Superpixel Meshes for Fast Edge-Preserving Surface Reconstruction

András Bódis-Szomorú[1] Hayko Riemenschneider[1] Luc Van Gool[1,2]
[1]Computer Vision Lab, ETH Zurich, Switzerland [2]PSI-VISICS, KU Leuven, Belgium
{bodis,hayko,vangool}@vision.ee.ethz.ch

Abstract

Multi-View-Stereo (MVS) methods aim for the highest detail possible, however, such detail is often not required. In this work, we propose a novel surface reconstruction method based on image edges, superpixels and second-order smoothness constraints, producing meshes comparable to classic MVS surfaces in quality but orders of magnitudes faster. Our method performs per-view dense depth optimization directly over sparse 3D Ground Control Points (GCPs), hence, removing the need for view pairing, image rectification, and stereo depth estimation, and allowing for full per-image parallelization. We use Structure-from-Motion (SfM) points as GCPs, but the method is not specific to these, e.g. LiDAR or RGB-D can also be used. The resulting meshes are compact and inherently edge-aligned with image gradients, enabling good-quality lightweight per-face flat renderings. Our experiments demonstrate on a variety of 3D datasets the superiority in speed and competitive surface quality.

1. Introduction

Automatic 3D modelling of real urban environments has been a long-term challenge [40], and has a wide range of applications such as cultural heritage, entertainment, real estate business, navigation and urban planning. Companies like Google, Apple and Acute3D have automated 3D urban model creation using Multi-View Stereo (MVS) technologies. These solutions rely on aerial data, as to meet the large scale requirements. The resulting, textured meshes lack facade details, however. Yet, many applications would benefit from facade-level details or the related semantics.

The alternative is to use street-level acquisition. In this vein, image-based Multi-View Stereo (MVS) has also been used, at least at a city district scale [29, 38, 18, 57]. These methods tend to be too slow for larger scenes for several reasons. They often enforce pixelwise photo-consistency across multiple views, and deliver very dense point clouds or depth maps that need extra meshing steps to obtain more compact surfaces. Leaving them as such comes with a high toll on the required storage space and visualization workload. In case of volumetric MVS, like the GPU-based CMP-MVS [29], a street scene can cost multiple hours.

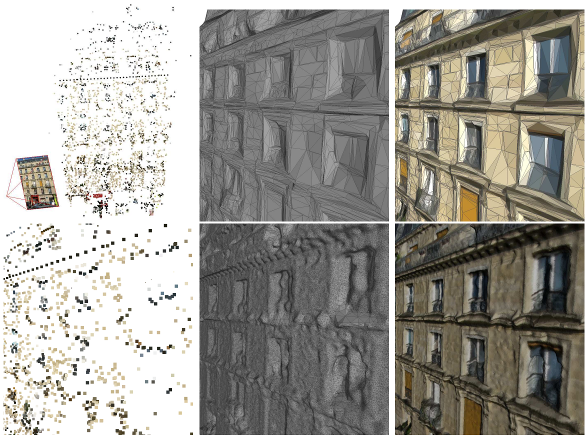

Figure 1. Given only sparse SfM data and a single view (left), our lightweight method produces a mesh aligned to image contents fast (seconds) (top). The approach is general but particularly promising for scalable LoD-3 street-level city modeling. Bottom: output of a state-of-the-art multi-view stereo software [29].

In turn, our aim is to develop a fast image-based surface meshing approach to cover large city areas at the level of a CityGML LoD-3 model, i.e. the geometry of buildings to be captured precisely, with indications of architectural components like windows being set back or balconies sticking out. However, we additionally experiment with reliefs, and believe that our meshing could be applied in other scenarios, e.g. indoor scenes, replacing SfM with LiDAR points, or when dense 3D data (e.g. RGB-D) is already available.

In this paper, we present a novel approach to *directly* obtain a 3D surface mesh from a set of images and from sparse Structure-from-Motion (SfM) data, which we assume to have been pre-calculated, e.g. from the same set of images. This is in contrast to the classical approach of reconstructing a highly detailed pixelwise dense model via MVS and then apply simplification and meshing at the expense of even more processing time. As we demonstrate, SfM can already give sufficient 3D data to treat the problem as many single-view surface fitting problems with small overlap between the views. By incorporating superpixels and

978-1-4673-6964-0/15/$31.00 ©2015 IEEE

image edges, our method produces compact meshes as it ensures that mesh edges (and vertices) are aligned with image gradients, unlike in existing solutions. Edge-alignment enables semantic segmentation [47] and visually pleasing renderings with simple per-face coloring, thus eliminating complicated texturing procedures (e.g. atlases) in many applications. Surface fitting is formulated in a novel way to be solved with an efficient linear solver. As a result, depths can be computed in a split-second on a CPU, and despite the sparse 3D input, the quality of the resulting meshes is comparable to that of a state-of-the-art pipeline, e.g. [29]. See Figure 1 for a first qualitative comparison of the surfaces.

Treating the surface reconstruction problem as many single-view ones has several advantages in itself. First, a single observing view is sufficient to reconstruct a certain surface area (no need for pairwise rectifications). Second, the overall runtime is linear in the number of views or sublinear in case a view selection is run beforehand. Third, it is parallelizable per view. To model large scenes, our single-view meshes can be simply concatenated and trimmed, like in [41], as they are already quite consistent.

The traditional bottleneck of dense depth estimation, including photoconsistency calculations, is avoided altogether. As we show, our method outperforms pixelwise MVS methods in speed, and by a natural per-view parallelization, it can be further speeded up by 1-3 orders of magnitude, thereby bringing large-scale ground-level city modelling within reach.

2. Related Work

Structure-from-Motion (SfM) has been scaled up to handle entire city districts [2, 18, 67, 11]. Herein, we give an overview of existing surface modeling techniques, assuming SfM data at hand.

Multi-View Stereo (MVS). Volumetric methods partition the scene into cubic voxels [52, 58, 23], tetrahedra [29, 60] or polyhedral cells [9], and classify these as inside/outside. Volumetric methods are known to produce watertight but non-smooth surfaces [33] and suffer from poor scalability [29]. Still, they found their application for large-scale urban scenes via aerial images [9, 23]. Recently, Hoppe *et al.* [28, 27] found a way to locally update their tetrahedralization, and the inside/outside reasoning as new evidence arrives from SfM, and obtain an updated mesh on-the-fly.

CMP-MVS [29] is a free, state-of-the-art volumetric tool producing high-quality meshes. It takes hours to reconstruct a street-level scene on a GPU from hundreds of images.

Due to its better scalability, depth map based MVS [45] has been demonstrated on ground-level images of urban scenes at city block scale [44, 38].

