Multi-planar Monocular Reconstruction of Manhattan Indoor Scenes

Seongdo Kim Roberto Manduchi
University of California, Santa Cruz
1156 High St, Santa Cruz, CA 95064, USA
{seongdo,manduchi}@soe.ucsc.edu

Abstract

We present a novel algorithm for geometry and camera pose reconstruction from image sequences that is specialized for indoor Manhattan scenes. Unlike general-purpose SfM/SLAM, our system represents geometric primitives in terms of canonically oriented planes. The algorithm starts by computing multi-planar segmentation and motion estimation from image pairs using constrained homographies. It then proceeds to recover the relative scale at each frame and to determine chains of match clusters, where each cluster is associated with a plane in the scene. Motion and scene geometry (expressed in terms of planar models) are then optimized using a novel formulation of Bundle Adjustment. Compared with other state-of-the-art SfM/SLAM algorithms, our technique is shown to produce superior and realistic surface reconstruction for a monocular indoor scene.

1. Introduction

The problem of joint reconstruction of camera motion and 3-D scene geometry from images (called Structure from Motion (SfM) or SLAM, depending on the context) has been studied for decades. Impressive results have been obtained, both with vast collections of unordered images [13], and with video sequences taken from a moving camera [10]. Rather than attempting to raise the state of the art in general-purpose SfM or SLAM, this work proposes a new approach for a very specific scenario: indoor scenes characterized by a Manhattan World (MW) geometry. The MW geometry assumption is appropriate for most indoor environments. Scenes with vertical walls not intersecting at right angles can be modeled by weak MW [12], which inherits many of the general properties of the MW geometry. Of course, there are cases in which the MW geometry would be inadequate, such as in the presence of curved surfaces, ramps, or generic objects or people visible in the scene; in these cases, our technique would not be directly applicable.

The MW geometry is inherently simple, which facilitates reconstruction. For example, by estimating the three vanishing points (an operation that is feasible in edge-rich indoor scenes), one obtains the camera orientation with respect to the "canonical" directions (plane normals) [7]. The homography induced on images of the same plane seen by a moving camera has only three degrees of freedom, which facilitates multi-planar segmentation and estimation [12]. The images of multiple parallel and coplanar lines can be characterized by an invariant descriptor ("characteristic line") that enables robust co-planar line clustering [4]. Our work builds on these previous results, and proposes a technique for SfM/SLAM that makes careful use of the intrinsic properties of the MW geometry.

The main characteristic of our system lies in the fact that all surface elements are represented in terms of canonically oriented planes. Although we use feature points, matched across image pairs, to estimate the plane locations and to validate geometric reconstruction, we never maintain a representation of individual points in space. This is a major departure from traditional reconstruction techniques. The advantage of this approach is highlighted by our novel formulation of Bundle Adjustment, which uses planar primitives, jointly optimized with the camera poses by minimization of a specially designed reprojection error. The output of our algorithm is a set of canonically oriented planes, together with the reconstructed camera poses and a sparse set of back-projected feature points. This information can be used for realistic patch-based reconstruction. Our algorithm takes around a second (end-to-end) per image on a GPU-enabled computer. In our experiments, it produced camera trajectories comparable to the state of the art, with superior geometric reconstruction. An expanded version of this article was published in [6].

2. Related Work

SfM and visual SLAM algorithms can be roughly divided in three categories: those that match specific features across images (typically points, e.g. [10], or lines, e.g. [4]); those that use direct image alignment to track the camera pose (e.g. [1]); and those that use a volumetric representation of space (e.g. [11]). Some works explicitly represent the presence of planes [14, 3]. For example, ORB-SLAM [10] includes a "model selection" component that decides whether the scene is planar; in this case, an homography is more effective at describing view changes than the associated epipolar geometry. Pop-up SLAM [17] detects the extent of the visible ground plane (floor) for each image using a convolutional network [16], and uses this information to extract the visible vertical planes.

