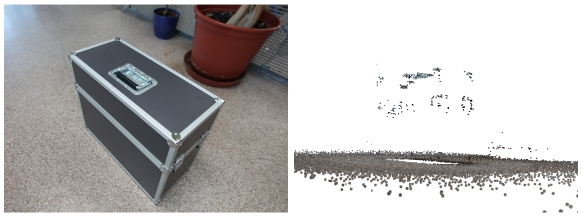
Improving Sparse 3D Models for Man-Made Environments Using Line-Based 3D Reconstruction

Manuel Hofer, Michael Maurer, Horst Bischof
Institute for Computer Graphics and Vision
Graz University of Technology, Austria
{hofer,maurer,bischof}@icg.tugraz.at

Abstract

Traditional Structure-from-Motion (SfM) approaches work well for richly textured scenes with a high number of distinctive feature points. Since man-made environments often contain textureless objects, the resulting point cloud suffers from a low density in corresponding scene parts. The missing 3D information heavily affects all kinds of subsequent post-processing tasks (e.g. meshing), and significantly decreases the visual appearance of the resulting 3D model. We propose a novel 3D reconstruction approach, which uses the output of conventional SfM pipelines to generate additional complementary 3D information, by exploiting line segments. We use appearance-less epipolar guided line matching to create a potentially large set of 3D line hypotheses, which are then verified using a global graph clustering procedure. We show that our proposed method outperforms the current state-of-the-art in terms of runtime and accuracy, as well as visual appearance of the resulting reconstructions.


(a) (b)

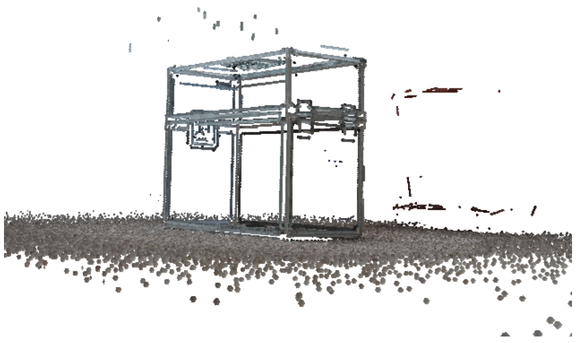

(c)

Figure 1. (a) Example image of the *BOX* sequence. (b) A traditional sparse 3D reconstruction [14] based on SIFT [18] features. (c) Additional reconstructed 3D line segments as complementary image features using our proposed method. As can be seen, the metal box is hardly represented in the sparse point cloud, while it is clearly recognizable when using 3D line segments.

1. Introduction

Extracting 3D information from a set of 2D images has become a very popular and widely studied field over the last few years [1, 9, 19, 20, 14, 21, 23]. Most of these Structure-from-Motion (SfM) approaches are based on detecting and matching distinctive keypoints over several images, by using invariant descriptors such as SIFT [18] or SURF [3]. Therefore, we can only expect to obtain a high amount of correct 3D information for scene parts which correspond to highly textured and non-repetitive regions in the underlying images. This is especially crucial for man-made environments, which often contain weakly textured- (e.g. facades) or even wiry objects (e.g. power pylons or fences). Such structures contain a very low amount of keypoints, which additionally suffer from a low discriminability. Hence, these objects are often poorly represented in the resulting 3D point cloud.

Since man-made objects often consist of linear structures, line segments can be exploited to reconstruct their boundaries and make them reappear in the 3D model. Traditionally, the task of matching 2D line-segments across images requires some appearance or color information (e.g. by using histogram-based [2] or patch-based [25, 24, 16, 4, 5, 10] line descriptors), which would not be of much help for very challenging scenarios, such as the occurrence of wiry structures or see-through objects. Not only do line-segments located on such structures have a very high self-similarity, but also the usage of appearance information is usually not possible since the local surroundings are highly viewpoint dependent.

In this paper we propose a novel approach to line-based 3D reconstruction, based on the output of any conventional SfM pipeline. To be robust against occlusions, imprecise line-segment detections, or unreliable appearance information, we solely use weak epipolar matching constraints to generate a large set of potentially matching 2D line segments among neighboring images, and compute a 3D hypothesis for each of these matches as the intersection of their viewing planes. To identify correct matches, we create a sparse affinity matrix among all 2D segments, and estimate their potential of belonging to the same physical 3D, line by analyzing the coincidences among their 3D hypotheses. To detect corresponding segments robustly, we apply an efficient graph-based clustering method. The final set of 3D line segments is then computed from these correspondences. Figure 1 shows an example SfM result for a metal box, with and without additional 3D information generated by our proposed method. As can be seen, the box is only recognizable when using complementary 3D line features, which significantly improves the overall appearance of the reconstruction.

2. Related Work

While a lot of line matching methods have been presented over the years (e.g. [25, 24, 5]), actual line-based 3D reconstruction approaches are surprisingly rare. This may be due to the fact that relative pose estimation using line-segments is only possible for matched segments in very special spatial configurations (e.g. [7]), or because the vast majority of line matching approaches are not able to handle the real challenging cases, such as wiry objects. Nevertheless, a few methods which are capable of handling such objects as well have been presented recently [15, 13, 12, 11].

