Guided 2D Modeling of 3D Buildings using Oriented Photos

Lisa Kellner*
Supervised by: Michael Schwärzler†

VRVis Research Center
Vienna / Austria

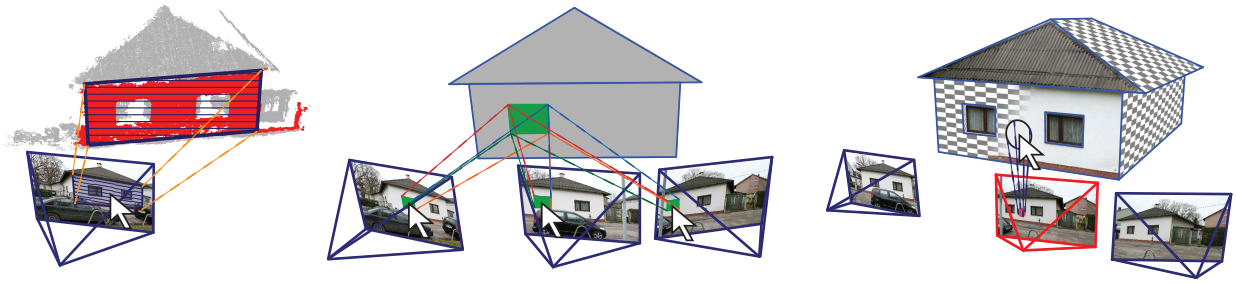

Figure 1: Modeling operations taking both oriented images and point cloud data into account. Left: Point Cloud-supported Single Shot sketching, exploiting planar structures in the data. Middle: Multi-View Shot View sketching. Right: Texturing the generated polygons using an interactive brushing method

Abstract

Capturing urban scenes using photogrammetric methods has become an interesting alternative to laser scanning in the past years. For the reconstruction of CAD-ready 3D models, two main types of interactive approaches have become prevalent: One uses the generated 3D point clouds to reconstruct polygonal surfaces, while the other focuses on 2D interaction in the photos to define edges and faces.

We propose a novel interactive system that combines and enhances these approaches in order to optimize current reconstruction and modeling workflows. Our main interaction target are the photos, allowing simple 2D interactions and edge-based snapping. We use the underlying segmented point cloud to define the 3D context in which the sketched polygons are projected whenever possible. An intuitive Visual Guiding interface gives the user feedback on the accuracy to expect with the current state of modeling to keep the necessary interactions at a minimum level.

Keywords: 3D-Modeling, Guidance, Photogrammetry

1 Introduction

The use of three-dimensional point cloud data of buildings gathered with different sensors is becoming part of the standard workflow in surveying and mapping – may it be from tachymetric devices, laser scans, or photogrammetry. While solutions for acquisition, storage and viewing of the generated point clouds have become commercially available, the derivation of low-polygonal, CAD-ready models still poses a mostly unsolved challenge.

Only recently, researchers have started tackling this problem by not only using meshing algorithms to triangulate the point cloud, but to detect and use underlying structures first in order to use geometric primitives to represent the building (see Section 2). By doing so, the emerging 3D models correlate a lot more to the way a human artist would reconstruct a building in a 3D modeling tool: Not only is the number of polygons usually considerably lower, they also contain sharp edges and hierarchical definitions, and are therefore a lot easier to manipulate interactively.

Still, reconstructing complete data sets is often hardly possible due to limitations in the point data: Laser scanners cannot be arbitrarily positioned, so that it is common that parts of the data are missing. Tachymetric point clouds are too sparse for reconstruction, and photogrammetric point clouds often have large holes caused by uniformly colored areas in them (see Section 3). This shortcoming in photogrammetric point clouds can be partly compensated by considering the source photographs as well, making it possible to identify and model sharp building edges using line features that can be identified in images more easily.

In this paper, we pursue the concept of using both point cloud and photo data together, but combine it with the approaches from Sinha et al. [18] or Arikan et al. [2] of simplifying interactive 3D modeling to a 2D problem whenever possible: We propose an interactive modeling approach based on oriented photos, in which the user

sketches the desired geometry directly on the image using simple 2D operations, supported by an edge-based snapping feature. Based on the available data quality of the underlying point cloud, the 2D polygon is either directly projected to 3D space whenever possible, or the user is required to define the polygon geometry in multiple further photos, until a unique mapping to 3D space can be defined. This interactive process is supported and guided by providing suggestions for potential polygon candidates in neighboring images, as well as by giving visual feedback on the estimated accuracy for the projection to 3D space. This allows the user to make use of both point cloud and image data, while relying on an optimal, flexible workflow with minimal manual intervention.

