Sketch-based Guided Modeling of 3D Buildings from Oriented Photos

Michael Schwärzler[*] VRVis Research Center
Lisa-Maria Kellner[*] VRVis Research Center
Stefan Maierhofer[*] VRVis Research Center
Michael Wimmer[†] TU Wien

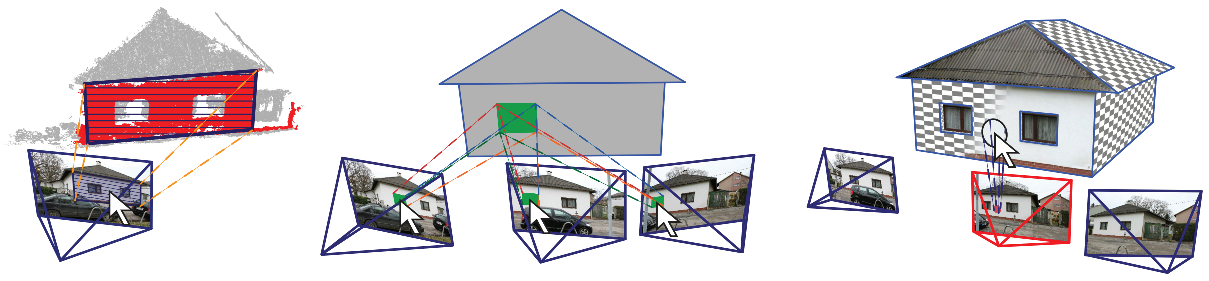

Figure 1: *Modeling operations taking both oriented images and point cloud data into account. Left: Point cloud-supported single shot sketching, exploiting planar structures in the data. Middle: Multi-view shot view sketching. Right: Texturing the generated polygons using an interactive brushing method.*

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

Concepts: •**Computing methodologies** → **Shape modeling; Mesh models;**

1 Introduction

The use of three-dimensional point cloud data of buildings gathered with different sensors has become part of the standard workflow in surveying and mapping – may it be from tachymetric devices, laser scans, or photogrammetry. However, also the games and movie industries resort to captured 3D data more and more frequently, as games are set in real-world scenarios, and special effects in movies require a full 3D reconstruction of a scene.

While solutions for the acquisition, storage and viewing of the raw point clouds have become commercially available, most applications, especially the games industry, requires polygonal models with lower detail. In particular, the planarity present in architecture needs to be preserved, which still poses a significant challenge.

Only recently, researchers have started tackling this problem by not only using meshing algorithms to triangulate point clouds, but to detect and use the underlying structure first, in order to obtain the main geometric primitives that represent a building (see Section 2). By doing so, resulting 3D models correspond 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 respect the sharp edges in the original scene.

One problem that complicates reconstruction is that data sets are rarely complete: 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).

In this paper, we propose an interactive modeling approach that remedies the problem of missing data by leveraging the available oriented 2D photos, which hold much more information than the point cloud or any reconstruction using only the point cloud: the user sketches the desired geometry directly on the image using simple 2D operations, both following the visual cues provided by a consistent image, which are much stronger than the features of a reconstructed point cloud, as well as supported by snapping to automatically extracted image edges. This allows recovering the missing structural information while integrating naturally into a simple sketch-based 2D workflow. From the user sketches, the 2D polygons are either directly projected to 3D space, or the user provides additional sketches in further views until a unique mapping to 3D space can be defined, depending on the available data quality of the underlying point cloud. 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

[*]e-mail:{msc|kellner|sm}@vrvis.at
[†]e-mail:wimmer@cg.tuwien.ac.at

I3D '17, February 25 - 27, 2017, San Francisco, CA, USA
ISBN: 978-1-4503-4886-7/17/03...$15.00
DOI: http://dx.doi.org/10.1145/3023368.3023374

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. Our main contributions are:

- An intuitive 3D modeling approach for photogrammetric datasets, that relies on simple, sketch-based interactions on 2D photos based on snapping to edges in the image, and therefore even usable on mobile devices.
- A polygon-sketching method that obtains the required 3D plane automatically from the point cloud.
- A multi-view sketching method allowing polygons to be defined where no point cloud data is available, and that reduces user interaction by proposing suitable polygons in further views once an initial polygon has been sketched.
- A visual indicator guiding the user through the polygon sketching process by giving feedback on achieved accuracy.