What all these approaches have in common is that they rely on given camera poses. This is necessary to omit explicit line matching, since without any kind of pose- or reliable appearance information a 3D reconstruction would be impossible. While this seems to be a drawback, accurate camera pose estimation alone is usually much easier than generating dense 3D models, and also works well around complex- and weakly textured objects. In [15] they try to estimate the correct depth of all 2D line segments individually, by computing a score for each possible 3D location within a certain sweeping range. This is done by backprojecting all 3D hypotheses into neighboring images, and computing a gradient score (since physical 3D lines appear as strong gradients in images). After computing the most plausible 3D location for each 2D segment, they apply a direct spatial clustering procedure with a fixed radius, to fuse segments which belong to the same physical line and discard outliers. While their approach produces visually very pleasant results, it is comparably slow due to the huge amount of potential locations that have to be verified. To overcome this limitation, Hofer et al. [13] used weak epipolar matching constraints among neighboring images, to limit the possible 3D locations for each line segment to a much smaller and discrete set. They adopted the idea of gradient scoring and spatial clustering, and showed significant runtime improvements without sacrificing the accuracy. The same group of authors also presented two incremental approaches [12, 11], which are able to generate 3D line models in conjunction with any online SfM system directly on-the-fly. They additionally got rid of the time consuming gradient scoring step, and demonstrated how 3D line hypotheses can be successfully verified using scale-invariant spatial clustering on its own.

While all of these approaches deliver reasonable results, the offline algorithms have an impractically high runtime [15, 13], while the online approaches often struggle to obtain complete and accurate reconstructions (due to their incremental and greedy nature). Additionally, they either rely on appearance information (to achieve a reasonable performance) [11], or suffer from gross outliers when using a larger spatial grouping radius [12] (which is often necessary to obtain complete reconstructions with a reasonable number of images). To overcome these drawbacks, we build up on the concept of weak epipolar-guided matching, to generate a large set of potentially matching 2D line segments. For each 2D segment we compute a set of 3D hypotheses corresponding to these matches, and compute a score for each of them using the spatial hypotheses distribution rather than time-consuming gradient scoring. This is based on the idea that if we find a correct match for a 2D segment in several neighboring images, the resulting 3D hypotheses will be close in space, while outlier matches are at an arbitrary position. Hence, correct hypotheses are in general closer to each other than incorrect ones. We use this information to select the most plausible 3D hypothesis for each 2D segment, which we then group together using a global graph-clustering algorithm.

In Section 3 we will explain our pipeline in more detail, before we conclude with extensive experiments on synthetic- as well as real-world datasets in Section 4.

3. Line-based 3D Reconstruction

Given an unordered set of images $I = \{I_1, \ldots, I_N\}$ and their corresponding camera poses $C = \{C_1, \ldots, C_N\}$, we want to reconstruct an accurate 3D line model. Therefore, we first need to detect a set of 2D line segments $L_i = \{l_1^i, \ldots, l_{n_i}^i\}$ for each image I_i, where each line segment $l_n^i = \{p_n^i, q_n^i\}$ consists of two 2D points p and q in the image space. For this purpose we use the popular Line Segment Detector (LSD) [22] algorithm, which offers accurate detections with a very low number of false positives. We furthermore define $\hat{l}_n^i$ to be the infinite line passing through l_n^i. Since it only makes sense to match images with an overlapping field of view, we define a visual neighbor set $\zeta_i^m \subset \{1, \cdots, N\} \backslash \{i\}$ for each image I_i, which holds the indices of its m nearest neighbors (images) with respect to the number of common worldpoints (obtained by the sparse SfM beforehand).

The computing steps for our approach are as follows. In Section 3.1 we show how to compute a large set of potential pairwise 2D line segment matches among neighboring images. In Section 3.2 we explain how we create 3D hypotheses for each of these matches, and in Section 3.3 we show how to select the most suitable hypothesis for each 2D segment. Finally, in Section 3.4 we demonstrate how to compute pairwise affinities among potentially matching 2D segments using their selected hypotheses in a scale invariant way, and how to cluster corresponding segments together using graph clustering.

3.1. 2D Line Segment Matching

The first step in our pipeline is to find all potentially matching 2D line segments among all neighboring images. Since we do not want to use appearance information (to be able to handle wiry structures as well as see-through objects), we exploit weak epipolar constraints, similar to [12]. These constraints roughly state that for two line segments $l_n^i \in L_i$ and $l_m^j \in L_j$ (where $j \in \zeta_i^m$) at least one of the endpoints of l_m^j must be relatively close to one of the epipolar lines defined by the endpoints of l_n^i (*endpoint similarity*), and the opposite endpoint of l_m^j must be located in the same direction as the other epipolar line (*orientation*). To avoid any kinds of heuristics regarding the definition of how close a segment endpoint has to be to its nearest epipolar line, we change the matching constraints such that we only require a slightly modified orientation constraint. Therefore, we first compute the intersection points x_1 and x_2 with $\hat{l}_m^j$ and the epipolar lines of l_n^i. We then assign the endpoint-intersection pair (p, x) $\big(p \in l_m^j$ and $x \in \{x_1, x_2\}\big)$ with minimal Euclidian distance to each other, as well as the remaining two points (q, y) $\big(q \in l_m^j, q \neq p$ and $y \in \{x_1, x_2\}, y \neq x\big)$. We define a binary sparse matching matrix Π (to store the potential correspondences for each image pair) as

$$\Pi_{i,j}(n,m) = \begin{cases} 1 & \text{if } \langle \overrightarrow{pq}, \overrightarrow{xy} \rangle > 0 \\ 0 & \text{else} \end{cases}, \tag{1}$$

where $\langle \cdot, \cdot \rangle$ is the dot product. This simple constraint discards matches with an improper orientation with respect to the epipolar geometry, while simultaneously ensuring maximum robustness regarding imprecisely detected or occluded segment endpoints.

