Line3D: Efficient 3D Scene Abstraction for the Built Environment

Manuel Hofer, Michael Maurer, Horst Bischof

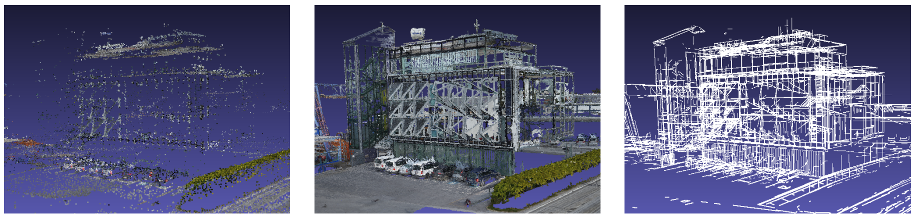
Institute for Computer Graphics and Vision
Graz University of Technology - Graz, Austria
http://www.icg.tugraz.at

Abstract. Extracting 3D information from a moving camera is traditionally based on interest point detection and matching. This is especially challenging in the built environment, where the number of distinctive interest points is naturally limited. While common Structure-from-Motion (SfM) approaches usually manage to obtain the correct camera poses, the number of accurate 3D points is very small due to the low number of matchable features. Subsequent Multi-view Stereo approaches may help to overcome this problem, but suffer from a high computational complexity. We propose a novel approach for the task of 3D scene abstraction, which uses straight line segments as underlying features. We use purely geometric constraints to match 2D line segments from different images, and formulate the reconstruction procedure as a graph-clustering problem. We show that our method generates accurate 3D models, with a low computational overhead compared to SfM alone.

1 Introduction

Recovering 3D information from an image sequence used to be a very challenging and time consuming task. Today, thanks to freely available software such as Bundler [24] or VisualSfM [26], even non-expert users are able to generate accurate 3D models from arbitrary scenes within hours. Since these so-called Structure-from-Motion (SfM) approaches operate on a sparse set of distinctive feature points (e.g. SIFT [18] features), the resulting 3D point cloud is usually quite sparse as well. The more important part of the SfM result are the obtained camera poses for each input image, which enable subsequent Multi-View Stereo (MVS) pipelines (e.g. PMVS [8] or SURE [21]) to create a (semi-) dense point cloud.

While the first part of this two-step procedure (pose estimation via SfM) can be computed very efficiently even for large crowd-sourced datasets [7, 10], the second part (dense reconstruction via MVS) is still computationally expensive and can take up to several days even on modern desktop computers. Moreover, the resulting 3D point cloud might easily consist of millions of points and just viewing it in a point-cloud viewer quickly becomes a very tedious task. The same holds for any kind of automatic data analysis or post processing (e.g. meshing [17]). This is due to the nature of using point clouds as a representation of a 3D



SfM [15]
46,572 points
(a) runtime: 2.25 hours

PMVS [8]
12,156,664 points
(b) runtime: > 11 hours

Line3D (proposed)
13,489 lines
(c) runtime: **375.63 sec**

Fig. 1: Three different 3D representations of the *BUILDING* sequence (344 images). (a) Sparse 3D model [15]. (b) Semi-dense point-cloud (PMVS [8]). (c) 3D line model using *Line3D*. As we can see, it is hardly possible to recognize the building in the sparse 3D model, while it is clearly recognizable in both the semi-dense- and the line-based 3D model. Compared to PMVS, our method has much lower runtime- and memory requirements.

model. On the one hand, shapes of arbitrary complexity can be described by a set of 3D points, but on the other hand, the number of points needed to do so can quickly exceed the capabilities of your system.

What would be desirable is an efficient way of abstracting the 3D model, so that as much 3D information as possible can be encoded with only as much data as really necessary. A natural choice would be to use more complex geometric primitives as data representation, such as planes (e.g. [20]) or lines (e.g. [12]). While this might not be sufficient for natural scenes (e.g. forests, etc.), it is especially useful for the built environment, where most of the structures are piece-wise planar/linear.

