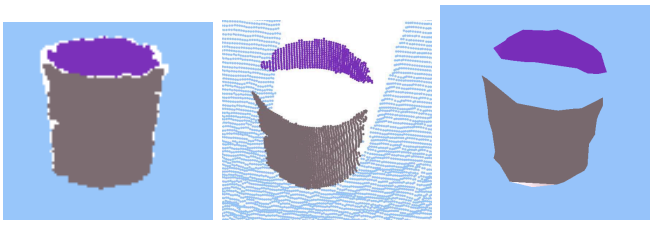
2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Quadtree-based Polynomial Polygon Fitting

Joshua Hampp[1] and Richard Bormann[1]

***Abstract*— In this paper, we present a novel method for surface reconstruction with a low execution time for segmenting and representing scattered scenes accurately. The surfaces are described in a memory-efficient fashion as polynomial functions and polygons. Segmentation and parameter determination is done in one pass by using a quadtree on ordered point clouds, which results in a complexity of $\mathcal{O}(\log n)$.**

This paper includes an evaluation with respect to reconstruction accuracy, segmentation precision, execution time and compression ratio of everyday indoor scenes. Our surface reconstruction algorithm outperforms comparable approaches with respect to execution time and accuracy. More importantly, the new technique handles curved shapes accurately and enables complex tasks like 3D mapping for mobile robots in an unknown environment.



(a) Segmentation (b) Labeled input point cloud (c) Reconstructed and labeled point cloud

Fig. 1: Surface reconstruction of a bin from segmentation to parametric description ($n = 2$)

I. Introduction

Recently the mobile computing segment has expanded significantly as a consequence of the smart phone boom. Also mobile devices are steadily extended with new sensors like cameras, gyroscopes and in future 3D cameras. The combination of mobility and sensors enable new applications like indoor mapping and indoor navigation. As mobile devices have a low power consumption they are predestined to be used in mobile robotics.

Yet, the computational power of mobile devices is quite limited, whereas many 3D mapping algorithms have excessive computational and storage requirements. For example KinectFusion [1] showed a convincing implementation of 3D indoor mapping that was achieved by high computational effort. The enormous demands on the system stem from the complex and discrete representation of the surfaces. To enable efficient 3D applications like mapping on mobile robots, it is necessary to use a compact and efficient surface representation.

In this paper, we introduce a novel approach for surface reconstruction of curved surfaces, including segmentation. The surface model is described as polynomial function which is obtained by least-squares regression within a quadtree (Fig. 2). After region growing, the outlines of the segments are converted to polygons based on a binary pattern similar to [2]. Hence, each surface segment is described in a memory efficient fashion by its hull and a polynomial function (Fig. 1). The main contributions are:

a) a segmentation with low execution time through an efficient data structure,
b) general and accurate representation of curved surfaces with polynomial functions,
c) and a memory efficient storage through hull polygons.

In the first half of Section III, the surface model and segmentation is outlined. The second half of Section III describes the polygon extraction. Our approach is compared with RANSAC, Marching Cubes, Multi Plane, JPEG and PNG in Section IV. We conclude in Section V with a summary and an outlook for future work.

II. Previous Work

The first step of surface reconstruction is to find segments which describe different surfaces. Segmentation is generally divided into edge-based and region-based approaches. E.g. Sappa and Devy [3] use the first one to extract boundary points of segments through jump edges. Edge-based segmentation fails to recognize smooth transitions between surfaces while region-based approaches overcome this limitation by combining segmentation and model fitting in one step.

Region-based segmentations like [4], [5] use normals at each point as region growing criterion for locally planar patches. As Holz et al. [6] state normal estimation on a complete point cloud can be quite expensive. Hence, our approach directly compares the depth values with the model of curved surfaces and avoids normal computation.

Mörwald et al. [5] showed an approach to reconstruct curved geometries with NURBS. First the point cloud is over-segmented into planar patches. Then the planes are merged together and represented by NURBS [7] which represent curved surfaces accurately. In contrast, the proposed algorithm segments and determines curved surfaces in one pass.

*The research leading to these results has received funding from the German Federal Ministry of Economics and Technology (BMWi) within the project AutoPnP (01MA11005).

