J Intell Robot Syst (2011) 61:169–180
DOI 10.1007/s10846-010-9478-8

Meshing and Simplification of High Resolution Urban Surface Data for UAV Path Planning

Florian-M. Adolf · Heiko Hirschmüller

Received: 1 February 2010 / Accepted: 1 September 2010 / Published online: 22 October 2010

Abstract This work presents an approach to utilize high resolution surface data as a-priori information for three dimensional path planning at very low altitude. The major challenge is preserve features while reducing the amount of data to a minimum. Non significant height points are eliminated by a neighbor search, performed within a data structure generated from pseudo 3D Delaunay meshing. A comparison to an alternatively implemented simplification shows which inherent building features can be preserved. For highlighting the feasibility of the approach, the processing results of real urban surface data from the inner city of Berlin is presented and used for sampling based path planning of an unmanned helicopter.

Keywords Surface meshing · Mesh simplification · Mission planning

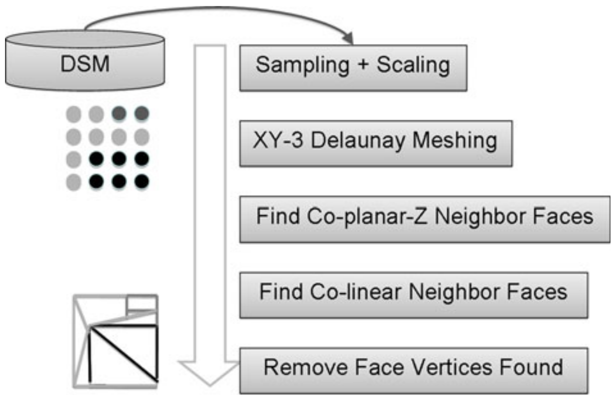
1 Introduction

Surveillance and reconnaissance missions inside urban environments show a great potential for small, low flying unmanned aircraft. Small vehicles may fly quickly into narrow areas or into buildings for providing additional situational awareness. A challenging problem is autonomous obstacle avoidance, low above the ground, without relying on a real-time, highly reliable data link to a remote aircraft operator. Hence, such mission profiles demand automated path planning in all three dimensions. One approach for path planning in such environments is sampling based motion planning

F.-M. Adolf (✉)
Department of Unmanned Aircraft, DLR Institute of Flight Systems,
Lilienthalplatz 7, 38108 Braunschweig, Germany
e-mail: florian.adolf@dlr.de

H. Hirschmüller
Department of Perception and Kognition, DLR Institute of Robotics and Mechatronics,
Münchner Straße 20, 82234 Oberpfaffenhofen-Wessling, Germany
e-mail: heiko.hirschmueller@dlr.de

Fig. 1 The proposed DSM conversion process and reduction into a polygonal surface


DSM
Sampling + Scaling
XY-3 Delaunay Meshing
Find Co-planar-Z Neighbor Faces
Find Co-linear Neighbor Faces
Remove Face Vertices Found

[1, 2]. However, such systems need to perform rapid collision tests when their initial, sampling based free space representation is constructed. Rapid collision checking against arbitrary shaped, unclosed obstacle surfaces is often achieved using polygonal obstacle representations. Online obstacle update methods that use initial knowledge about surfaces during an update process, have been presented by [3, 4]. However, unmanned aircraft operators may wish to obtain initial path plans that do not change drastically during mission execution. Hence, the more is known a-priori, the lower the chance of large path plan deviations from the initial solution.

A conservative meshing and simplification process is proposed that converts Digital Surface Models (DSM's) to polygonal surfaces suitable for sampling based path planning. A general scheme of the proposed approach is shown in Fig. 1. Basically, any set of points representing a surface profile can be fed into the five step process, at the end of which the output is a simplified set of adjacent polygonal faces that interpolate the surface.

A practical problem is to provide a polygonal obstacle representations for realistic mission environment scales (e.g. city center area of 4 km^2). Obstacles can be reconstructed manually for a set of selected objects like buildings[1,2] or other man made objects like airports.[3] But for large urban areas this can become an error prone task and may yield wrong simplifications or displacements. Another approach is to seize detailed surface information that is obtained as high resolution DSM [5, 6]. Although obstacles are constrained to be represented as height points (e.g. no information about free space under bridges), such top view obstacles are a feasible and precise a-priori knowledge for path planning.