Simplifications. Pollefeys *et al.* [44] present a real-time system using plane-sweep stereo on the GPU [21], given orientation and plane priors derived from SfM point clouds. Micusik and Kosecka [38] extend plane sweeping by aggregating surface data per superpixel to stabilize textureless areas. Furukawa *et al.* [17] provide speedup by simplifying to piecewise-planar Manhattan-world geometry. Gallup *et al.* [20] segment planar/non-planar (mostly vegetation) regions to identify areas that can be simplified.

Aiming at city visualization, [46] exploits the 2.5D height map assumption for urban scenes, whereas [22] relaxes this to an n-layer height map. Cornelis *et al.* [10] reconstruct a simplified canyon-like street model with planar ground and ruled vertical surfaces for the facades, thereby allowing for real-time processing.

These methods work fast yet rely on the Manhattan-world assumption or simplify facade models to an extend that important (LoD-3) details are lost.

Another line of work starts from ortho-rectified facade images [56, 37] or uses SfM multi-plane fits for rectification [48, 69], and perform a semantic/procedural analysis [47, 36] of these to obtain LoD-3 facade details.

Mesh Alignment. Given a coarse mesh, missing details can be recovered by re-aligning the mesh to the images. Morris *et al.* [39] perform a search for the most photo-consistent triangulation over a sparse set of 3D control points. Surface evolution techniques [59, 60] have a similar objective but are driven by expensive variational formulas. These works enforce photo-consistency over the mesh faces, but do not align the edges of mesh triangles to the image content: triangles may cross geometric edges and semantic boundaries. [43, 12] propose methods to refine details of a surface using photometric stereo or bundle adjustment, and manage to recover incredibly fine details. Recently, Salman and Yvinec [49] proposed a MVS algorithm that samples mesh edges in the vicinity of image edges. Unlike ours, their method operates on dense MVS points, enforces points as hard constraints, and is orders of magnitude slower, even without computing the dense input.

Direct Fitting to SfM. Several methods have been introduced to model the scene directly from sparse SfM data. Most of these assume piecewise planarity and do not handle non-planar regions [64, 54, 15, 53, 4]. In addition, similarly to Manhattan-world methods and plane sweeping stereo, many rely on a prior analysis of the SfM point cloud, that involves computation and clustering of normals or robust multi-plane fitting. In our experience, these preprocessing steps have serious stability issues over a sparse point cloud.

Probably the most well-known way to convert a point cloud to a mesh is via the Poisson method [30]. It assumes a uniform point distribution, and tends to perform poorly on sparse SfM data, which is inherently non-uniform. In turn, Multi-Scale Compactly Supported Radial Basis Function (MSCS-RBF) [42, 8] are applied successfully by New-

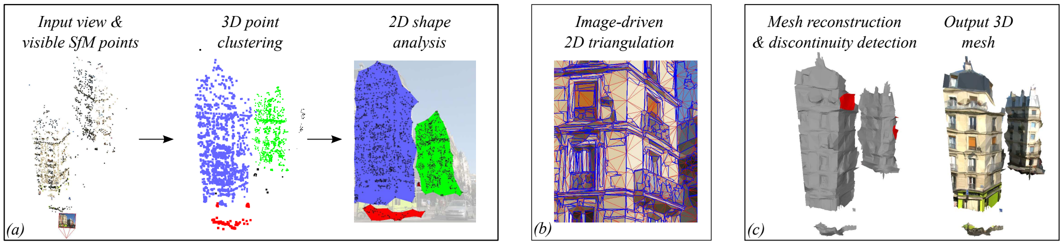

Figure 2. Our approach for single-view mesh reconstruction from sparse SfM data. (a) 3D/2D point cloud analysis returns 2D outlines of the areas populated by SfM points, (b) 2D base mesh extraction from the image, (c) reconstructs submeshes of (b) in the polygons of (a).

come and Davison [41] to fit a coarse base mesh directly to SfM points from each reference view. Similarly, several works from the stereo literature perform surface fits to sparse Ground Control Points (GCPs). Geiger *et al.* [24] compute a 2D Delaunay-triangulation over GCPs to reduce the stereo search space, while Wang and Yang [61] apply GCPs as hard constraints to interpolate a pixelwise disparity map in a linear formulation. A pioneering work in stereo [16] solves this interpolation with soft constraints. The recent video summarization work by Kopf *et al.* [32] fits a proxy mesh to SfM data for image-based rendering, but their mesh is not subdivided according to image content, and they use a smoothness prior favoring fronto-parallel planes (a 1st-order constraint).

Another line of work attempts to learn depth cues in a training stage to reconstruct the scene from a single image [26, 51, 34]. To render the task feasible, they rely on strong assumptions about the scene (e.g. Manhattan world). Particularly related to our work is that by Saxena *et al.* [50], where image cues and sparse GCPs within superpixels are combined to infer a piecewise-planar 3D model. These approaches come at the expense of geometric detail, however.

In contrast to these methods, we propose to reconstruct the scene directly by fitting meshes from single views to the sparse SfM points, without assuming a few dominant orientations, a Manhattan-world or piecewise planarity. Inspired by GCP-based interpolation-style linear methods [16, 24, 61, 32], we propose a sparse linear method using GCPs as *soft* constraints. However, our method operates over a triangular mesh per view, rather than on a pixelwise map, and it uses a 2nd-order smoothness prior, i.e. it favors planarity (penalizes curvature), instead of fronto-parallel planes. Our meshes are based on the images (e.g. superpixels), and are inherently edge-aligned with image gradients.

3. Single-view 3D mesh reconstruction

In this section we detail our single-view mesh reconstruction method. We propose an image-driven mesh, i.e. a mesh that is adapted to the image content. Mesh edges get aligned to image edges, and mesh faces tend to cover homogeneous image areas only as densely as needed. These properties lead to a quite compact surface model, and good visual quality, especially well-suited for textureless per-face flat rendering, i.e. when each triangle is colored according to its mean color in a base image.

Our method consists of three major stages, as summarized in Figure 2. First, the SfM points (our GCPs) are clustered to group and identify disconnected 3D structures (2a). Second, independently from SfM, we extract a 2D base mesh aligned to image gradients (2b). Third, vertex depths are calculated using the GCPs as soft-constraints and favoring local smoothness (2c). These steps are detailed in the next sections, starting with the base mesh.

3.1. 2D base mesh extraction

We propose an image-driven reconstruction which uses the image gradients to partition each image independently, similarly to superpixelization. The partitioning is embodied in a 2D base mesh, that is later reconstructed in 3D. Contrary to related work [63, 71], we do not limit the final mesh to a single depth per partition (fronto-parallel). Note that we start from an image-aligned 2D mesh, then reconstruct vertex depths, instead of first obtaining a 3D mesh (e.g. from a dense depth map) and then tuning it to the images [59, 60].