To limit the number of outlier matches as much as possible we also perform the reverse matching procedure, and only keep potential matches for which $\Pi_{i,j}(n,m) = \Pi_{j,i}(m,n) = 1$. Since this procedure is purely local and only affects two images at a time, it can be heavily parallelized for maximum efficiency. Once we have obtained all pairwise matches we can proceed to the task of 3D hypotheses computation.

3.2. 3D Hypotheses Estimation

Having a potentially huge set of correspondences, we want to rank them according to a score which directly reflects the probability of correctness. Since appearance cannot be used reliably in many cases (as stated above), we want to model the quality of a 2D matching hypothesis based on the spatial configuration of its corresponding 3D line (which can be easily computed with given camera poses). The main idea is that all 3D hypotheses defined by correct matches for a segment l_n^i have to be close in space, since (ideally) the intersection of their viewing planes (defined by the rays through the segment endpoints and the camera center) has to be the actual physical 3D line, which does not hold for outlier matches. This is of course not exactly true for realistic scenarios, since there are always small uncertainties in the pose estimation- as well as the line segment detection steps. Nevertheless, we can exploit this property to define which hypotheses do make sense, and which do not.

First, we have to compute a 3D hypothesis $h_{n,m}^{i,j} = \big\{P_{n,m}^{i,j}, Q_{n,m}^{i,j}\big\}$ for each potential match l_n^i and l_m^j $(\Pi_{i,j}(n,m) = 1)$, which is the 3D line segment located on the intersection line between the viewing planes of l_n^i and l_m^j, whose endpoints coincide with the 2D segment l_n^i. While we could use ordinary triangulation to compute this 3D segment [13, 12, 11] (which requires a singular value decomposition), we do this in a more efficient way by simply intersecting the viewing planes for both 2D segments in 3D. To achieve this, we compute the normal vectors $v_{i,n}$ and $v_{j,m}$ of the viewing planes through l_n^i and l_m^j, as well as normal vectors $\tilde{p}_{i,n}$ and $\tilde{q}_{j,m}$ for the planes passing through p_n^i and q_n^i perpendicular to l_n^i (all normal vectors are normalized to unit length). These vectors can be easily computed as

$$v_{k,\cdot} = \left(R_k^T K_k^{-1} p\right) \times \left(R_k^T K_k^{-1} q\right), \tag{2}$$

where R_k and K_k are the rotation matrix and intrinsics of camera C_k, and p and q $(p \neq q)$ are two arbitrary 2D projections of the desired 3D plane (in homogeneous coordinates).

Finally, the 3D segment endpoints $P_{n,m}^{i,j}$ and $Q_{n,m}^{i,j}$ can be computed as

$$P_{n,m}^{i,j} = \frac{-d_1(v_{j,m} \times \tilde{p}_{i,n}) - d_2(\tilde{p}_{i,n} \times v_{i,n}) - d_3^p(v_{i,n} \times v_{j,m})}{\langle v_{i,n}, (v_{j,m} \times \tilde{p}_{i,n}) \rangle} \tag{3}$$

$$Q^{i,j}_{n,m} = \frac{-d_1(v_{j,m} \times \tilde{q}_{i,n}) - d_2(\tilde{q}_{i,n} \times v_{i,n}) - d^q_3(v_{i,n} \times v_{j,m})}{\langle v_{i,n}, (v_{j,m} \times \tilde{q}_{i,n})\rangle} \tag{4}$$

where $d_1 = -\langle v_{i,n}, Z_i\rangle$, $d_2 = -\langle v_{j,m}, Z_j\rangle$, $d^p_3 = -\langle \tilde{p}_{i,n}, Z_i\rangle$, $d^q_3 = -\langle \tilde{q}_{i,n}, Z_i\rangle$, and Z_k is the camera center of camera C_k.

With this simple procedure we can directly compute the 3D hypotheses during the matching procedure on the GPU for maximum efficiency. We have evaluated the accuracy of this approach compared to ordinary triangulation and found the differences to be in the range of 10^{-6}. To obtain the inverse hypothesis $h^{j,i}_{m,n}$ (with respect to l^j_m) we perform the same procedure as above with exchanged indices.

Once we have computed the set of all possible hypotheses, we can rank them according to their score, which we define in the following section.

3.3. Hypothesis Selection & Outlier Removal

To estimate which 3D hypotheses are more likely to be correct than others, we define a score based on the spatial proximity of hypotheses which share a 2D segment. Similar to [12, 11] we do not want to count the number of spatially close hypotheses, but rather the number of different cameras which we could connect within a certain grouping radius r.