[1]The authors are with the Fraunhofer Institute for Manufacturing Engineering and Automation IPA, 70569 Stuttgart, Germany <first name>.<last name> at ipa.fraunhofer.de; www.ipa.fraunhofer.de

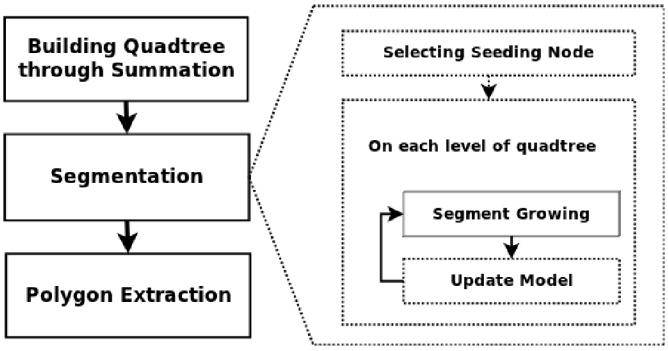


Fig. 2: Pipeline for Quadtree-based Polynomial Polygon Fitting on range images

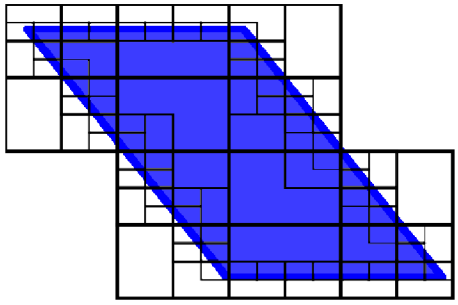


Fig. 3: Quadtree-based representation of a segment (blue)

Another approach is to incorporate color information from RGB-D sensors like Kinect. Erdogan et al. [8] combine color and depth information into one weighting function for a graph-based approach. Furthermore Erdogan et al. show that least squares fitting for planar patches on the 1D domain is more robust than on 3D points by adapting all three dimensions. Therefore our surfaces also represent the 1D depth value of the depth image in dependence of the lasting coordinates.

In contrast to the presented surface reconstruction methods RANSAC [9] fits an initial model to the point cloud without further segmentation beforehand. Random samples are selected from a point cloud to retrieve a first model, which can be verified by the remaining data. Repeating those steps, the input cloud is segmented into different surfaces.

Besides of planes and NURBS triangles are a common representation of surfaces in computer graphics [10]. One way to extract triangles from a segmented point cloud is Marching Cubes [2]. It divides the covered space into cubes and converts local surface intersections into binary patterns. The binary patterns are replaced through a corresponding triangle list. The idea of binary patterns is found again in this work for polygon extraction from border points.

III. Methodology

In indoor environments, a robot encounters diverse scenes with a plethora of different geometries. Most applications in service robotics need a representation which describes everyday geometries, e.g. for grasping, collision avoidance and object recognition. Our approach represents point clouds through curved surfaces including the outline and holes. The outline and holes are represented as polygons, which are generated from the segments. The surface model describes the depth value in dependence of the remaining coordinates.

A. Segmentation

A segment is a region or set of points which can be described through the same parameterized model. A model specifying the depth value z of each point from a point cloud can be stated as function $d(x, y)$ of x and y. A general representation of an unknown function is given by a Taylor series which is expressed as polynomial function with one parameter for each term. The accurate representation of a segment through a Taylor series depends on the degree n of d and how well the point set can be reproduced by the function. Although an exact fitting is desirable, system immanent noise should be eliminated through segmentation. A Taylor series up to the linear term is sufficient for planes; curved objects need a higher degree for the approximation.

The polynomial function d_n of degree n with the parameters p is stated as follows:

$$d_n(x, y) = z = \sum_{i=0}^{n} \sum_{j=0}^{i} p_{(j+\sum_1^i i)} x^j y^{i-j} = p^T a$$

The parameters p for the function d_n are unknown, but the output and the input of $z_i = d_n(x_i, y_i)$ are given by the point cloud. The measurement z_i is the depth at the world coordinate x_i and y_i, which is given from the depth image for point i. Therefore, the parameters p can be estimated by linear least-squares fitting [11].

$$R = \sum_i \left(d_n\left(x_i, y_i\right) - z_i\right)^2$$

Minimizing the residuals R leads to:

$$\arg\min_p \sum_i \left(p^T a_i - z_i\right)^2 \tag{1}$$

The parameter p is solved by linear regression. For example singular value decomposition [12] or QR decomposition can be used. For our implementation we used singular value decomposition with respect to a lower execution time.