We propose a novel approach for the task of 3D scene abstraction, denoted as *Line3D*, which makes use of straight line segments as data representation. Our method works as an efficient SfM post-processing tool and positions itself in between sparse and dense 3D reconstruction. We build on recent methods [11–14], which use epipolar-guided line segment matching and formulate the 3D reconstruction as a clustering problem. Our main contributions are the reformulation of the scoring procedure of matched 2D segments in a less restrictive way, the replacement of the simple graph-clustering procedure in [11, 12] with a more recent matrix-diffusion based method [4], as well as the computation of affinities between potentially matching segments using a linear function of their estimated depth and user specified regularization parameters in the pixel space. These modifications ultimately result in more complete 3D models without negatively influencing the runtime.

Figure 1 shows a comparison between a sparse-, dense-, and a line-based 3D model for an urban scene. As we can see, our reconstruction provides a high amount of 3D information, despite its sparsity compared to the dense model. Moreover, running our method is only a low computational overhead, even for this relatively large-scale dataset. The source code of our method is publicly available and can be downloaded from **http://aerial.icg.tugraz.at**.

2 Related Work

While line segments have been used for tasks such as image registration or 3D reconstruction for a long time (e.g. [2]), in recent years image-based 3D reconstruction has been dominated by the use of image feature-points and their invariant descriptors (e.g. SIFT [18]). Only quite recently, the principles of feature-point descriptors have been successfully ported to the task of line segment matching (e.g. [28–30]), but line-based 3D reconstruction for real-world scenarios is still rarely used. While earlier methods have severe limitations (e.g. Manhattan-world assumption [23]), more recent approaches [11–14, 16] have successfully been deployed on challenging datasets. They all require known camera poses (e.g. by running a conventional SfM pipeline beforehand), since pose estimation using line segments can only be done in special scenarios (e.g. by using triplets of two parallel and one orthogonal lines [5]), with given 3D lines [27], or when explicit endpoint correspondences can be established [19].

Jain et al. [16] proposed a method that does not require explicit correspondences between line segments from different images, which enables 3D reconstruction under difficult lighting conditions or around highly non-planar objects (such as power pylons), where patch-based line descriptors would fail. They formulate the reconstruction procedure as an optimization problem, where the unknown depth of the endpoints of 2D line segments in the images is modelled as a random variable. They compute the most probable 3D locations for the segment endpoints by minimizing the reprojection error among several neighboring views, and compute a final 3D model by merging individual 3D hypotheses that are sufficiently close together. While their approach generates visually pleasant results, the continuous optimization of the endpoint depths, in a potentially large range, renders the method inefficient for large-scale datasets.

To overcome these issues, Hofer et al. [13, 14] replaced the continuous depth estimation with epipolar guided line segment matching, to limit the number of possible 3D locations to a discrete set. They further replaced the greedy line-merging from [16] with a scale invariant graph clustering formulation [12], which can also be evaluated on-the-fly for incremental SfM applications [11].

We build up on the core principles presented in [11–14], which are appearance-less line segment matching and global graph-clustering of corresponding segments across images. We demonstrate how the resulting 3D reconstructions can be improved by making several adaptions to their original formulation, without sacrificing runtime performance.

3 3D Reconstruction Using Line Segments

Given an (unordered) image sequence $I = \{I_1, \ldots, I_N\}$, we first run an arbitrary SfM pipeline to obtain the corresponding camera poses as well as a sparse set of

3D points $X = \{X_1, \dots, X_K\}$, which is needed solely to define which images are visual neighbors. We further define $X(i) \subset X$ to be the set of 3D points which are visible in image I_i. We require a set of 2D line segments $L_i = \{l_1^i, \dots, l_{m_i}^i\}$ for each image, where each segment l_m^i simply consists of two endpoints $p_m^i, q_m^i \in \mathbb{R}^2$. The line segments can be obtained by any line segment detector, such as LSD [9] or EDL [1].

Similar to [11, 12], our method consists of several steps: (1) establishing potential correspondences between line segments from different images, (2) evaluating these correspondences based on their support in neighboring views, (3) selecting the most plausible correspondence for each 2D segment as its 3D position hypothesis, and (4) clustering 2D segments based on their spatial proximity in 3D to obtain the final correspondence set and 3D model.