The question is of course how to define such a radius in a scale-invariant way. We adapt the idea of deriving a meaningful value for r from a user defined maximum uncertainty σ in the image space from [12, 11], but additionally incorporate depth information to compensate for larger uncertainties farther away from the camera. We therefore compute a characteristic radius $r^{i,j}_{n,m}$ for each hypothesis $h^{i,j}_{n,m}$ by computing the average distance of its endpoints to a plane through an orthogonally shifted version of l^i_n by a distance σ. We further define $d_{P^{i,j}_{n,m}}$ and $d_{Q^{i,j}_{n,m}}$ to be the distances of the endpoints of $h^{i,j}_{n,m}$ to the camera center Z_i. Afterwards, we set a view specific characteristic grouping radius r_i to be the median over all values $r^{i,\cdot}_{n,\cdot}$, and d_i is the average endpoint-to-camera distance of the selected element (corresponding to the median).

We now define a binary decision function as

$$\varphi\left(h^{i,j}_{n,m}, C_k\right) = \begin{cases} 1 & \text{if} \quad \exists h^{i,k}_{n,\cdot} : \begin{array}{c} \overline{P^{i,j}_{n,m} P^{i,k}_{n,\cdot}} < w_{P^{i,j}_{n,m}} \cdot r_i \wedge \\ \overline{Q^{i,j}_{n,m} Q^{i,k}_{n,\cdot}} < w_{Q^{i,j}_{n,m}} \cdot r_i \end{array} \\ 0 & \text{else} \end{cases} \tag{5}$$

with

$$w_{X^{i,j}_{n,m}} = \begin{cases} \frac{d_{X^{i,j}_{n,m}}}{d_i} & \text{if } d_{X^{i,j}_{n,m}} < 2d_i \\ 2 & \text{else} \end{cases}, \quad X \in \{P, Q\} \tag{6}$$

being a linear scale factor to allow higher uncertainties in the distance, and smaller uncertainties closer to the camera.

We now compute the local score for each hypothesis as

$$\psi\left(h^{i,j}_{n,m}\right) = \sum_{y \in \zeta^m_i} \varphi\left(h^{i,j}_{n,m}, C_y\right), \tag{7}$$

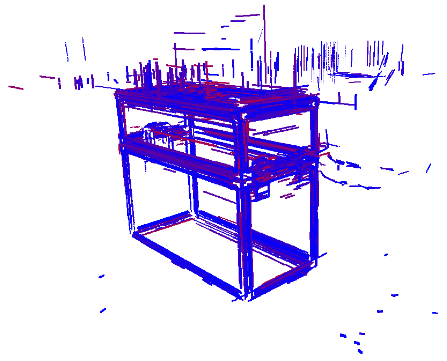


Figure 2. This figure illustrates the set of most plausible hypotheses $h^*_{i,n}$ for each 2D line segment l^i_n from the *BOX* sequence (Figure 1). The colors indicate the hypotheses scores $\hat{\psi}$ in the range of 1 (blue) to 0 (red). As we can see, most of the line segments vote for their correct position in 3D, with a low number of gross outliers.

which coincides with the number of cameras agreeing on $h^{i,j}_{n,m}$, with respect to the allowed spatial uncertainties. We additionally discard hypotheses for which the score is below a visibility threshold α. Afterwards we normalize the scores to $[0, 1]$ by dividing $\psi(h^{i,j}_{n,m})$ by the locally highest score

$$\psi^*_{i,n} = \max_{j,m}\left(\psi\left(h^{i,j}_{n,m}\right)\right). \tag{8}$$

To be more robust against outlier hypotheses, we assign the same final score $\hat{\psi}$ for pairwise hypotheses defined as

$$\hat{\psi}\left(h^{i,j}_{n,m}\right) = \min\left(\psi\left(h^{i,j}_{n,m}\right), \psi\left(h^{j,i}_{m,n}\right)\right), \tag{9}$$

to prevent that hypotheses gain importance which are only plausible for one of their 2D residuals.

Finally, we select the most plausible hypothesis $h^*_{i,n}$ for each 2D segment l^i_n to be the hypothesis $h^{i,\cdot}_{n,\cdot}$ with the maximum score. Figure 2 shows all selected hypotheses for the *BOX* sequence. As we can see, most of the line segments already vote for their correct 3D position with a very low number of outliers. To reconstruct a 3D line model using this information, we can now proceed to the task of line segment clustering.

3.4. Graph Clustering

After the hypothesis selection step we end up with one 3D hypothesis for each 2D segment. To actually cluster corresponding 2D segments together we do not want to apply a direct greedy clustering algorithm as in previous approaches [15, 13, 12], but rather want to employ a global clustering algorithm which takes into account the whole hypotheses constellation (similar to [11], but on the 2D segment and not the 3D hypotheses level). This prevents the clustering of hypotheses which are just close in space coincidentally, and suppresses the occurrence of gross outliers in the final model even for larger values for σ. We therefore define a global sparse affinity matrix A among all 2D segments. The majority of the matrix entries are zero (since only potentially matching 2D segments can have non-zero affinities). The affinity for matching segments l_n^i and l_m^j (with $\Pi_{i,j}(n,m) = 1$) is defined as

$$A_{\eta(i,n),\eta(j,m)} = \begin{cases} \hat{\psi}\left(h_{i,n}^*\right) \cdot e^{-\frac{\lambda \mu_{n,m}^{i,j}}{r_{i,j}}} & \text{if } \mu_{n,m}^{i,j} < r_{i,j} \\ 0 & \text{else} \end{cases} \tag{10}$$

where $\eta(\cdot,\cdot)$ assigns consecutive and unique row indices to 2D line segments, $r_{i,j} = 0.5(r_i + r_j)$, and $\mu_{n,m}^{i,j}$ is the maximum distance between the endpoints of $h_{i,n}^*$ to the infinite line $\hat{h}_{j,m}^*$ and vice versa. The scaling factor λ is by default chosen such that the affinity weight $e^{-\frac{\lambda \mu_{n,m}^{i,j}}{r_{i,j}}}$ is still 0.5 if the maximally allowed distance $r_{i,j}$ between the 3D hypotheses is reached.