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. [2013]. Instead, we put our focus on methods that incorporate *photos* in the geometric reconstruction pipeline, or that try to simplify 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 the systems presented by Debevec et al. [1996] or Sinha et al. [2008]: These interactive tools also rely on image-based modeling operations, and use photogrammetric data sets to calculate geometric correspondences in order to 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 [Kobbelt et al. 2001; Boissonnat and Oudot 2005; Kazhdan et al. 2006; Alliez et al. 2007; Salman et al. 2010]. Wang et al. [2015] use additional image 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 create 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 geometric editing or semantic classification cumbersome.

To tackle these problems, Arikan et al. [2013] have proposed an interactive method that identifies 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. In unclear cases, users 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. Especially in the case of photogrammetric data, often-occurring large holes cannot be accurately reconstructed. Reisner-Kollmann et al. [2011] propose using image information for automatically filling holes in the surface. In our approach, we combine both approaches: planar surfaces in the point cloud are identified and used as a sketching plane for 2D modeling interaction – but by using photos as additional input in the 2D domain, 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 her artistic freedom.

3 Photogrammetric Data

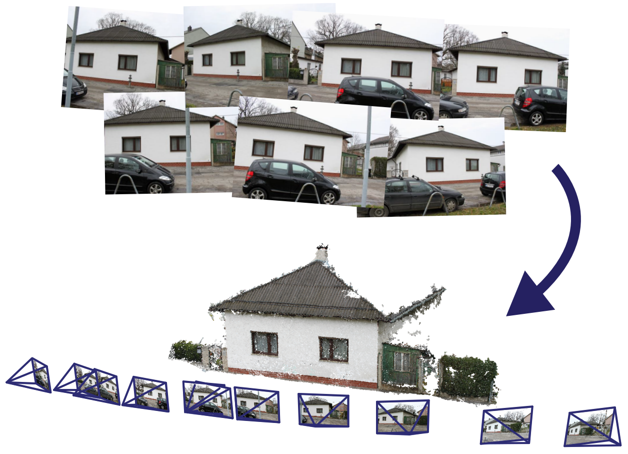

Figure 2: *Photogrammetric Network (bottom), consisting of a 3D point cloud and photos (top), for which their relative positions and orientations have been computed. We refer to them as* shots.

As this work focuses on interactive modeling using photogrammetric data, we describe the properties and distinctive characteristics of this input type: A *Photogrammetric Network* is obtained from a set of overlapping photos of an object by using *Structure from Motion (SfM)* techniques, which gives the locations and orientations from which the photos were taken, as well as a 3D point cloud consisting of matching image features that have been reprojected to 3D space (see Figure 2). The point cloud can be further densified using algorithms proposed by Furukawa et al. [2010; 2010]. Since such points have not been measured but were calculated using image features, photogrammetrically generated points may not be as dense and – more importantly – not as uniformly distributed as point clouds from laser scans, leading to more holes in the data. For example, it is difficult to extract features from completely flat, featureless walls. We therefore strive for compensating such 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 not only makes the modeling tools less complex to handle, but also corresponds to human intuition, since humans are 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 relatively 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. We describe these three methods in the following subsections.

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 triangulation method based on homogeneous direct linear transformation (DLT) as described by Hartley and Zisserman [2004], resulting in a least-squares optimal solution. This approach is just one possible solution to this intersection problem. We opted for it is robust and easily generalizes to triangulation in more than 2 views. We exploit this 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 works well, it is still an overhead for the user to have to sketch in two shots. To allow sketching in just a single shot, we exploit the available point cloud data: Similar to Arikan et al. [2013], we segment the point cloud into planar segments using the RANSAC algorithm by Schnabel et al. [2007]. After an initial polygon has been sketched, we try to find on which planar segment it was most likely intended to be drawn. For this, we transform the points of each planar segment from 3D world space into the 2D image space of the shot used for sketching, 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).