2 Related Work

The field of urban reconstruction has gained a lot of scientific attention in the last years. A complete overview is out of scope for this paper, and we refer the interested reader to the recent state-of-the-art report by Musialski et al. [14]. Instead, we put our focus on methods that incorporate photos in the geometric reconstruction pipeline, or that try to simplify the 3D modeling by either reducing the interaction dimensions or providing suggestions to the user.

From a user-oriented perspective, our novel system is most closely related to systems presented by Debevec et al. [5] or Sinha et al. [18]: These interactive tools also rely on image-based modeling operations, and use photogrammetric data sets to calculate geometric correspondences in order the reconstruct 3D geometry. While operations like snapping to edges in images or multi-view texture generation have been integrated in these tools as well, neither of them exploits the availability of the underlying photogrammetric point cloud in order to simplify the sketching progress as in our method. Furthermore, we introduce an additional *guiding indicator* in the graphical user interface that operates as a feedback provider to give the user an easy-to-grasp preview on how much more modeling work is needed (see Section 5.3).

The derivation of polygonal meshes from point clouds has been intensively studied in recent years [11, 3, 10, 1, 16], Wang et al. [20] use additional image-based data in their interactive tool in order to regularize the building by proposing a scaffold-like structure. Still, especially in the domain of urban reconstruction, the resulting 3D buildings differ significantly from typical models designed with CAD tools: While human users construct building models consisting primarily of geometric primitives with exact intersections, meshed point clouds are inherently noisy and contain holes. Additionally, the absence of hierarchical relations makes operations like geometric editing or semantic classification cumbersome.

To tackle these problems, Arikan et al. [2] have proposed an interactive method that first identifies the basic planar shapes in a point cloud, on which initial coarse polygons are created. Holes between the polygons are automatically closed by an optimization step whenever possible. In unclear cases, the user can edit, fix or add polygons using simple 2D operations on the corresponding segmented plane. Unfortunately, this approach relies on point clouds that resemble nearly the whole surface, and especially in the case of photogrammetrically generated data, the often-occurring large holes cannot be accurately reconstructed. Reisner-Kollmann et al. [15] proposes using image information for filling holes in the surface in an automatic method. In our approach, we employ the ideas of both these approaches: planar surfaces in the point cloud are identified and used as a sketching plane for 2D modeling interaction – but by using the photos as additional input in the 2D domain, the sketched polygons can additionally snap to image edges, and polygons for which no point cloud data is available can be accurately reconstructed.

Our novel work is therefore a combination and extension of the before-mentioned interactive modeling tools, striving for simplicity in terms of modeling operations (2D image-based sketching and snapping) and exploiting structural information (planar point cloud segments, image edges) from all data sources available – while helping and guiding the user through the process and leaving all decisions to his artistic freedom.

3 Photogrammetric Data

Figure 2: Photogrammetric Network (bottom), consisting of a 3D point cloud and photos (top), for which their relative positions and orientations have been computed. We refer to them as *Shots* in this work.

As this work focuses on interactive modeling using photogrammetric data, we describe the properties and distinctive characteristics of this input type: A *Photogrammetric Network* consists of a 3D point cloud and images, which are overlapping photos of an object or – like in our case – buildings. By applying *Structure from Motion (SfM)* techniques, the relative positions (i.e. the location and the ori-

entation) on which the photos were taken from, as well as a 3D point cloud, consisting of matching image features that have been reprojected to 3D space, can be computed (see Figure 2). The point cloud can be further densified using algorithms proposed by Furukawa et al. [8, 7], but since these points have not been measured, but were calculated using image features, photogrammetrically generated points may not be as dense and – more important – not as uniformly distributed as point clouds from laser scans, leading to more holes in the data. For example, it is difficult to extract robust features and therefore closely spaced 3D points from completely flat, featureless walls. We therefore strive for compensating this missing information by defining polygons in multiple photos, see Section 4.1.