We have experimented with three methods to create 2D triangle meshes adapted to the image gradient information (as shown in Figure 3): (a) Delaunay triangulation of corner-like keypoints (e.g. Harris-corners), (b) Constrained Delaunay triangulation of polygonized binary image edges, and (c) Constrained Delaunay triangulation of polygonized superpixel boundaries. The constrained Delaunay triangulation in cases (b) and (c) enforces the extracted polygon edges into the triangulation. In case (a) the triangle count is moderated by only retaining the n-best corners. We apply Douglas-Peucker polygon simplification with a low tolerance (1-3 pixels), prior to constrained triangulation. This drastically reduces the number of polygon vertices while still preserving image edges. Increasing the triangle count typically improves the edge-adherence of the triangulation but increases the number of vertices to reconstruct.

3.2. Depth reconstruction

We consider the problem of dense single-view depth (mesh) reconstruction, given the image, the perspective

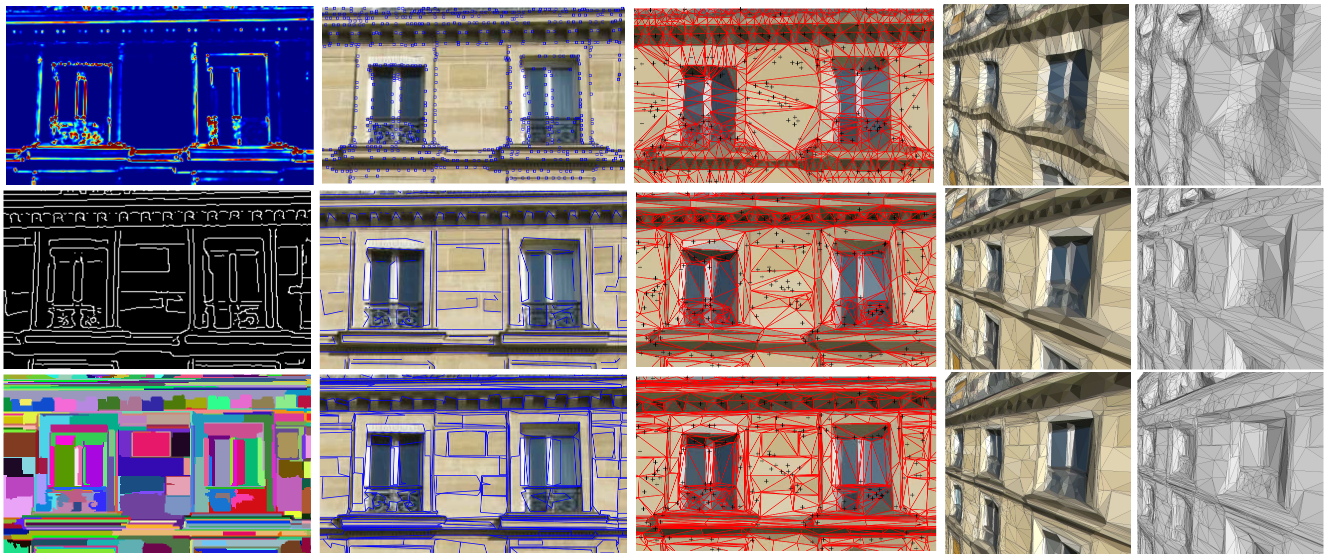

Figure 3. Different ways to obtain 2D image-aligned triangle mesh. Rows: (a) Delaunay Triangulation (DT) over image corners, (b) constrained DT over polygonized gradient edges, (c) constrained DT over polygonized superpixel boundaries. Columns: (1) Harris cornerness score / Canny edge map / superpixel label map, (2) original image with Harris corners and polygonized boundaries in blue, (3) constrained Delaunay triangulation over its mean-color rasterization (visible SfM points as black crosses), (4-5) 3D reconstruction obtained from sparse SfM data with our method introduced in Section 3.2.

camera model, and known depths at sparse 3D locations, i.e. Ground Control Points (GCPs). Our GCPs are points of an inhomogeneous sparse or semi-dense SfM point cloud, and we exploit the image-driven 2D triangulation described in Section 3.1. Unlike most superpixel-stereo methods, we relax the requirement that each superpixel (polygonal image region) observes a single plane or parametric 3D patch. Instead, our elementary units are triangles *derived* from such polygons, and we aim to reconstruct the depths at triangle vertices, while additionally favoring smoothness. This idea saves us from the need of explicitly enforcing continuity across individual 3D patches, since it is naturally and *exactly* satisfied in the proposed approach (please refer to the supplementary material for a visual comparison).

We assign GCPs to triangles efficiently via triangle rasterization and by sampling the raster map at GCP locations[1]. To reconstruct the depths $\{\hat{d}_i\}$ at vertices $\mathcal{V} = \{v_i\}_{i=1}^{V}$, we rely on the GCPs $\mathcal{P} = \{p_i\}_{i=1}^{N}$ with known respective depths $\{d_i\}$. We require the 3D triangles to fit the observed GCPs and that triangles are smoothly connected. We formulate this as the minization of a fitting and a smoothness term:

$$E(\hat{\mathbf{d}}) = E_{fit}(\hat{\mathbf{d}};\mathbf{d}) + \lambda E_{smooth}(\hat{\mathbf{d}}), \qquad (1)$$

where $\hat{\mathbf{d}}$ and $\mathbf{d}$ are the vectors of vertex depths and GCP depths, respectively, and λ is a scalar balance between fitting quality and smoothness.

We propose to use a least-squares formulation and linear interpolation via barycentric coordinates to obtain a simple quadratic form for (1), which can be efficiently minimized. Any GCP p_i belongs to a triangle t_i defined by some vertices $\{v_p, v_q, v_r\}$. Hence, p_i can be written as $p_i = \alpha_{pi}v_p + \alpha_{qi}v_q + \alpha_{ri}v_r$, where $(\alpha_{pi}, \alpha_{qi}, \alpha_{ri})$ are the barycentric coordinates of p_i w.r.t. t_i. The linearly interpolated (known) depth of the GCP from the (unknown) depths of the vertices is $\tilde{d}_i = \alpha_{pi}\hat{d}_p + \alpha_{qi}\hat{d}_q + \alpha_{pi}\hat{d}_r$. Collecting the equations for all GCPs, we obtain the matrix form $\tilde{\mathbf{d}} \triangleq (\tilde{d}_1, \ldots, \tilde{d}_N)^T = \mathbf{A}\hat{\mathbf{d}}$, where $\mathbf{A}$ is a $N \times V$ sparse matrix with up to $3N$ non-zeros. The unary term writes as

$$E_{fit}(\hat{\mathbf{d}};\mathbf{d}) = (\mathbf{d} - \mathbf{A}\hat{\mathbf{d}})^T \mathbf{\Sigma}^{-1}(\mathbf{d} - \mathbf{A}\hat{\mathbf{d}}), \qquad (2)$$

where $\mathbf{\Sigma}$ is the covariance matrix of the GCP depths $\mathbf{d}$.