To determine 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, and express this in a heuristic $h \in [0, 1]$. The uniformity is estimated by rasterizing all points in the segment as splats over the polygon and then determining a fill ratio r, where 1 means fully filled and 0 not filled at all. The heuristic is then computed as

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

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

The splat size q used for splatting 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

It is often obvious 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

In this section, we describe how we integrated the described polygon-sketching methods in an interactive modeling workflow. All interactive concepts described in the following only guide and support the user. Despite all the suggestions of our system, we assume that the user “knows best” what her 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 sketching operations are performed within 2D photos for reasons of simplicity. However, it is of utter importance that the user also implicitly knows about the current view location in the 3D world, so that the spatial context can be used to sketch polygons in multiple shots without mixing them up.

During sketching, users are presented a 2D view of the current photo. Nevertheless, since the corresponding shot also includes 3D information, we allow the opacity value to be changed arbitrarily, such that 3D content (e.g. already modeled polygons) can be made visible. Furthermore, switching between shots is designed to help the user retain his spatial orientation: Instead of switching the displayed photo immediately, the camera performs a flying animation to show where the user is "virtually going".

5.2 Sketching and Snapping in Shot View

We facilitate sketching in a selected shot by a snapping feature which lets sketched lines snap to edges detected in the image. Thus, the user only needs to create a rough sketch. For this, we use an implementation of the Line Segment Detector described in the work of von Gioi et al. [2010] to find edges in the underlying image. The outline of the initial polygon is compared to the set of lines in the image. Two lines are considered matching if they are nearly parallel and spatially close. If no match is found, the sketched edge will be used as-is. Figure 3 shows this polygon-snapping workflow.

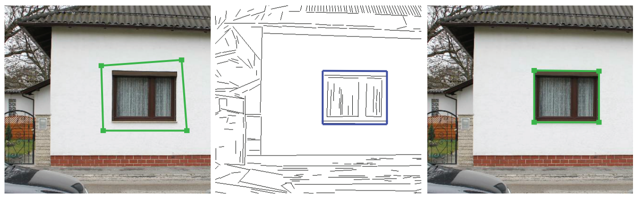
Figure 3: *Polygon Snapping: sketched 2D polygon on an image (left), extracted image lines with matchings in blue (middle), snapped polygon (right).*

If the user opts for using the multi-view shot modeling mode, the workflow requires the same polygon to be available in different images. Instead of having to sketch the corresponding polygon again, our proposed system tries to minimize this effort: As soon as the user switches to the next shot (defined by a nearest spatial neighborhood heuristic), the initial polygon is projected into the new image and repositioned to "fit the same sketched object". For example, if the polygon snapped to window edges in the initial image, the projected polygon in the neighbor image also tries to snap to the same window edges.

This is achieved by computing normalized color histograms at the edges of the initial polygon in the underlying photo. Then, the histograms are compared with histograms of edges found in the target image to find matching lines. The best match is used as starting point for the polygon in the other shot. Finally, adjacent edges are added step-by-step.

Our system makes no assumptions about specific structures or spatial arrangement of the underlying geometry to be reconstructed. We only assume a planar polygon viewed from different vantage points under arbitrary affine transformations. We found that histogram-based matching of polygon edges works reliably in this general case. Of course, this does not exclude the possibility to combine our approach with existing specialized methods for constrained scenarios, like highly repetitive structures [Musialski et al. 2012] or hierarchical block-like arrangements [Xiao et al. 2008].

Especially in the case of buildings, it becomes obvious why an interactive approach with minimized input, which mostly consists of deciding on proposed suggestions, is important: Architectural objects often consist of extremely similar and repetitive patterns, and an initially sketched window can be found multiple times on the next photo by our algorithm. We therefore display all found polygon candidates as suggestions in the target image. Since the user is able to retain a global spatial overview more easily, the correct window can then be picked with a single click. This process is repeated