The oriented photos – we refer to them as *Shots* – in the Photogrammetric Network are positioned around the point cloud. By having access to intrinsic and extrinsic camera parameters, transformations from the 2D image space to the 3D world space and vice versa can be achieved. In the case of our work, this is necessary for simple 2D editing and sketching steps and their according impact on the 3D world space, see Section 5.

Another advantage of the availability of Shots is their use in further reconstruction steps, as for interactive line snapping or texture generation. Furthermore, the acquisition process can be done with a consumer-level photo camera and freely available SfM-Tools, making it a cheap and easy solution compared to other methods.

4 Definition of Polygons using Shots

The primary interaction and sketching target in our framework is a Shot, selected from a photogrammetric network as described above. Sketching directly in a Shot photo for the purpose of creating 3D geometry has two major advantages in terms of usability:

- The user immediately grasps the scene to reconstruct, as a photo is a very close approximation of what one perceives when looking at an object.
- The interaction is performed in a 2D environment. This does not only make the modeling tools less complex to handle – humans are usually used to sketching or drawing on a flat sheet of paper since their early childhood.

While defining the approximate outline of a flat polygon in 2D space is therefore comparably easy to achieve, the derivation of the corresponding representation in 3D space requires additional information: The *3D plane* on which the 2D outline has to be projected from the photo is completely unknown at first, but can be calculated by taking additional constraints into account. We therefore propose three methods to estimate this needed information in an intuitive way with the least possible user effort, and without having to leave the 2D sketching domain.

4.1 Multi Shot Sketching

One method to obtain the 3D positions for the vertices of a sketched polygon in 2D image space is to define it not only in one, but in multiple photos. Since the orientation of the Shots is known in 3D space, each pixel on the image plane can be used to define a ray from the focal point of the camera through the pixel position in world space. If this is done for a polygon vertex in multiple images, the intersection point of the corresponding rays defines its 3D position (see Figure 1, middle). This is repeated for all vertices, and the unknown plane can then be estimated using the least-squares method.

We implemented these ray intersections using the linear triangulation method based on homogenous direct linear transformation (DLT) as described by Hartley and Zisserman [9] resulting in a least squares optimal solution. This approach is just one possible solution to this intersection problem. We opted for it as the authors state, that the "homogenous linear method [...] often provides acceptable results. Furthermore, it has the virtue that it generalizes easily to triangulation when more than two views of the point are available". We take this into account during our guided sketching feedback, where we encourage the user to define the polygon in more than 2 shots (see Section 5.3).

4.2 Point Cloud Supported Single Shot Sketching

Even though the multi-view approach described above is an algorithmically well-working solution, a human user would prefer to minimize ones efforts and wants the system to "understand" what one intended to do after sketching a polygon in a single Shot, and reproject it into 3D space. This can in fact be made possible by exploiting the point cloud data: Similar to Arikan et al. [2], we segment the point cloud into planar segments using the RANSAC algorithm by Schnabel et al. [17]. After an initial polygon has been sketched, we transform the points of each segment from 3D world space into 2D image space, and test which points of each segment lie inside the polygon. Note that in our current implementation, we perform this test for all segments, which could be easily optimized by performing a culling step (e.g. by using the bounding boxes of the segments).

We compute a heuristic $h \in [0,1]$ which gives us an estimation of how well a polygon fits a planar segment. We use the number of points lying *inside* the 2D polygon as well as the *uniformity* of the distribution of these points, i.e. whether the projected point cloud segment has "holes" in it. The uniformity is estimated by rasterizing all points as splats over the polygon and then determining a fill ratio r, where 1 means fully filled and 0 not filled at all.

$$h = \frac{mr}{n}$$

where n is the total number of points of the segment and m is the number of points inside the polygon.

Splat size q is based on the average point distance, where d_i is the distance between point i and its nearest neighbor.

$$q = \frac{1}{n}\sum_{i=1}^{n} d_i$$

If there is at least one segment that passes the (adjustable) acceptance threshold, we choose the one with the highest result as the potential candidate, and inform the user about the outcome (see Section 5.3). If the user decides to make use of it, the polygon is projected onto the plane that has been fit to the point cloud segment (see Figure 1). Otherwise, the user continues sketching the polygon in further views, and the Multi-view Shot Sketching algorithm is applied. Nevertheless, the initially found candidate segment can still be helpful: if the normal of the polygon calculated using the multi-view method differs only 10 degrees from the segment plane normal, the polygon is adjusted to it accordingly.