3.1 Establishing Line Segment Correspondences

To generate a line-based 3D model we need to establish correspondences between 2D line segments from different images. Theoretically, this could be done by one of the numerous line-matching approaches presented in the past (e.g. [28–30]). However, most of these approaches are patch-based and are therefore only suitable for line segments located on planar surfaces. Most of the line segments in natural images correspond to depth discontinuities, which results in line descriptors describing the potentially far away background. To overcome this drawback, recent methods have demonstrated how correspondences can be established and verified using purely geometric principles, without any kind of appearance [12–14], or with color histogram-based line descriptors [3] as weak support [11].

We follow [11–14] and use epipolar matching constraints to establish a set of potential correspondences for each line segment l_m^i individually. Since it would be infeasible (and unnecessary) to match all images with each other, we first compute a set of visual neighbors $V_i \subset \{1, \dots, N\} \backslash \{i\}$ for each image I_i, by finding its M nearest neighbors in terms of Dice's similarity coefficient

$$S_I(i, j) = \frac{2 \cdot |X(i) \cap X(j)|}{|X(i)| + |X(j)|}, \tag{1}$$

which sets the number of common worldpoints in relation to the total number of worldpoints for each image (the higher the more similar).

We then match all segments in L_i to all segments in L_j (if $j \in V_i$). For a specific segment pair, $l_m^i \in L_i$ and $l_{\bar{m}}^j \in L_j$, we compute the epipolar lines of their endpoints in the opposite image. We then simply intersect the infinite lines passing through the segments l_m^i and $l_{\bar{m}}^j$ with the epipolar lines, and compute the overlap of the region between the intersection points with the original segments. If both relative overlaps (normalized by the length of the respective segment l_m^i or $l_{\bar{m}}^j$) are above a fixed threshold τ, we consider l_m^i and $l_{\bar{m}}^j$ to be potentially matching ($\tau = 0.25$ in all our experiments).

As shown in [11–14], we can transform each 2D correspondence into a 3D line $H^{i,j}_{m,\bar{m}}$ by intersecting the two planes passing through the respective camera centers $C_i, C_j \in \mathbb{R}^3$, and the 2D segments. We compute two 3D line segment hypotheses ($h^{i,j}_{m,\bar{m}}$ and $h^{j,i}_{\bar{m},m}$) for each correspondence, which are defined as 3D line segments on $H^{i,j}_{m,\bar{m}}$, whose projected endpoints coincide with the endpoints of the 2D line segments l^i_m and $l^j_{\bar{m}}$ respectively. Similar to the 2D case, a 3D line segment consists of two 3D points ($h^{i,j}_{m,\bar{m}} = \{P^{i,j}_{m,\bar{m}}, Q^{i,j}_{m,\bar{m}}\}$). Note that $H^{i,j}_{m,\bar{m}} = H^{j,i}_{\bar{m},m}$, while in general $h^{i,j}_{m,\bar{m}} \neq h^{j,i}_{\bar{m},m}$ (due to occlusions and imprecise 2D segment detections).

3.2 Evaluating Line Segment Correspondences

The matching procedure enables us to establish a potentially large set of correspondences, most of which are of course incorrect. Since we only use weak epipolar constraints, it is not possible to distinguish correct from incorrect matches during matching. However, we can assign confidence values for correspondences after L_i has been matched with all visual neighbors. This can either be done using gradient-based backprojection and scoring of the 3D hypotheses over multiple images [14, 16] (which is time consuming), or by directly analysing their 3D similarity to each other [11–13] (which requires some scale information). Both methods are based on the observation that correct hypotheses of a 2D segment always support each other (e.g. they are close together in 3D space and project to similar locations in the images), while this does not hold for incorrect ones.

To be scale invariant and fast, we use a novel similarity measure based on positional- and angular reprojection errors between a 3D hypothesis and 2D segments. We assign a confidence

$$c(h^{i,j}_{m,\bar{m}}) = \sum_{x \in V_i \setminus \{j\}} \max_{y \in \{1,\dots,m_x\}} \left\{ A_{2D}(\mathit{\Gamma}_x(h^{i,j}_{m,\bar{m}}), l^x_y) \right\}, \qquad (2)$$

to a correspondence $h^{i,j}_{m,\bar{m}}$, where $\mathit{\Gamma}_x$ projects a 3D line segment into an image I_x, and A_{2D} computes a truncated affinity between two 2D segments. This affinity is defined as

