Computer Vision and Image Understanding 172 (2018) 107–123

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Single-View Reconstruction using orthogonal line-pairs☆

Aamer Zaheer*,[a], Maheen Rashid[b], Muhammad Ahmed Riaz[a], Sohaib Khan[c]

[a] Department of Computer Science, Lahore University of Management Sciences, Lahore, Pakistan
[b] University of California, Davis, United States
[c] Science and Technology Unit, Umm Al Qura University, Makkah Al Mukarramah, Saudi Arabia

ARTICLE INFO

Keywords:
3D Reconstruction
Single-View Reconstruction
Multi-planar scenes
Orthogonal angles
Angle regularity
Urban environment

ABSTRACT

Multi-planar buildings and man-made structures are characterized by a profusion of parallel and orthogonal lines. In this paper, using orthogonal line-pairs as the primary feature, we describe an automatic algorithm to recover 3D structure of a multi-planar scene from a single image. First, we show how the presence of such regular angles can be used for 2D rectification of an image of a plane to a fronto-parallel view. Next, by exploiting this ability to rectify scene planes, we propose an automatic Single-View Reconstruction (SVR) method, assuming there are enough orthogonal line-pairs available on each plane. This angle regularity is only imposed on physically intersecting line-pairs, making it a local constraint. Furthermore, we also describe a novel algorithm to automatically segment planes within a scene, and discover their extents and adjacency relationships, using only orthogonal line-pairs. Unlike earlier literature, our approach does not make restrictive assumptions about the orientation of the planes or the camera view, and works for both indoor and outdoor scenes. Results are shown on challenging images which would be difficult to reconstruct for existing automatic SVR algorithms.

1. Introduction

In this paper, we propose an automatic Single View Reconstruction algorithm for urban environments by exploiting their frequently occurring orthogonal angles. Urban building environments are full of geometric regularities and can be represented as a combination of linear primitives, such as lines and planes, in a piecewise fashion. Moreover, the angles between these linear primitives are not randomly distributed, unlike those in natural scenes. In fact, most of the line-pairs and plane-pairs are mutually parallel or orthogonal. These regularities have been used to solve several problems including Structure from Motion (Bartoli and Sturm, 2003; Furukawa et al., 2009), Camera Calibration (Wilczkowiak et al., 2005), and Metric Rectification (Liebowitz and Zisserman, 1998; Zaheer and Khan, 2014) in addition to Single View Reconstruction. Here we focus on line-based Single View Reconstruction of a multi-planar environment using the angle regularity between line-pairs.

Our goal is to reconstruct scenes with arbitrary plane orientations that may not conform to a strict global model of angle regularity. Previous line-based approaches of Single View Reconstruction (SVR) assume the angle regularity between line-pairs to be a global phenomenon, that is, all lines on a plane are assumed to meet at mutually orthogonal vanishing points. Similarly, the plane orientations are assumed to follow either the Manhattan Model, that is all planes are aligned to the principal directions (Hedau et al., 2009; Lee et al., 2009; Park et al., 2015), or the Ground-Vertical Model, that is, all planes are orthogonal to a common ground plane (Barinova et al., 2008; 2010; Hoiem et al., 2005). These global models for angle regularity are highly restrictive as all the planes and lines must follow a strict global constraint. In contrast, we propose a statistical approach to reconstruct planes with arbitrary orientations and use local angles between line-pairs rather than global vanishing points based grouping. Computing arbitrary plane orientations instead of a fixed set of orientations expands the search space considerably, but we propose a robust algorithm that is shown to work exceptionally well in practice. Some examples are shown in Fig. 1.

As a motivation for the local angle regularity constraint, consider Fig. 2 (Left) which shows histogram of angles between pairs of lines on over 5000 patches from the Doersch et al. dataset (Doersch et al., 2012). The dataset contains rectified local image patches of multiple urban areas around Europe. The histogram at top-left shows two large peaks at zero and ninety degrees corresponding to parallel and orthogonal line-pairs. The histogram at bottom-left only uses angles between line-pairs that intersect within the image patch, thus removing the parallel line-pairs. It is apparent that in this dataset, parallel and orthogonal line-pairs are much more frequent than line-pairs meeting at

☆ The name of the Editor in chief Nikos Paragios.

* Corresponding author.
E-mail address: formanite98@gmail.com (A. Zaheer).

https://doi.org/10.1016/j.cviu.2017.11.014
Received 21 February 2017; Received in revised form 23 November 2017; Accepted 27 November 2017
Available online 01 February 2018
1077-3142/

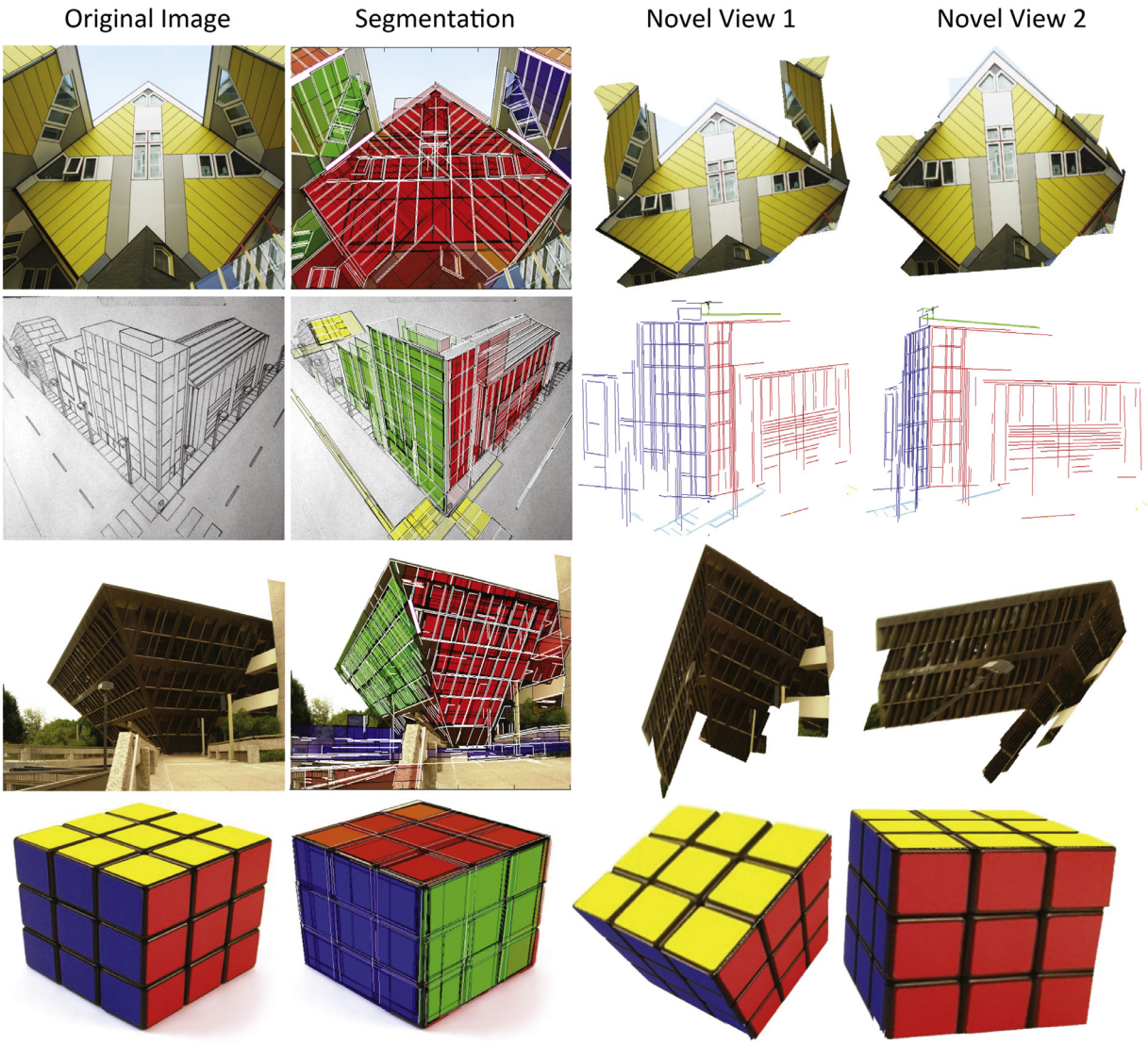


Fig. 1. Reconstruction for the previously unsolved scenarios: Our solution can reconstructed arbitrarily oriented planes, where lines may not converge on orthogonal vanishing points. It also reconstructs line-sketches and images with atypical viewpoints.

any other angle. However, each image patch covers a small local area and the angle distribution of an image containing a large area might be different. In order to understand the relationship between a local constraint over a small area versus a global constraint over a large area, we use a satellite image of a large urban area (Fig. 2 Right) and compute the angle histogram between all lines (top) versus just the locally intersecting line-pairs (bottom). This illustrates that line-pairs are much more likely to be orthogonal in a local patch than the whole image. Hence, we base our approach on locally intersecting line-pairs as a primary building block and remove any global restriction on line and plane orientations.