Equation 1 is minimized to formulate an additive expression for parameter determination:

$$\sum_i a_i z_i = p^T \sum_i a_i a_i^T$$

By use of the additive behavior of the contributing values $a_i z_i$ and $a_i a_i^T$, the parameter only has to be determined once for all points of the segment. It is necessary that the summarized points are part of the same model. Otherwise the result will not be correct.

B. Quadtree Representation

The contributing values $a_i z_i$ and $a_i a_i^T$, which are needed to determine the parameters of the model, are represented by a quadtree (Fig. 3) in an efficient fashion. The point set of a segment consists of neighboring points from the depth image, which can therefore be summarized beforehand through their local proximity without consideration of different segments. Segmentation is done by a top-down strategy within the tree. For parameter determination only the summarized nodes have to be evaluated instead of all single points.

Each point is a leaf of the quadtree. Four neighboring points are quadrants of the parent node. Each node has four children and contains the sums of $\sum az$ and $\sum aa^T$ from the lower nodes or leaves. The values $\sum az$ and $\sum aa^T$ of the leaves are given by the corresponding points.

After building the quadtree, segmentation starts with the selection of a random seed node. We used every second node starting from top left corner. From the starting point, the segment grows and the parameter model is refined until a termination condition occurs.

Because the seed node should describe the model of the segment accurately, the starting node has to be completely part of the segment which is true for all leaves. Yet, a single point is not sufficient as starting node. To ensure that the seeding node is completely part of one single surface it is required that the surrounding nodes are also a valid part of the segment. For an accurate approximation of the model the more points the better the result. Thus, the starting node is selected for our application at the 3rd and 4th level of the quadtree, which have 4^2 and 4^3 leaves.

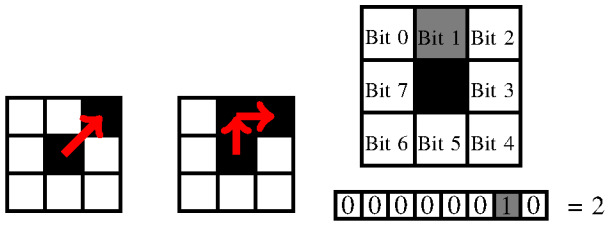
The termination condition states if another node is also a part of the model. The condition is a combination of the allowed residuals (2) of the nodes and edge-based (3) criterion.

$$e_{i,1}^2 = \frac{\left(\overbrace{p^T a}^{\text{estimated depth}} - \overbrace{\begin{pmatrix} 1 \\ 0 \\ \vdots \end{pmatrix} \cdot (az)}^{\text{observed depth}} \right)^2}{m^2} \tag{2}$$

For the residuals $e_{i,1}$ the averaged z value of the node and its four children i are individually compared against the approximated model p with m leaves. If all $e_{i,1}$ lie within 1.5σ (86.6% of valid points) of the expected standard deviation σ of the depth image, the node is valid.

$$(e_2)^2 = (z_{\max} - z_{\min})^2 \tag{3}$$

For the edge based criterion the difference between the maximum $z_{\max}$ and minimum depth value $z_{\min}$ within the node is checked against the expected difference of the camera error model and the local gradient from the model. This value should lie within 3σ (99.7% of the cases). Both factors of σ are chosen thresholds and should be adjusted in dependence of the application and sensor.



(a) One neighbor (b) Two neighbors (c) Direction estimation of neighboring points

Fig. 4: Direction estimation of neighboring points

Segment growing within the quadtree is done on the same level until no node fulfils the validation rule. After one full pass, region growing is repeated, because the model was updated and refined in the first run (Fig. 2). Nodes which could not be validated in a second pass are marked as border nodes. Border nodes are split up to their children and processed further on the lower level to refine the segment. On lower levels the growth is limited (in our case to 9 steps from the initial parent node) to prevent high computational costs in some special cases (see IV-G). If the leaves are reached and a leaf is not valid part of the model, it is a border point of the segment, and therefore part of the outline of the segment.