4.3 Sketching Using the Plane of Existing Polygons

Since it is obviously possible for a human user to assign a semantic meaning to polygons that are being sketched, it is often an easy task to recognize that some elements lie on the same plane in 3D space. This is especially the case for elements like windows, doors or balconies on a facade. We therefore allow the user to simply define the polygon of an existing element as the 3D sketching plane for the next polygon, and can therefore reproject the 2D outline to 3D space immediately.

5 Guided Polygon Creation

After providing information on the theoretical background on the View-based Shot Sketching in the previous Section, we describe how we integrated these concepts in interactive workflows that are designed to give the user an optimal modeling experience. All interactive concepts described in the following Sections only guide and support the user – despite all suggestions of our system, the assume that the "user knows best" what his intentions are. Every suggestion and guidance step in our system can therefore also be safely ignored by the user.

5.1 Shot View Navigation

As described above, all shot view sketching operations are performed in the 2D photos of the Shots for reasons of simplicity. Although the sketching takes place in the images, it is of utter importance that the user implicitly always knows about the current view location in the 3D world, so that the spatial context can be used to sketch polygons in multiple Shots and not mix them up.

During sketching, the user is presented a 2D view of the current photo. Nevertheless, since the corresponding Shot incorporates 3D information, we allow the opacity value to be changed arbitrarily, so that the 3D content (e.g. already modeled polygons or even the point cloud) can be made visible. Furthermore, the navigation between the shots has been designed to help to retain the information of the current user position: Instead of changing the displayed photo immediately, the camera starts a flying animation to show where the user is going. We also allow the user to leave the "Shot View" at any time and fly around in the 3D scene, and fly back to the current shot or the next shot with a smooth transition later on. For these reasons, the shots needs to be internally sorted according to their spatial neighborhood relation and not according to the time the photo was taken or even the file name. We therefore decided to sort the shots with a *Traveling Salesman* algorithm with the distance between the shot centers as weight function, resulting in an order that humans would intuitively describe as the proper natural way of describing the neighborhood relations.

5.2 Sketching and Snapping in Shot View

Once the user decides to start modeling a new polygon in a selected Shot, the initial polygon only needs to be sketched roughly on the photo, as we allow it to snap to near edges in the image. For this, we use an implementation of the Line Segment Detector described in the work of von Gioi et al. [19] to find edges in the underlying image. The outline of the initially sketched polygon is compared to the line set of the image. Two lines of these sets are matching if they are nearly parallel and spatially close. If no matching image line to a polygon edge is found, the initial edge will be used. To conclude the snapping process, the matching lines are intersected with each other, and the intersection points are the vertices of the new, snapped polygon. Figure 3 shows the polygon snapping workflow.

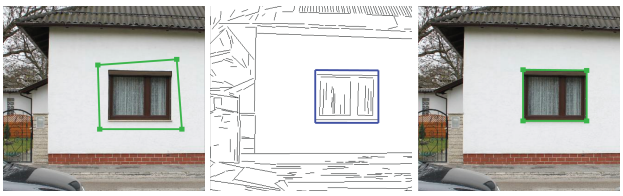

Figure 3: Polygon Snapping: sketched 2D polygon on an image (left), extracted image lines with matchings in blue (middle), snapped polygon (right)

If the user opts for using the Multi-View Shot Modeling mode, the modeling workflow requires the same polygon to be available in different images. Instead of having to sketch the corresponding polygon again, the proposed system tries to minimize the needed efforts: As soon as the user switches to the next Shot, the initial polygon is not only projected into the other image, but it is positioned to "fit the same sketched object". For example, if the polygon snapped to window edges in the initial image, the pro-

jected polygon in the neighbor image also tries to find and snap to the same window edges.

To achieve this, we transform the edges of the initial polygon outline into the underlying photo and compute normalized color histograms of them. Then the polygon outline histograms are compared with histograms of edges found in the target image to find matching lines. A match is a pair of lines, one from the polygon outline and its corresponding edge in the target image. The best polygon-outline image-edge pair is the basis of the polygon in the other shot. Further adjacent edges are stepwise added to the polygon depending on histogram matches, or approximated if no match was found.