Despite using a more flexible local constraint for angle regularity, the proposed algorithm works for both indoor and outdoor scenes with no constraints on the camera viewpoint. Previous literature typically makes further assumptions of indoor or outdoor context, ground level viewpoint, and visibility of ground plane. These assumptions make multi-planar segmentation easier by restricting the multi-planar layout of how and where the planes meet. However, it limits the scope of each algorithm to a particular restrictive scenario. For example, consider the Rubik's cube in the last row of Fig. 1. The cube follows the Manhattan world model perfectly but does not fit the indoor/outdoor context assumptions that require the horizontal planes to be floor, ceiling or ground only. Similarly, in the top row of Fig. 1, all the planes are mutually orthogonal but the lines do not follow three principal directions and the structure does not fit either the indoor or outdoor context assumptions used in earlier literature.

In contrast, our only assumption on the multi-planer layout is that it must be a 2.5D connected set of planes — an assumption implicit to all the previous work discussed in related work (Section 1.1). The 2.5D assumption means that each pixel corresponds to exactly one 3D point in the world. If the planes are not connected with each other, there will be no line-based constraint on their relative depths, that is, two mutually disconnected sets of planes will not have the correct ratio of scale or depth with respect to each other. Hence we propose a line-based SVR method for the urban environments that have local angle regularity and consist of a 2.5D connected set of planes with arbitrary plane orientations.

The key idea in exploiting angle regularity is that the image of a 3D plane can be rectified to a fronto-parallel view by searching for the homography that maximizes the number of orthogonal angles between transformed line-pairs. This rotation-induced homography yields the normal vector of the 3D plane. For scenes containing more than one 3D plane, our approach has four main steps: 1) orthogonal line-pairs are assigned plane memberships by iteratively computing plane orientation

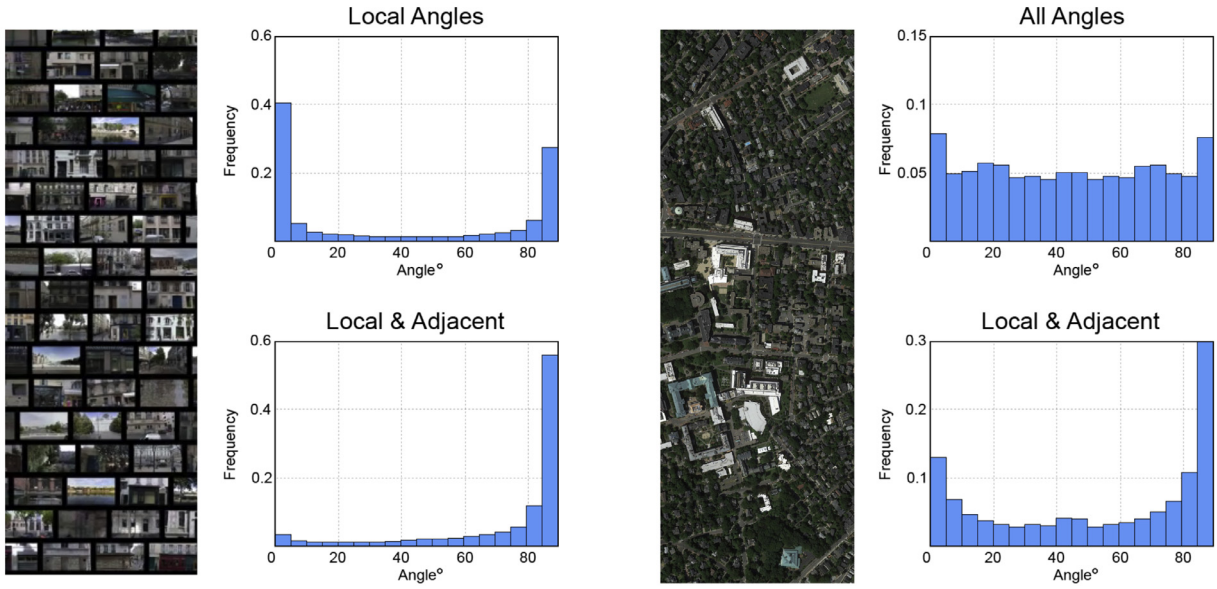


Fig. 2. Local angle regularity in the man-made world: Adjacent line-pairs are highly likely to be orthogonal. Left: Angle histograms for over 5000 local fronto-parallel patches in the Paris Dataset (Doersch et al., 2012), all pairs are used in top histogram while only the line-pairs that intersect within the patch are used in bottom histogram. Right: Angle histogram for all line-pairs in the satellite image in top histogram vs. only the line-pairs that intersect within a small neighborhood of each other are considered in the bottom histogram.

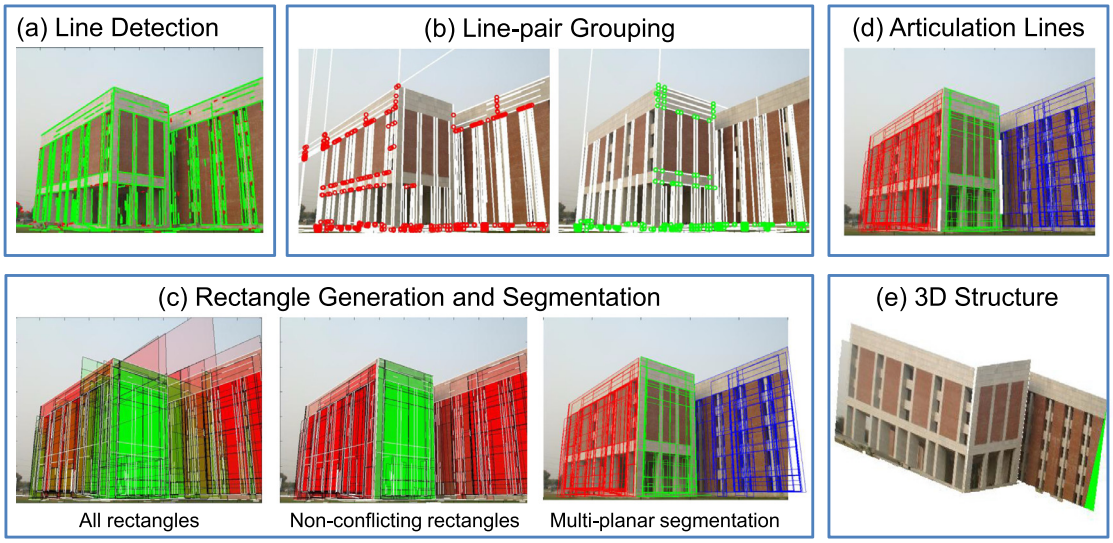


Fig. 3. Shape from angle regularity: (a) Original image superimposed with line detection. (b) Lines are extended to intersect, and two plane orientation hypotheses (red and green) are generated through RANSAC. (c) Line-pairs form rectangular regions and some overlapping rectangles have conflicting plane orientations. Three planar segments (red, green and blue) are identified after removing conflicts. (d) Articulation lines between planes are shown in white. (e) Novel view of 3D reconstruction. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

hypotheses through RANSAC (Fig. 3(a) and (b); 2) rectangles are generated from orthogonal line-pairs, and grouped into planar segments (Fig. 3(c); 3) the adjacency of planar segments and their shared articulating lines are computed, using global geometric analysis of all lines and plane segments (Fig. 3(d); and 4) the articulating lines and the plane normals are used to solve for the full 3D structure (Fig. 3(e)).

This paper extends the algorithm presented in an earlier version of our work (Zaheer et al., 2012), in implementation, analysis and experimentation. The rest of the paper is organized in three sections. First, we separate out the problem of automatic multi-planar segmentation by providing a manual segmentation and explain the rest of the steps in Section 2. In Section 3, we provide an algorithm for automatic multi-planar segmentation and reconstruction. Finally, detailed results including qualitative and quantitative evaluation of each module and comparisons with previous approaches are presented in Section 4.

1.1. Related work

Initial SVR research focused on interactive approaches to provide the multi-planar segmentation, using a global geometric constraint on lines and planes, and assuming the scene to be a 2.5D world. A classic example is *'Tour Into the Picture'* by Horry et al. (1997), which took the strict assumption of a Manhattan World and all camera axes being aligned with the world principal directions, leading to one finite and two infinite vanishing points. Kang et al. (2001) later relaxed the assumptions of *Tour into the picture* to work with two finite vanishing

points, that is, a vanishing line, while the geometric constraints were again specified manually. The underlying building block of most subsequent line based SVR methods has been the vanishing-point-based rectification technique proposed by Liebowitz et al. (1999) and Sturm and Maybank (1999). The idea is to first group lines according to their vanishing points and then use two orthogonal vanishing points for rectification and reconstruction of a plane. They used an interactive approach for marking plane boundaries as well as vanishing points (Liebowitz et al., 1999). Subsequent work on detection of orthogonal vanishing points removed the need for manually identifying the vanishing points (Almansa et al., 2003; Wildenauer and Vincze, 2007).