C. Polygon-based Border Representation

After the segmentation step the model is described through its parameters and its border points, containing the hull and holes. The border points are converted to a polygon for a memory-efficient representation. A polygon is the approximation of neighbouring points by a closed path composed of straight lines. The ordered structure of the point cloud is used to find neighbouring border points. In an image a line is sampled with pixels directed into a uniform direction. Therefore, any changes in direction introduce a new line segment of the polygon if it is not part of aliasing.

For each point a direction is assigned which is given through its neighboring border point as shown in Figure 4a. The direction of a line is the differential equation which is constant for linear equations.

$$f(x) = ax + b$$

$$f''(x) = 0$$

The second discrete derivation of the line f'' is equal to the direction drift of the sampled line. On the depth image the direction drift of successive pixels is the Manhattan distance between the expected position of the next point from the last direction, and the actual observed position. Because of quantization and noise the local direction drift is usually not 0. However, integrating the derivative yields $\int f'' = 0$ for an ideal line. For locally observed data an error Δ_s^e is given from starting point s to the endpoint e of a line segment:

$$\Delta_s^e = \int_s^e f''$$

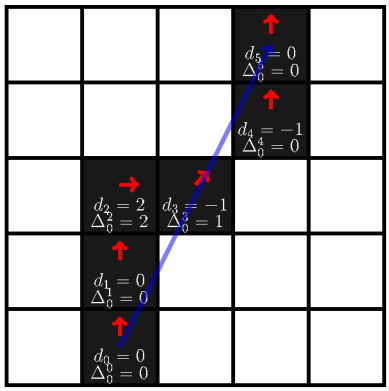


Fig. 5: Line extraction (blue) from border points (black)

If Δ_s^e exceeds a given threshold, the line segment cannot be expressed by a linear equation anymore and leads to a new line of the polygon.

In Figure 4a the direction estimation is shown. If there is more than one neighboring point, the leftmost in relation to the last direction is taken (Fig. 4b). This ensures that all points are passed in clock-wise direction. After visiting a point it is removed from the border point list and will not be revisited again. An example is given in Figure 5.

An initial direction for clock-wise circulation has to be set for the starting point, which has always two neighbors for a closed path. As starting point the topmost and leftmost is selected, which is followed by the next neighboring point on the right side. This enforces a clock-wise selection of the next point search.

D. Implementation of Polygon Fitting

A point in a binary 2D image has eight neighbors, shown in Figure 4c. The binary occupancy map of the neighbors is encoded in an 8-bit integer which can be used as index for a lookup table. The combination of the lookup value of the last point and the actual point yields – beside direction – the direction drift. With a previously generated lookup table with a 16-bit index the polygon extraction has a complexity of $\mathcal{O}(n)$ with n border points.

IV. Results

Our approach is evaluated with regard to execution speed, reconstruction accuracy, segmentation precision and compression ratio in comparison to RANSAC, Marching Cubes with a "greedy" voxelization algorithm, an organized multi plane approach (Multi Plane), JPEG and PNG. The test system was equipped with an i7-2600, at which only a single core was used, and 4 GB RAM. The algorithm was integrated in ROS and is using the PCL [13]. The results for the whole pipeline were averaged over all frames and give an insight into potential fields of application.

TABLE I: Averaged execution time for algorithms and processing steps of our approach

degree	quadtree	region growing	polygon	sum
QPPF n=1	8.12 ms	10.6 ms	1.70 ms	20.4 ms
QPPF n=2	19.8 ms	13.5 ms	1.80 ms	35.0 ms
QPPF n=3	94.5 ms	25.5 ms	10.0 ms	130.0 ms
QPPF n=4	101 ms	37.4 ms	18.9 ms	157.3 ms
Multi Plane	-	-	-	90.0 ms
RANSAC	-	-	-	412 s
Marching Cubes	-	-	-	2.00 s

A. Scenes

As datasets we used the publicly available IPA dataset[1] and TOCS dataset[2]. Both were captured with an Asus Xtion Pro Live with a resolution of 640×480. The IPA dataset contains eight real world scenes from various cluttered indoor environments from far and close distances (Fig. 6). In addition the IPA dataset contains 36 simulated scenes which were generated with Blensor [14]. They contain labels for continuous surfaces like planes, spheres, cylinders and NURBS in different sizes, distances and viewing angles. The labeled and simulated dataset contains noise-free - but with disparity error - and noisy point clouds which were both used as groundtruth to evaluate the segmentation precision. Every 10th frame (213 point clouds) of the TOCS dataset was used with an all-round scenery, from table scenes to far walls.