$$A_{2D}(l_1, l_2) = \begin{cases} S^a_{2D}(l_1,l_2) \cdot S^p_{2D}(l_1,l_2) & \text{if } S^a_{2D}(l_1,l_2) \cdot S^p_{2D}(l_1,l_2) > 0.5 \\ 0 & \text{otherwise} \end{cases}, \qquad (3)$$

with S^a_{2D} being an *angular* similarity, and S^p_{2D} being a *position* similarity defined as

$$S^a_{2D}(l_1,l_2) = \exp\left(-\frac{\angle(l_1,l_2)^2}{2\sigma_a^2}\right) \qquad S^p_{2D}(l_1,l_2) = \exp\left(-\frac{d_{\max}(l_1,l_2)^2}{2\sigma_p^2}\right), \qquad (4)$$

where $\angle(l_1, l_2)$ denotes the angle between the two line segments (in degrees), and $d_{\max}(l_1, l_2)$ is the maximum normal distance between the endpoints of l_1 to the infinite line passing through l_2, and vice versa. σ_a and σ_p are user specified regularization parameters.

With this formulation we are able to determine whether a matching hypothesis makes sense or not. We only keep hypotheses for further processing for which $c(h^{i,j}_{m,\bar{m}}) > 1$, which means that at least two segments from two additional images (apart from I_i and I_j) have to support $h^{i,j}_{m,\bar{m}}$. We end up with a much sparser set of correspondences, with a significantly lower number of outliers, while correct hypotheses are only seldom removed.

3.3 Assigning 3D Locations to 2D Segments

As in [11–13], given all hypotheses $h^{i,j}_{m,\bar{m}}$ for a 2D segment l^i_m, we want to estimate its most probable 3D position, since each 2D segment can only be a projection of one specific 3D structure. We then use this 3D information for the following clustering procedure, as first shown in [11, 12]. For each 2D segment l^i_m we define its 3D location as

$$\hat{h}^i_m = \underset{h^{i,j}_{m,\bar{m}}}{\operatorname{argmax}} \left\{ c(h^{i,j}_{m,\bar{m}}) \right\} , \tag{5}$$

which is simply its 3D hypothesis with the highest confidence. We additionally normalize the associated confidence $c(\hat{h}^i_m) = \min\{1, c(\hat{h}^i_m)/2\}$, such that $c(\hat{h}^i_m) = 1$ means a hypothesis is supported by ≥ 4 images (see Section 3.2). In contrast to [11, 12], where confidences are always normalized linearly by the locally highest confidence value (per image), we normalize by a fixed value. We have seen that 3D segment hypotheses verified by 4 or more images are almost never incorrect, which can also be observed for SfM point-clouds on the 3D point level. This enables correct matches which are only found in a low number of visual neighbor images (due to occlusions, etc.) to obtain a high confidence, despite the potential occurrence of other correspondences from the same image which might be occluded less often. Since this procedure is purely local, it can be easily done even for large-scale datasets.

3.4 Clustering 2D Segments Across Images

To perform the segment clustering we need an affinity matrix W, which holds the pairwise similarities between all potentially matching 2D segments. Since we only need to consider segment pairs which have been matched before, this matrix is usually very sparse. The question is how these similarities should be computed. We could use the same metric as for the hypothesis confidence above, by projecting 3D segments into images and evaluating the projective score (see Eq. 3). The problem with this procedure is that the reprojection error is not necessarily an appropriate indicator for a good correspondence, since it might be small despite a large spatial displacement. Hence, it is desirable to compute similarities directly in the 3D space.

To achieve this, we need some scale information. Since it is not possible to obtain a metric 3D reconstruction from a conventional SfM pipeline (unless further knowledge about the scene is provided, e.g. ground control points [22]), we have to find a way to derive a scale estimate from the reconstruction. Motivated by [11–13], we use user defined uncertainty thresholds in the pixel space, which are then brought into the local 3D space of the reconstruction. Unlike in these approaches, where an estimate about spatial uncertainty thresholds is made using all potential 3D line segment hypotheses (correct as well as incorrect ones), we formulate the uncertainty estimation as a linear function of the scene depth with respect to the underlying camera geometry, which is more robust to outlier hypotheses.