In Section 2, we develop an analogous interactive algorithm using local line-pair constraints to compute arbitrary plane orientations rather than vanishing points or Manhattan model. It includes: A 2D rectification constraint, an automatic method for 2D rectification without assuming two orthogonal vanishing points which yields the plane pose for each plane individually, and finally a reconstruction method that uses the connected world assumption to constrain the relative depths of planes.

In order to automatically segment the image into multiple planes, further contextual assumptions regarding the multi-planar layout of the scene have been used in literature. Previous methods typically assume either an outdoor or an indoor scenario which restricts the total number of horizontal planes to one (ground only) or two (floor and ceiling) and require their visibility. The whole world is either assumed to be Manhattan or, all the planes are required to be orthogonal to the ground plane.

Hoiem et al. proposed a ground-vertical model for outdoor world in *Automatic Photo Popup* where they assumed the world to be made up of connected vertical planes, all intersecting a common ground plane. They combined a learning based segmentation approach with geometric constraints in order to compute a *popup* model of the world (Hoiem et al., 2005). Their vertical plane reconstruction was mainly based on the always visible ground-vertical boundary line. Using line-based constraints and the outdoor ground-vertical model of photo-popup, Barinova et al. proposed an automatic line and appearance based method for SVR (Barinova et al., 2008). They extended the *Tour Into Picture using a Vanishing Line* version by Kang et al. and removed the assumption of any infinite vanishing direction in the outdoor scenario. Instead, they corrected for vertical tilt followed by reconstruction using a vanishing line. A texture based approach, *Transform Invariant Low Rank Textures(TILT)*, and a recent line-based method by Pan et al. compute a vertical facade model similar to *Photo Popup* but, importantly, do not require the presence of a ground plane (Pan et al., 2015; Zhang et al., 2012). Pan et al. combine geometric features such as horizon and vanishing points to compute a surface normal map using a line-sweep algorithm similar to Lee et al. (2009) and combine it with a semantic building segmentation to build vertical facade models. However, these methods only reconstruct vertical planes.

For the indoor scenarios, a learning-based approach similar to *Automatic Photo Popup* was used by Delage et al. under a Manhattan world assumption (Delage et al., 2006). However, the most influential work in the indoor world assumes the room layout to be a box scenario in order to handle the indoor clutter problem (Hedau et al., 2009; 2010). These works combine line and appearance based constraints to construct the restricted indoor models. Another direction in line-based SVR is an analysis of the 2D and 3D junctions between lines. The seminal work in this direction was Kanade's theory of *Origami World* (Kanade, 1980). Later, *Indoor World* work by Lee et al. (2009) successfully combined the junction analysis paradigm with an Indoor Manhattan World assumption to recover the underlying structure of a cluttered indoor scene automatically. In addition to restricting the scene to be Manhattan, they further require that the only horizontal surfaces are floor and ceiling, and their intersection with the walls is visible. The most interesting work based on junction analysis "lifts" 2D lines to a 3D Manhattan world (Ramalingam and Brand, 2013). While this is the only other line-based method that works in both indoor and outdoor world, it is limited to a Manhattan world model. A recent line-based method used Manhattan world assumption and graph-cut optimization to compute a simpler indoor reconstruction (Park et al., 2015).

Most state-of-the-art line-based SVR algorithms, e.g. Lee et al. in indoor scenario and Barinova et al. in the outdoor scenario, depend on the two-step approach which groups the dominant vanishing points globally and assumes them to be mutually orthogonal in 3D (Liebowitz et al., 1999). Yu et al. showed that this global grouping may be ambiguous even in Manhattan environments, and proposed a local check called *Spatial Coherence* to protect against it (Stella et al., 2008). They further computed rectangles in the scene and grouped them according to the order of their depth, but did not extend it to a full 3D reconstruction. Junction analysis is also an effective way to impose local constraints.

There is a large body of work on SVR that either assumes additional prior information, such as "Rent3D", which uses floor plans to combine multiple "box" models of an apartment, or depend completely on a machine learning framework to "learn" the depth map, such as the seminal work by Saxena et al. (2009) and recent deep learning based approach that recovers surface normals from a single indoor image (Wang et al., 2015). However, for the purposes of this paper, we limit the scope of our discussion and comparisons to methods that use geometric line-based constraints in a significant way (Barinova et al., 2008; Hedau et al., 2009; Hoiem et al., 2005; Lee et al., 2009).

We outline an automatic scene segmentation and reconstruction method in Section 3 which only requires local angle regularity assumption and a 2.5D connected multi-planar scene. Unlike previous automatic methods, our method requires neither an indoor, nor an outdoor scenario; does not place any restriction on plane orientations; works with any number of horizontal planes; and does not restrict the camera to be necessarily viewing a horizontal-vertical boundary. We are able to compete with earlier automatic SVR work in terms of accuracy while assuming less about the camera and world structure.

2. Interactive reconstruction

Given an image of a multi-planar 3D scene, Single View Reconstruction answers two questions:"Where are the planes in this 2D image?", and "What is the 3D structure of these planes?" Our solution to the first question, that of automatic plane segmentation, is delayed till Section 3. Here, we assume that plane segmentation is available through user input, and answer the second question of estimating the 3D structure of these plane segments. The steps of this interactive algorithm are outlined in Fig. 4.

2.1. Line-pair detection and grouping

The process of interactive reconstruction begins by detecting locally adjacent line-pairs. After reading an input image and its EXIF data, line segments are detected using LSD, a fast line segment detector (Von Gioi et al., 2010). Typical problems with LSD output and its cleanup is shown in Fig. 5(a)–(c). Since we are interested in a local angle regularity constraint, we only consider adjacent line-pairs whose intersection point lies on or near both the line segments (see Fig. 5(d)). Moreover, parallel line-pairs might seem to intersect due to line detection inaccuracies. Therefore, a line-pair is considered *adjacent* only if the angle between the lines in the image is more than 10°.

A piecewise planar environment can be described by planar regions meeting each other in straight lines in $\mathbb{R}^3$. Therefore, we approximate our typical multi-planar urban environments by polygonal plane boundaries meeting at common lines. These polygonal plane boundaries are marked manually using an intuitive graphical interface along with reusing the common points and lines between them. The line-pairs detected as described in the previous paragraph are then grouped into their corresponding planar as described in the as described in the

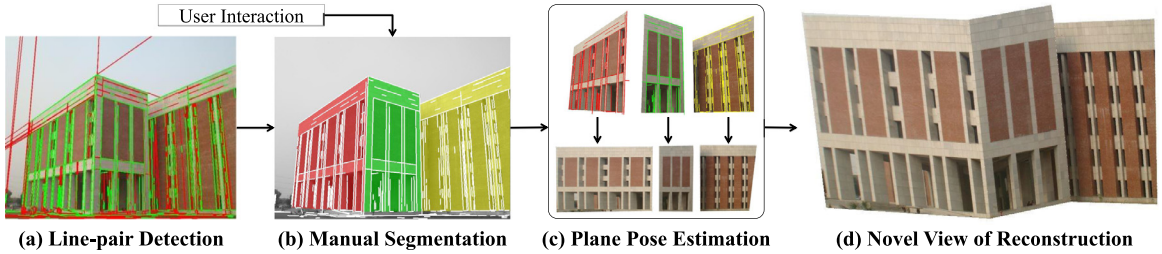


Fig. 4. Visualization of the steps involved in Interactive Reconstruction: (a) Lines intersecting after extension (red) are considered adjacent line-pairs, (b) user marks the plane boundaries, (c) individual planes are rectified independently to compute their pose relative to the camera, and (d) plane poses and common points are used to reconstruct the multi-planar structure. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

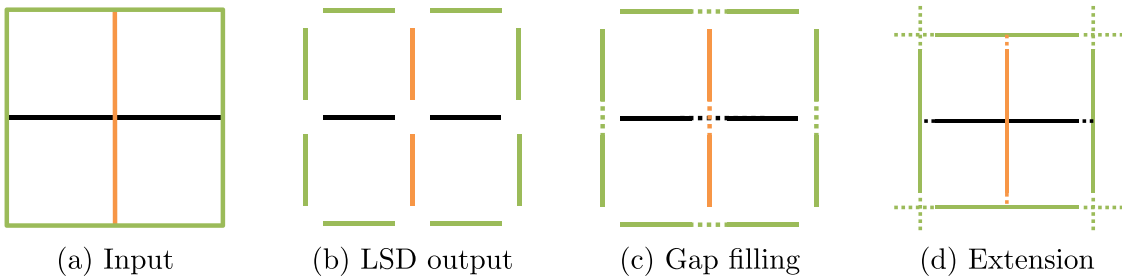


Fig. 5. Line-pair detection: (a) Input image contains six lines, (b) twelve line segments detected by LSD (Von Gioi et al., 2010) which breaks line segments into smaller pieces at the intersection points, (c) gap-filling (dashed) between collinear line segments merges broken line segments but does not work for line segments ending in an intersection, (d) line extension (dashed) to cover the full length of such line segments and compute neighboring pairs. Lines are extended until they hit a detected line segment, or the extension becomes longer than the gap-filled line segment. All adjacent line-pairs are recovered by simply checking for line segment intersections after the line extension process.

segments for pose estimation algorithm in Section 2.2, where the pose of every plane is computed independently from the rest of the scene.