B. Parameter Settings

The camera error model is essential to distinguish between edges and noise. For the datasets we used the error model according to [15], which expresses the standard deviation σ in dependence of the distance z of the measurement:

$$\sigma = 1.425 \times 10^{-3} z^2$$

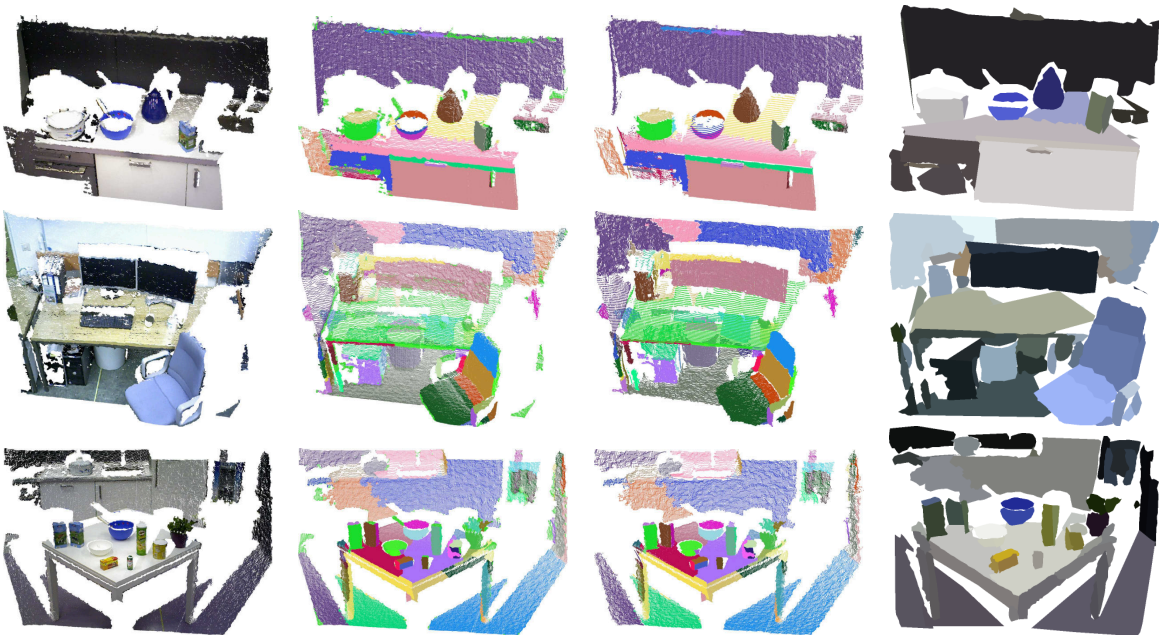
For polygon extraction from the border points, a direction drift threshold of 3 was used, which results in a very precise substitute for the border points.

For RANSAC we used the implementation from PCL with planes, spheres and cylinders with a distance threshold of 3 cm. Marching Cubes was also used from PCL with the parameters of 0.025 for the leaf size and 0.02 for the ISO level. Multi Plane is a flood fill approach from PCL which is similar to [6]. The used parameters are 2.5° for the angular threshold, 0.001 for the maximum curvature, a distance threshold of 1 cm and a minimum cluster size of 100 points.

C. Execution Time

The execution time was measured for each step, namely building the quadtree, segmentation and polygon extraction, on the above mentioned computer system. As Table I shows surface reconstruction for second degree polynomials is possible with over 28 Hz. The most expensive step for non-linear polynomials is building the quadtree with around 85% as a

[1]https://github.com/ipa320/cob_test_dataset
[2]http://svn.pointclouds.org/data/Toyota



(a) Input point cloud (b) Labeled input point cloud (c) Reconstructed and labeled point cloud (d) Reconstructed polygons

Fig. 6: Different representations of a point cloud with second degree (kitchen, office, table)

result of the growing number of data with increasing degree. Yet, this step can be parallelized well as it is straightforward and does not depend on other results.