Given this affinity matrix, we exploit the efficient graph clustering approach by Felzenswalb and Huttenlocher [8], which works in a completely unsupervised way and does not require a priori knowledge about the expected number of clusters. As an input it only requires a region preference parameter. Since this parameter roughly corresponds to the desired minimal cluster size, we set it to α (which is the minimum number of cameras which have to support a valid cluster).

Given the clustering result, we compute the final set of 3D line segments by merging the most plausible hypotheses for all clustered 2D segments. Similar to [15] we find the direction of the underlying 3D line using principal component analysis of the scatter matrix containing all endpoints of the 3D hypotheses. A point on the line is computed as the center of gravity among these endpoints. We finally project all hypotheses endpoints onto the newly defined 3D line and consider all sub regions on this line to be valid 3D line segments, if they are visible in at least α cameras (based on the projected 3D hypotheses).

Using these steps we are now able to generate accurate and complete 3D line models from arbitrary sparse 3D reconstructions (as seen in Figure 1). In the following section we will evaluate our approach on several challenging datasets, and compare with several existing methods.

4. Experimental Results

To demonstrate the capabilities of our proposed method, we compare ourselves to state-of-the-art methods [15, 13, 12], by performing several qualitative and quantitative experiments. To preprocess our datasets (i.e. perform pose estimation) we use an offline SfM pipeline [14], which uses SIFT features [18]. Since [12] is originally an incremental approach, we feed the already computed camera poses sequentially into the algorithm for a fair comparison. Our test system is a standard desktop PC equipped with a *GeForce GTX780Ti* graphics card. Since most of the processing steps in our algorithm are parallelizable (except for the final clustering step), we exploit the compute capabilities of *CUDA* for maximum efficiency.

All parameters are set to default values and remain unchanged for all experiments ($m = 10$, $\sigma = 10$, and $\alpha = 4$). Since the LSD algorithm (line segment detection) suffers from large image sizes, we scale down all images to approximate FullHD resolution (as suggested in [12]). This does not affect the accuracy and quantity of the detections in a negative way, but decreases the runtime significantly. We operate on the smaller images for all competitive algorithms.

4.1. Quantitative Evaluation

To evaluate the accuracy of our proposed method compared to state-of-the-art methods [15, 13, 12], we use the synthetic *Timberframe* [1] dataset. As an error metric we compute the Hausdorff distance between densely sampled points along the 3D line segments, and the groundtruth CAD model. Figure 3 shows the reconstruction results for all competing methods with a visualization of the root mean squared error (RMSE).

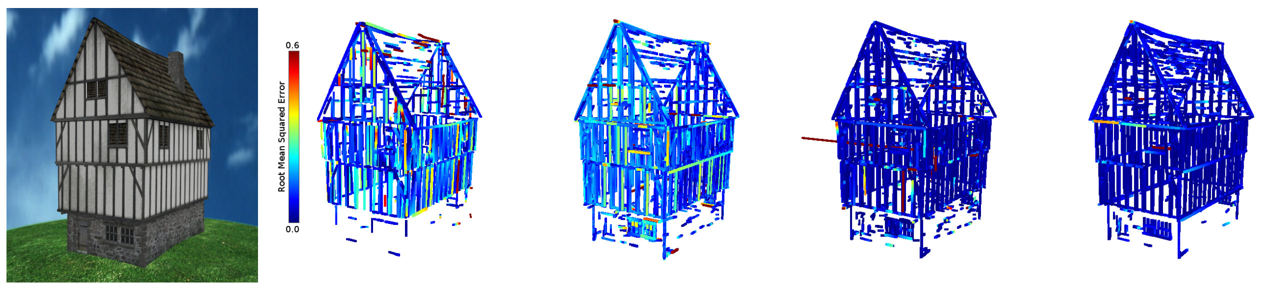
As can be seen, our approach achieves the highest accuracy among the competing algorithms ($RMSE = 0.046$, vs. second best [13] $RMSE = 0.064$). Additionally, with an average runtime of $\varnothing t = 0.21$ seconds per image, our approach is more than three times as fast as [12] ($\varnothing t = 0.74$ seconds). This is mainly due to the exploitation of massive parallelism using the GPU, and the usage of efficient graph clustering [8]. Please note that the runtimes include all necessary steps, including data I/O and line segment detection.

4.2. Qualitative Evaluation

To further demonstrate the effectiveness of our algorithm, we performed additional qualitative evaluations against [13] and [12]. For this purpose we have chosen several challenging test sequences, including wiry structures. Figure 4 shows the reconstruction results, the relevant numbers (e.g. runtime) can be found in Table 1.