Especially in the case of buildings, it becomes obvious why an interactive approach with minimized input that mostly consists of deciding on proposed suggestions is important: Architectural objects often consist of extremely similar and repetitive patterns, and an initially found window can be found multiple times on the next photo, resulting in the same amount of candidates for the correspondence. Since the user is able to retain a global spatial overview more easily, the correct window can be picked with a single click.

5.3 Visual Guidance Feedback

As stated before, we want to minimize the needed user interaction while keeping the possibility to influence any design decision at the user level. It is therefore important for the user to be provided with feedback on whether the polygon should be sketched in further views, or if enough information on the needed 3D plane is available to compute the world space position of the object.

During each polygon creation process, the user gets permanent feedback via our novel Visual Guidance interface to realize this: In the user interface, a state bar appears as soon as the initial polygon is sketched. The states can switch between *red* (not enough information), *yellow* (the system can suggest a 3D polygon, but it may be inaccurate or ambiguous) and *green* (an accurate polygon can be provided, and no other plane candidates can interfere). See Figure 4 for a visualization of the guidance element in the user interface.

Concretely, we use the red state whenever an initial polygon has been sketched, and no plane to project the 2D outline onto is available. This is the case when no neighboring 3D polygon has been selected (see Section 4.3) and no fitting planar point cloud segment can be found (i.e. the metric returns no value above a certain user-definable threshold for all point cloud segments, see Section 4.2). The yellow state is used when either two or more potential planar point cloud segment candidates that are of equal quality are available, or, when using Multi-view Shot Sketching mode, the polygon has only been defined in two Shots yet (which may be inaccurate, see Section 4.1). The user can stop sketching anytime this quality estimator is not in red state, and a 3D polygon is created – but has to be aware that the result may not be as accurate as needed when he stops too early.

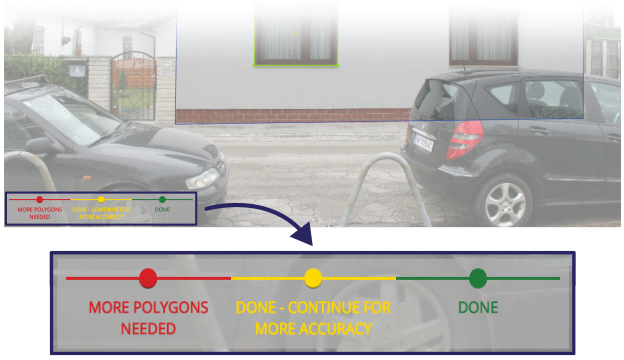

Figure 4: The Visual Guidance interface shows whether enough polygons have already been sketched to compute a 3D polygon, or if the user should continue sketching.

6 Additional Photo-based Modeling

Apart from sketching the initial polygons, we have integrated further possibilities that demonstrate the combined use of photo data, point clouds and geometry in a single environment.

6.1 Model refinement

All ordinary 3D polygon-modeling tasks in our system can also be performed via the Shot view. This is especially true for existing polygons that have not been modeled in this particular view, but can be reprojected and edited in the corresponding photo anyway. Following the same principle, polygons snap to edges, and can be aligned according to the image content: Model refinement through the shot view is more intuitive and accurate for a user than, for instance, fitting a polygon to the point cloud.

Figure 5: Left: By defining hierarchical relations, holes and side faces are automatically extracted. Right: Interactive removal of occluding objects from the texture by overlaying the photo semi-transparently using the Shot view.

In addition to the Shot-based sketching operations described in the previous Sections, our framework supports regular manipulation and polygon creation known from other 3D modeling packages. This comes handy in cases when not the whole building has been photographed, but

the surface needs to be closed (even though this cannot be seen as part of the reconstruction anymore). We also included an optimization-based snapping approach to close the small gaps between the polygons as proposed by Arikan et al. [2], which moreover takes further constraints like parallelism or orthogonality of edges into account – an often-needed requirement for CAD-ready models. Moreover, we allow the definition of hierarchical relations: This makes it easily possible to define "holes" for windows and doors in the facades (the corresponding side faces are added automatically), like in Figure 5, left.