We aim at converting an uncertainty threshold t from the pixel space into the 3D space, for each image I_i individually. Therefore, we define z_i to be the center point of I_i, and $\tilde{z}_i$ to be z_i shifted by t (in any direction). We unproject z_i from the image at a distance of 1, and obtain a 3D point Z_i. We then shoot a 3D ray through $\tilde{z}_i$ and compute the normal distance k_t^i between Z_i and this ray. We use this distance as the slope of our linear uncertainty function

$$u_i(X,t) = k_t^i \cdot \|C_i - X\|_2, \tag{6}$$

where $\|C_i - X\|_2$ is the Euclidean distance between a 3D point X and the camera center C_i of I_i (i.e. its scene depth along the viewing ray). In other words, $u_i(X,t)$ assigns a spatial uncertainty to a 3D point X, with respect to a maximally allowed reprojection error t, in the image I_i.

To avoid the possibility that the allowed spatial uncertainty grows too large for points far away from the camera center, we analyse the configuration of the final 3D hypotheses of all segments in L_i, to obtain a depth range in which this estimation makes sense. We therefore compute the median scene depth D_i over all final 3D hypotheses $\hat{h}_m^i$, by using both segment endpoints, and truncate our uncertainty function at the median. We obtain a modified uncertainty estimator

$$\hat{u}_i(X,t,D_i) = \begin{cases} u_i(X,t) & \text{if } \|C_i - X\|_2 < D_i \\ k_t^i \cdot D_i & \text{otherwise} \end{cases}, \tag{7}$$

which can then be finally used to estimate similarities between clusterable 2D segments.

To compute the pairwise segment affinities, we use two separate uncertainty thresholds t_l (lower bound) and t_u (upper bound), with $t_l < t_u$. Since we always have small inaccuracies throughout the reconstruction procedure (e.g. in the SfM or the line segment detection), we cannot assume we will have perfect 3D hypotheses with zero distance to each other. We therefore do not punish deviations below t_l, and fit a Gaussian model between t_l and the cutoff value t_u. For two potentially matching 2D segments l_m^i and $l_{\bar{m}}^j$, their similarity is computed as

$$W(l_m^i, l_{\bar{m}}^j) = \frac{1}{2}\left(c(\hat{h}_m^i) + c(\hat{h}_{\bar{m}}^j)\right) \cdot A_{3D}(\hat{h}_m^i, \hat{h}_{\bar{m}}^j). \tag{8}$$

The similarity function A_{3D} is defined in a similar way as for the 2D case (Eq. 3):

$$A_{3D}(\hat{h}_m^i, \hat{h}_{\bar{m}}^j) = S_{3D}^a(\hat{h}_m^i, \hat{h}_{\bar{m}}^j) \cdot \min\left\{S_{3D}^p(\hat{h}_m^i, \hat{h}_{\bar{m}}^j), S_{3D}^p(\hat{h}_{\bar{m}}^j, \hat{h}_m^i)\right\}, \tag{9}$$

where the angular similarity S^a_{3D} is equivalent to its 2D counterpart S^a_{2D} (Eq. 4), and the position similarity S^p_{3D} is defined as

$$S^p_{3D}(\hat{h}^i_m, \hat{h}^j_{\bar{m}}) = \min\left\{E(\hat{P}^i_m, \hat{h}^j_{\bar{m}}), E(\hat{Q}^i_m, \hat{h}^j_{\bar{m}})\right\}, \tag{10}$$

with the point-to-line affinity E being computed as

$$E(X,h) = \begin{cases} 1 & \text{if } dist(X,h) < \hat{u}_i(X,t_l,D_i) \\ \exp\left(-\frac{(dist(X,h)-\hat{u}_i(X,t_l,D_i))^2}{2\sigma^2_{i,X}}\right) & \text{otherwise} \end{cases}, \tag{11}$$

where $dist(X,h)$ is the Euclidean distance between a point X and a line h. The distance regularisation parameter $\sigma_{i,X}$ is derived from t_l and t_u, such that the affinity E drops to 0.01 if the maximum allowed distance $\hat{u}_i(X,t_u,D_i)$ is reached.

The resulting affinity matrix could now be directly fed to an arbitrary graph clustering algorithm, which takes a simple pairwise affinity matrix as an input. Related methods [11, 12] used [6] as a clustering algorithm, which delivers visually pleasant results for the general case. To further improve the clustering result, we deploy a more recent clustering strategy [4], which is based on diffusing the given affinity matrix W, by implicitly considering the underlying data manifold. Compared to [6], there is virtually no computational overhead, since the diffusion procedure can be efficiently computed in parallel on the GPU.