Multi Plane shows also a low execution time. In comparison, RANSAC has at least a linear complexity with $\mathcal{O}(I(E+n))$ (number of iteration I, complexity for initial model estimation E and number of points n). The complexity of Marching Cubes depends on the number of cubes m, as the complexity is stated as $\mathcal{O}(m \log n)$. With the evaluated configuration both approaches do not reach 1 Hz.

D. Reconstruction Accuracy

Accuracy is stated as minimal distance between the model and the observed point. The distance between a point and a polynomial manifold in n-dimensional space is not trivial and there is no straightforward mathematical solution. Therefore we used Levenberg-Marquardt optimization to iteratively find the minimal distance between the surface and a point (x_0, y_0, z_0) with the following distance model:

$$F(x,y) = (x_0 - x)^2 + (y_0 - y)^2 + (z_0 - d(x,y))^2$$

$$\arg\min_{x,y} \left\| F(x,y) + \begin{pmatrix} \frac{\partial F}{\partial x} \\ \frac{\partial F}{\partial y} \end{pmatrix} \Delta \right\|$$

The accuracy results are shown in Figure 7 and Table II. For further use second degree is recommend to represent curved shapes while retaining a high performance regarding accuracy, coverage and execution time. With the stated parameter settings Multi Plane reaches also a high accuracy. But the trade-off is a low coverage.

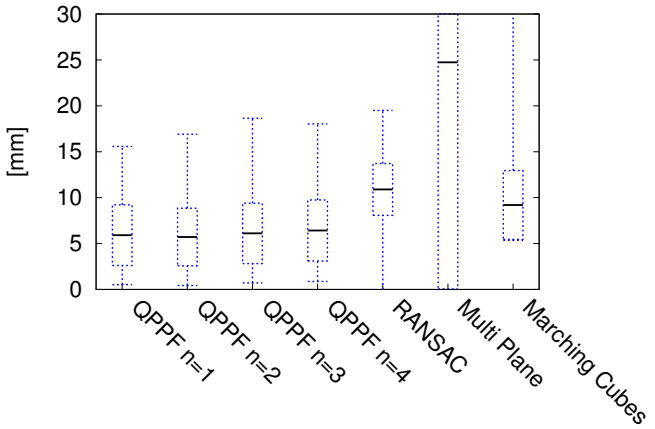


Fig. 7: Averaged accuracy results for evaluated datasets (boxplot)

TABLE II: Averaged distance between point cloud and reconstructed surfaces for evaluated datasets

method	median [mm]	mean [mm]	variance [mm]	coverage [%]
QPPF n=1	5.8010	5.9118	**0.0095**	95.3
QPPF n=2	**5.2395**	**5.7108**	0.0127	**97.1**
QPPF n=3	5.5545	6.1075	1.1138	93.2
QPPF n=4	5.9705	6.4237	11.055	88.6
RANSAC	11.829	10.896	0.1051	91.6
Multi Plane	7.2755	24.736	14.425	78.9
March. Cubes	8.1285	9.1998	0.0784	76.9

TABLE III: Segmentation quality with F_{os} (left) and F_{us} (right) in percent

method	planes		cylinders		sphere		NURBS		other	
QPPF n=1	16.5	16.2	48.0	47.7	79.3	78.2	58.1	57.7	33.6	30.8
QPPF n=2	12.3	11.9	9.5	9.3	7.2	7.0	9.8	9.5	**8.9**	**6.4**
QPPF n=3	17.2	16.8	**4.3**	**4.2**	8.7	8.6	**8.2**	**8.2**	10.0	8.0
QPPF n=4	14.0	13.7	6.6	6.5	**7.0**	**6.9**	12.0	11.9	9.0	6.8
RANSAC	**9.2**	**5.1**	41.6	32.7	66.6	54.9	52.2	43.8	50.4	29.6
Multi Plane	44.0	35.4	26.2	9.1	36.8	16.8	53.8	40.3	65.8	58.6

Cases of failure are especially surfaces which are nearly parallel to the viewing direction. Those surfaces are sampled with a low resolution which induces an unstable model fitting and an increased deviation.