2.2. Plane pose estimation

To recover the orientation of each manually marked planar segment, we attempt to rectify the plane such that it becomes fronto-parallel to the camera as shown in Fig. 6. For simplicity, we consider an image with only one plane orientation and propose a solution for 2D image rectification by recovering the plane pose. The same process can be applied to each planar segment iteratively.

The original image of the plane and its rectified view are related by a homography which is generated by the camera rotating about its center (Hartley and Zisserman, 2004). Therefore, the camera rotation with respect to the planar segment can be recovered from the rectifying homography. Any rotation induced homography, H, that relates two rotated views is given by:

$$\mathrm{H} = \mathrm{K}\mathrm{R}_Z^{\gamma}\mathrm{R}_Y^{\beta}\mathrm{R}_X^{\alpha}\mathrm{K}^{-1}, \tag{1}$$

where K is the 3×3 matrix of intrinsic parameters, and R_X^{α}, R_Y^{β}, and R_Z^{γ} denote rotations about the *X*, *Y*, and *Z* axes of the camera by α, β, and γ respectively.[1]

We assume square pixels and the image origin at camera's principal point, which reduces K to diag$[f, f, 1]$, containing a single focal length parameter, *f*. Under these assumptions, $\mathrm{K}\mathrm{R}_Z^{\gamma}$ simplifies to a similarity transform, which does not affect rectification and can therefore be ignored. Hence, the search space for the rectifying homography is reduced to just three parameters, α, β, and *f*. For most of our experiments, the focal length is known from the camera's EXIF data,[2] leaving only two parameters: α and β. The rectifying homography is now given by:

$$\mathrm{H}_{\alpha,\beta} = \mathrm{R}_Y^{\beta}\mathrm{R}_X^{\alpha}\mathrm{K}^{-1}. \tag{2}$$

We have empirically found that these simplifying assumptions do not qualitatively degrade results. Having a simplified search space, however, means that we require fewer number of constraints in order to compute the plane pose. In the following, we formulate and compare some angle constraints that may be utilized in order to recover the plane pose.

2.2.1. Orthogonal angle constraint

If a pair of lines $(\mathbf{l}_i, \mathbf{l}_j)$ represents the image of mutually orthogonal line-pairs in 3D space, then the following cost function can be used to compute the fronto-parallel plane pose:

$$(\alpha, \beta) \;=\; \mathrm{argmin}_{(\alpha,\beta)} \sum_{(i,j)} \left(\widetilde{\mathbf{v}}'^{\top}_i \widetilde{\mathbf{v}}'_j \right)^2, \tag{3}$$

where $\widetilde{\mathbf{v}}'_i$ is the unit normal vector of the line $\mathrm{H}_{\alpha,\beta}^{-\top}\mathbf{l}_i$, that is, it is the vector formed by the first two elements of the line $\mathrm{H}_{\alpha,\beta}^{-\top}\mathbf{l}_i$ normalized to unit length. This is because after the correct rectifying transformation, $\mathrm{H}_{\alpha,\beta}$, the two lines will have the parameters, $\mathbf{v}_i = \mathrm{H}_{\alpha,\beta}^{-\top}\mathbf{l}_i$ and $\mathbf{v}_j = \mathrm{H}_{\alpha,\beta}^{-\top}\mathbf{l}_j$. The first two elements of these transformed lines, written as $\widetilde{\mathbf{v}}'_i$ and $\widetilde{\mathbf{v}}'_j$ specify their orientation, and should be orthogonal after correct rectification.

This cost surface is not convex and some incorrect local minima may arise but we rarely have to try more than three random initializations to reach the global minimum. In the following RANSAC based algorithm, we simply take a random initialization for (α, β) with each random sample and the local minima get discarded because they do not have enough inliers supporting them.

2.2.2. Regular Angle Consensus

The method for rectification described above can be used for plane pose computation if known orthogonal line-pairs are given but in order to automate it, we need to identify the orthogonal line-pairs

[1] We follow a mathematical notation similar to Hartley and Zisserman (2004); x is a scalar; $\mathbf{x}$ and $\mathbf{X}$ are homogeneous vectors in $\mathbb{P}^2$ and $\mathbb{P}^3$ respectively; $\bar{\mathbf{x}}$ and $\widetilde{\mathbf{X}}$ are their inhomogeneous versions; X denotes a matrix; while $\bar{\mathbf{x}}$ is used for a column vector.

[2] If focal length is not available through EXIF data, it can be computed if at least two vanishing points are at a finite location (Sturm and Maybank, 1999).

Fig. 6. A typical tilted viewpoint is shown in (a) where vertical lines do not appear to be parallel in the image. The same surface seen from a fronto-parallel square-pixel camera shows no angle distortion in (b).

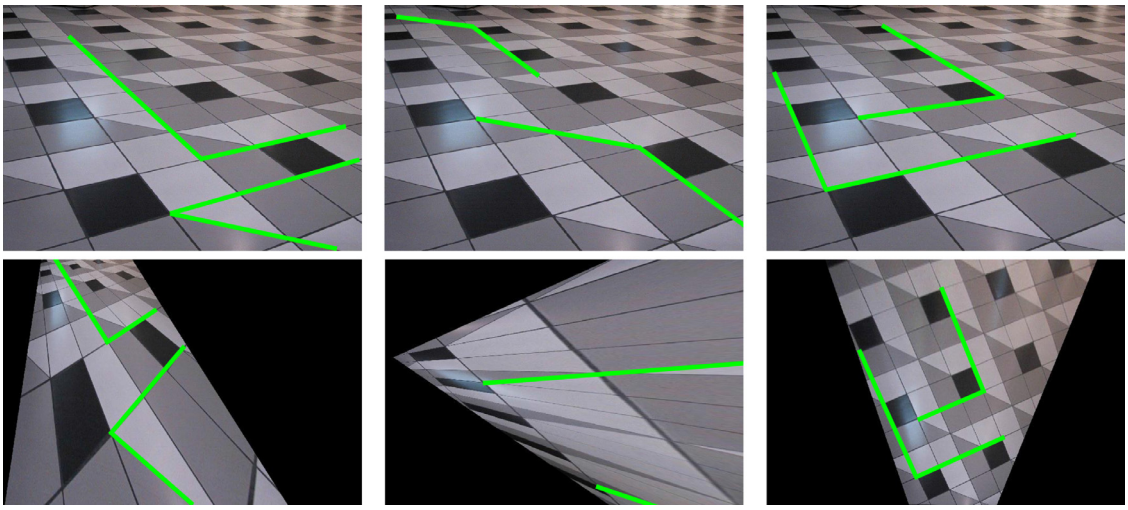


Fig. 7. Visualization of different random samples in Regular Angle Consensus over 978 line-pairs. The top row shows the line-pairs selected in the input image. The bottom row shows the rectified line-pairs and image. The illustration shows output rectification when (a) one, (b) none, or (c) both the selected line-pairs in the input random sample were correct.

automatically. Typical urban environments are characterized by an abundance of locally intersecting orthogonal line-pairs as discussed in Section 1. We use the local angle regularity assumption together with the nonlinear rectification algorithm described above to automatically identify the orthogonal line-pairs and rectify the image. Using the high frequency of orthogonal line-pairs, we propose a random sampling algorithm based on RANSAC (Fischler and Bolles, 1981) which finds the plane orientation that maximizes the number of orthogonal line-pairs in the rectified image.

Random Sample Consensus, or RANSAC (Fischler and Bolles, 1981), is a popular robust estimation technique capable of rejecting large percentages of outliers. We compute a rectifying homography for two randomly selected line-pairs and then see how many other line-pairs have also become orthogonal. The rectifying homography that rectifies the most number of line-pairs after a number of trials is the most probable solution. This *Random Sample Consensus* framework is naturally applicable to our assumption of angle regularity where 90° angle is assumed to be frequent but no global assumption about the line orientations is needed.

We show three different initial random samples and corresponding rectifications in Fig. 7. It shows that the largest set of inliers is automatically achieved for the correct random sample. The number of corresponding inliers (out of total 978 in this example) is provided below the images. The rectified image in Fig. 7(b) is not shown to full extents as wrong line-pairs generate a close to degenerate homography. Since the rectifying homography and the orthogonal line-pairs are computed simultaneously, there is no need to know in advance which pairs of lines are mutually orthogonal in 3D. The complete algorithm is described in Algorithm 1.

Two input parameters are needed for Algorithm 1, the inlier threshold t and the maximum number of trials to perform. Note that orthogonality cost is normalized between zero and one so it is easy to find a threshold that works for all realistic scenarios. We empirically chose $t = 10^{-2}$ which tolerates the typical noise in line detection (i.e. up to 5.7°) while minimizing the accidental alignment of non-orthogonal line pairs. For simplicity, the maximum number of random trials to be performed is also assumed to be fixed in Algorithm 1 as proposed by Fischler and Bolles (1981) but a subsequent improvement allows for adaptive computation of this number automatically as described in Hartley and Zisserman (2004). Using the latter method we eliminate the number of trials parameter.