The clustering result from [4] is post-processed by removing all clusters which do not contain 2D segments from at least four different images. We estimate the final 3D line for each remaining cluster from the 3D segments of the contained 2D residuals, as first shown in [16]. The line direction can be computed by a Singular Value Decomposition of the scatter matrix containing all endpoints of clustered 3D segment hypotheses, and a point on the line can easily be obtained by computing the center of gravity among all these endpoints. We finally project all individual segments onto the averaged 3D line, and compute a set of 3D line segments on this line, such that each of these segments is fully covered by at least three of the projected hypotheses. Figure 2 visualises the different steps of the reconstruction procedure for the *BUILDING* sequence.

4 Experimental Results

We demonstrate the capabilities of our algorithm on two challenging real-world datasets, and quantitatively compare our results to the state-of-the-art [12] on a publicly available dataset with ground truth. We further set our line-based reconstructions in relation to conventional dense point-clouds, obtained from PMVS [8], to give an idea of the pros and cons of both methods in terms of runtime vs. level of abstraction.

The parameters are kept fixed for all datasets. We set the 2D confidence regularisation parameters to $\sigma_p = 2px$ and $\sigma_a = 5°$, and the uncertainty thresholds to $t_l = 2px$ and $t_u = 6px$. As a line segment detector we use LSD [9], and as

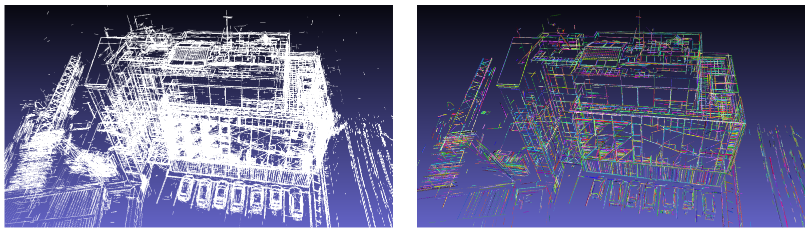


Fig. 2: Visualisation of the reconstruction procedure. (a) Individual 3D hypotheses $\hat{h}_m^i$ for all segments l_m^i. (b) Result of the graph-clustering [4] using random colors (one per cluster).

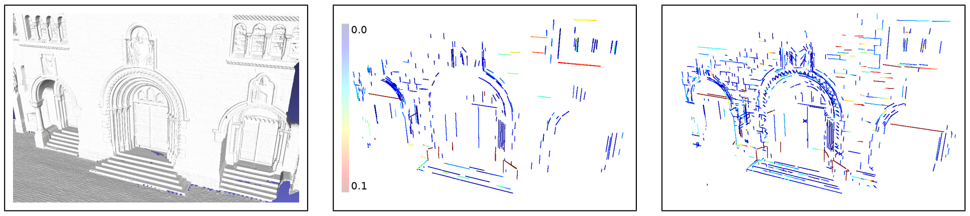
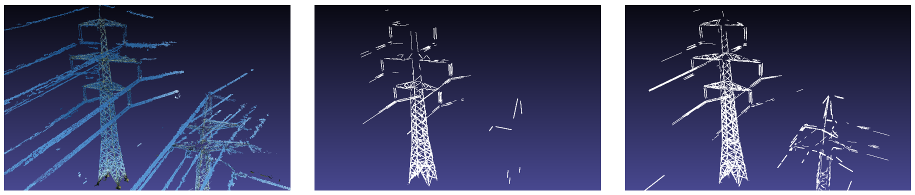

0.0
0.1
Ground truth (laser scan)
Hofer et al. [12] RMSE: 0.0598 6.61 seconds
Line3D (proposed) RMSE: 0.0568 3.64 seconds

Fig. 3: Quantitative evaluation on the *Herz-Jesu-P8* [25] dataset.

an SfM pipeline we use [15]. Our algorithm is implemented in C++ and CUDA, making use of parallel computing whenever possible.