3. Multi-planar Fitting from Two Views

3.1. Manhattan-constrained Homography Computation and Multi-Planar Clustering

Saurer et al. [12] introduced the concept of constrained homographies for weak Manhattan World scenes (i.e. scenes that contain horizontal planes and vertical planes, where the latter are not necessarily mutually parallel or orthogonal). In the more restrictive Manhattan World case (all visible planes either orthogonal or parallel to one another), the homography $\mathbf{H}$ relating two images of the same plane has three (rather than eight) degrees of freedom. One useful characteristic of Manhattan wold scenes is that the rotation aligning the camera with the *canonical Manhattan orientation* (i.e., with axes mutually parallel to the three planes normals) can be computed from the three vanishing points [7]. This in turns provides a convenient way to estimate $\mathbf{R}$.

When multiple planar structures are visible in the image, a (constrained) homography can be computed for each such structure, assuming that the point matches across the two images have been properly clustered. Kim and Manduchi [5] proposed an extension to the T-linkage algorithm that accounts for the reduced degree of freedom of the homographies to be computed. Minimal sets of two matches are sampled. Each such set determines one plane for each canonical direction (homographies $\mathbf{H}^1, \mathbf{H}^2, \mathbf{H}^3$). Then, for each canonical direction, agglomerative clustering of matches is conducted using T-linkage.

3.2. Translation Vector Regression

Each cluster of point matches $\mathcal{C}_{m,k}^{r(m,k)} = \{(\mathbf{x}_m, \mathbf{x}_{m+1})\}$ (where m is the image pair index, k is the cluster index, and $r(m,k)$ identifies the canonical direction of the k-th planar model) determines its own homography, defined by the scaled translation vector $\mathbf{t}_{m,k}/d_{m,k}$. Let us denote by $\bar{\mathbf{t}}_{m,k}$ the unit-norm normalized translation, and by $\bar{d}_{m,k}$ the scaled distance as defined earlier. Since the actual camera translation is unique, all unit vectors $\bar{\mathbf{t}}_{m,k}$ estimated for the same image pair should be identical. We enforce this constraint, and find a common unit-norm translation vector $\bar{\mathbf{t}}_m$ while simultaneously refining the planes' location, by solving an appropriate constrained minimization problem for each image pair.

4. Multi-frame Integration

4.1. Recovering Relative Scale

The translation vectors computed for each frame pair are defined up to an unknown scale. To recover the relative scale of each translation vector, we define a metric normalized by the distance between the camera locations in the first two frames ($\mathbf{t}_0$). Under this metric, the translation vectors and plane distances are $\mathbf{t}_m = \sigma_m \bar{\mathbf{t}}_m$ and $d_{m,k} = \sigma_m \bar{d}_{m,k}$, respectively (as a reminder, m is the frame number and k is the index of the cluster, which is associated with a plane with normal $\mathbf{n}_{m,k}$). We can then recover the sequence of *scale factors* $\{\sigma_1, \sigma_2, \dots\}$ recursively.

4.2. Cluster Chain Determination

Determining chains of pairwise image matches is a critical step in classical SLAM algorithms [8], as it allows one to associate a point in space with a number of feature points detected over multiple images, and thus to compute the reprojection error for a putative set of poses. In our case, we are interested in not only finding match chains, but also *cluster chains*, which identify the same planar model across subsequent images. We need to ensure that two separate planar models don't get mistakenly merged into one, or that the same model gets split in two. This is achieved through a suitable clustering of the directed N-partite graph $\mathcal{G}$, whose m-th partition contains nodes associated with points detected in the m-th image. Each node in the m-th partition may be connected to at most one node associated with a matching point in the previous image (partition m-1) or in the next image (partition m+1).