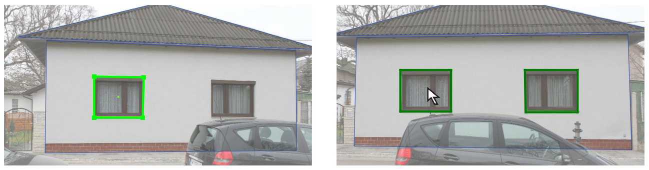
Figure 4: *Multi-view sketching with suggestions. Left: The window is sketched in the first view and snapped to the image edges. In this case, no suitable planar point cloud segment is found, which is why the sketched 2D polygon cannot be projected to 3D space. Right: In a neighboring view, a search for similar polygons is performed in image space by using the histograms of the polygon edges of the source image. Two candidate polygons that resemble the same type of window are found. They are suggested to the user, and the left one is selected by a click.*

in each image the user navigates to. In images where the correct polygon cannot be found (e.g., due to occlusions), the wrong suggestions can be completely skipped.

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 get feedback on whether the polygon should be sketched in further views, or if enough information on the needed 3D plane is already available to compute the world space position of the object.

During each polygon creation process, the user gets continuous 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 state can vary 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 5 for a visualization of the guidance element in the user interface.

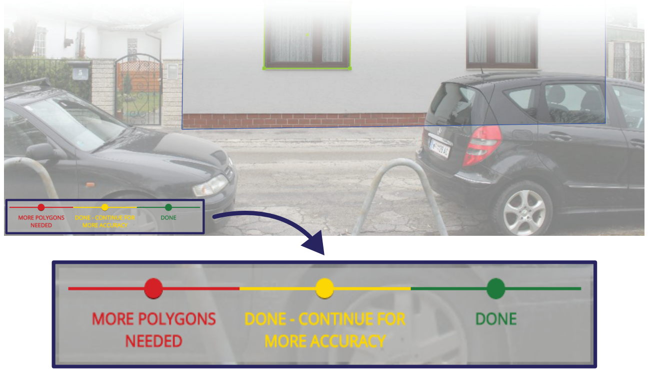

Figure 5: *The visual guidance interface shows whether enough polygons have already been sketched to compute a 3D polygon, or if the user should continue sketching.*

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, after sketching the initial polygon, 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 green state is shown as soon as the polygon is defined in at least three different views, or when a single significant plane to project the 2D outline to is available (single-shot 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 typical 3D polygon-modeling tasks in our system – like adding and removing vertices, translation or scaling – can 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.

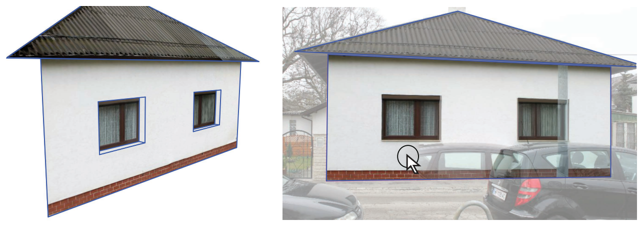

Figure 6: *Left: By defining hierarchical relations, holes and side faces are automatically extracted. Right: Interactive removal of occluding objects from the texture by overlaying the photo semi-transparently using the shot view.*

In addition to shot-based sketching operations described in previous sections, our framework supports standard polygon editing features known from other 3D modeling packages. For this, the camera can be moved around freely in 3D space. Switching to this mode is especially useful when a surface needs to be closed due to missing photos. We also included optimization-based snapping to close small gaps between polygons as proposed by Arikan et al. [2013], which takes into account further constraints like parallelism or orthogonality of edges – 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 6, left.

6.2 Texture Brushing

Shots can be used to generate textures by reprojecting associated photos onto the polygons. In order to obtain a suitable texture, we use a technique proposed by Musialski et al. [2010], where a single polygon is textured from multiple photos, each pixel colored according to the best-fitting shot. While the initial source of each pixel is selected automatically based on angle and distance, arbitrary parts of the texture can be "repainted" with the photo of a user-selected shot in order to remove occluders or artifacts. Figure 7 shows textures with different coloring according to shots.