To obtain a comparably simple form for the smoothness term, most existing methods use a simple squared penalty for the depth differences [16, 61, 32]. This favors fronto-parallel planes. In turn, we present a different formulation that penalizes curvature, yet in a linear fashion. We consider, for each vertex v_i, the triangle t_{ij} formed by a neighboring vertex v_j and its previous and next adjacent vertex v_j^- and v_j^+ in the oriented 1-ring neighborhood of v_i (see the right side of Figure 4), and linearly interpolate the (unknown) depth $\hat{d}_i$ of v_i from the (unknown) depths of v_j, v_j^- and v_j^+ via barycentric coordinates, namely,

$$\hat{d}_{ij} = \beta_1^{ij}\hat{d}_j + \beta_2^{ij}\hat{d}_j^+ + \beta_3^{ij}\hat{d}_j^- \qquad (3)$$

In case the triangles on the two sides of the edge $\{v_i, v_j\}$ are in the same plane, the interpolated depth $\hat{d}_{ij}$ equals $\hat{d}_j$.

[1] Our method is robust to the assignment errors due to rasterization.

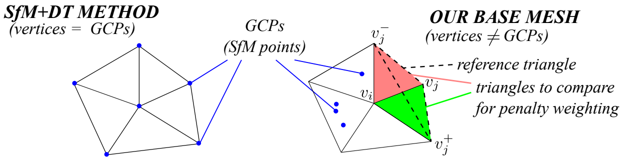

Figure 4. Distinction between Delaunay triangulation over GCPs (SfM+DT, left) and our base mesh (right). 1-ring vertex neighbors are used to calculate our smoothness term penalizing curvature. The depth difference of v_i w.r.t the dashed triangle is penalized.

To favor local smoothness, we penalize for the difference between these. A weighting term w_{ij} can be used additionally, based on the observed color difference of the two triangles meeting in edge $\{v_i, v_j\}$. This is useful to relax the planarity requirement at crease edges, which are likely to be observed in the image. A single penalty term formulates as

$$E_{ij} = w_{ij}^2(\hat{d}_i - \hat{d}_{ij})^2 = w_{ij}^2(\mathbf{b}_{ij}^T\hat{\mathbf{d}})^2, \tag{4}$$

where $\mathbf{b}_{ij} \in \mathbb{R}^V$ is a vector containing only the 4 non-zero elements $\{1, -\beta_1^{ij}, -\beta_2^{ij}, -\beta_3^{ij}\}$. Collecting all constraints in a matrix form, the pairwise term in (1) becomes

$$E_{smooth}(\hat{\mathbf{d}}) = \hat{\mathbf{d}}^T\mathbf{B}^T\mathbf{W}^2\mathbf{B}\hat{\mathbf{d}}, \tag{5}$$

where $\mathbf{B}$ is a sparse matrix formed of the row vectors $\mathbf{b}_{ij}^T$, and $\mathbf{W} = diag\{w_{ij}\}$ is the color-weighting matrix. After substitution of (2) and (5) into (1), and differentiation, the minimization of (1) boils down to solving

$$(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{B}^T\mathbf{W}^2\mathbf{B})\hat{\mathbf{d}} = \mathbf{A}^T\mathbf{d}. \tag{6}$$

for the vertex depths $\hat{\mathbf{d}}$ given the GCP depths $\mathbf{d}$. This is a sparse linear $V \times V$ system (V the number of vertices), which can be solved efficiently by using a linear solver.

3.3. Point clustering and α-shapes

Reconstruction of a 2D base mesh covering the whole image with the method of Section 3.2 results in a watertight "blanket" over the full field of view, undesirably closing discontinuities. Therefore, we follow a conservative strategy: we identify (probably concave) image regions with sufficient SfM support per view, and reconstruct (the submesh observed within) each of these regions separately.

First, we cluster the subset of 3D points observed per view (Figure 2a), rather than clustering the full point cloud at once. This highly increases the chance that point clusters on surfaces separated in depth end up in different groups. A region growing in the 3D k-nearest neighborhood, with a greedy choice of seeds (preferring points with denser neighborhood) proved to be sufficient. It identifies outlier SfM points as unassigned points, which we remove prior to fitting. Other clustering algorithms could also be used here.

Second, the observed 2D shape of each point cluster is extracted via standard α-shapes [13]. This returns a set of triangles of the 2D Delaunay-triangulation over the points. This triangulation, defined over projected SfM points, is different than our 2D base mesh (see Figure 4). The lowest α value resulting in only manifold vertices is found automatically by trying out increasing values $\{\alpha_i\}$. This guarantees that the outline of each α-shape component can be unambiguously extracted as the longest polygonal boundary. The largest (worst-case) α_i value is set to ∞, which corresponds to the convex hull of the 2D point cluster. As a result, each region sufficiently populated by SfM point observations is circumscribed by a polygon (Figure 2a). These polygons are used to cut out submeshes of our image-aligned 2D base mesh, which are then subject to separate reconstruction.

3.4. Identifying discontinuities and occlusions

The methods in Section 3.3 already handle major discontinuities, but surfaces with self-occlusions may survive that procedure as a single point cluster. This results in self-occlusion discontinuities closed. If discontinuity seams over consecutive edges were known, the 2D mesh could be simply modified to incorporate these seams into the reconstruction. Unfortunately, they are difficult to locate at edge precision, and mislocating them can lead to unpleasant boundary artifacts. Thus, we rather identify discontinuities as groups of faces that are observed in sharp angles in an initial reconstruction (see Figure 5 for an example).

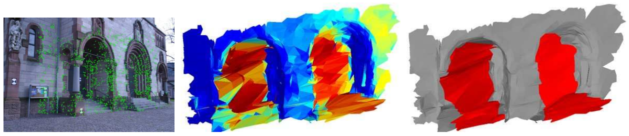

Figure 5. Example for discontinuity detection: input image and SfM points (left), reconstructed model after point clustering, observation angles color-coded (middle), segmentation result (right).

We formulate this problem as a binary segmentation over all F mesh faces $\{f_i\}$ by minimizing

$$E(\mathcal{L}) = \sum_{i=1}^{F} E_i^{disc}(l_i) + \lambda_{disc} \cdot \sum_{\{f_i, f_j\}} e_{ij}\mathbb{I}[l_i \neq l_j] \tag{7}$$

over a labelling solution $\mathcal{L} = \{l_1, \ldots, l_F\}$, where $l_i \in \{0, 1\}$ is a binary label for face f_i, indicating whether it is a discontinuity. $e_{ij} \cdot \mathbb{I}[l_i \neq l_j]$ is a weighted Potts penalty where e_{ij} is the relative edge length between face f_i and f_j. The unary term measures the observed angle of a face as $E_i^{disc}(0) = \phi_i$ and $E_i^{disc}(1) = 1 - \phi_i$ with

$$\phi_i = \exp\{-\tfrac{1}{2\sigma^2}(\mathbf{v}_i^T\mathbf{n}_i)^2)\} \tag{8}$$

where $\mathbf{n}_i$ is the unit-normal of face f_i, and $\mathbf{v}_i$ is the viewing direction of the centroid $\mathbf{c}_i$ of f_i. We solve (7) via graph-cuts [6, 5, 31], remove the segmented discontinuity triangles, and recompute the depths at the remaining vertices for each α-shape separately.