However, a set of challenges need to be considered for using high resolution DSM's with a sampling based three dimensional path planner. Firstly, common polygon reduction algorithms were designed with real time performance improvements in mind [7, 8] or a general data reduction [9–11]. They do not aim at conservative preservation of important surface characteristics that is essential for collision free paths (e.g. around thin walls or pillars). This is different from the requirements

[1] Google Earth 3D objects warehouse, http://sketchup.google.com/3dwarehouse/

[2] Bing Maps 3DVIA, http://www.microsoft.com/maps/product/3dvia.aspx

[3] Virtual Terrain Project, http://www.vterrain.org

in computer graphics. Secondly, reducing the overall number of data points is one key factor to achieve feasible planning speed. Finally, in case of three dimensional planning, a DSM needs to be transformed into polyhedral surfaces. Note that a key property of the used mesh data structure allows to represent gaps or displacements between adjacent polygons which can become the case in realistic world mapping scenarios of the intended application [12]. Many existing simplification algorithms assume a closed surface or purely triangular polygons. Hence, they cannot easily be applied to the simplification problem at hand.

In this paper, a method for creating a simplified polygonal mesh from high-resolution surface data as a-priori knowledge for the application of path planning is presented. It tries to reduce the amount of data, while preserving all features that are relevant for the application.

In Section 2, the data acquisition is highlighted on which this work is based on. Thereafter, in Section 3, the used surface meshing procedures are explained. In Section 4, the implementation details on the mesh simplification are presented, which can be applied to any surface mesh data. Finally, Section 5 shows first results of how the simplified data can be used for 3D mission planning of an unmanned helicopter. The work is concluded in Section 6 with an outlook on future directions.

2 Acquisition of a Digital Surface Model

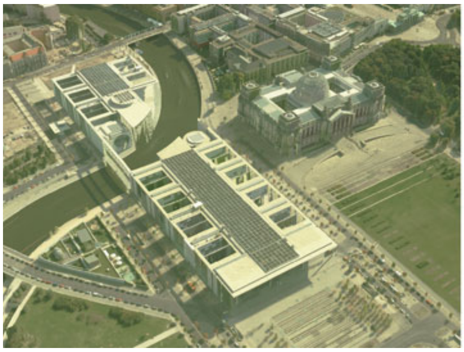
Digital Surface Models are created from images of classical aerial push broom or full frame cameras, but can equally be generated from satellite imagery. The used camera is assumed to be intrinsically calibrated. The extrinsic parameters like position and orientation at every imaging location are typically recorded by highly sophisticated IMU/DGPS systems and corrected by corresponding feature points with bundle adjustment using off-the-shelf photogrammetric tools.

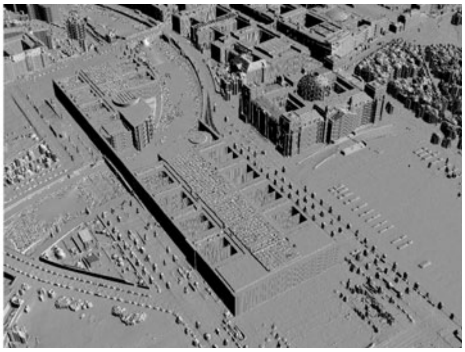
Overlapping images are matched by the Semi-Global Matching (SGM) method [5, 6] that performs pixel-wise stereo matching using a radiometrically robust cost like Mutual Information or Census [13]. Pixel-wise matching is supported by a global smoothness constraint that prefers neighboring depth values to be similar, but allows sharp depth discontinuities at any pixel. The SGM method is quite accurate at a relatively low runtime.

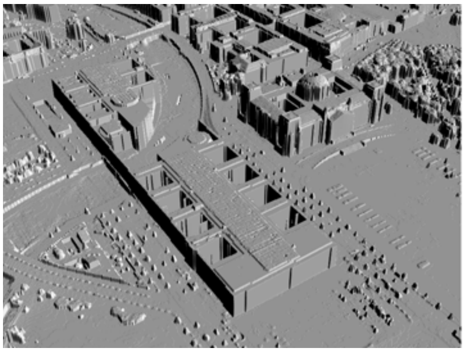
Stereo matching of images is performed pairwise to all overlapping images in flight-direction as well as perpendicular to it. Each image pair is also matched in reverse direction and both depth images are tested for consistency, which is known as the left/right consistency check. This results in a highly redundant and complex geometric representation of the scene by many perspective depth images that correspond from their geometry to the original images. For simplification of the representation, all pixels of all depth images are reconstructed and projected into a geo-referenced, orthographic DSM. The redundancy is removed by using the median of depth values for each cell of the DSM, which typically removes any remaining outliers.