In Fig. 8, we show a comparison between our 2D rectification method and the Direct Metric Rectification approach by Liebowitz and Zisserman (1998) - the first approach to use orthogonal line-pairs for 2D Rectification. For this comparison, we generated fifty line-pairs per experiment with varying percentages of orthogonal angles and

Input: L, a set of lines in the image
A, line adjacency matrix
K, the camera calibration matrix
t, the RANSAC threshold

```
bestinliers ← {}
bestH_{α,β} ← I_{3×3}
repeat
  pick two random adjacent line-pairs
  find the rectifying orientation H_{α,β}
       by minimizing orthogonality cost
  inliers ← {}
  for all l_i, l_j ∈ L and A(i, j) > 0 do
    if (ṽ'_i^⊤ ṽ'_j)^2 < t then
      add (i, j) to inliers
    end if
  end for
  if |inliers| > |bestinliers| then
    bestH_{α,β} ← H_{α,β}
    bestinliers ← inliers
  end if
until maximum number of trials are done
```

Algorithm 1. Regular Angle Consensus.

distorted them using a random rotation induced homography. We also simulated line detection error by adding zero mean Gaussian noise of 0°–3° standard deviation in one line orientation for each pair. After rectification by either algorithm, we compare the rectified line-pair angles with the original angles and compute the Root Mean Squared Error for each experiment. Two hundred such experiments were averaged to generate each data point plotted in Fig. 8. We do not show errors above 25° to improve readability of the graphs.

Liebowitz' algorithm requires a minimal random sample of five line-pairs and we use the same RANSAC method with a 5-pair minimal set as described above and both methods use an inlier threshold of 0.0349 which corresponds to 2° error in rectified angle. Our implementation of the 5-pair algorithm is based on the steps specified in Hartley and Zisserman (2004). Our 2-pair method is always more accurate than the 5-pair method given the same percentage of inliers and noise distribution, as shown in Fig. 8. We use non-linear optimization while the 5-pair algorithm has a linear closed form solution. However, the 2-pair method requires much fewer trials in RANSAC as number of trials in RANSAC increase exponentially with an increase in the minimal sample set size. Also note that the 5-pair algorithm does not explicitly compute the plane pose, as required for Single View Reconstruction. Moreover, the 5-pair algorithm fails when all the input lines are aligned in a grid because the angles in a grid pattern are invariant to non-uniform scaling. In contrast, explicit plane pose recovery works well for a grid pattern.

2.3. Reconstruction

Let R_i be the 3×3 rotation that makes the i-th plane fronto-parallel to the camera, that is, it rotates the plane normal, $\widetilde{\mathbf{N}}_i$, to the camera optical axis $\mathbf{Z}$.

$$[0, 0, 1]^\top = \mathrm{R}_i \widetilde{\mathbf{N}}_i \tag{4}$$

$$\widetilde{\mathbf{N}}_i = \mathrm{R}_i^\top [0, 0, 1]^\top. \tag{5}$$

While the rectification process provides us the plane normal for each plane, it does not impose any constraint on the depth of the plane. In fact, the plane can have an arbitrary depth in front of the camera with a corresponding arbitrary scale in the real world. However, if we have two planes that meet at a common 3D point, they cannot have an arbitrary depth relative to each other. Hence the relative depth of a pair of connected planes is constrained by the common point between them. The same argument applies to a larger set of connected planes since the depth of a plane in a connected set is constrained by the common points with the rest of the set.

2.3.1. Linear algorithm for depth recovery

Let π_1 and π_2 be two planes with normal vectors $\widetilde{\mathbf{N}}_1$ and $\widetilde{\mathbf{N}}_2$, respectively; that is, $\boldsymbol{\pi}_j = \begin{bmatrix} \widetilde{\mathbf{N}}_j^\top, & d_j \end{bmatrix}^\top$, where d_j represents the depth of the plane. Assume that the two planes share a common 2D point $\mathbf{x}$. Given that the camera is in the canonical view and the camera intrinsic matrix K is known, we may back-project the point into a 3D ray $\widetilde{\mathbf{X}} = \mathrm{K}^{-1}\mathbf{x}$. The 3D point $\mathbf{X}$ imaged as $\mathbf{x}$ may lie anywhere on the ray $\alpha\widetilde{\mathbf{X}}$ for some positive α, that is, $\mathbf{X} = [\alpha\widetilde{\mathbf{X}}^\top, 1]^\top$. In our particular case, however, this ray must intersect both the planes π_1 and π_2 at the same 3D point, therefore

$$\boldsymbol{\pi}_1^\top \begin{bmatrix} \alpha\widetilde{\mathbf{X}} \\ 1 \end{bmatrix} = \boldsymbol{\pi}_2^\top \begin{bmatrix} \alpha\widetilde{\mathbf{X}} \\ 1 \end{bmatrix} = 0. \tag{6}$$

Equating the two α values and rearranging yields

$$\begin{bmatrix} \widetilde{\mathbf{N}}_2^\top\widetilde{\mathbf{X}} & , & -\widetilde{\mathbf{N}}_1^\top\widetilde{\mathbf{X}} \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} = 0. \tag{7}$$

Generalizing to multiple planes and allowing for the possibility of more than one articulating point for each plane-pair, the set of constraints on relative depths can be written as a linear system,

$$\mathrm{A}\overline{\mathbf{d}} = \overline{\mathbf{0}}, \tag{8}$$

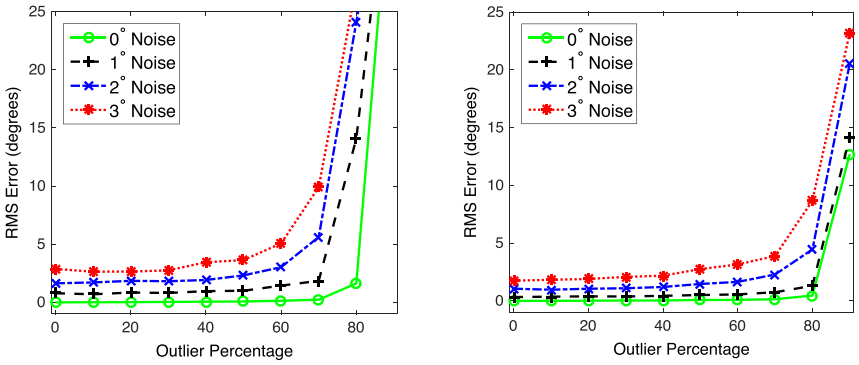
where $\overline{\mathbf{d}} = [d_1, ..., d_p]^\top$ contains the relative depths of p planes and every row of A contains one common point constraint between the j-th and k-th planes such that $a_{i,j} = \widetilde{\mathbf{N}}_k^\top\widetilde{\mathbf{X}}_i$, $a_{i,k} = -\widetilde{\mathbf{N}}_j^\top\widetilde{\mathbf{X}}_i$ and rest of the elements in the i-th row are zeros.

The vector of relative depths, $\overline{\mathbf{d}}$, is the right null vector of A and can be computed through SVD. It is recovered up to an arbitrary scale because Eq. (8) represents a homogeneous system. In order to fairly weigh the constraints, all back-projection rays are normalized to unit norm. Note that relative depths of planes can only be computed correctly for a

Input: L, set of rectified line-pairs
R, set of oriented rectangles

```
Compute goodness scores for all rectangles in R
repeat
  Compute weighted conflict scores for all rectangles in R
  Remove the rectangle with highest conflict score from R
until No overlapping pair of rectangles has conflicting orientations
```

Algorithm 2. Oriented Rectangle Consensus.



(a) 5-pair Algorithm by Liebowitz *et al.* (b) 2-pair Algorithm (Ours)

Fig. 8. 2D Rectification error comparison with the Direct Metric Rectification Algorithm by Liebowitz and Zisserman (1998): Our 2-pair method always outperforms the 5-pair algorithm for the same outlier percentage and line orientation noise.

set of planes if they are connected as a group, that is, they are either directly adjacent through some common points or by association through intermediate adjacent planes.

Our solution has similarities to the interactive approach used by Sturm and Maybank (1999), which was based on finding plane normals through vanishing points, and then solving for the common points and relative depths together. Though we only compute relative depths, and use a different constraint, our linear system turns out to be similar.

2.3.2. Nonlinear optimization

During plane pose estimation, we minimize the orthogonality cost for rectified line-pairs for each plane independently. The focal length may be known through EXIF data or computed from any of the planes. Plane orientations and focal length contain some error as they are computed from noisy line detections. During plane depth estimation, we assume the plane orientations and focal length to be fixed and use the common point constraints alone to find the relative depths which results in further error accumulation. However, these results can serve as an initial guess which can be further improved by refining all the plane parameters and focal length simultaneously while enforcing both the orthogonality and common point constraints.