As we can see, we significantly outperform the state-of-the-art in terms of runtime throughout all test sequences (up to ten times faster as [12]). Additionally, our approach generates much cleaner and more accurate results. While [13] frequently manages to generate complete reconstructions, it often suffers from severe outliers (e.g. in the *BOX* sequence). In contrast, [12] does not generate any gross outliers in our experiments, but often misses relevant parts of the underlying scene. This is due to the very small spatial grouping radius (derived from a maximum reprojection error of only 1 pixel), which cannot handle larger triangulation- and pose uncertainties. We did not increase

[1] http://www.mpi-inf.mpg.de/resources/LineReconstruction



(a) Example (b) [15] (c) [13] (d) [12] (e) Ours

Figure 3. (a) Example image of the synthetic *Timberframe* sequence (240 images). (b) Original result by [15] (runtime of several hours, $RMSE = 0.189$, $mean = 0.137$). (c) Result by [13] ($\varnothing t = 3.14$, $RMSE = 0.064$, $mean = 0.037$). (d) Result by [12] ($\varnothing t = 0.74$, $RMSE = 0.080$, $mean = 0.044$). (e) Our results ($\boldsymbol{\varnothing t = 0.21}$, $\boldsymbol{RMSE = 0.046}$, $\boldsymbol{mean = 0.029}$). All runtimes $\varnothing t$ are in seconds. The reconstructions have been colored to visualize their root mean squared error (RMSE) with respect to the groundtruth CAD model.

		[13]			[12]			Ours		
Sequence	#img.	#seg.	t [s]	$\varnothing t$ [s]	#seg.	t [s]	$\varnothing t$ [s]	#seg.	t [s]	$\varnothing t$ [s]
PYLON	106	4261	680.91	6.42	1017	213.68	2.02	2950	29.07	**0.27**
BOX	139	1131	294.53	2.12	331	171.83	1.24	484	39.59	**0.28**
FACADE	317	6067	4319.20	13.63	2263	572.05	1.80	5009	64.32	**0.20**
PYLON II	35	1951	303.75	8.68	187	92.63	2.65	889	10.07	**0.29**

Table 1. Evaluation for the used test sequences (Figure 4). *#seg.* stands for the number of generated 3D line segments, t is the total runtime in seconds, and $\varnothing t$ is the average runtime per image.

this value in [12] for our evaluations, since this quickly leads to extremely noisy reconstruction results, with a severe amount of outliers. We do not suffer from this problem since we do not use σ for direct clustering, but rather for estimating probabilities for 3D line hypotheses. Figure 5 illustrates the robustness of our approach against different values for σ for the *FACADE* sequence. As can be seen, our approach can successfully handle smaller- (with only few 3D lines disappearing), as well as significantly larger values (without introducing outliers).

To demonstrate the positive effects of having additional complementary 3D information in the final 3D model, we have computed two separate 3D meshes for the *BOX* sequence, one by using the sparse point-cloud only, and the other one by using the improved 3D model with additional 3D line segments. Figure 6 shows the simple textured meshes for both cases. As we can see, more 3D information is definitely beneficial, and the visual appearance of the 3D mesh is significantly improved.

5. Conclusion

We have proposed a novel line-based 3D reconstruction approach, which operates on sparse SfM results. We have shown that adding complementary 3D information, in the form of 3D line segments, significantly improves the overall visual appearance of the resulting reconstructions. We significantly outperform the current state-of-the-art in terms of runtime, as well as accuracy and completeness of the 3D models. The usage of a robust maximum uncertainty parameter in the image space allows scale invariant 3D reconstruction, with a low probability of outliers. Global graph-clustering and massive parallelism lead to a robust and efficient line-based 3D reconstruction method, which can handle solid- as well as complex wiry objects.

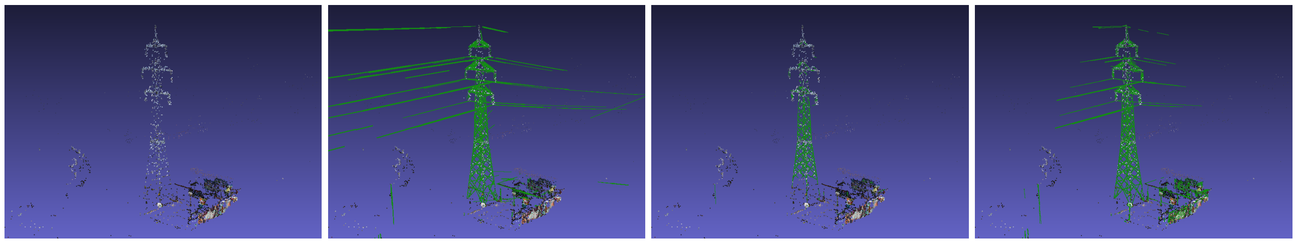
Since we do not only encounter linear edges in man-made environments, we want to investigate the incorporation of more complex object boundaries into our pipeline in the near future. Furthermore, we want to evaluate the possibility of integrating our method into a real-time SLAM [6] or PTAM [17] system. We strongly believe that incorporating complementary 3D information will lead to significant improvements for all kinds of visual navigation tasks in urban environments. We are positive that our concepts can be adapted for real-time computation, without sacrificing the accuracy and robustness.