The whole processing chain works fully automatic. Figure 2a and b show the reconstruction of a small part of Berlin that has been captured by UltraCam-D[4]

[4]http://www.microsoft.com/ultracam


(a) Textured Model from UltraCam, 7 cm/pixel


(b) DSM from UltraCam, 7 cm/pixel


(c) DSM from UltraCam, downscaled to 1 m/pixel

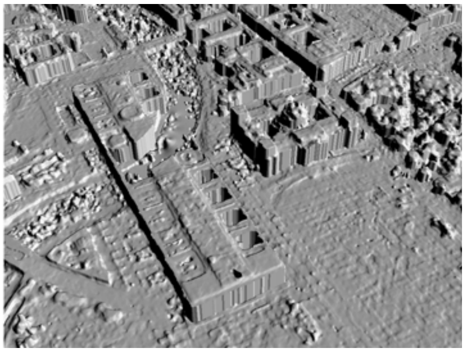

(d) DSM from World View, 0.5 m/pixel

Fig. 2 Textured and untextured DSM of Berlin from UltraCam and world view data

with an overlap of 80% in flight direction and 70% perpendicular to it. The ground resolution of the images and the resolution of the DSM is 7 cm/pixel. Texturing of the DSM is also done fully automatically from the aerial images. Since the resolution of the model is very high, the DSM has been downsampled for all tests of this paper to a resolution of 1 m/pixel, which is shown in Fig. 2c. It is important to note that due to the low resolution requirements, even satellite imagery would have been sufficient for the purpose of path planning. Figure 2d shows the same scene as reconstructed from World View data that has been captured with 0.5 m/pixel.

3 Surface Meshing

For converting DSM data into a polygonal surface, a meshing step generates graph-like connections between all points, such that a closed set of adjacent polygons interpolates the point set. The DSM is meshed using the *XY-3* method from [14], which is specifically designed for triangulation of height points into a triangular mesh. Furthermore, a Delaunay triangulation [15] is used which maximizes the minimum angle of all the angles of the triangles in the triangulation in order to avoid skinny triangles.

Table 1 Triangulation of three test sections from inner city of Berlin without simplification

Section	x/y-Res. [m]	Vertices	Faces	Size [MB]
Mitte	1	2,450,000	4,893,702	730
Mitte	2	612,500	1,221,852	182
Mitte	3	272,728	543,356	81
Mitte	4	153,300	305,026	45
Potsdamer	1	276,051	550,000	80
Potsdamer	2	69,000	136,950	20
Potsdamer	3	30,728	60,756	9
Potsdamer	4	17,250	33,976	5
Reichstag	1	113,176	225,000	33
Reichstag	2	28,388	56,100	8
Reichstag	3	12,600	24,750	4
Reichstag	4	7,144	13,950	2

The vertical resolution is always 10 cm

Triangulation in three dimension can be constrained into a 2D problem, by considering that the DSM is a 2.5D representation. A 2D Delaunay triangulation is created based on an Euclidean metric for the vertical projection of the surface model. That is, each of the points has an elevation such that they are 3D points, but the predicates used to build the 2D triangulation are computed using only the x and y coordinates of these points. Thus, it is also referred to as XY-3 triangulation. Figure 3 shows an example of the initial mesh output. In this work, a safety distance of two meters around the unmanned aircraft is used during path planning. Thus, from a collision avoidance perspective, the level of detail for planar building surfaces, e.g. walls or roofs, can be simplified within the needed path planning precision. Since the initial mesh needs to be altered to perform the simplification, the used implementation allows to efficiently change vertices and facets. Note that altering the mesh does not invalidate the Delaunay property of each vertex relation. If vertices are modified or removed, this framework aims at re-meshing affected vertices to Delaunay partitions again.