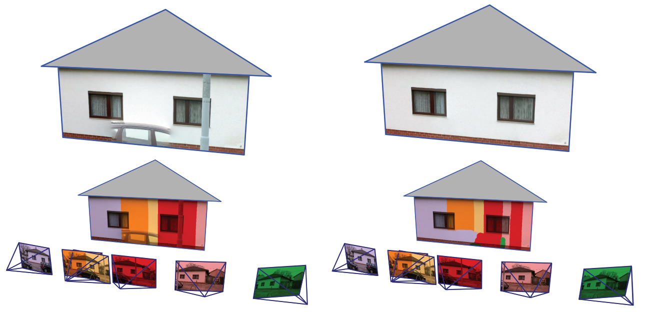

Figure 7: *Top left: The initial texture containing occluders. Bottom Left: The associated shots, visualized using a false color mask. Top right: The cleaned texture after interactive brushing. Bottom right: The correspondingly modified mask.*

We extended the original method, in which users could only brush a polygon in 3D view, to be also used via the shot view, in which transparency can be adjusted interactively. This way, users simultaneously have access to the current state of the textured polygon in the 3D world, and the 2D photo content. In 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 one can easily find proper image parts for specific texture positions (see Figure 6, right). Switching between shots and leaving the shot view is possible at any time as described in Section 5.1.

6.3 Touch-based Interaction

As navigating between the individual shots and sketching on photos – the main interaction tasks needed in our approach – do not require any pixel-accurate clicks or complicated keyboard commands, we also investigated the possibility to use our system on a touch-based interface. As demonstrated in the accompanying video and in Figure 8, the concept of roughly sketching on photos is well-suited for defining the planar shapes of a building. Right now, touch-based modeling is limited to sketching in the shots, as adding arbitrary polygons freely (see Section 6.1) has not been implemented in a touch-based manner in our prototype yet.

Although the touch input is less precise than mouse and keyboard, this is often alleviated by the snapping mechanism, and we see the possibility of using a touch-based interface as an interesting first step towards the development of specialized mobile apps: A complete on-site 3D building reconstruction (taking photos, computing orientations in the cloud, and modeling the final textured model) could be made possible using just a small hand-held device.

7 Implementation

Our novel modeling and reconstruction framework has mainly been implemented using an existing 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 [2011]. The interactive texture-brushing method makes use of a Poisson solver implemented in OpenCL. For polygon snapping, we were given access to the original implementation of Arikan et al. [2013].

The tool currently supports photogrammetric data sets generated with either the PMVS/CMVS toolkit [CMVS 2010] or with the

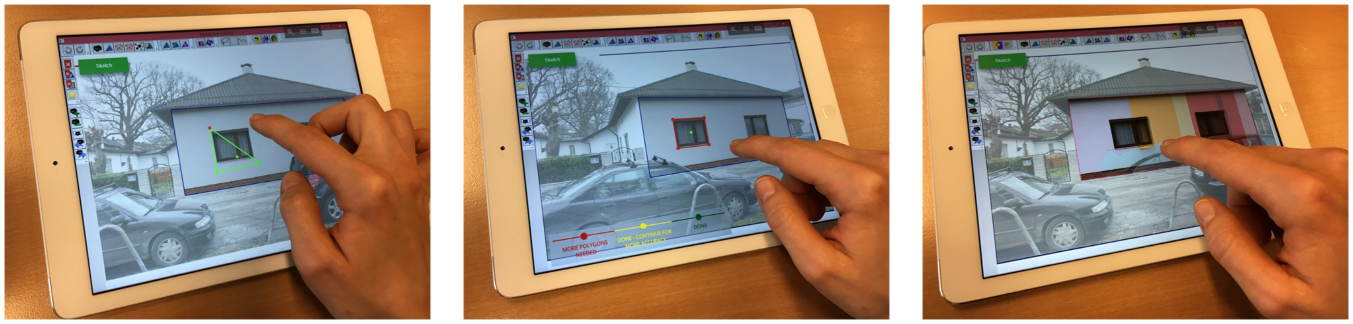

Figure 8: *Evaluation of our proposed system on a touch-based device. Left: Coarse sketching of the polygon in the initial view. Middle: The polygon has been snapped to image edges and been reprojected to a neighboring image in the multi view sketching mode. The visual guidance feedback indicates that enough views have been used. Right: Interactive brushing.*

commercially available software Agisoft Photoscan [2016]. 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 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 Evaluation and Results

All described interaction methods are fully interactive. Real-time frame rates allow fluent work on consumer-level hardware. We have evaluated our novel framework by gathering feedback from five users we let try to reconstruct several buildings from photogrammetric datasets. All had some background knowledge in the field of 3D modeling or interactive editing in 3D scenarios. We intentionally chose datasets where the photogrammetric reconstruction produced point clouds that did not fully represent the building structure (i.e., some planar surfaces like white walls or reflective windows that are clearly visible in the photos are not depicted in the point cloud). Thus, both single shot and multi shot sketching could be applied.