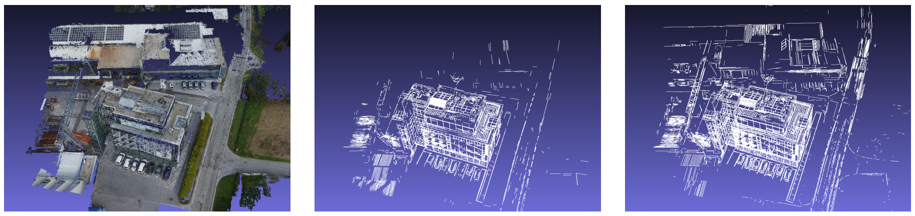
Figure 3 shows a quantitative comparison between our method and the method by Hofer et al. [12] on the *Herz-Jesu-P8* [25] dataset. The lines are colored by their root-mean-square error (RMSE) to the ground truth surface. As we can see, both approaches have a comparably high accuracy while our method manages to reconstruct more 3D segments. Please note that not all valid 3D lines are actually contained in the ground truth. This is especially notable on the railings at the main entrance (colored in dark red).

Figure 4 shows qualitative results for two real-world test sequences. Please note that the runtime for PMVS is measured in hours, while for [12] and *Line3D* it is in seconds. As can be seen, both line-based approaches generate virtually outlier-free results very efficiently, but our method in general manages to reconstruct more 3D segments. This is mainly due to the different uncertainty- and confidence estimation procedures, as well as the modified clustering process, which enable 3D segments that are not visible in many images to be reconstructed more likely. The comparison to the dense point-clouds underlines once more how a lot of 3D information can be extracted in a very short time when only straight line segments are used as features. Our 3D line models give the



$390,762$ points | $1,689$ lines | $2,697$ lines
0.83 hours | 50.67 seconds | 55.85 seconds

PYLON, 66 images, 4320×3240px



$12,156,664$ points | $12,565$ lines | $13,489$ lines
11.34 hours | 368.28 seconds | 375.63 seconds

BUILDING, 344 images, 4912×3264px

Fig. 4: Qualitative reconstruction results. Left column: PMVS [8], Middle column: Hofer et al. [12], Right column: *Line3D* (proposed method).

viewer a very good impression of what is going on in the scene, but in a compact way and requires a very short amount of computational time.

5 Conclusion

We proposed a new method to generate abstract 3D models for built environments. We have shown how a significant amount of 3D information about a scene can be encoded very efficiently, by using line segments in contrast to a large point-cloud. However, our goal was not to replace dense 3D reconstruction, but rather to provide an alternative for all scenarios in which 3D edge information is preferred over a point-cloud.

At the moment, our method can be seen as an SfM post-processing tool, which takes camera poses and images as an input, and returns a 3D model. In our future work, we intend to use the obtained 3D line segments (and their 2D residuals) to refine the camera poses from the SfM. We believe that using a combination of points and lines has the potential to improve SfM for indoor- and urban environments, where distinctive feature-points are rare.

Acknowledgements: This work has been supported by the Austrian Research Promotion Agency (FFG) project FreeLine (Bridge1/843450) and OMICRON electronics GmbH.