Throughout this work, three example sections have been extracted from DSM data of the inner city of Berlin. First, "Mitte" refers to a major section of the inner

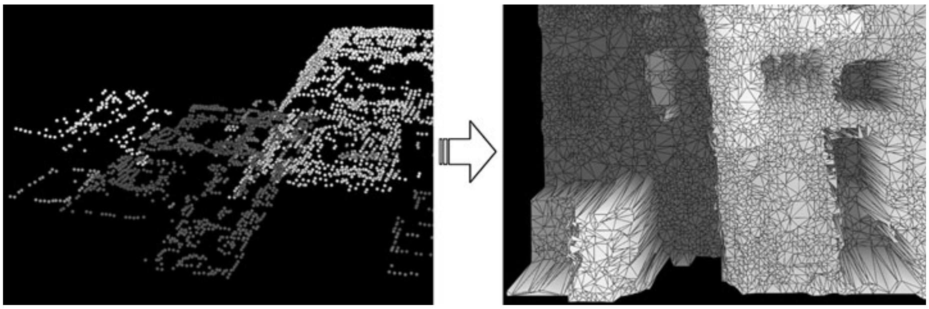


Fig. 3 Example of an urban building's height points (*left*) with 10,000 m^2 size, meshed using XY-3 technique

district area with 1,750 × 1,400 m^2 in size. Within this area two smaller section sites of different building density are defined. The Reichstag building site with 500 × 300 m^2 area size has a wide open field and some narrow inner building courtyards. The Potsdamer-Platz site with 500 × 650 m^2 area size covers dense building structures and narrow street corridors. From the initial meshing results in Table 1, it can be observed that even the smallest example section (i.e. Reichstag) can exceed 100,000 polygon faces. For non-specialized computing hardware, it is a challenge to perform polygon intersection tests in acceptable runtime during path searches.

4 Mesh Simplification

The goal of a simplification process is to reduce the triangle set by joining adjacent similar faces together while preserving the object characteristic (e.g. corner vertices). One way to achieve this goal is to search the mesh for vertices that could be removed. In this case, mesh simplification problems can be considered as a problem of initial values. The output quality of a simplified mesh depends on the initial set of vertices that are marked for removal. The proposed simplification process aims at a conservative removal of vertices. That is, preserving any features relevant for collision detection within a tolerance margin (e.g. two meter safety distance to the unmanned aircraft). This approach relates to ideas from [16] where local neighborhood searches identify data points to be discarded. However, the approach in this work performs a search on the mesh rather than the DSM directly. The mesh uses the half edge data structure from [17] which allows significantly more efficient searches for candidates in adjacent vertices.

Given the initial dense mesh, a search for insignificant vertices is performed. As opposed to iterative approaches focusing on real-time performance, a subset of vertices is searched in one step over the whole mesh. Also, the size of the candidate sub-set does not depend on a global threshold, e.g. a fixed number of polygons to be removed from the mesh. Obviously, it makes sense to get rid of small height differences first. Hence, the first measure explores vertical differences, as shown in Fig. 4. To find insignificant vertices a local measure circulates through each direct neighbor of a vertex and compares each height difference. If all neighbors are within a given tolerance Δz, the vertex is marked for removal. During the visit of all vertex neighbors, some vertices have been visited or even marked already. This can slow down the search process significantly. Hence, during the search process a set of unique search hits is updated and used to ignore redundant results.

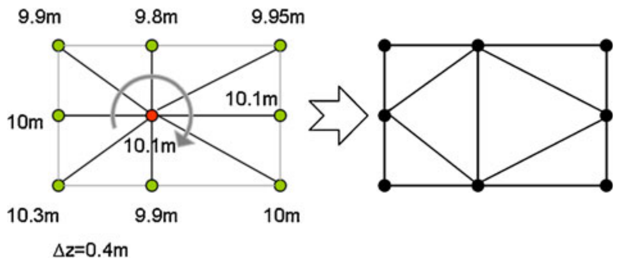


Fig. 4 Scheme of the horizontal plane simplification: all height values of the neighbors around the center vertex must be within the tolerance interval Δz (*left*). If a vertex has been removed, an automatic re-triangulation of the affected faces is performed

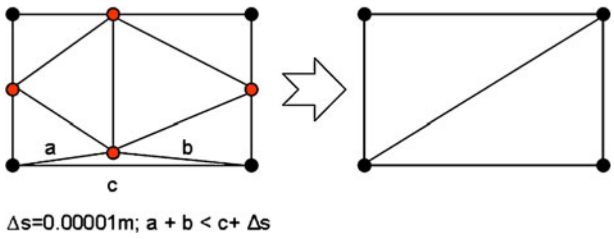


Fig. 5 Scheme of the co-linear plane simplification. Vertex B is removed if it is co-linear within a given tolerance on an edge between two neighbor vertices. Automatic re-triangulation

The second measure explores neighbor vertices within a co-linearity tolerance ΔCo. That is, the path length $a + b$ over a vertex A between two other vertices B and C is not longer than a direct connection c between B and C (Fig. 5). Note that local vertex searches and removal from the mesh are separated into two steps. All found vertices are deleted at the end of the selection process. Otherwise, global measures must be taken to avoid an accumulation of local tolerance errors. Additionally, based on the experiences reported in [7], the development of a global simplification measure is not considered as it is hard to implement with reasonable runtime complexity. Note that this particular implementation of the algorithm works with triangles only. However, nothing is lost by this inherited constraint from the Delaunay triangulation as any polygon with more than three points could be triangulated. Based on experimental experiences on the three test sections, it turns out that the two local simplification steps are more feasible when applied consecutively with an additional constraint.