Acknowledgements

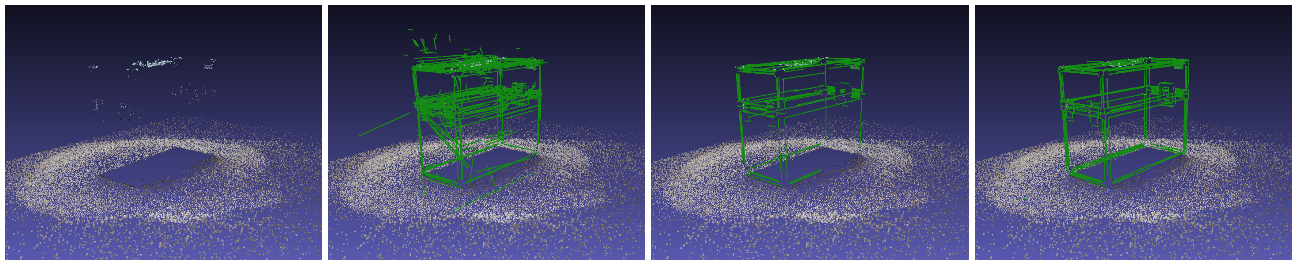
This work has been supported by the Austrian Research Promotion Agency (FFG) project FreeLine (Bridge1/843450) and OMICRON electronics GmbH.

References

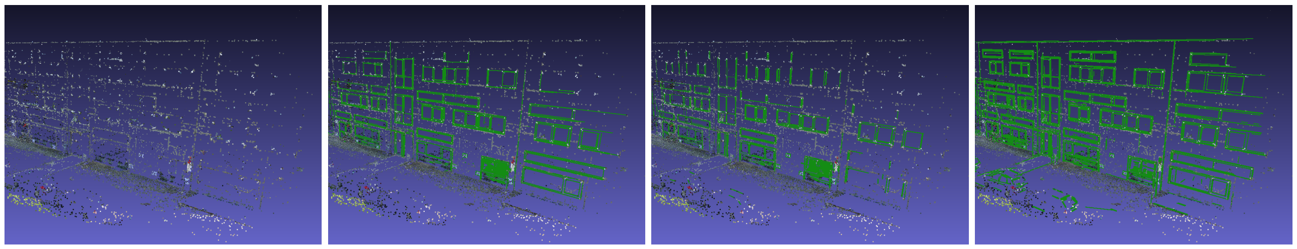
[1] S. Agarwal, N. Snavely, I. Simon, and S. Seitz. Building rome in a day, 2009. International Conference on Computer



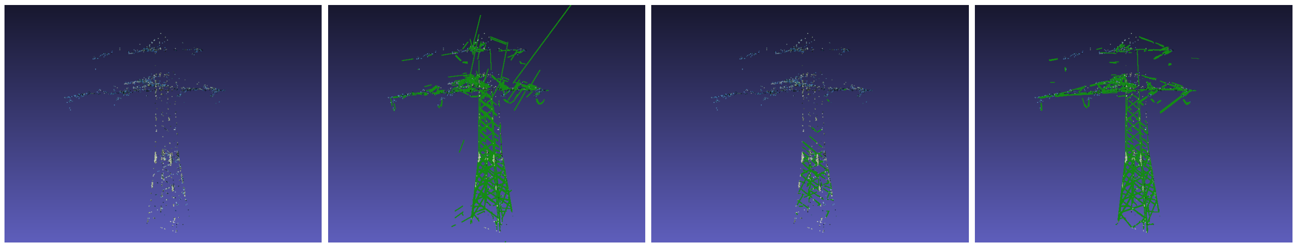
(a) PYLON sequence (106 images)



(b) BOX sequence (139 images)

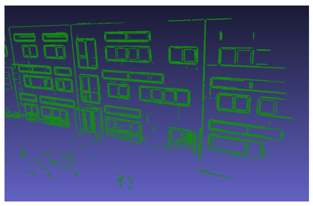


(c) FACADE sequence (317 images)

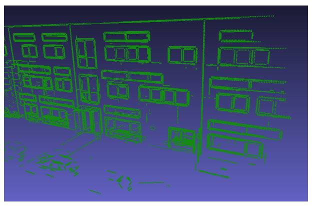


(d) PYLON II sequence (35 images)

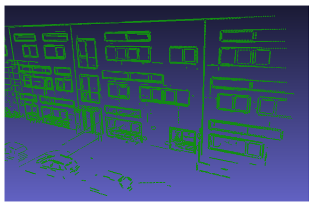
Figure 4. *Left:* Sparse point-cloud using SIFT [18] features, *Middle left:* Results by [13], *Middle right:* Results by [12] , *Right:* Results using our proposed method. As can be seen, our approach produces more accurate 3D reconstructions with a significantly lower computing time (see Table 1).