6.2 Texture Brushing

The Shots can furthermore be used to generate textures for the created polygons by reprojecting the photos onto the 3D plane. We employed the technique proposed by Musialski et al. [13], where the texture is a composition of the photos, and each pixel gets colored according to the image of the best fitting shot. While the initial source shot of each pixel is initially selected automatically based on angle and distance, arbitrary parts of the texture can be "repainted" with the content of a user-selected shot photo in order to remove occluders or artifacts. Figure 6 shows textures with different coloring according to shots.

Figure 6: Top left: The initial texture containing occluders. Bottom Left: The associated Shots, visualized using a false color mask. Top right: The cleaned texture after interactive brushing. Bottom right: The correspondingly modified mask.

We extended the original method, in which the user could brush a polygon only directly in 3D view, to be also used via the Shot view, in which the transparency can be adjusted interactively. This way, the user simultaneously has access to the current state of the textured polygon in the 3D world, and the 2D Shot photo. In the Shot view brushing mode, the brush paints over the texture with the content of the shot image the user is actually looking at, so that the user can easily find the proper image part for a specific texture position (see Figure 5, right). Switching between Shots and leaving the Shot view is possible at any time as described in Section 5.1.

7 Implementation

Our novel modeling and reconstruction framework has mainly been implemented using an internal rendering framework based on the .NET framework and OpenGL. The Visual Guidance feedback element has been realized using a web-based overlay that was created using HTML5 and the D3.js toolkit [4]. The interactive texture brushing method makes use of a Poisson solver implemented in OpenCL. For the polygon snapping, we were given access to the original implementation of Arikan et al. [2].

The tool currently supports photogrammetric data sets generated with either the PMVS/CMVS toolkit [6] or with the commercially available software Agisoft Photoscan [12]. During the import process, the Shot neighborhood relations as described in Section 5.1 are computed, and the image edges for snapping are extracted in preprocessing steps.

We believe that our proposed workflows and interaction methods can technically be integrated into existing 3D modeling packages, but it has to be carefully evaluated whether the interactions described in this paper conflict with the standards established there.

8 Results

We have evaluated our novel modeling framework by trying to reconstruct several buildings from photogrammetric data sets. All operations can be performed completely interactive once the data set is imported, and the real-time frame rates allow fluent work on a consumer-level computer.

As can be seen in Figure 7, the targeted goal of creating low-polygonal, textured, CAD-ready 3D buildings in just a few minutes could be reached: The modeling times for the buildings lie between five and fifteen minutes – including the generation of textures for the polygons. It is important to notice that especially the side parts of the buildings, where no complete point cloud was available due to the limited access for the photographer to the area, could be accurately reconstructed using our image-based approach. The front facades, where the point cloud is usually quite dense, could be successfully modeled with the single Shot method described in Section 4.2. Once a single window of a certain type was modeled, all the others on the same facade could be created using the same plane and the edge-based snapping feature within seconds.

8.1 Limitations

Even though we have shown that using both photos and point clouds from photogrammetric data sets in an interactive workflow makes it possible to reconstruct more areas accurately, our approach still suffers from the fact that objects that are hidden, occluded or only visible in a single

Figure 7: Three textured 3D building models generated with our approach. In the left column, the photogrammetric point cloud is visualized, followed by the geometric reconstruction including hierarchical definitions in the middle column. In the right column, the final model with textures generated from multiple photos and using interactive occluder removal are shown. While parts that are not depicted in the point cloud could be reconstructed using the photos, that the backsides of the houses were modeled freely, as they were not accessible for the photographer. Modeling time including texture generation: Top row 5 minutes, seconds row 15 minutes, third row 10 minutes, fourth row 20 min.

photo require manual, inaccurate modeling steps. Furthermore, we are (similar to the methods proposed by Arikan et al. [2] and Sinha et al. [18]) limited to the reconstruction of planar surfaces. Even though curved surfaces can be approximated using multiple polygons, the handling of such primitives is more challenging than it is for planar shapes.

9 Conclusion & Future Work

We have demonstrated how to combine interactive techniques from both image-based and point cloud-based methods to reconstruct CAD-ready 3D models of buildings within a few minutes. 3D planes, on which sketched 2D polygons are reprojected, can not only be computed from multiple views, but also from planar segments detected in the corresponding point cloud. Image-based snapping features and suggestions further improve the sketching workflow.