The initial approach, referred in this work as *variant one*, is to find adjacent co-planar vertices in the horizontal plane and then merge remaining co-linear ones. Additionally, *variant two* has a slightly different second merge step. It merges only those co-linear vertices if they are within a tolerance and if co-linearity is not in the horizontal plane. The necessity for this additional constraint becomes clearer as it

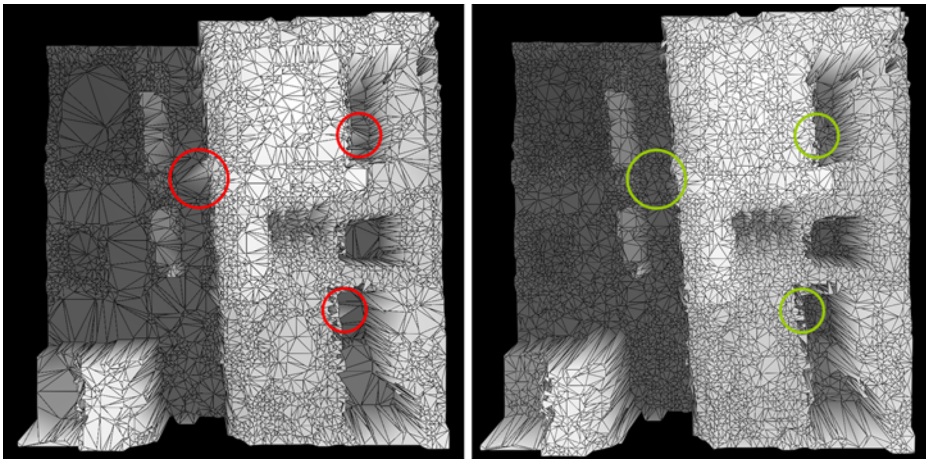


Fig. 6 Example of the output for consecutive face merging in approach variant one (*left*) vs variant two (*right*). The markers highlight critical simplification side effects

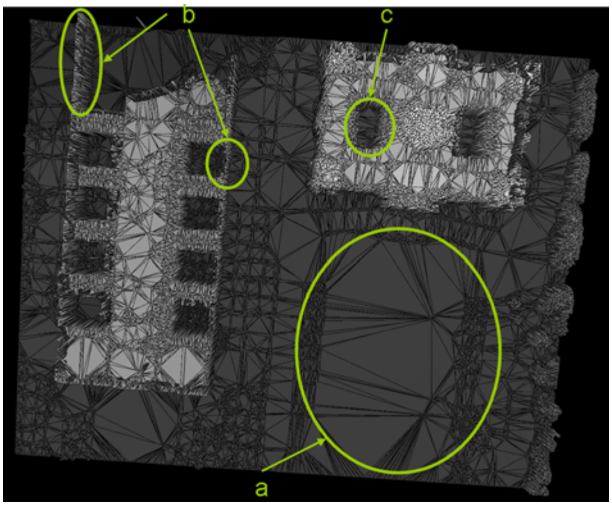


Fig. 7 Example of the output for face merging using approach variant two. The markers highlight proper simplification of flat areas (**a**) into fewer triangles while preserving comparably thin walls (**b**) and inner building courtyards (**c**)

yields an undesired accumulation of tolerances for ground planes. The first merge step allows fairly high differences in height such that any further tolerance in the horizontal plane should be avoided. An example of side effects can be seen in the marked areas of Fig. 6. Moreover, Fig. 7 shows how the additionally constrained search avoids those side effects. Note that parameters were manually tuned to $\Delta z = 0.4$ m and $\Delta Co = 10^{-5}$ m to avoid drastic simplification errors; these remain constant for all tests.

Table 2 shows the simplification results achieved by variant two. Although additionally constrained and with conservative tolerances in height and co-linearity, a considerable reduction of the data can be achieved. Naturally, the larger the sampling step size over the initial DSM, the less vertices are within the given tolerances such that the reduction of the initial mesh drops from more than two third to approximately one third.