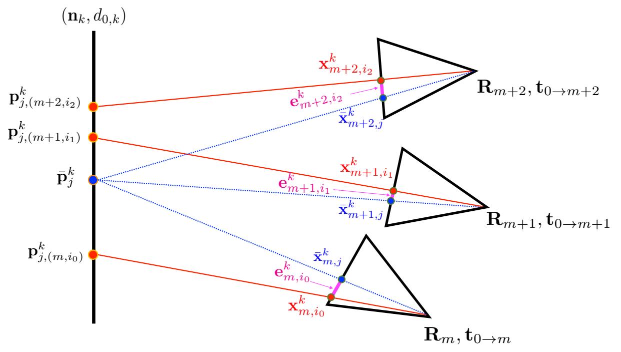
4.3. Plane-Constrained Bundle Adjustment

Bundle adjustment (BA) modifies a vector of model parameters (normally, the set of 3-D points, camera poses, and possibly intrinsic parameters) with the goal to ensure that observations are consistent with the model under an appropriate metric. Unlike typical BA, we do not attempt to optimize the location of individuals 3-D points. Rather, we modify the location (but not the orientation) of the K planes to which these points belong.

An image point $\mathbf{x}_{m,i}^k \in \mathcal{M}_j^k$ associated with the plane $(\mathbf{n}_k, d_{0,k})$ defines a 3-D point by the intersection of the line of sight through the $\mathbf{x}_{m,i}^k$ in the m-th camera and the plane. In the reference frame of the first camera, this point can be expressed as:

$$\mathbf{p}_{j,(m,i)}^k = \mathbf{R}_{0\to m}^T \left(\frac{d_{m,k}\, \mathbf{K}_m^{-1} \tilde{\mathbf{x}}_{m,i}^k}{\mathbf{n}_k^T \mathbf{K}_m^{-1} \tilde{\mathbf{x}}_{m,i}^k} - \mathbf{t}_{0\to m} \right) \quad (1)$$

In the equation above, $d_{m,k} = d_{0,k} + (\mathbf{n}_k)^T \mathbf{t}_{0\to m}$ is the distance of the k-th plane to the m-th camera location.



$(\mathbf{n}_k, d_{0,k})$
$\mathbf{x}^k_{m+2,i_2}$
$\mathbf{p}^k_{j,(m+2,i_2)}$
$\mathbf{e}^k_{m+2,i_2}$
$\mathbf{R}_{m+2}, \mathbf{t}_{0\to m+2}$
$\bar{\mathbf{x}}^k_{m+2,j}$
$\mathbf{p}^k_{j,(m+1,i_1)}$
$\bar{\mathbf{p}}^k_j$
$\mathbf{x}^k_{m+1,i_1}$
$\mathbf{e}^k_{m+1,i_1}$
$\bar{\mathbf{x}}^k_{m+1,j}$
$\mathbf{R}_{m+1}, \mathbf{t}_{0\to m+1}$
$\mathbf{p}^k_{j,(m,i_0)}$
$\bar{\mathbf{x}}^k_{m,j}$
$\mathbf{e}^k_{m,i_0}$
$\mathbf{x}^k_{m,i_0}$
$\mathbf{R}_m, \mathbf{t}_{0\to m}$

Figure 1: Computation of the reprojecton error for Bundle Adjustment (Sec. 4.3)

$\mathbf{K}_m^{-1}\tilde{\mathbf{x}}^k_{m,i}$ represents the line of sight through the pixel, expressed in terms of the m-th camera frame.

Ideally, the lines of sight through all pixels in $\mathcal{M}^k_j$ should intersect at the same point, $\mathbf{p}^k_j$, in the k-th plane. In practice, some amount of dispersion of the points $\{\mathbf{p}^k_{j,(m,i)}\}$ should be expected. Our BA procedure is designed to minimize a measure of such dispersion, defined as follow (see Fig. 1). We first compute the mean over the indices (m,i) in $\mathcal{M}^k_j$ of the back-projected points $\{\mathbf{p}^k_{j,(m,i)}\}$. We then reproject this mean point, $\bar{\mathbf{p}}^k_j$, onto the individual cameras at their estimated poses, obtaining:

$$\bar{\mathbf{x}}^k_{m,j} = EN\left(\mathbf{K}_m(\mathbf{R}_{0\to m}\bar{\mathbf{p}}^k_j + \mathbf{t}_{0\to m})\right) \tag{2}$$

Finally, we compute the reprojection errors $\mathbf{e}^k_{m,i} = \bar{\mathbf{x}}^k_{m,j} - \mathbf{x}^k_{m,i}$ for all $\mathbf{x}^k_{m,i} \in \mathcal{M}^k_j$. We minimize, over camera poses and plane locations (and, optionally, camera focal lengths), the norm of $\mathbf{e}^k_{m,i}$ as measured by the Huber loss, summed over all planes and all matched point chains.

