a scene of 4 real cylinders (left) and in a scene of 10 cylinders (right).

	4P-RANSAC	Ours
a	100%	80.2%
b	100%	100%
c	90.7%	87.1%
d	99.6%	38.8%
e	93.3%	94.4%

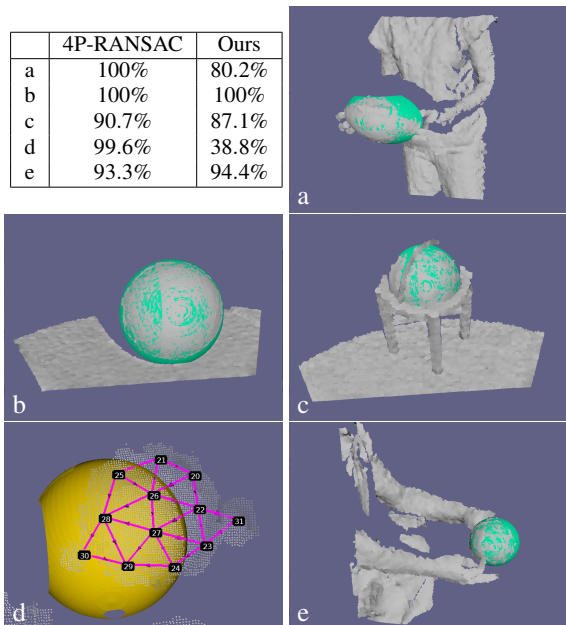


Figure 13: Accuracy for detecting an ellipsoid within a point cloud of the data set of (Choi et al., 2016) with ours and 4P-RANSAC (Birdal et al., 2019).

operate at a frame rate of 1 Hz for point clouds of $> 300,000$, which is by magnitudes faster than existing RANSAC based approaches with similar accuracy for scenes consisting of quadric surfaces. Future improvements can be made on the fitting procedure on our segment graph. Fitting is the only part of our technique whose computation time depends on the number of previously processed point clouds. We plan to use an incremental fitting approach that only recomputes the surface patches associated with segments that have been influenced by the current point cloud. Furthermore, nodes from our segment graph that belong to a quadric surface can be used to compute an approximation of the surface's boundary to form a more accurate reconstruction that could also be used for common CAD exchange formats.

ACKNOWLEDGEMENTS

This work has partly been supported by the Deutsche Forschungsgemeinschaft (DFG) under grant agreement He2696/16 HandCAD.

REFERENCES

- Alehdaghi, M., Esfahani, M. A., and Harati, A. (2015). Parallel RANSAC: Speeding up Plane Extraction in RGBD Image Sequences using GPU. In *5th International Conference on Computer and Knowledge Engineering*, pages 295–300. IEEE.
- Andrews, J. and Séquin, C. H. (2013). Type-Constrained Direct Fitting of Quadric Surfaces. *Computer-Aided Design and Applications*, 11(1):107–119.
- Arbeiter, G., Fuchs, S., Hampf, J., and Bormann, R. (2014). Efficient Segmentation and Surface Classification of Range Images. In *IEEE International Conference on Robotics and Automation*, pages 5502–5509. IEEE.
- Beale, D., Yang, Y.-L., Campbell, N., Cosker, D., and Hall,