Table 2 Simplification using variant two: results for processing initial meshes of the three urban example areas

Section	x/y-Res. [m]	Vertices		
		Simplified	Co-planar	Co-linear
Mitte	1	944,873 (−62%)	1,505,127	24,284
Mitte	2	338,182 (−45%)	268,247	6,071
Mitte	3	177,749 (−35%)	92,552	2,427
Mitte	4	109,757 (−29%)	42,301	1,242
Potsdamer	1	86,448 (−69%)	187,893	1,710
Potsdamer	2	33,151 (−52%)	35,109	740
Potsdamer	3	17,981 (−41%)	12,410	337
Potsdamer	4	11,303 (−35%)	5,769	178
Reichstag	1	27,163 (−76%)	85,615	398
Reichstag	2	10,770 (−62%)	17,510	108
Reichstag	3	5,855 (−53%)	6,619	126
Reichstag	4	3,692 (−48%)	3,382	70

Finally, a comment on the expected complexity of the algorithm is stated. In worst case the incremental local search marks not a single data point for removal from the mesh. In such a case:

1. n nodes are searched with O(n).
2. A binary search in a candidate set C is performed in order to check if the local neighborhood of current node m has been examined already. Thus, C with potentially increasing set size k are searched with O(k*logk).
3. If node m is not a candidate for removal, check all j neighbors of the current node m with C (again O(k*logk)). Moreover, one can assume a fixed number of neighbors for triangular mesh surfaces with $j = 3$.
4. Hence, the approximate complexity is n*j*2*O(k*logk), with k potentially increasing while iterating through 0..j node neighbors.

5 Surface Polygons for Path Planning

The path planner presented in [18] is used which implements a multi-query strategy based on a free space search graph representation also known as *road map*. It comprises a Probabilistic Road Map (PRM) which has become a popular method to implement path planners primarily designed for multiple query motion planning problems. Regardless of which sample distribution technique is applied to construct the initial road map, it is important to validate such a planing system with high resolution data. As mentioned in the beginning, full 3D path planning systems can deal with arbitrarily complex obstacles by which e.g. the open space under bridges, windows and alike can be represented. As closed polygonal surfaces can be considered as a subset of such obstacles, it is desirable to use the surface directly in the most general representation. The used path planning system treats each polygon face individually such that facets do not differ between inside and outside. The drawback of this design is, that adjacent facets store points redundantly. However, for the tests with the given urban sections this increase in memory allocation does not have a major impact on the initial road map construction phase.

Results for effective planning times are particularly interesting when obstacles are dense, like in narrow street canyons. A planning result visualization shown in Fig. 8

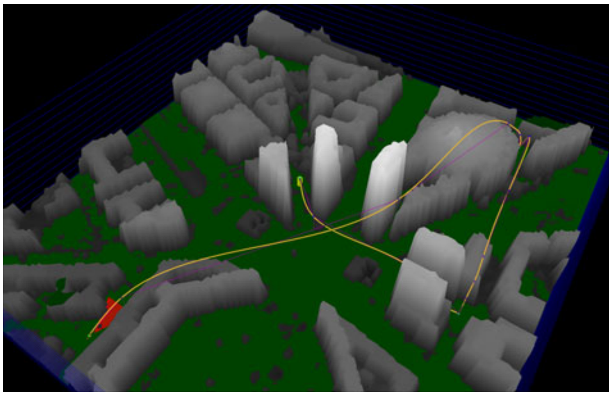


Fig. 8 Visualized path planning result of the test section Potsdamer Platz

shows a snapshot from one of the planning tests performed inside an urban scene covered densely with high buildings.

In Table 3, processing times for all example sections are shown. The runtime for the simplified mesh is compared with the time needed for a non-simplified mesh. The tests are performed with a single threaded C++ planner implementation, run in 32-bit mode on a Pentium Core 2 processor with 2 GB working memory. In these tests, the path planner has to find a collision free path between four widely distributed waypoints. Each of the waypoints is about 5 m close to walls or to the ground in order to simulate intensive intersection tests during the connection to the roadmap. The roadmap is based on Quasi-Random Halton sampling [19] with an approximate relative sample distance of 20 m and 34.5 m maximum edge length. Although the road map construction times shown include the sample distribution, times of approximately less than a second have an insignificant share. Times shown for planning represent an A*-based path search. The time measurements include the local connection of each user waypoint to the road map. The sample connection strategy is based on a locally bounded neighbor search around each user waypoint, followed by intersection tests of straight line connections with each neighbor sample.