For an estimated pair of planes π_q and π_r, their 3D intersection line can be projected into the image as $\mathbf{l}_{q,r}$. Let $\mathbf{x}_s^{q,r}$ be the common points between planes π_q and π_r, where $s \in (1, k)$ if there are k observed common points between the planes. The function $d(\mathbf{l}, \mathbf{x})$ measures the orthogonal distance between point $\mathbf{x}$ and line $\mathbf{l}$. The orthogonality cost for a plane π_k with orthogonal line pairs $(\mathbf{l}_i, \mathbf{l}_j)$ is defined in Eq. (3). Adding these two costs results in the combined cost function

$$\mathscr{E}(f, \boldsymbol{\pi}_1, ..., \boldsymbol{\pi}_p) = \sum_{\forall(\boldsymbol{\pi}_q, \boldsymbol{\pi}_r)} \sum_{s=1}^{k} d(\mathbf{l}_{q,r}, \mathbf{x}_s^{q,r}) + \sum_{\forall \boldsymbol{\pi}_k} \sum_{(i,j)} \left(\widetilde{\mathbf{v}'}_i^{\top} \widetilde{\mathbf{v}'}_j \right)^2. \tag{9}$$

We minimize this cost function over all the orthogonal line-pairs for all the rectified planes and all common point constraints for all the adjacent planes to find the optimal plane parameters and focal length. Note that the 3D reconstruction is only possible up to a scale, therefore we fix the depth of the first plane during optimization. The focal length is optimized even if it was available through EXIF data. Fig. 9 illustrates the advantage of the non-linear optimization described in this section. Before optimization (Fig. 9(a)), the planes do not stitch well along their intersection line but after optimization (Fig. 9(b)), the planes are more symmetrically reconstructed, and join along their common line to create a qualitatively superior reconstruction. Fig. 10 shows reconstruction of some such structures from single images where segmentation is provided.

3. Automatic reconstruction

In the previous section, we have shown the recovery of multi-planar structure from orthogonal angle constraints, provided the segmentation of each plane is manually specified. However, automatically segmenting planes in an arbitrary scene is a non-trivial problem in itself. This section deals with recovering the extents of each plane automatically, as well as identifying which planes meet each other, and if they do, identifying the articulating line between them. Keeping with the theme of the paper, our approach for automatic scene segmentation is also based primarily on the feature of orthogonal line pairs, hence demonstrating the power of exploiting this characteristic of man-made scenes.

Given a set of detected lines in an image of a multi-planar environment, our task is to automatically figure out the number of planes to be reconstructed, their boundaries in the image, and their 3D structure. Automatic SVR is an ill-posed problem as we have to deduce the 3D structure from a single 2D observation. Additional assumptions are needed to constrain the output structure. Most previous approaches restrict the space of possible structures by: 1) Assuming a global orientation constraint for the reconstructable lines and planes under Manhattan or Ground-Vertical model, 2) limiting the multi-planar layout to restricted indoor or outdoor world models with special planes where no horizontal planes other than the floor and ceiling are allowed, and 3) limiting the camera viewpoints by requiring the boundaries between vertical and the special horizontal planes to be visible.

In contrast, our objective is to reconstruct multi-planar structures with arbitrary plane orientations in both indoor and outdoor scenarios without requiring any special reference planes, and without imposing any restrictions on the camera viewpoints. Our only assumptions are angle regularity and that the reconstructed structure is 2.5D, that is, each pixel has at most one depth assignment, and planes must be connected, either directly sharing an intersection line or connected through intermediate adjacent planes. This 2.5D multi-planar layout assumption is implicitly required by all the related line-based SVR approaches, whether automatic or interactive. The strength of our approach lies in the angle regularity assumption, which is a flexible local constraint for arbitrary plane pose estimation and extends naturally to bottom-up segmentation of a 2.5D multi-planar environment, as we will discuss in this section. Our automatic SVR algorithm has four separate parts:

1. **Pose**: All the plane orientations are recovered first using angle regularity assumption as outlined in Section 3.1.
2. **Segmentation**: The contiguous regions belonging to each plane and their boundaries are identified in Section 3.2.
3. **Layout**: We figure out whether and where each pair of planar

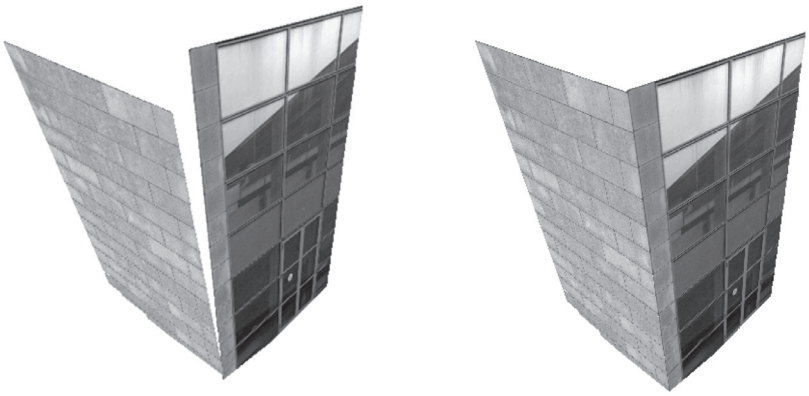


Fig. 9. Improvement achieved by nonlinear optimization which ensures that the planes actually meet at the articulation line through a minimal parametrization.

segments meet in Section 3.3.

4. **Reconstruction**: Lastly, the plane depths and 3D boundaries of planar segments are computed and the 3D structure is texture-mapped in Section 3.4.

The automatic reconstruction process is visualized for a simple image in Fig. 3.

3.1. Multi-planar pose estimation

In order to recover all the plane orientations, we apply the *Regular Angle Consensus* (Algorithm 1) in multiple iterations. In each iteration, a dominant plane orientation is recovered, its inlier line-pairs are removed, and the process is repeated to find the dominant plane orientation that rectifies the largest number of remaining line-pairs. This process of detection and removal of dominant consensus sets is repeated until the dominant plane orientation does not rectify a substantial number of line-pairs. If the dominant plane orientation in an iteration rectifies less than 10 line-pairs or less than 10% of total line-pairs, it is considered invalid and the repetition process is stopped. Note that the line-pairs belonging to multiple parallel planes will be grouped together since a rectifying homography only depends on the orientation of the plane and not on its depth. This strategy is able to recover arbitrary plane orientations independently of other planes and no global plane orientation constraint is needed.

The progression of the algorithm is shown in Fig. 3(b) where the first iteration recovers the building front orientation with line-pairs on both front-facing planes, while the second iteration recovers the side-facing wall. The third iteration is unable to rectify at least 10% of the total line-pairs and the algorithm is terminated.

A single orthogonal line-pair does not define a plane orientation uniquely. In Fig. 3, the horizontal lines on the vertical planes become parallel to each other near the horizon. The line-pairs formed by these lines are rectified by both orientations. Therefore, we recompute the inlier set of every plane orientation using all line-pairs. It allows such line-pairs to belong to both plane orientations. The line-pairs that are not rectified by any plane orientation are removed from consideration once the algorithm is terminated.

3.2. From line-pairs to regions: Multi-planar segmentation

At this stage we have recovered the plane orientations and assigned non-unique orientation membership to line-pairs. Now we need to move from line-pairs to image regions, by finding the support of each plane hypothesis in the image. This is a challenging step and forms the core of our automatic SVR algorithm. The key idea in automatic segmentation is that most locally adjacent orthogonal line-pairs arise from a supporting rectangular region in urban environments. This simple observation allows us to use angle regularity as a cue for segmentation as each plane contains many orthogonal line-pairs which can be associated to their supporting rectangles. We use the following three-step approach to exploit this observation and segment planar regions in the image:

1. **Region hypotheses**: Generating oriented region hypotheses from rectified line-pairs. Each pixel may be a member of multiple such regions with conflicting labels.
2. **Unique pixel labels**: Computing unique orientation labels for pixels, by removing conflicts between overlapping regions. This is a global analysis guided by the angle regularity assumption.
3. **Region grouping**: Grouping contiguous groups of uniformly labeled pixels to form the plane segments. This also gives us the total number of planes to be reconstructed.

3.2.1. Oriented rectangle hypotheses

In this section, we discuss the formation of oriented rectangle hypotheses which will be grouped into plane segments. We use rectified line-pairs from Section 3.1 to generate region hypothesis because it allows us to transfer plane orientations to pixel regions. Since orthogonal line-pairs typically support rectangular regions in urban environments, we use rectangles as our primary primitive, derived from the orthogonal line-pairs as a basic unit of segmentation.

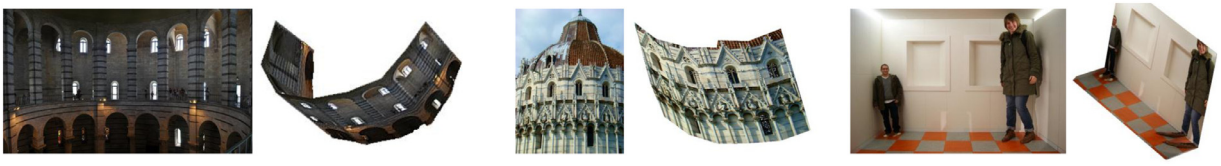


Fig. 10. Reconstruction with manual segmentation: The curved surfaces were treated as piece-wise planar, and their planar segments were identified. The right-most figure shows Ames Room illusion, where our reconstruction mimics human perception.