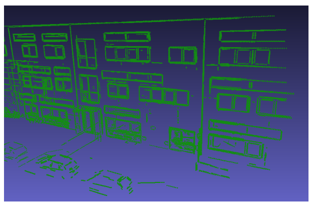
(a) $\sigma = 5$



(b) $\sigma = 10$



(c) $\sigma = 15$

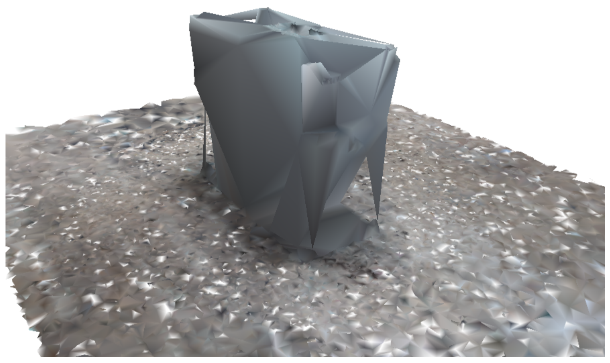


(d) $\sigma = 20$

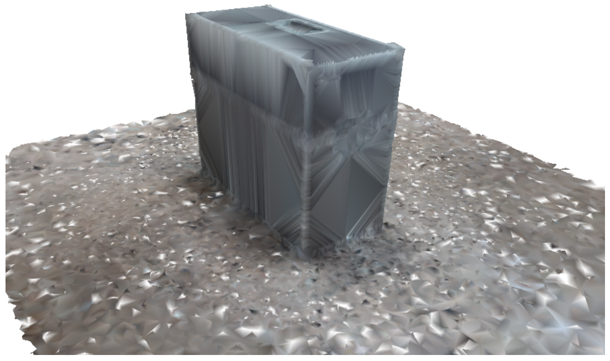
Figure 5. This figure illustrates the robustness of our method with respect to the parameter choice for σ. As we can see, the results do not vary significantly and larger values do not introduce severe outliers.

Vision (ICCV).

[2] H. Bay, V. Ferrari, and L. van Gool. Wide-baseline stereo matching with line segments, 2005. International Conference on Computer Vision and Pattern Recognition (CVPR).



(a) 3D mesh using points only



(b) 3D mesh using points and lines

Figure 6. This figure illustrates how complementary 3D information affects subsequent post-processing tasks, such as meshing. (a) A 3D mesh generated using the sparse point-cloud only. (b) A 3D mesh generated using the original sparse point-cloud, as well as the 3D line segments obtained by our proposed method (densely sampled as points). As can be seen, the additional 3D information along the edges (see Figure 1) significantly improves the meshing result.

[3] H. Bay, T. Tuytelaars, and L. van Gool. SURF: Speeded up robust features, 2006. European Conference on Computer Vision (ECCV).

[4] F. Bin, W. Fuchao, and H. Zhanyi. Line matching leveraged by point correspondences, 2010. International Conference on Computer Vision and Pattern Recognition (CVPR).

[5] F. Bin, W. Fuchao, and H. Zhanyi. Robust line matching through line-point invariants, 2011. Pattern Recognition.

[6] A. Davison. Real-time simultaneous localization and mapping, 2002. International Conference on Computer Vision (ICCV).

[7] A. Elqursh and A. Elgammal. Line-based relative pose estimation, 2011. International Conference on Computer Vision and Pattern Recognition (CVPR).

[8] P. Felzenszwalb and F. Huttenlocher. Efficient graph-based image segmentation, 2004. International Journal of Computer Vision (IJCV).

[9] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day, 2010. European Conference on Computer Vision (ECCV).

[10] K. Hirose and H. Saito. Fast line description for line-based SLAM, 2012. British Machine Vision Conference (BMVC).

[11] M. Hofer, M. Donoser, and H. Bischof. Semi-global 3D line modeling for incremental structure-from-motion, 2014. British Machine Vision Conference (BMVC).

[12] M. Hofer, A. Wendel, and H. Bischof. Incremental line-based 3D reconstruction using geometric constraints, 2013. British Machine Vision Conference (BMVC).

[13] M. Hofer, A. Wendel, and H. Bischof. Line-based 3D reconstruction of wiry objects, 2013. Computer Vision Winter Workshop (CVWW).

[14] A. Irschara, C. Zach, and H. Bischof. Towards wiki-based dense city modeling, 2007. International Conference on Computer Vision (ICCV).

[15] A. Jain, C. Kurz, T. Thormaehlen, and H. Seidel. Exploiting global connectivity constraints for reconstruction of 3D line segments from images, 2010. International Conference on Computer Vision and Pattern Recognition (CVPR).

[16] B. Khaleghi, M. Baklouti, and F. Karray. SILT: Scale-invariant line transform, 2009. Computational Intelligence in Robotics and Automation (CIRA).

[17] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces, 2007. International Symposium on Mixed and Augmented Reality (ISMAR).

[18] D. Lowe. Distinctive image features from scale-invariant keypoints, 2004. International Journal of Computer Vision (IJCV).

[19] M. Pollefeys, L. van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera, 2004. International Journal of Computer Vision (IJCV).

[20] N. Snavely, S. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D, 2006. ACM SIGGRAPH.

[21] N. Snavely, S. Seitz, and R. Szeliski. Modeling the world from internet photo collections, 2008. International Journal of Computer Vision (IJCV).

[22] R. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: A fast line segment detector with a false detection control, 2010. Transactions on Pattern Analysis and Machine Intelligence (PAMI).

[23] C. Wu. Towards linear-time incremental structure from motion, 2013. International Conference on 3D Vision (3DV).

[24] Y. Zhang, H. Yang, and X. Liu. A line matching method based on local and global appearance, 2011. International Congress on Image and Signal Processing (ICISP).

[25] W. Zhiheng, W. Fuchao, and H. Zhanyi. MSLD: A robust descriptor for line matching, 2009. Pattern Recognition.