While all users grasped the modeling interactions after a short introduction and managed to fulfill the requested task – to reconstruct and texture the given building model as fast and accurately as possible – within 10 to 20 minutes, their feedback differed greatly based on their background and experiences:

- Two experts from the field of surveying were immediately happy with the idea of defining polygonal shapes using single points from a shot, as it resembles their typical workflow when taking geodetic measurements. They were eager to see such techniques applied to their currently used GIS software. One expert was rather skeptical regarding the achievable accuracy of the photogrammetric approach, and requested a prior registration of the point cloud to geodetic data.

- One user working as content creator for the gaming industry first intended to coarsely sketch the ground plan of the building, extrude the walls from it, and wanted to align the walls to fit the photos. While this seems to be a feasible approach at first, it would require the rotation of the defined planes – an operation we have not allowed in our prototype (but would definitely be worth to evaluate). After adapting to our proposed workflow, it was pointed out that the shot-based creation of occluder-free building textures from multiple photos was a lot faster than manual stitching that is usually done.

- We asked a user who was familiar with the O-Snap system [Arikan et al. 2013] for creating buildings solely from point clouds to use our system. Since the optimization-based snapping is also integrated in our prototype, the user tried to make use of it as often as possible. This sometimes failed when there were only very few planes sketched in the beginning of the modeling process, as the O-Snap algorithm tried to close holes when the basic structure was not defined yet, overriding the results from the image-based edge snapping. As soon as the main planes of a building had been modeled, the two algorithms complemented each other very well, though. We therefore removed the accessibility of the geometric optimization tool during shot view sketching, so that it is now only available as a post-processing option.

- In order to verify the feasibility of the touch-based version, we asked a developer of mobile VR/AR apps to evaluate our prototype implementation. Even though he had several remarks on how to improve the usability (especially concerning the used gestures), he was convinced that the proposed modeling concept could be applied in a dedicated mobile app.

As can be seen in Figure 9, the targeted goal of creating low-polygonal, textured, CAD-ready 3D buildings in just a few minutes could be reached: The average modeling times for the buildings lie between five and fifteen minutes – including the generation of textures for the polygons. All users used the feedback from the guidance indicator during the polygon sketching progress, and even pointed us towards interaction workflows in tools they frequently use, where similar guidance approaches could be useful.

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 photo require manual, inaccurate modeling steps. Furthermore, we are (similar to the methods proposed by Arikan et al. [2013] and Sinha et al. [2008]) 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.

In the future, we plan to further fine-tune the user-oriented interaction methods – we are especially interested in bringing the touch-based interaction to a level that a complete 3D building modeling process is possible on mobile devices. Together with the camera capabilities of modern smartphones and server-based photogrammetric computation, a complete reconstruction workflow on a single device seems to be within reach.

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. Furthermore, we were inspired from the user feedback we got from the expert users, and plan to enhance the modeling workflow by also integrating additional data sources like geodetic measurements or ground plans.

Acknowledgements

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

AGISOFT, 2016. Photoscan. http://www.agisoft.com/. Accessed: 2016-10-23.

ALLIEZ, P., COHEN-STEINER, D., TONG, Y., AND DESBRUN, M. 2007. Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of the fifth Eurographics symposium on Geometry processing*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 39–48.

ARIKAN, M., SCHWÄRZLER, M., FLÖRY, S., WIMMER, M., AND MAIERHOFER, S. 2013. O-snap: Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics 32* (Jan.), 6:1–6:15.