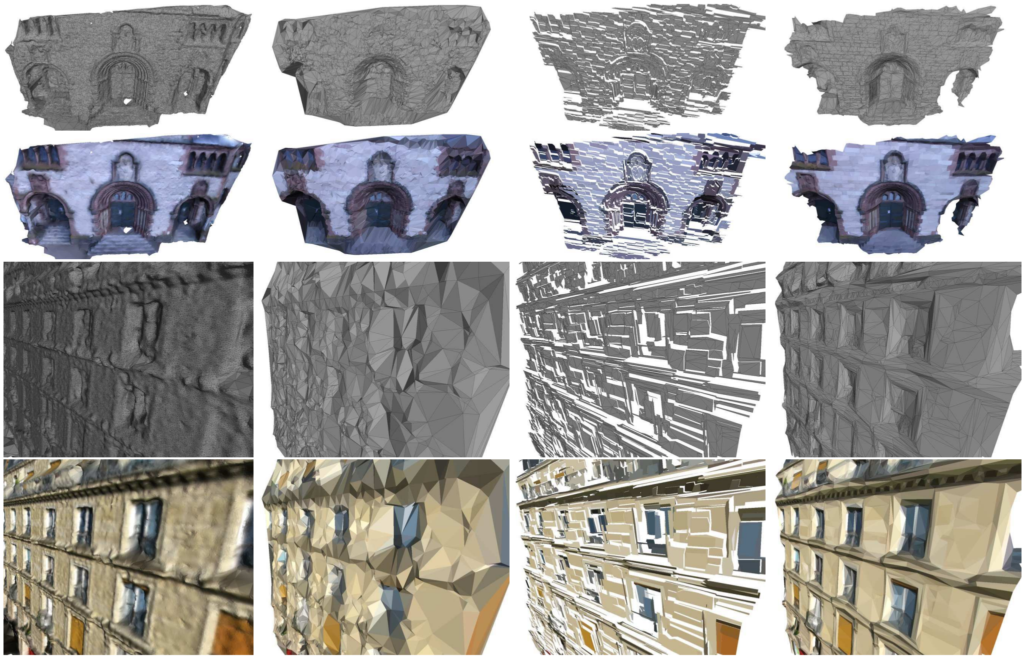

Figure 6. Results for different methods on the Herz-Jesu (top) and Mirbel datasets (bottom). Columns: (1) CMP-MVS mesh, (2) SfM+DT: 2D Delaunay triangulation over SfM points, (3) FS+Fronto+Soft: depths of FS superpixels with fronto-parallel assumption and SfM points as soft-constraints, (4) our method over FS superpixels. Rows: bare geometry and untextured mesh with per-face mean coloring. Note the level of detail our method can capture from only the sparse SfM data (inlier SfM points are at DT vertices in the 2nd column).

			SfM	Ours	PMVS2	CMP-MVS	Input		8×	GPU	1×	
Dataset	Images(±)	Resolution	#pts	#tri/im	#pts	#tri	t_{match}	t_{sfm}	t_{pmvs2}	t_{cmp}	t_{ours}	t_{par}
Street Z	630 (630)	800×1200	238.9k	15.4k	1 620k	3 927k	16051s	207s	1624*	24061s+	1171s	1.9s
Street P	428 (10)	800×1067	365k	13.4k	1 630k	4 697k	710s	811s	1290s*	16143s+	843s	2.0s
Street M	26 (26)	800×1067	19.5k	14.2k	93.3k	1 253k	45s	4s	49s*	1083s+	50.9s	2.0s
LeuvenCastle	28 (28)	800×600	16.2k	9.1k	29.6k	586.0k	69s	3s	28s*	825s+	32.3s	1.2s
Medusa	15 (15)	800×640	16.3k	9.4k	30.8k	536.1k	19s	1s	16s*	470s+	18.0s	1.2s
Fountain	11 (11)	1024×683	13.5k	10.1k	44.5k	707.9k	8s	1s	20s*	468s+	16.3s	1.5s
HerzJesu	8 (8)	1024×683	8.3k	10.4k	35.2k	492.5k	7s	1s	16s*	334s+	10.6s	1.3s
Dionysos	8 (8)	800×600	4.1k	7.3k	17.4k	243.3k	7s	1s	16s*	229s+	10.0s	1.3s
MertonVI	6 (6)	1024×768	2.1k	13.3k	10.8k	180.5k	2s	1s	9s*	123s+	9.9s	1.7s

Table 1. Datasets and timings. *8-core parallelism, +GPU usage. t_{ours} is given for single CPU core. ± indicates the number of previous and next images considered for matching. t_{par} is the runtime of our method for a single view (views fully paralellizable after SfM).

4. Results

We tested our method on landmark scenes (MertonVI [64], Herz-Jesu [55] and LeuvenCastle [45]), our medium to large-scale street scenes M, Z, P ranging to over 600 images, as well as on relief sequences (Medusa [45], Fountain [55] and Dionysos), see Table 1. We used SfM data produced by VisualSfM [66, 68] with SiftGPU [65].

Qualitative evaluation. Figure 6 compares our single-view method to the state-of-the-art multi-view stereo tool CMP-MVS [29], and to two baseline methods that estimate a depth map given sparse depths at GCPs in a single view. The first method (SfM+DT) uses a Delaunay-triangulation over 2D observations of SfM points, which is lifted to 3D via the known SfM depths to obtain a 3D mesh (see also Fig. 4, left). This is exploited in [24] for stereo and in [49] for MVS. The second method (FS+Fronto+Soft) operates over (FS [14]) superpixels, assumes a single depth per superpixel (fronto-parallel plane), estimates the depths by using the known SfM point depths as soft-constraints while smoothing by penalizing for color-weighted depth differences between superpixels. Different aspects of FS+Fronto+Soft are motivated by [71, 16, 32]. For a fair evaluation, all methods are applied after the point clustering proposed in Section 3.3, i.e. after removing SfM outliers. Figure 6 demonstrates that meshes of SfM+DT have triangles crossing actual crease edges as the method is not aware of observed edge locations. Also, it is fully susceptible to noise in the SfM points, as it uses them as hard-constraints. SfM+DT has the advantage over FS+Fronto+Soft that it keeps the mesh naturally watertight. The piecewise-continous assumption of FS+Fronto+Soft makes it difficult to produce a watertight mesh. This also holds for slanted-plane stereo methods by principle (e.g. [62, 70]), the fronto-parallel assumption [63, 71] only emphasizes the problem. In turn, our method respects geometric edges, produces a connected mesh (per α-shape), and the mesh captures the important crease edges of the structure, which are observable as image edges.