Results in Table 3 indicate clearly that simplification can reduce processing times by more than 50% for tasks that comprise many intersection tests with polygonal faces. It also shows that sampling and planning occurs within seconds even for fairly large urban environments (esp. "Mitte" scenario) although detailed surface data is used for path planning. Especially for larger sites like "Mitte", the experiments show that the presented simplification allows to use a higher resolution (e.g. two meter) with similar processing times as for non-simplified mesh at a lower resolution (e.g. three meter). Though time savings for fairly open and small sections like "Reichstag" are comparably low, when areas of similar size become densely built ("Potsdamer") or areas become much larger ("Mitte") the simplification becomes a more important feature.

Likewise the effective planning time can decrease with less polygonal details "disturbing" the graph's edge connection, such that less nodes need to be searched on the roadmap. However, planning time results also indicate that (quasi-)randomness

Table 3 Comparison of planning times in simplified vs. original urban meshes

Section	x/y-Res. [m]	Runtimes [s]					
		Loading		Roadmap		Planning	
Mitte	1	96.10	(−61%)	122.50	(−88%)	5.33	(−18%)
Mitte	2	33.66	(−46%)	36.10	(−27%)	3.87	(−19%)
Mitte	3	17.48	(−36%)	25.76	(−11%)	3.28	(−32%)
Mitte	4	10.86	(−28%)	19.50	(−13%)	3.58	(−8%)
Potsdamer	1	8.94	(−69%)	46.59	(−36%)	2.92	(−25%)
Potsdamer	2	3.33	(−56%)	33.63	(−10%)	2.95	(−20%)
Potsdamer	3	1.78	(−41%)	32.70	(−5%)	2.97	(−14%)
Potsdamer	4	1.14	(−40%)	28.28	(−2%)	2.53	(−27%)
Reichstag	1	2.77	(−75%)	7.56	(−74%)	0.69	(−42%)
Reichstag	2	1.06	(−60%)	3.75	(−41%)	0.50	(−6%)
Reichstag	3	0.56	(−60%)	2.86	(−41%)	0.54	(−19%)
Reichstag	4	0.39	(−45%)	3.19	(−4%)	0.46	(−18%)

Note, the "Mitte" scenario used a lower sampling density (40 m rel. distance) in roadmap construction avoids deterministic estimates on how much the planning time can decrease for a specific mission.

Certainly, the shown processing times shown in Table 3 may vary compared to other path planning implementations. Nonetheless most implementations share the property that collision checks with polygons have a major share in the most expensive computations. For the roadmap based path planner used in this work, collision tests are performed during the initial roadmap construction phase. Additionally, after the graph search algorithm found a path on the roadmap, intersection tests are performed for path smoothing (e.g. short cuts and spline-based interpolation). Local path planners like Rapidly-exploring Random Trees (RRTs) [20, 21] can also be used, especially for solving the case of non-holonomic motion. In this case, no roadmap would be constructed a-priori and thus all collision checks would be performed during an instantaneous local sampling and sample connection step. Thus, it can be assumed that other path planning approaches are likely to scale similarly during their collision intensive part, if they use the same obstacle representation and intersection test technique.

6 Conclusion

This work presents an integrated approach to use high resolution DSM data converted into polygonal obstacles for 3D path planning. Original surface data from an urban environment is tested with a path planning of an unmanned helicopter system. This is achieved by balancing between detail preservation and surface simplification. The primary advantage of the presented simplification algorithm is the quality of the produced mesh both geometrically and visually. Although simplification parameters are conservatively tuned, the preservation of surface features is achieved while reducing the amount of polygon faces by more than 60%. This way, it is possible to automatically convert a DSM into an adjacent polygon set for the application of path planning. As in case of the planner used in the presented examples, many 3D path planning systems use polygonal obstacle information. Having polygons available as a-priori obstacle knowledge for very low altitude missions, bridges the gap to have DSM's in a polygonal world representation. Tests with a sampling-based path planning system indicate that the simplification process reduces collision test intensive processes (i.e. roadmap construction and path smoothing) by more than a half, too. Furthermore, precise a-priori surface models allow larger scale online updates, e.g. from polygonal world mapping [12].