BOISSONNAT, J.-D., AND OUDOT, S. 2005. Provably good sampling and meshing of surfaces. *Graph. Models 67* (September), 405–451.

BOSTOCK, M., OGIEVETSKY, V., AND HEER, J. 2011. D^3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics 17*, 12, 2301–2309.

CMVS, Y. F., 2010. Clustering views for multi-view stereo (CMVS). http://www.di.ens.fr/cmvs/. Accessed: 2016-10-23.

DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *SIGGRAPH*, 11–20.

FURUKAWA, Y., AND PONCE, J. 2010. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell. 32*, 8 (Aug.), 1362–1376.

FURUKAWA, Y., CURLESS, B., SEITZ, S. M., SZELISKI, R., AND INC, G. 2010. R.: Towards internet-scale multiview stereo. In *In: Proceedings of IEEE CVPR*.

HARTLEY, R. I., AND ZISSERMAN, A. 2004. *Multiple View Geometry in Computer Vision*, second ed. Cambridge University Press, ch. 12.2, 312f.

KAZHDAN, M., BOLITHO, M., AND HOPPE, H. 2006. Poisson surface reconstruction. In *Proceedings of the 4th Eurographics symposium on geometry processing*, Eurographics Association, Aire-la-Ville, Switzerland, SGP '06, 61–70.

KOBBELT, L. P., BOTSCH, M., SCHWANECKE, U., AND SEIDEL, H.-P. 2001. Feature sensitive surface extraction from volume data. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, SIGGRAPH '01, 57–66.

MUSIALSKI, P., LUKSCH, C., SCHWÄRZLER, M., BUCHETICS, M., MAIERHOFER, S., AND PURGATHOFER, W. 2010. Interactive multi-view façade image editing. In *VMV 2010*, 131–138.

MUSIALSKI, P., WIMMER, M., AND WONKA, P. 2012. Interactive coherence-based façade modeling. *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2012) 31*, 2 (May), 661–670.

MUSIALSKI, P., WONKA, P., ALIAGA, D. G., WIMMER, M., VAN GOOL, L., AND PURGATHOFER, W. 2013. A survey of urban reconstruction. *Computer Graphics Forum 32*, 6 (Sept.), 146–177.

REISNER-KOLLMANN, I., LUKSCH, C., AND SCHWÄRZLER, M. 2011. Reconstructing buildings as textured low poly meshes from point clouds and images. In *Eurographics 2011 - Short Papers*, N. Avis and S. Lefebvre, Eds., 17–20.

SALMAN, N., YVINEC, M., AND MERIGOT, Q. 2010. Feature preserving mesh generation from 3D point clouds. *Computer Graphics Forum 29*, 5, 1623–1632.

SCHNABEL, R., WAHL, R., AND KLEIN, R. 2007. Efficient ransac for point-cloud shape detection. *Computer Graphics Forum 26*, 2 (June), 214–226.

SINHA, S. N., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. 2008. Interactive 3d architectural modeling from unordered photo collections. *ACM Trans. Graph. 27*, 5, 159.

VON GIOI, R. G., JAKUBOWICZ, J., MOREL, J.-M., AND RANDALL, G. 2010. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis & Machine Intelligence 32*, 4, 722–732.

WANG, J., FANG, T., SU, Q., ZHU, S., LIU, J., CAI, S., TAI, C.-L., AND QUAN, L. 2015. Image-based building regularization using structural linear features. *Transactions on Visualization and Computer Graphics 1*, 99, 1.

XIAO, J., FANG, T., TAN, P., ZHAO, P., OFEK, E., AND QUAN, L. 2008. Image-based façade modeling. *ACM Trans. Graph. 27*, 5 (Dec.), 161:1–161:10.