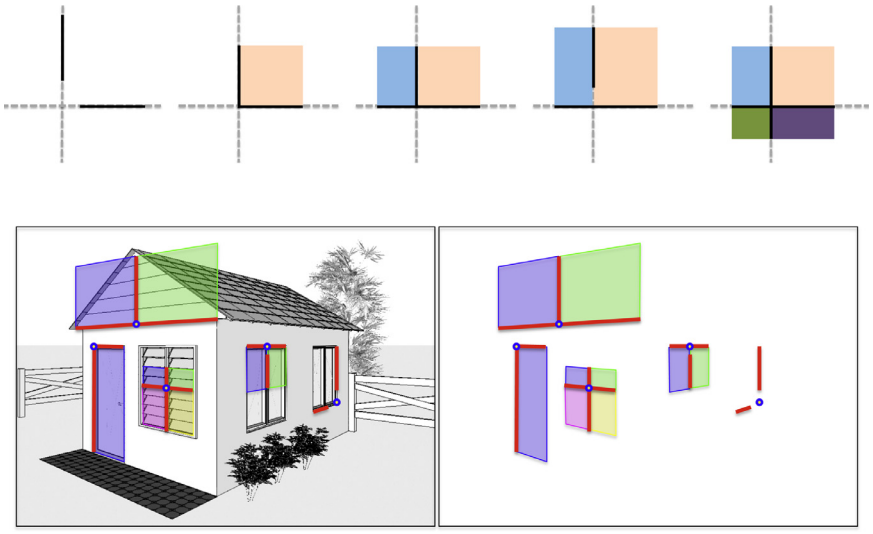


Fig. 11. Rectangles formed by line intersections. *Top:* Detected line segments (solid black) and their supporting lines (dashed gray) generate the rectangles in the rectified view (colored boxes). Five different cases of intersection are illustrated from left to right, resulting in zero, one, two, two and four rectangles respectively. *Bottom:* The same five cases of line intersections shown for oriented rectangles in a non-rectified illustrative image. The rectangle orientation is given by the plane hypothesis that the line-pair belongs to.

We create rectangles using the support line-pairs in the rectified view. A line-pair which is orthogonal in the rectified view, but appears at an arbitrary angle in the original image, may intersect in five different ways as illustrated in Fig. 11. For each case, rectangular regions with two sides supported by the line-pair are added to the region hypotheses and assigned the orientation label of the line-pair. If a line-pair has more than one orientation labels, rectangles are computed for each of the orientations. A frequent example of a line-pair associated with two orientations happens when one of the lines lies on the horizon - or very close to it - as demonstrated in Fig. 13.

Note that the line-pairs are orthogonal in the rectified view but distorted in the image. Hence, their corresponding rectangles are also distorted by the same rotation-induced homography. These regions in the un-rectified image are called “oriented rectangles”. We illustrate the process of generating oriented rectangles from line-pairs in Fig. 12, where the orientation of each rectangle is represented by its color. The sides of these rectangles that are not supported by detected line segments typically represent missed lines. We call them *hallucinated* lines and use them while computing the Multi-planar Layout in Section 3.3.

The oriented rectangles allow us to associate the orientation labels from line-pairs to their pixel neighborhoods. Since line-pairs may share lines, may intersect one another, and may be close enough to share their pixel neighborhoods, their corresponding rectangles may overlap. Furthermore, by the angle-regularity assumption, planar segments will be characterized by a profusion of line-pairs with the same orientation label. This clustering of line-pairs with similar orientation causes the creation of many overlapping rectangles with the same orientation label. Consequently, at this stage, planar segments are characterized by a number of overlapping oriented rectangles with the same orientation label.

However, there typically are some incorrect oriented rectangles whose orientation does not match the label of the underlying plane segment. Such oriented rectangles may arise out of errors in line-pair detection, or ambiguity and errors in line-pair grouping. However, the number of conflicting oriented rectangles are much smaller compared to overlapping rectangles with the correct orientation label of the planar segment they lie inside. For example, see the oriented rectangles generated in Fig. 3. The wrong rectangles here are usually generated by ambiguous labeling of line-pairs near the horizon (Fig. 13). Since we want to compute a 2.5D structure where each pixel is assigned to at most one plane, our goal is to remove the incorrectly oriented rectangles so no pair of overlapping rectangles will have a conflicting orientation.

3.2.2. Oriented Rectangle Consensus

Rectangle generation results in many overlaps between correctly

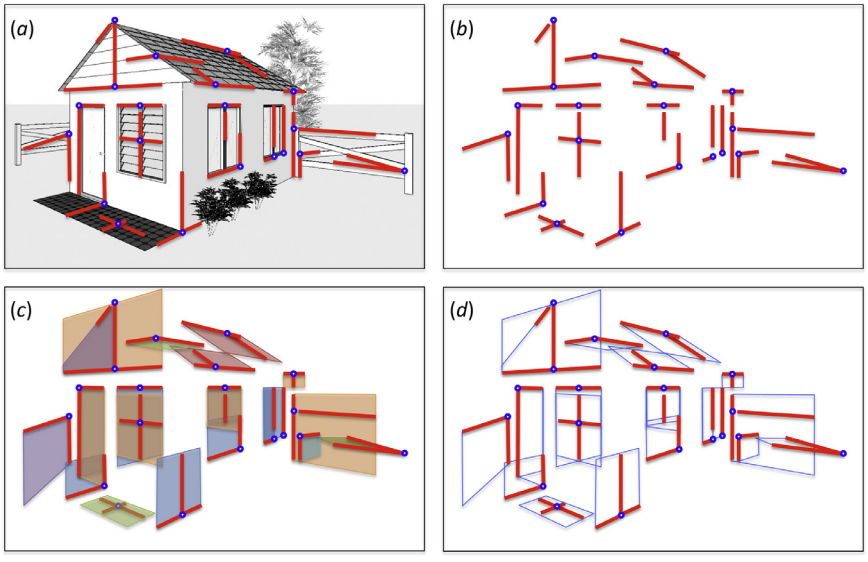


Fig. 12. Illustration of generating region hypotheses: (*a*) Detected line pairs overlaid on the original image. Note that not all of them represent actual orthogonalities in the world. (*b*) Detected line-pairs shown without the image, to emphasize the data that the algorithm is based on. (*c*) Oriented rectangles, generated from the line-pairs. Color of rectangle denotes its orientation. (*d*) Oriented rectangles are bounded by detected line-pairs (red) and hallucinated lines (blue). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

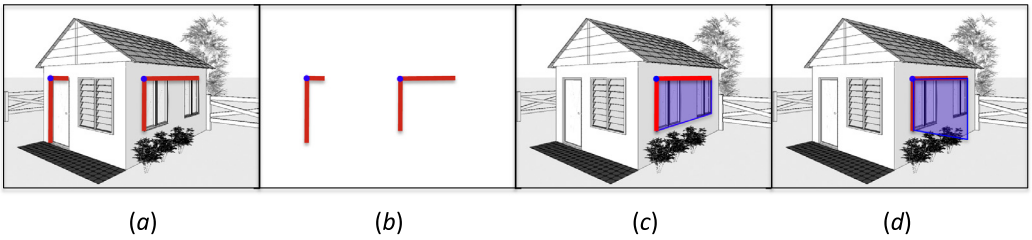


Fig. 13. Ambiguity in line-pair orientation: (*a*) Two line-pairs with the same image angle that are aligned at the horizon create ambiguity and appear exactly identical in the image (*b*). These line-pairs can belong to the plane hypothesis of either the front face of the hut or the side wall. (*c* and *d*) For one of those line-pairs, two different but equally plausible orientation hypotheses are shown.

oriented rectangles and relatively fewer overlaps with rectangles of conflicting orientation. We use these relative frequencies to remove orientation conflicts so that each pixel can be assigned a unique orientation. We formulate a simple greedy algorithm which repeatedly ranks all the oriented rectangles by the number of conflicts they cause, and removes the worst rectangle. The algorithm terminates when no overlapping pair of rectangles has a conflicting orientation. However, it treats all conflicts equally, that is, an incorrectly oriented rectangle adds the same conflict penalty to other rectangles as a good one.

Since each rectangle is associated to an orientation and contains many line-pairs that belong to the underlying planar segment, an incorrect rectangle will contain a large number of line-pairs that are rectified by the correct orientation. We use this observation to compute a measure of a rectangle's quality, called *goodness score*. The goodness score is a ratio of the number of line-pairs inside a rectangle that conform to its orientation and the total number of inliers inside the rectangle. A rectangle with the same orientation as the underlying plane will have a high goodness score because of line-pairs originating from the underlying plane. Here, a line-pair is considered inside a rectangle if the intersection point of the line-pair is inside the oriented rectangle boundary. The goodness scores are used to weigh the number of conflicts each rectangle causes.