In addition to accuracy, the number of valid points of the input cloud is compared against the reconstructed points to show that most surfaces in indoor environments can be represented by our model. Third and fourth degree have a lower coverage than first and second degree. The combination of the lower coverage and good accuracy results with increasing degree indicates over-fitting of noisy or incomplete surfaces which limits region growing.

E. Segmentation Precision

In order to measure the segmentation precision, the application is crucial for labeling of the groundtruth. Our desired application is 3D mapping of environments which consist of continuous surfaces. Therefore, different surfaces are labeled as separated segments in the groundtruth.

The precision of the segmentation was evaluated by comparing the computed segments with the groundtruth to provide a measure for under- and over-segmentation. The segments with the best match are true positives N_{true} on point basis while the incorrect segmented points are false positives N_{false}. With the number of all points N_{all} over-segmentation F_{os} and under-segmentation F_{us} is defined as:

$$F_{\text{os}} = 1 - \frac{N_{\text{true}}}{N_{\text{all}}}$$

$$F_{\text{us}} = \frac{N_{\text{false}}}{N_{\text{all}}}$$

In Table III over- and under-segmentation is stated. Low values are preferable. RANSAC shows in general better results for planes in the used dataset because it merges divided segments which are handled as separated surfaces in the proposed approach. For more complex surfaces like spheres, cylinders and free form surfaces RANSAC and our surface model with degree 1 show similar results. From second degree on, even complex surfaces can be represented precisely. Multi Plane handles noise-free data well while noisy point clouds lead to over-segmentation.

F. Compression Ratio

A good compression is especially needed for point cloud transmission with a limited bandwidth like mobile Internet. The compression ratio is stated for all evaluated surface reconstruction methods and two image compression methods (JPEG and PNG). For the surface reconstruction algorithms the resulting memory usage is compared with the original point cloud. PNG and JPEG [16] are lossless and lossy image compression algorithms which are applied and compared with the size of the disparity image.

TABLE IV: Compression ratio in relation to original size (point cloud, bitmap)

method	compression ratio [%]	size [kB]	orig. size [kB]
PNG	7.00	63.0	901
JPEG	4.47	40.2	901
Marching Cubes	57.8	2080	3600
RANSAC	23.2	837	3600
Multi Plane	2.28	82.1	3600
QPPF n=1	**0.017**	**0.63**	3600
QPPF n=2	0.028	1.0	3600
QPPF n=3	0.020	0.70	3600
QPPF n=4	0.024	0.85	3600

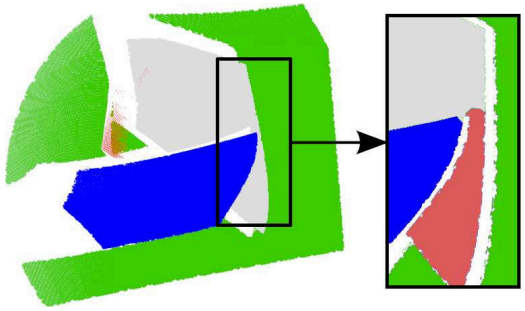


Fig. 8: Narrow passage between gray and red segment is divided

The input point cloud had a resolution of 640×480 pixels with a size of 3600 kB. In Table IV the compression ratio in relation to the size of the input point cloud or rather the size of the disparity image saved as bitmap is stated. The resulting file size of our surface reconstruction is by factor 50 smaller than those of the image compression algorithms. Polygon extraction was not performed for RANSAC. Otherwise similar compression results to our approach could be expected. Especially for teleoperation, a bandwidth load of around $32\,\text{kB}\,\text{s}^{-1}$ at $30\,\text{Hz}$ is preferable over a full point cloud transport with $105\,\text{MB}\,\text{s}^{-1}$.

G. Discussion

The over-segmentation of planes in Figure 8 is apparent. It is a result of the limited growth in the lower levels of the quadtree. Narrow passages prevent further spreading and region growing stops here. A possible solution is an additional optimization step which merges similar neighbouring segments. Merging is not necessarily done as part of the surface reconstruction because following applications like mapping either have no need for optimized surfaces or use their own merging strategy.