- P. (2016). Fitting quadrics with a Bayesian prior. *Computational Visual Media*, 2(2):107–117.
- Beraldin, J.-A., Picard, M., El-Hakim, S. F., Godin, G., Valzano, V., and Bandiera, A. (2005). Combining 3D technologies for cultural heritage interpretation and entertainment. In Beraldin, J.-A., El-Hakim, S. F., Gruen, A., and Walton, J. S., editors, *Videometrics VIII*, volume 5665, page 108.
- Birdal, T., Busam, B., Navab, N., Ilic, S., and Sturm, P. (2019). Generic Primitive Detection in Point Clouds Using Novel Minimal Quadric Fits. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.
- Buonamici, F., Carfagni, M., Furferi, R., Governi, L., Lapini, A., and Volpe, Y. (2018). Reverse engineering modeling methods and tools: a survey. *Computer-Aided Design and Applications*, 15(3):443–464.
- Choi, S., Zhou, Q.-Y., Miller, S., and Koltun, V. (2016). A Large Dataset of Object Scans. *arXiv:1602.02481*, pages 1–7.
- Denker, K., Hagel, D., Raible, J., Umlauf, G., and Hamann, B. (2013). On-Line Reconstruction of CAD Geometry. In *2013 International Conference on 3D Vision*, pages 151–158. IEEE.
- Drost, B., Ulrich, M., Bergmann, P., Hartinger, P., and Stenger, C. (2017). Introducing MVTEC ITODD — A Dataset for 3D Object Recognition in Industry. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, volume 2018-Janua, pages 2200–2208. IEEE.
- Fischler, M. a. and Bolles, R. C. (1981). Random Sample Paradigm for Model Consensus: Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395.
- Frahm, J.-M. and Pollefeys, M. (2006). RANSAC for (Quasi-)Degenerate data (QDEGSAC). In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06)*, volume 1, pages 453–460. IEEE.
- Gotardo, P., Bellon, O., and Silva, L. (2003). Range Image Segmentation by Surface Extraction using an Improved Robust Estimator. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2. IEEE Comput. Soc.
- Kaiser, A., Ybanez Zepeda, J. A., and Boubekour, T. (2019). A Survey of Simple Geometric Primitives Detection Methods for Captured 3D Data. *Computer Graphics Forum*, 38(1):167–196.
- Kim, Y. M., Mitra, N. J., Yan, D.-M., and Guibas, L. (2012). Acquiring 3D Indoor Environments with Variability and Repetition. *ACM Transactions on Graphics*, 31(6):1.
- Lukács, G., Martin, R., and Marshall, D. (1998). Faithful Least-squares Fitting of Spheres, Cylinders, Cones and Tori for Reliable Segmentation. In *Computer Vision - ECCV'98, 5th European Conference on Computer Vision*, volume 1.
- Makhal, A., Thomas, F., and Gracia, A. P. (2018). Grasping Unknown Objects in Clutter by Superquadric Representation. In *Second IEEE International Conference on Robotic Computing*, pages 292–299. IEEE.
- Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohli, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). KinectFusion: Real-time dense surface mapping and tracking. *2011 10th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2011*, pages 127–136.
- Oesau, S., Lafarge, F., and Alliez, P. (2016). Planar Shape Detection and Regularization in Tandem. *Computer Graphics Forum*, 35(1):203–215.
- Papon, J., Abramov, A., Schoeler, M., and Worgotter, F. (2013). Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2027–2034.
- Rusu, R. B., Blodow, N., and Beetz, M. (2009). Fast Point Feature Histograms (FPFH) for 3D registration. In *2009 IEEE International Conference on Robotics and Automation*, pages 3212–3217. IEEE.
- Sand, M. (2019). *Inkrementelle Rekonstruktion von planaren Volumenmodellen mit handgehaltenen Tiefenkameras (German)*. PhD thesis, University of Bayreuth, Bayreuth.
- Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum*, 26(2):214–226.
- Tasdizen, T., Tarel, J. P., and Cooper, D. B. (1999). Algebraic curves that work better. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2(February):35–41.
- Taubin, G. (1991). Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(11):1115–1138.
- Vanco, M. and Brunnett, G. (2002). Direct Segmentation for Reverse Engineering. In *First International Symposium on Cyber Worlds, 2002. Proceedings.*, pages 24–31. IEEE Comput. Soc.
- Vempati, A. S., Kamel, M., Stilinovic, N., Zhang, Q., Reusser, D., Sa, I., Nieto, J., Siegwart, R., and Beardsley, P. (2018). PaintCopter: An Autonomous UAV for Spray Painting on Three-Dimensional Surfaces. *IEEE Robotics and Automation Letters*, 3(4):2862–2869.
- Whelan, T., Kaess, M., and Fallon, M. (2012). Kintinuous: Spatially extended kinectfusion. *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, page 7.