5. Implementation and Experiments

5.1. Implementation Details

Vanishing points detection. Vanishing points are computed using the technique of [15] on line segments detected by the LSD algorithm [2]. Following [5], line segments with length of 20 pixels or more are clustered using T-linkage [9] (implemented on GPU). A candidate set of vanishing points is found, which is then refined by minimizing a form that penalizes geometric discrepancies.

Multi-planar clustering. SIFT features are matched across subsequent frames, then plane-constrained T-linkage (implemented on GPU) is run starting from non-minimal sets of matches identified using the region-based sample selection scheme of [5].

Non-linear minimization. Minimization of the reprojection errors with Huber loss (Sec. 4.3) is accomplished using the “Schur complement trick” as implemented by the Ceres solver.

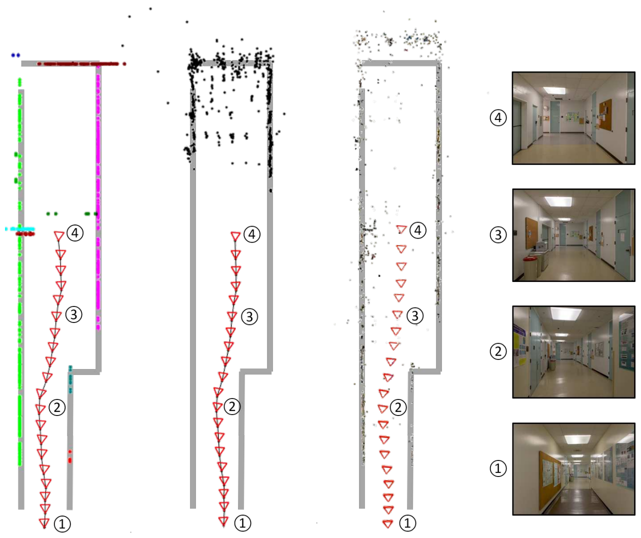


Figure 2: Bird-eye view of reconstructed points from the Corridor 1 sequence. Left: our algorithm (points in different detected planes are shown with different color). Center: ORB-SLAM [10]. Right: SfM Revisited [13]. Note that the sky blue and cinnamon points in our result represent planar surface induced by frontal surfaces of trashcan and printer.

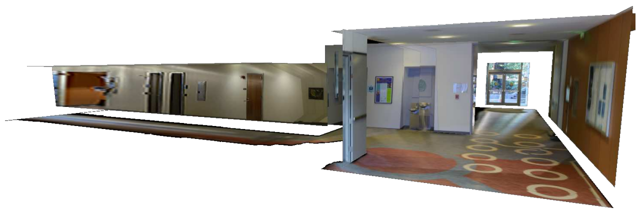
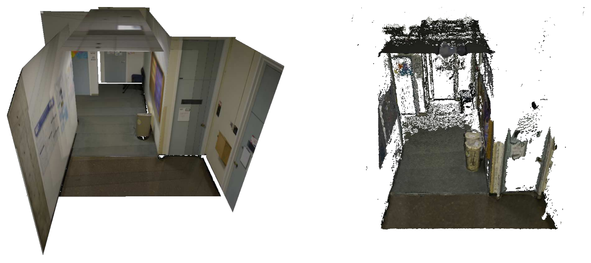


Figure 3: 3-D textured rendering of one of the walls of the reconstructed Corridor 2 scene.

Bundle Adjustment sequence. We first run a round of BA optimizing only plane locations and camera locations. Then, the plane merging procedure described at the end of Sec. 4.3 is applied. Finally, another round of BA is run, this time optimizing all parameters (plane locations, camera locations, camera rotation, and optionally focal lengths).