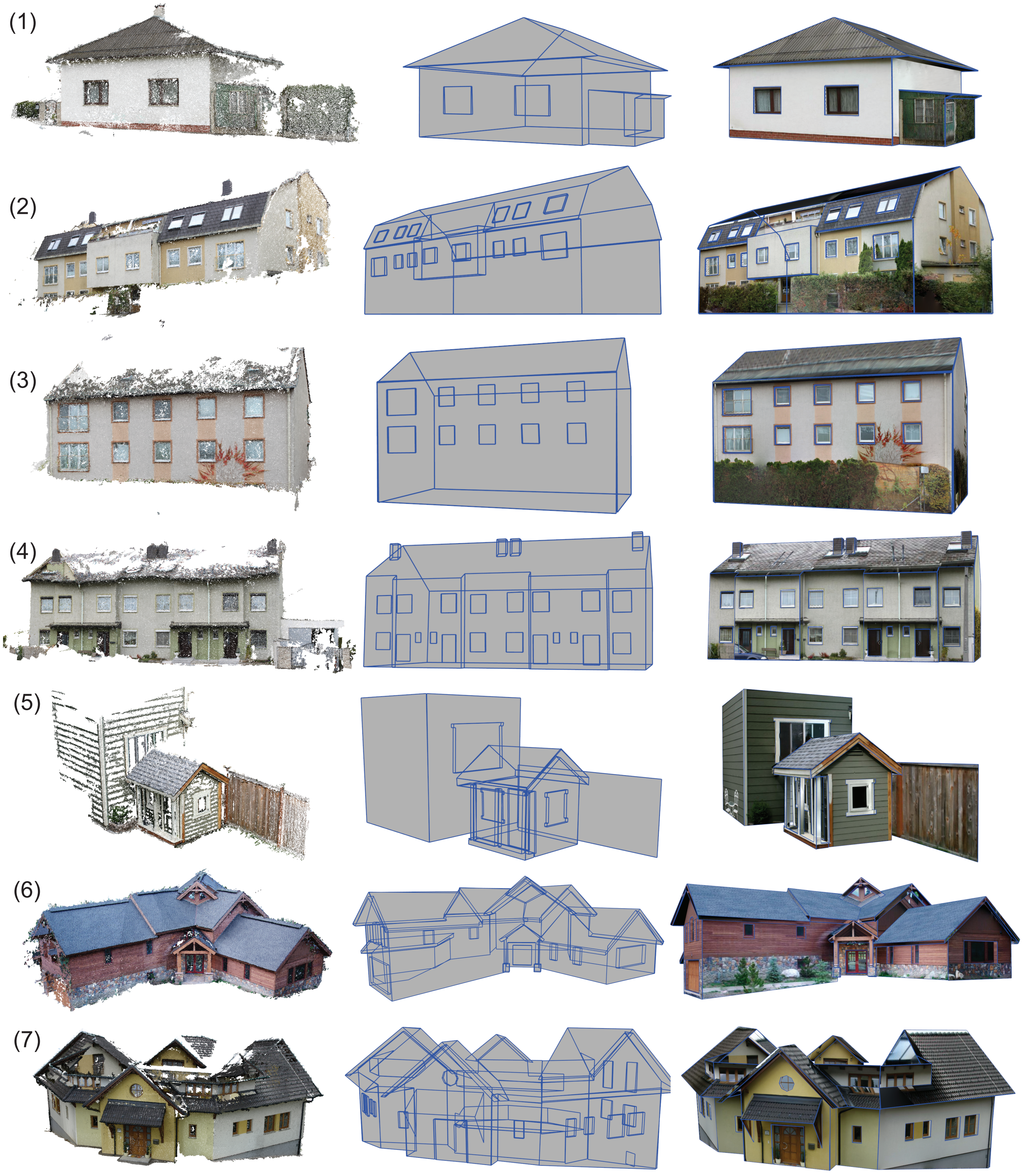

Figure 9: *Textured 3D building models generated with our approach. Left: Photogrammetric point cloud. Middle: Geometric reconstruction including hierarchical definitions. Right: Final model with textures generated from multiple photos using the interactive occluder removal. Parts that are not depicted in the point cloud could be accurately reconstructed using the photos. The backsides of the houses 1-4 were modeled freely, as they were not accessible for the photographer. Buildings 6 and 7 were modeled using data sets with complete photo coverage from all sides, but required significantly more modeling time due to their complexity. For building 7, not enough photos were taken to model the roof completely. Modeling time including texture generation in minutes, from top to bottom: 5, 15, 10, 20, 15, 40, 45.*