References

1. Akinlar, C., Topal, C.: EDLines: Real-Time Line Segment Detection by Edge Drawing (2011), International Conference on Image Processing (ICIP)
2. Ayache, N., Faverjon, B.: Efficient Registration of Stereo Images by Matching Graph Descriptions of Edge Segments (1987), International Journal of Computer Vision (IJCV)
3. Bay, H., Ferrari, V., van Gool, L.: Wide-Baseline Stereo Matching with Line Segments (2005), International Conference on Computer Vision and Pattern Recognition (CVPR)
4. Donoser, M.: Replicator Graph Clustering (2013), British Machine Vision Conference (BMVC)
5. Elqursh, A., Elgammal, A.: Line-Based Relative Pose Estimation (2011), International Conference on Computer Vision and Pattern Recognition (CVPR)
6. Felzenszwalb, P., Huttenlocher, F.: Efficient Graph-Based Image Segmentation (2004), International Journal of Computer Vision (IJCV)
7. Frahm, J.M., Fite-Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y.H., Dunn, E., Clipp, B., Lazebnik, S., Pollefeys, M.: Building Rome on a Cloudless Day (2010), European Conference on Computer Vision (ECCV)
8. Furukawa, Y., Ponce, J.: Towards Internet-Scale Multi-View Stereo (2010), International Conference on Computer Vision and Pattern Recognition (CVPR)
9. von Gioi, R., Jakubowicz, J., Morel, J.M., Randall, G.: LSD: A Fast Fine Segment Detector With a False Detection Control (2010), Transactions on Pattern Analysis and Machine Intelligence (PAMI)
10. Havlena, M., Schindler, K.: VocMatch: Efficient Multiview Correspondence for Structure from Motion (2014), European Conference on Computer Vision (ECCV)
11. Hofer, M., Donoser, M., Bischof, H.: Semi-Global 3D Line Modeling For Incremental Structure-from-Motion (2014), British Machine Vision Conference (BMVC)
12. Hofer, M., Maurer, M., Bischof, H.: Improving Sparse 3D Models for Man-Made Environments Using Line-Based 3D Reconstruction (2014), International Conference on 3D Vision (3DV)
13. Hofer, M., Wendel, A., Bischof, H.: Incremental Line-based 3D Reconstruction using Geometric Constraints (2013), British Machine Vision Conference (BMVC)
14. Hofer, M., Wendel, A., Bischof, H.: Line-based 3D Reconstruction of Wiry Objects (2013), Computer Vision Winter Workshop (CVWW)
15. Irschara, A., Zach, C., Bischof, H.: Towards Wiki-Based Dense City Modeling (2007), International Conference on Computer Vision (ICCV)
16. Jain, A., Kurz, C., Thormaehlen, T., Seidel, H.: Exploiting Global Connectivity Constraints for Reconstruction of 3D Line Segments from Images (2010), International Conference on Computer Vision and Pattern Recognition (CVPR)
17. Labatut, P., Pons, J., Keriven, R.: Efficient Multi-View Reconstruction of Large-Scale Scenes using Interest Points, Delaunay Triangulation and Graph Cuts (2007), International Conference on Computer Vision (ICCV)
18. Lowe, D.: Distinctive Image Features from Scale-Invariant Keypoints (2004), International Journal of Computer Vision (IJCV)
19. Micusik, B., Wildenauer, H.: Structure from Motion with Line Segments Under Relaxed Endpoint Constraints (2014), International Conference on 3D Vision (3DV)
20. Raposo, C., Antunes, M., Barreto, J.: Piecewise-Planar StereoScan: Structure and Motion from Plane Primitives (2014), European Conference on Computer Vision (ECCV)

21. Rothermel, M., Wenzel, K., Fritsch, D., Haala, N.: SURE: Photogrammetric Surface Reconstruction from Imagery (2012), LCD Workshop
22. Rumpler, M., Daftry, S., Tscharf, A., Prettenthaler, R., Hoppe, C., Mayer, G., Bischof, H.: Automated End-to-End Workflow for Precise and Geo-accurate Reconstructions using Fiducial Markers (2014), Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)
23. Schindler, G., Krishnamurthy, P., Dellaert, F.: Line-Based Structure from Motion for Urban Environments (2006), International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT)
24. Snavely, N., Seitz, S., Szeliski, R.: Photo Tourism: Exploring image collections in 3D (2006), ACM Transactions on Graphics (SIGGRAPH)
25. Strecha, C., von Hansen, W., Van Gool, L., Fua, P., Thoennessen, U.: On Benchmarking Camera Calibration and Multi-View Stereo for High Resolution Imagery (2008), International Conference on Computer Vision and Pattern Recognition (CVPR)
26. Wu, C.: Towards linear-time Incremental Structure-from-Motion (2013), International Conference on 3D Vision (3DV)
27. Zhang, L., Koch, R.: Line Matching using Appearance Similarities and Geometric Constraints (2012), Lecture Notes in Computer Science: Pattern Recognition
28. Zhang, L., Xu, C., Lee, K.M., Koch, R.: Robust and Efficient Pose Estimation from Line Correspondences (2012), Asian Conference on Computer Vision
29. Zhang, Y., Yang, H., Liu, X.: A Line Matching Method based on Local and Global Appearance (2011), International Congress on Image and Signal Processing (ICISP)
30. Zhiheng, W., Fuchao, W., Zhanyi, H.: MSLD: A Robust Descriptor for Line Matching (2009), Pattern Recognition