For further models produced by our method, we refer to Figure 9 and the supplementary material.

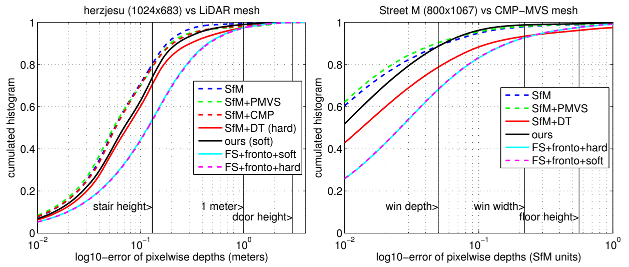

Figure 7. Cumulated histograms of pixelwise depth errors over all images of Herz-Jesu w.r.t. the LiDAR mesh [55] (left) and Street M w.r.t. the CMP-MVS [29] mesh (right). Three vertical markers are at the size of a stair, 1 meter and door height for Herz-Jesu, and window depth, width and floor height for Street M.

Depth accuracy. We evaluate the accuracy of our meshes w.r.t. a high-resolution reference mesh $\mathcal{M}^r$. In comparison, we evaluate the methods SfM+DT, FS+Fronto+Soft, CMP-MVS introduced above, and also compare to the accuracy of the source SfM point cloud, of the dense point cloud produced by Patch-based Multi-View Stereo, PMVS2 [19], and of the method FS+Fronto+Hard which is similar to FS+Fronto+Soft but applies the known sparse point depths as hard constraints (motivated by [61]).

The discrepancy between a mesh $\mathcal{M}$ and the reference mesh $\mathcal{M}^r$ is measured by comparing the depth maps rendered of the two meshes from the same viewpoint i, knowing the camera matrix $\mathbf{P}_i^r$. In the same vein, the depths of points w.r.t. view i are compared to the depth maps of $\mathcal{M}^r$ in the views according to the visibility information from SfM/PMVS2. We collect the pixelwise/pointwise absolute depth errors for all views of a dataset, and compute the cumulated density function of the errors. Figure 7 shows the error curves for the Herz-Jesu and Street M datasets. For Street M, we use the high-resolution CMP-MVS mesh as reference mesh $\mathcal{M}^r$, whereas for Herz-Jesu, the publicly available LiDAR mesh [55] is used as ground-truth. To compare the reconstructed meshes given in the SfM frame to the LiDAR mesh given in a metric frame, a precise registration procedure is carried out between the frames via the known SfM points, ground-truth cameras and depth maps.

In Figure 7, vertical markers are placed at meaningful scales, e.g. stairstep/door/floor height. The higher the curve of a method, the higher the precision. The plots consistently show that state-of-the-art dense-MVS methods (CMP-MVS, PMVS2) provide the highest precision. The proposed single-view surface fitting method (black curve) has higher accuracy than all tested single-view baseline methods (SfM+DT, FS+Fronto+Soft, FS+Fronto+Hard), and has comparable accuracy of dense-MVS methods in the scales particularly interesting for city scenes (e.g. starting around the depth of a window or the height of a stair).

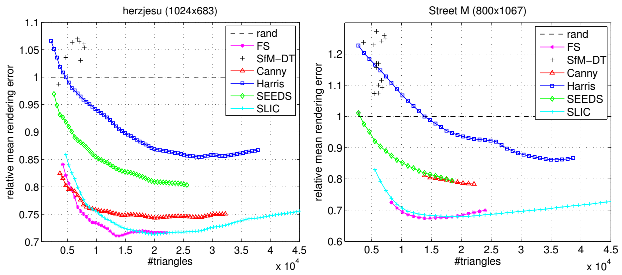

Figure 8. Errors of a per-face flat rendering for different 2D triangulation methods in function of the triangle count per image, for two different datasets. Errors are relative to those of a triangulation over uniform random points. The curves are averages over all images at each triangle count. For SfM-DT (+), each marker corresponds to the error of a single triangulation per image.

Rendering quality. In Figure 3, we present different ways to obtain our 2D base mesh. To evaluate these, we rasterize the 2D mesh with mean image colors per triangle – which is equivalent to rendering the 3D mesh from the particular viewpoint with per-face flat coloring – and compare the rasterized image to the original one. We measure the discrepancy by averaging absolute differences over color channels, pixels and all images. This rendering quality implicitly indicates how well the decomposition is aligned with image edges, e.g. for semantic classification or visualization.

Figure 8 compares Delaunay triangulations over Harris corners [25], polygonized Canny edges [7], and polygonized FS [14], SEEDS [3] and SLIC [1] superpixel boundaries, for the Herz-Jesu and Street M datasets. For each method, we vary its core parameter (e.g. Canny threshold) to obtain different triangle counts. As a baseline, we evaluated triangulations on top of n uniform random points over the image (*rand* method), and report all accuracies relative that of *rand*. For comparison, we also show the result of SfM+DT introduced earlier, i.e. a triangulation over projected SfM points (originating from SIFT [35] points in VisualSFM). We conclude from Figure 8, that FS and SLIC superpixels (slightly slower) approximate the image best at any number of triangles, while Harris corners and the SfM+DT give the lowest accuracy. The fastest methods, SEEDS and Canny reside in between.

Timings. Particularly positive aspects of our method are its speed and potential for parallelization. Table 1 shows timings for different datasets processed on a single 3.4 GHz