5.2. Experiments

We collected a number of sequences of corridors in our buildings using an iPhone 6 (1024 × 768 pixels). Each sequence contains 20 images, which were taken by hand at each step of walking approximately 0.4 meters of distance from each other. Our reconstructions have been computed without optimization of the camera’s focal length. We show reconstruction results for two scenes: Corridor



(a) (b)

Figure 4: 3-D textured renderings of the reconstructed Corridor 3 scene. (a) our algorithm. (b) SfM Revisited [13] with dense reconstruction option turned on.

1 (Figs. 2, bird-eye views) and Corridor 2 (Fig. 3, 3-D view). For Corridor 1, we also show the result using the open source implementation[1] of the ORB-SLAM algorithm [10], as well as the result using the open source implementation[2] of the "SfM Revisited" algorithm of [13]. The reconstructed points are shown on top of the floor plan, which was manually adjusted in all three cases to best fit the points. A 3-D textured rendering of a third scene (Corridor 3) is shown in Fig. 4 and compared with the result of SfM Revisited.

6. Conclusions

We have introduced a technique for motion recovery and surface reconstruction that makes use of the Manhattan World geometry at every step of the way. Our approach relies on pairwise matching of feature points, but represents geometric primitives in terms of planes. This enables a novel formulation of Bundle Adjustment that optimizes plane locations, rather than point locations. The result is expressed in terms of planar structures, a natural representation for indoor scenes.

Our algorithm has shown very promising results in relatively short sequences with a few dozen images. Further work will be needed to evaluate its performances in very long data sets (including loop closure), as well as in situations with multiple non-planar objects.

References

[1] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014. 1

[2] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: a Line Segment Detector. *Image Processing On Line*, 2012. 3

[3] N. Jiang, D. Lin, M. N. Do, and J. Lu. Direct structure estimation for 3D reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2655–2663, 2015. 2

[4] C. Kim and R. Manduchi. Indoor Manhattan spatial layout recovery from monocular videos via line matching. *Computer Vision and Image Understanding*, 157, April 2017. 1

[5] S. Kim and R. Manduchi. Multi-planar fitting in an indoor Manhattan world. In *IEEE Winter Conference on Applications of Computer Vision*, pages 11–19. IEEE, 2017. 2, 3

[6] S. Kim, R. Manduchi, and S. Qin. Multi-planar monocular reconstruction of manhattan indoor scenes. In *Proc. 3DV*, 2018. 1

[7] J. Košecká and W. Zhang. Video compass. In *European Conference on Computer Vision*, pages 476–490. Springer, 2006. 1, 2

[8] A. Levin and R. Szeliski. Visual odometry and map correlation. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2004. 2

[9] L. Magri and A. Fusiello. T-linkage: a continuous relaxation of J-linkage for multi-model fitting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 3

[10] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. 1, 2, 3, 4

[11] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *2011 IEEE International Conference on Computer Vision*, pages 2320–2327. IEEE, 2011. 2

[12] O. Saurer, F. Fraundorfer, and M. Pollefeys. Homography based visual odometry with known vertical direction and weak manhattan world assumption. In *Proc. IEEE/IROS Workshop on Visual Control of Mobile Robots*, 2012. 1, 2

[13] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4104–4113, 2016. 1, 3, 4

[14] R. Szeliski and P. H. Torr. Geometrically constrained structure from motion: Points on planes. In *European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, pages 171–186. Springer, 1998. 2

[15] J. P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *International Conference on Computer Vision*, 2009. 3

[16] S. Yang, D. Maturana, and S. Scherer. Real-time 3D scene layout from a single image using convolutional neural networks. In *IEEE International Conference on Robotics and Automation*, pages 2183–2189. IEEE, 2016. 2

[17] S. Yang, Y. Song, M. Kaess, and S. Scherer. Pop-up slam: semantic monocular plane slam for low-texture environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1222–1229. IEEE, 2016. 2

[1]https://github.com/raulmur/ORB_SLAM2
[2]https://github.com/colmap/colmap