Future research should find a precise error metric as well as a model for the two simplification parameters. Also the applicability of less conservative polygon reduction methods should be exploited. One possible direction is presented in [22] which is based on progressive edge collapses. A second promising approach in [23] extends the presented idea of this work by a vertex clustering step and facet constrictions.

Acknowledgement This work benefited from algorithms and data structures of the CGAL library.

References

1. Pettersson, P.D.P.O.: Probabilistic roadmap based path planning for an autonomous unmanned helicopter. J. Intell. Fuzzy Syst. **17**(4), 395–405 (2006)

2. Plaku, E., Bekris, K., Chen, B.Y., Ladd, A.M., Kavraki, L.E.: Sampling-based roadmap of trees for parallel motion planning. IEEE Trans. Robot. **21**, 597–608 (2005)
3. Hrabar, S., Corke, P., Sukhatme, G., Usher, K., Roberts, J.: Combined optic-flow and stereo-based navigation of urban canyons for a uav. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 302–309 (2005)
4. Andert, F., Goormann, L.: Combined grid and feature-based occupancy map building in large outdoor environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2065–2070 (2007)
5. Hirschmüller, H.: Accurate and efficient stereo processing by semi-global matching and mutual information. In: Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 2, pp. 807–814. Washington, DC, USA (2005)
6. Hirschmüller, H.: Stereo processing by semi-global matching and mutual information. IEEE Trans. Pattern Anal. Mach. Intell. **30**(2), 328–341 (2008)
7. Garland, M., Heckbert, P.S.: Fast polygonal approximation of terrains and height fields. In: SIGGRAPH (1996)
8. Lindstrom, P., Koller, D., Ribarsky, W., Hodges, L.F., Faust, N., Turner, G.A.: Real-time, continuous level of detail rendering of height fields. In: SIGGRAPH '96: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, pp. 109–118. ACM, New York, NY, USA (1996)
9. Garland, M., Heckbert, P.S.: Surface simplification using quadric error metrics. In: SIGGRAPH '97: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, pp. 209–216 (1997)
10. Frey, P., Borouchaki, H.: Surface mesh evaluation. In: 6th International Meshing Roundtable, pp. 363–374 (1997)
11. Davis, C.: Statistics and Data Analysis in Geology, 3rd edn. John Wiley and Sons (2002)
12. Andert, F., Goormann, L.: A fast and small 3-d obstacle model for autonomous applications. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2883–2889 (2008)
13. Hirschmüller, H., Scharstein, D.: Evaluation of stereo matching costs on images with radiometric differences. IEEE Trans. Pattern Anal. Mach. Intell. **31**(9), 1582–1599 (2009)
14. Brönnimann, H., Fabri, A., Giezeman, G.-J., Hert, S., Hoffmann, M., Kettner, L., Schirra, S., Pion, S.: 2d and 3d kernel, in cgal editorial board (2006)
15. Preparata, F.P., Shamos, M.I.: Computational Geometry—An Introduction. Springer (1985)
16. Mello, F., Strauss, E., Oliveira, A.A.F., Rocha Gesualdi, A.: Non-uniform mesh simplification using adaptative merge procedures. In: GeoInfo, pp. 61–378 (2004)
17. Kettner, L.: Halfedge data structures, in cgal editorial board (2007)
18. Adolf, F., Langer, A., Melo Pontes e Silva, L., Thielecke, F.: Probabilistic roadmaps and ant colony optimization for UAV mission planning. In: 6th IFAC Symposium on Intelligent Autonomous Vehicles (2007)
19. Branicky, M.S., LaValle, S.M., Olson, K., Yang, L.: Quasi-randomized path planning. In: IEEE International Conference on Robotics and Automation, pp. 1481–1487 (2001)
20. Bruce, J., Veloso, M.: Real-time randomized motion planning for multiple domains. In: RoboCup 2006: Robot Soccer World Cup X, pp. 532–539. Springer, Berlin (2007)
21. Neto, A.A., Macharet D.G., Campos, M.F.M.: On the generation of trajectories for multiple UAVs in environments with obstacles. J. Intell. Robot. Syst. **57**(1–4), 123–141 (2010)
22. Jia, S., Tang, X., Pan, H.: Fast Mesh Simplification Algorithm Based on Edge Collapse. Springer Berlin / Heidelberg (2006)
23. Hussain, S., Grahn, H., Persson, J.: Feature-preserving mesh simplification: a vertex cover approach. In: IADIS International Conference on Computer Graphics and Visualization (CGV 2008), pp. 270–275. Amsterdam, The Netherlands (2008)