One drawback of region growing methods is the dependence of the result from the seeding node. By means of the quadtree the dependence is reduced because the first nodes

TABLE V: Standard deviation of segmentation precision with random seed nodes (30 iterations) with $\mathrm{std}(F_{\mathrm{os}})$ (left) and $\mathrm{std}(F_{\mathrm{us}})$ (right) in percent

method	planes		cylinders		sphere		NURBS		other	
QPPF n=1	**5.5**	13.4	**0.9**	1.3	0.9	1.7	2.4	2.2	4.0	3.6
QPPF n=2	6.8	**13.0**	1.3	**1.1**	**0.5**	**0.5**	**2.3**	**2.0**	**2.5**	**2.2**
QPPF n=3	10.7	13.2	7.3	7.2	7.3	7.4	9.2	8.9	8.3	8.2
QPPF n=4	10.8	14.4	12.3	11.5	12.3	11.7	14.5	13.6	11.6	10.2

already contain many points of the segment. We used random seed nodes and evaluated the simulated dataset 30 times for each degree. Table V shows the standard deviation of F_{os} and F_{us} for random seed nodes. In general, the segmentation precision is stable for first and second degree. However, smooth transitions of surfaces like in the plane dataset are difficult to distinguish. Therefore, segmentation precision of planes shows an increased deviation. Higher degree tends to overfitting. Traditional region growing methods start at a single point and are therefore more prone to noise.

V. Conclusions

This paper presented a new approach for surface reconstruction and segmentation for ordered point clouds. By introducing a novel data representation based on a quadtree, both – segmentation and surface reconstruction – are accelerated to meet real-time requirements. In addition, our approach compresses the input data to a compact representation while filtering noise and clutter. Therefore, our approach opens up new application areas in mobile robotics and consumer markets of mobile devices by representing unknown environments by parametric surfaces.

Future work will integrate color information to extract boundaries with greater robustness. In the next step, we will incorporate curved surfaces and 3D mapping applications to enable indoor navigation on mobile devices.

References

[1] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, 2011, pp. 559–568.

[2] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 163–169, 1987.

[3] A. D. Sappa and M. Devy, "Fast Range Image Segmentation by an Edge Detection Strategy," in *3DIM*, Quebec City, Que., Canada, 2001, pp. 292–299.

[4] J. Strom, A. Richardson, and E. Olson, "Graph-based Segmentation for Colored 3D Laser Point Clouds," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Kobe, Japan, October 2010.

[5] T. Mörwald, A. Richtsfeld, J. Prankl, M. Zillich, and M. Vincze, "Geometric data abstraction using B-splines for range image segmentation," in *IEEE International Conference on Robotics and Automation*, Karlsruhe, Germany, 2013.

[6] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, "Real-Time Plane Segmentation using RGB-D Cameras," in *RoboCup Symposium*, Istanbul, Turkey, 2011.

[7] W. Wang, H. Pottmann, and Y. Liu, "Fitting B-spline curves to point clouds by curvature-based squared distance minimization," *ACM Trans. Graph.*, vol. 25, no. 2, pp. 214–238, 2006.

[8] C. Erdogan, M. Paluri, and F. Dellaert, "Planar segmentation of rgbd images using fast linear fitting and markov chain monte carlo." in *CRV*. IEEE, 2012, pp. 32–39.

[9] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for Point-Cloud Shape Detection," *Computer Graphics Forum*, vol. 26, no. 2, pp. 214–226, 2007.

[10] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer graphics: principles and practice (2nd ed.)*. Boston, MA, USA: Addison-Wesley Longman, 1990.

[11] D. York, "Least squares fitting of a straight line with correlated errors," *Earth and Planetary Science Letters*, vol. 5, pp. 320 – 324, 1968.

[12] G. H. Golub and C. Van Loan, "An analysis of the total least squares problem," in *SIAM Journal on Numerical Analysis*, vol. 17, Ithaca, NY, USA, 1980, pp. 883–893.

[13] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *International Conference on Robotics and Automation*, Shanghai, China, 2011.

[14] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, "BlenSor: Blender Sensor Simulation Toolbox Advances in Visual Computing," ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2011, vol. 6939, ch. 20, pp. 199–208.

[15] K. Khoshelham, "Accuracy analysis of Kinect depth data," in *ISPRS workshop laser scanning*, Calgary, Canada, 2011.

[16] J. Miano, *Compressed image file formats: JPEG, PNG, GIF, XBM, BMP*, ser. SIGGRAPH series. Addison Wesley, 1999.