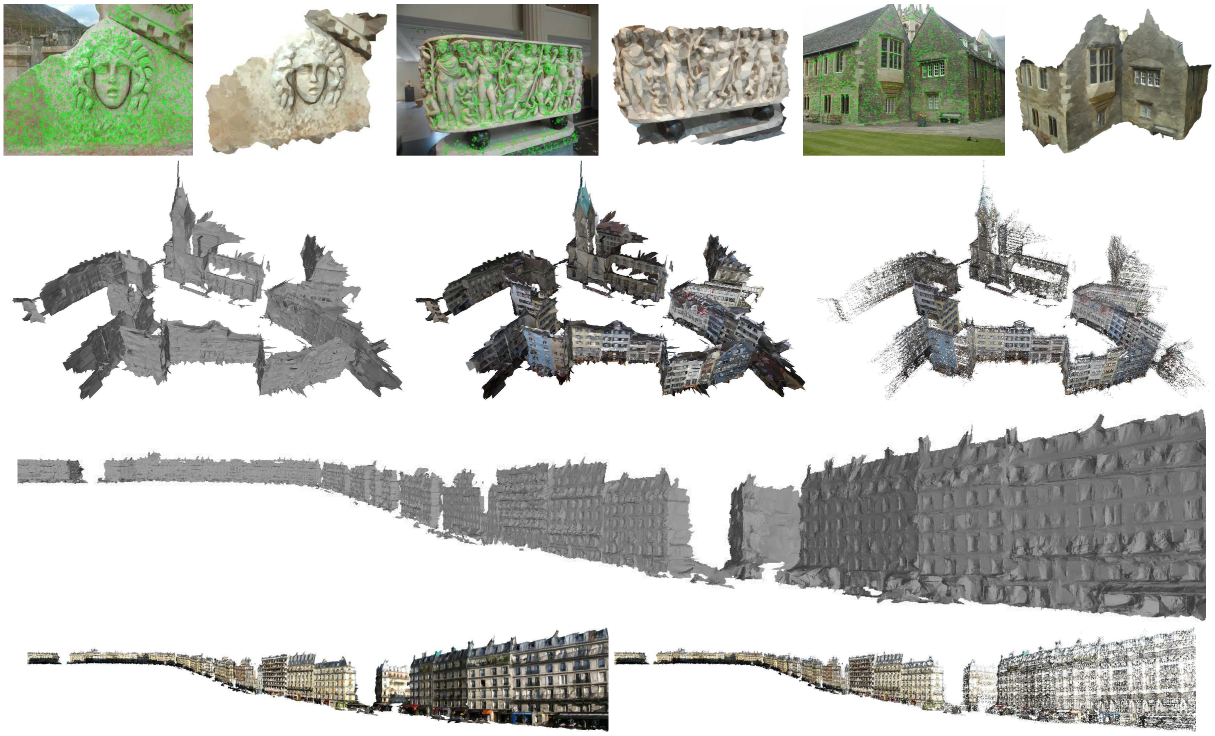

Figure 9. Some of our results (for close-up views or further models, please refer to the supplementary material). Top: output model (right) computed from the single input image and the overlaid SfM points shown on the left for Medusa, Dionysos and Merton VI. Below, two of our large-scale results: Street Z (middle) and Street P (bottom). Colored and uncolored single-view meshes are displayed jointly from all views without volumetric mesh fusion. The colored vertices of all single view meshes are also separately shown as point clouds. All models shown in this paper are rendered without texture, by using per-face flat coloring with mean colors taken from the images.

CPU core in Matlab[2]. After SfM, the largest dataset of 630 images of around 1 MPixel is processed in 28 minutes (non-parallel) vs. 6.7 hours of CMP-MVS, which uses a recent GPU. PMVS2 returns a semi-dense point cloud in 27 minutes but using 8-core parallelism. Our algorithm spends around 0.8-3 seconds per 1 MPixel image, depending on the triangulation method. The majority of this time is spent on 2D base mesh extraction, and the actual depth reconstruction (with discontinuity segmentation) runs in less than 0.3 seconds per view in Matlab. Moreover, our method is fully parallelizable per image (see last column of Table 1).

5. Conclusions

In this work, we proposed a novel method for direct mesh reconstruction from single images and sparse SfM data. Our method ensures that the reconstructed meshes are edge-aligned with image gradients yielding a fairly compact surface representation, yet allowing for good-quality textureless rendering. Unlike many other methods, our depth optimization operates on mesh vertices and penalizes curvature, rather than favoring fronto-parallel planes, which leads to cleaner surfaces and inherent watertightness where needed. Grace to our linear formulation, an efficient sparse linear solver already renders the method fast on CPU. Our approach is fully parallelizable per single view (unlike MVS approaches), and is linear in the number of views. It does not rely on dominant orientations (e.g. Manhattan-world), normal computation/clustering or direct plane fitting, which are hard to stabilize over sparse data. This work treats surface modeling as many single-view problems, and did not aim to do volumetric mesh fusion, as such methods would ruin good scalability or edge-alignment. If merging is needed anyway, we found that Poisson remeshing [30] works well over our meshes. As the meshes are already fairly consistent, we rather consider further speed-up by view selection and mesh trimming in the future.

Overall, our method avoids the computational bottlenecks of dense MVS, hence, we believe it is a good alternative to do lightweight modeling where there is no need and/or time to compute an extremely detailed pixelwise 3D model. In particular, our method makes large-scale ground-level urban modelling practically feasible, as we can reconstruct entire street scenes with LoD-3 details from ground-level images in a matter of minutes.

Acknowledgement. We thank Till Kroeger for his comments. This work was supported by the European Research Council (ERC) project VarCity (#273940).

[2] A majority of low-level tasks are implemented as C++ MEX.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *PAMI*, 34(11):2274–2282, 2012. 7

[2] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *ICCV*, 2009. 2

[3] M. Bergh, X. Boix, G. Roig, B. Capitani, and L. Van Gool. SEEDS: Superpixels extracted via energy-driven sampling. In *ECCV*, 2012. 7

[4] A. Bódis-Szomorú, H. Riemenschneider, and L. Van Gool. Fast, Approximate Piecewise-Planar Modeling Based on Sparse Structure-from-Motion and Superpixels. In *CVPR*, 2014. 2

[5] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):124–1137, 2004. 5

[6] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23(11):1222–1239, 2001. 5

[7] J. Canny. A computational approach to edge detection. *PAMI*, 8(6):679–698, 1986. 7

[8] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *SIGGRAPH*, 2001. 2

[9] A. Chauve, P. Labatut, and J. Pons. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *CVPR*, 2010. 2

[10] N. Cornelis, K. Cornelis, and L. Van Gool. Fast compact city modeling for navigation pre-visualization. In *CVPR*, 2006. 2

[11] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Sfm with mrfs: Discrete-continuous optimization for large-scale structure from motion. *PAMI*, 35(12):2841–2853, 2013. 2

[12] A. Delaunoy and M. Pollefeys. Photometric bundle adjustment for dense multi-view 3d modeling. In *CVPR*, 2014. 2

[13] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551559, 1983. 5

[14] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004. 6, 7

[15] F. Fraundorfer, K. Schindler, and H. Bischof. Piecewise planar scene reconstruction from sparse correspondences. *IJCV*, 24(4):395–406, 2006. 2

[16] P. Fua. A parallel stereo algorithm that produces dense depth maps and preserves image features. *Machine Vision and Applications*, 6(1):35–49, 1993. 3, 4, 6

[17] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski. Manhattan-world stereo. In *CVPR*, 2009. 2

[18] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *CVPR*, 2010. 1, 2

[19] Y. Furukawa and J. Ponce. Accurate, Dense, and Robust Multi-View Stereopsis. *PAMI*, 32(8):1362–1376, 2010. 7

[20] D. Gallup, J. Frahm, and M. Pollefeys. Piecewise planar and non-planar stereo for urban scene reconstruction. In *CVPR*, 2010. 2

[21] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *CVPR*, 2007. 2

[22] D. Gallup, M. Pollefeys, and J.-M. Frahm. 3d reconstruction using an n-layer heightmap. In *DAGM*, 2010. 2

[23] I. Garcia-Dorado, I. Demir, and D. G. Aliaga. Automatic urban modeling using volumetric reconstruction with surface graph cuts. *ComputersGraphics*, 37(7):896–910, 2013. 2