Our novel method is a natural extension of these related techniques, and does not interfere with their concepts, but improves them. By introducing an intuitive *Visual Guidance Indicator*, users can take shortcuts during the image-based modeling steps, while being aware of the quality impact this has.

As this project is ongoing work, we have to especially evaluate and fine-tune user-oriented interaction methods in the future. Not only will a decent user study be performed and more data sets be used, but we will also investigate if we can use learning algorithms to replace currently user-defined parameters and thresholds, as they may vary depending on input data. As we already managed to simplify and minimize the interactions to a level that allows reconstruction of a building with just a few mouse clicks, we will evaluate if these concepts can be used on a touch-based interface as well, opening the door for the use on mobile devices.

Acknowledgements

We wish to express our thanks to Thomas Ortner and Stefan Maierhofer from the VRVis Research Center for their valuable feedback. This work was supported by the Austrian Research Promotion Agency (FFG) through the FIT-IT project Replicate, project no. 835948. The competence center VRVis is funded by BMVIT, BMWFJ, and City of Vienna (ZIT) within the scope of COMET Competence Centers for Excellent Technologies. The program COMET is managed by FFG.

References

[1] Pierre Alliez, David Cohen-Steiner, Yiying Tong, and Mathieu Desbrun. Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 39–48, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[2] Murat Arikan, Michael Schwärzler, Simon Flöry, Michael Wimmer, and Stefan Maierhofer. O-snap: Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics*, 32:6:1–6:15, January 2013.

[3] Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graph. Models*, 67:405–451, September 2005.

[4] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2301–2309, 2011.

[5] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *SIGGRAPH*, pages 11–20, 1996.

[6] Yasutaka Furukawa. Clustering views for multi-view stereo (CMVS). http://www.di.ens.fr/cmvs/. Accessed: 2016-02-29.

[7] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, Richard Szeliski, and Google Inc. R.: Towards internet-scale multiview stereo. In *In: Proceedings of IEEE CVPR*, 2010.

[8] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(8):1362–1376, August 2010.

[9] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*, chapter 12.2, page 312f. Cambridge University Press, second edition, 2004.

[10] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the 4th Eurographics symposium on geometry processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, 2006. Eurographics Association.

[11] Leif P. Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 57–66, New York, NY, USA, 2001. ACM.

[12] Agisoft LLC. Photoscan. http://www.agisoft.com/. Accessed: 2016-02-29.

[13] Przemyslaw Musialski, Christian Luksch, Michael Schwärzler, Matthias Buchetics, Stefan Maierhofer, and Werner Purgathofer. Interactive multi-view façade image editing. In *VMV 2010*, pages 131–138, November 2010.

[14] Przemyslaw Musialski, Peter Wonka, Daniel G. Aliaga, Michael Wimmer, Luc van Gool, and Werner Purgathofer. A survey of urban reconstruction. *Computer Graphics Forum*, 32(6):146–177, September 2013.

[15] Irene Reisner-Kollmann, Christian Luksch, and Michael Schwärzler. Reconstructing buildings as textured low poly meshes from point clouds and images. In Nick Avis and Sylvain Lefebvre, editors, *Eurographics 2011 - Short Papers*, pages 17–20, April 2011.

[16] Nader Salman, Mariette Yvinec, and Quentin Merigot. Feature preserving mesh generation from 3D point clouds. *Computer Graphics Forum*, 29(5):1623–1632, 2010.

[17] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient ransac for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226, June 2007.

[18] Sudipta N. Sinha, Drew Steedly, Richard Szeliski, Maneesh Agrawala, and Marc Pollefeys. Interactive 3d architectural modeling from unordered photo collections. *ACM Trans. Graph.*, 27(5):159, 2008.

[19] Rafael Grompone von Gioi, Jrmie Jakubowicz, Jean-Michel Morel, and Gregory Randall. Lsd: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 32(4):722–732, 2010.

[20] Jinglu Wang, Tian Fang, Qingkun Su, Siyu Zhu, Jingbo Liu, Shengnan Cai, Chiew-Lan Tai, and Long Quan. Image-based building regularization using structural linear features. *Transactions on Visualization and Computer Graphics*, 1(99):1, 2015.