[24] A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *ACCV*, 2010. 3, 6

[25] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference (AVC)*, 1988. 7

[26] D. Hoiem, A. Efros, and M. Hebert. Recovering surface layout from an image. *IJCV*, 75(1), 2007. 3

[27] C. Hoppe, M. Klopschitz, M. Donoser, and H. Bischof. Incremental surface extraction from sparse structure-from-motion point clouds. In *BMVC*, 2013. 2

[28] C. Hoppe, M. Klopschitz, M. Rumpler, A. Wendel, S. Kluckner, H. Bischof, and G. Reitmayr. Online feedback for structure-from-motion image acquisition. In *BMVC*, 2012. 2

[29] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR*, 2011. 1, 2, 6, 7

[30] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics*, pages 61–70, 2006. 2, 8

[31] V. Kolmogorov and R.Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2):147–159, 2004. 5

[32] J. Kopf, M. F. Cohen, and R. Szeliski. First-person hyperlapse videos,. In *SIGGRAPH*, 2014. 3, 4, 6

[33] P. Labatut, J. Pons, and R. Keriven. Efficient Multi-View Reconstruction of Large-Scale Scenes using Interest Points, Delaunay Triangulation and Graph Cuts. In *ICCV*, 2007. 2

[34] L. Ladicky, J. Shi, and M. Pollefeys. Pulling things out of perspective. In *CVPR*, 2014. 3

[35] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 7

[36] A. Martinović, J. Knopp, H. Riemenschneider, and L. Van Gool. 3d all the way: Semantic segmentation of urban scenes from start to end in 3d. In *CVPR*, 2015. 2

[37] A. Martinovic, M. Mathias, J. Weissenberg, and L. Van Gool. A Three-Layered Approach to Facade Parsing . In *ECCV*, 2012. 2

[38] B. Micusik and J. Kosecka. Multi-view superpixel stereo in urban environments. *IJCV*, 89(1):106–119, 2010. 1, 2

[39] D. D. Morris and T. Kanade. Image-consistent surface triangulation. In *CVPR*, 2000. 2

[40] P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. Van Gool, and W. Purgathofer. A survey of urban reconstruction. In *EUROGRAPHICS*, 2012. 1

[41] R. A. Newcombe and A. J. Davison. Live dense reconstruction with a single moving camera. In *CVPR*, 2010. 2, 3

[42] Y. Ohtake, A. Belyaev, and H.-P. Seidel. A multi-scale approach to 3d scattered data interpolation with compactly supported basis functions. In *Shape Modeling International*, pages 153–161, 2003. 2

[43] J. Park, S. Sinha, Y. Matsushita, Y.-W. Tai, and In So Kweon. Multiview photometric stereo using planar mesh parameterization. In *ICCV*, 2013. 2

[44] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3d reconstruction from video. *IJCV*, 78(2-3):143–167, 2008. 2

[45] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *IJCV*, 59(3):207–232, 2004. 2, 6

[46] T. Pylvanainen, J. Berclaz, V. Hedau, T. Korah, M. Aanjaneya, and R. Grzeszczuk. 3d city modeling from street-level data for augmented reality applications. In *3DPVT*, 2012. 2

[47] H. Riemenschneider, A. Bodis-Szomoru, J. Weissenberg, and L. Van Gool. Learning Where To Classify In Multi-View Semantic Segmentation. In *ECCV*, 2014. 2

[48] H. Riemenschneider, U. Krispel, W. Thaller, M. Donoser, S. Havemann, D. Fellner, and H. Bischof. Irregular lattices for complex shape grammar facade parsing. In *CVPR*, 2012. 2

[49] N. Salman and M. Yvinec. Surface reconstruction from multi-view stereo of large-scale outdoor scenes. *International Journal of Virtual Reality*, 9(1), 2010. 2, 6

[50] A. Saxena, M. Sun, and A. Ng. 3-d reconstruction from sparse views using monocular vision. In *ICCV*, 2007. 3

[51] A. Saxena, M. Sun, and A. Ng. Make3d: Learning 3-d scene structure from a single still image. *PAMI*, 2008. 3

[52] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, 2006. 2

[53] S. Sinha, D. Steedly., and R. Szeliski. Piecewise Planar Stereo for Image-based Rendering. In *ICCV*, 2009. 2

[54] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH*, 2006. 2

[55] C. Strecha, W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *CVPR*, 2008. 6, 7

[56] O. Teboul, L. Simon, P. Koutsourakis, and N. Paragios. Procedural modeling and image-based 3d reconstruction of complex architectures through random walks. In *IJCV*, 2010. 2

[57] E. Tola, C. Strecha, and P. Fua. Efficient large scale multi-view stereo for ultra high resolution image sets. *Machine Vision and Applications*, 23(5):903–920, 2012. 1

[58] G. Vogiatzis, C. Hernandez, P. Torr, and R. Cipolla. Multi-view Stereo via Volumetric Graph-cuts and Occlusion Robust Photo-Consistency. *PAMI*, 29(12):2241–2246, 2007. 2

[59] G. Vogiatzis, P. Torr, and R. Cipolla. Bayesian stochastic mesh optimisation for 3d reconstruction. In *BMVC*, 2003. 2, 3

[60] H.-H. Vu, P. Labatut, J. Pons, and R. Keriven. High Accuracy and Visibility-Consistent Dense Multi-view Stereo. *PAMI*, 34(5):889–901, 2012. 2, 3

[61] L. Wang and R. Yang. Global stereo matching leveraged by sparse ground control points. In *CVPR*, 2011. 3, 4, 7

[62] Z.-F. Wang and Z.-G. Zheng. A region based stereo matching algorithm using cooperative optimization. In *CVPR*, 2008. 7

[63] Y. Wei and L. Quan. Region-based progressive stereo matching. In *CVPR*, 2004. 3, 7

[64] T. Werner and A. Zisserman. New techniques for automated architecture reconstruction from photographs. In *ECCV*, 2002. 2, 6

[65] C. Wu. SiftGPU: A GPU implementation of Scale Invaraint Feature Transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu`, 2007. 6

[66] C. Wu. VisualSFM: A Visual Structure from Motion System. `http://ccwu.me/vsfm/`, 2011. 6

[67] C. Wu. Towards linear-time incremental structure from motion. In *3DV*, 2013. 2

[68] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *CVPR*, 2011. 6

[69] J. Xiao, T. Fang, P. Tan, P. Zhao, E. Ofek, and L. Quan. Image-based facade modeling. In *SIGGRAPH Asia*, 2008. 2

[70] Q. Yang, L. Wang, R. Yang, H. Stewenius, and D. Nister. Stereo matching with color-weighted correlation, hierarchical belief propagation, and occlusion handling. *PAMI*, 31(3):492–504. 7

[71] C. Zitnick and S. Kang. Stereo for image-based rendering using image over-segmentation. *IJCV*, 75(1):49–65, 2007. 3, 6, 7