For each rectangle, its *weighted conflict score* is computed as the sum of the goodness scores of all overlapping oriented rectangles with a conflicting orientation label. Because of this weighting by the goodness scores, a good rectangle with many bad overlaps will have a low conflict score, while a bad rectangle with even a few good overlapping rectangles will have a high conflict score. Rectangles are ranked by their respective weighted conflict scores. This process of ranking the rectangles by their conflict scores and removal of the worst rectangle is repeated until no overlapping oriented rectangles have a conflicting orientation. Note that an incorrect rectangle will have a small inlier percentage and thus have limited effect on the rank of a correct rectangle.

The rectangle consensus process is illustrated in Fig. 14. The rectangles induced by rectified line-pairs result in the distorted rectangles shown in Fig. 14 (left). The color of each rectangle indicates its plane orientation label. Note that significant overlaps exist between rectangles of different colors before conflict removal. However, after conflict removal, the remaining overlapping rectangles do not have different colors as Fig. 14 (middle) demonstrates.

3.2.3. Oriented rectangle grouping

Rectangles with the same orientation label need not be part of the same planar segment; parallel but different planes still have the same labels as shown by two red planes in Fig. 3(c), middle.

In order to separate physical planar regions, we use a graph based connected components algorithm so that only overlapping rectangles have the same plane labels. A graph is constructed where all rectangles represent a node and two overlapping rectangles have an edge between them. Connected component analysis labels the nodes resulting in multiple non-overlapping groups of rectangles of the same orientation getting different labels. The physically contiguous regions get unique labels as shown in Fig. 3(c) on the right. These groups of rectangles are treated as planar segments in the subsequent steps of the algorithm.

3.3. Multi-planar layout

At this point we know the orientations of the planar segments but their relative depths are not known. In order to constrain the relative depth of any two planes, at least one point common to both planes must be known, as explained in Section 2.3. Furthermore, in order to reconstruct multiple planar segments using the same relative scale, all planar segments must be connected. Determining multi-planar layout can therefore be seen as the task of determining which planar segments physically meet, and where they meet.

Geometrically, planar segments intersect at lines. We therefore search for common lines between pairs of planes instead of common points. The search of common lines includes all the detected lines as well as the *hallucinated* lines, that is, those lines created during the rectangle generation process in Section 3.2.1. First, we prune out geometrically implausible common lines between each pair of planes and determine the best common line using 2.5D connectivity assumption, in Section 3.3.1. Next, in Section 3.3.2, the multi planar layout is computed by graph theoretic analysis of all the planar segments and their common lines. Note, however, that we do not impose any indoor or outdoor context or visibility of special planes on the layout.

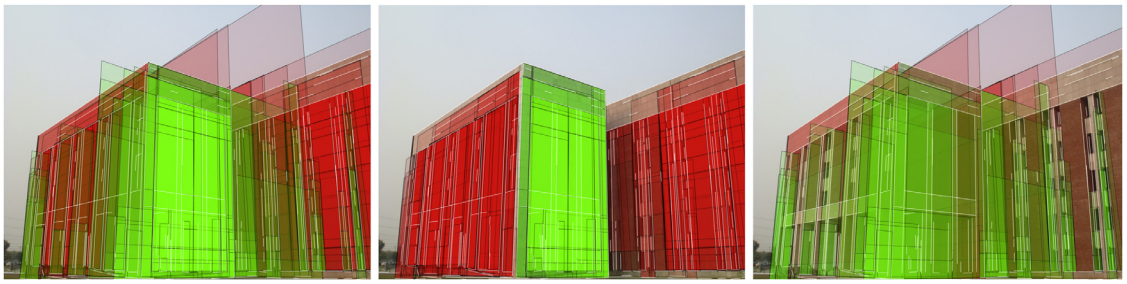


Fig. 14. Oriented Rectangle Consensus. Left: All oriented rectangles which also include conflicts as shown by overlap between red and green rectangles. Mid: Resulting consensus of rectangles with no conflicts. Right: The rectangles that were removed. They include mostly bad rectangles but also some good rectangles. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Fig. 15. Line of intersection being computed for the red and green planar segments on left. The yellow lines represent geometrically plausible lines. Since both planes are vertical, their intersection lines can only be vertical. On right, separation scores of all geometrically plausible lines are color coded from red (high) to blue (low). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3.3.1. Line of intersection

As a pair of non-parallel planes intersects at a line in 3D, we compute the best articulation line for each pair of non-parallel planes. Using a candidate set of all detected and hallucinated lines, a series of tests based on multi-planar geometry and 2.5D assumption are used to prune the candidates until at most a single best hypothesis is left.

Geometric plausibility: First, we consider each pair of non-parallel planes and prune out the geometrically implausible candidate lines. The direction of the intersection line must be orthogonal to both the plane normals and many candidate lines may not meet this criterion (see Fig. 15 for an example). In order to test a candidate line, we use one of its end-points to reconstruct the two planes using the algorithm described in Section 2.3. The 3D line of intersection is then projected into the image. The angle between the reprojected line and the candidate line denotes the reprojection error and should be zero if the 3D orientation of the candidate line is orthogonal to the two plane normals. If this angle is more than the threshold of 5°, the line is discarded. This process is repeated for all plane-pairs and a list of geometrically plausible intersection lines is computed for each pair of planes. The following tests are only performed on these selected candidate lines.

Separation: The assumption of a 2.5D multi-planar layout requires that a ray back-projected from any point in the image must intersect at most one 3D planar segment in the scene. Any line in the image bisects the image plane into two half-planes. If the two planar segments lie in separate half-planes induced by an intersection line, the resulting structure will be guaranteed to be a 2.5D structure, that is, no two scene planes may intersect any back-projected ray from the image. Therefore, we compute a separation score for each candidate line as a measure of how well it separates the two planar segments into its induced half-planes. In Fig. 15, each yellow candidate line on left is assigned a separation score as shown on right where red means a high separation score while blue represents a low separation score.

Proximity: Typically, multiple plausible lines separate the two planar segments equally well when the two planar segments are far from each other and do not meet physically. We use the separation score to pick the 5 best candidate lines and compute their distance to both the planar segments. The distance between a line and a planar segmented is the minimum orthogonal distance of the line from all the line segment end-points which form the planar segment. Among these, the line that has the minimum distance to both planar segments is chosen as the best line of intersection if the distance is less than a threshold (3% of image width). The pair of planar segments is considered non-adjacent otherwise.

3.3.2. A minimal connected multi-planar layout

We find the minimal set of lines of intersection using a graph search algorithm. Each planar segment is represented as a node and an edge connects a pair of planes with a valid intersection line. The edge is weighted by the larger distance of the intersection line from either planar segment. Larger distances, hence edge weights, arise from noise in the intersection line. Such a graph is typically a single connected component, that is, each node has a path to every other node. However, some noisy planar segments may be reconstructed due to clutter that are not adjacent to the correctly identified planar segments. Therefore, we first pick the largest connected component in this graph and discard other planar segments.

We only need a connected set of planes in order to constrain the relative depths of all the plane segments through the reconstruction algorithm in Section 2.3. Therefore, any cycles in the graph constructed above represent multiple paths between any two *connected* pair of planar segments. Due to noise in any of the previous steps, some of these alternate paths may not be consistent with each other, that is, they might lead to different solutions for depth. Therefore, we choose the best minimal subset of intersection lines that maintains the connectivity between planes. We compute the *minimum spanning tree* in this graph in order to pick the minimal set of articulation lines with the minimum sum of distances. The edges in this minimum spanning tree represent the most robust minimal set of adjacency constraints needed to constrain the relative depths of the planar segments.

Determining the multi-planar layout that is robust, and does not impose additional constraints on plane shape, orientation, or layout is a challenging task that has an infinite solution space. The approach described in this section is able to discard noisy plane segments, and compute a minimal set of intersection lines robustly. Furthermore, it uses simple tests derived directly from the assumptions of angle regularity and connected 2.5D multi-planar layout. In the process, no additional constraints are imposed on the orientation, layout, or shape of planar segments.

3.4. Reconstruction

At this stage, the normals of planar segments and the intersection lines between them are known. Using the algorithm described in Section 2.3, we constrain the relative depth between planar segments, hence recovering their complete plane equations. This is followed by the same non-linear optimization step to refine the plane equations and focal lengths while using both orthogonality and articulation constraints together.

Before texture mapping and rendering the complete 3D model, the extents of each planar segment must be determined. Ideally, a planar segment's extents should contain the entire planar region regardless of occlusion or noise, and should be bordered by the articulation lines with adjacent planar segments. The union of oriented rectangles belonging to each planar segment can be considered as its extents. However, due to occlusions, noisy line detection, or imperfect oriented rectangle conflict removal, the polygon denoting the union of oriented rectangles may have holes in it. Furthermore, the polygon may not extend to the lines common to the planar segment and its adjacent segments.

We recover planar extents by performing hole-filling on each planar segment in turn. Given a plane segment and its common line, we stretch each of the planar segment's oriented rectangles towards the extended common line until it goes beyond any of the common lines of the planar segment or hit an oriented rectangle belonging to another segment. This process is repeated for each extended common line of the planar segment. The union of all stretched rectangles denote the recovered extents of the planar segment. The recovered planar extents are used to texture map and render the final 3D reconstruction.

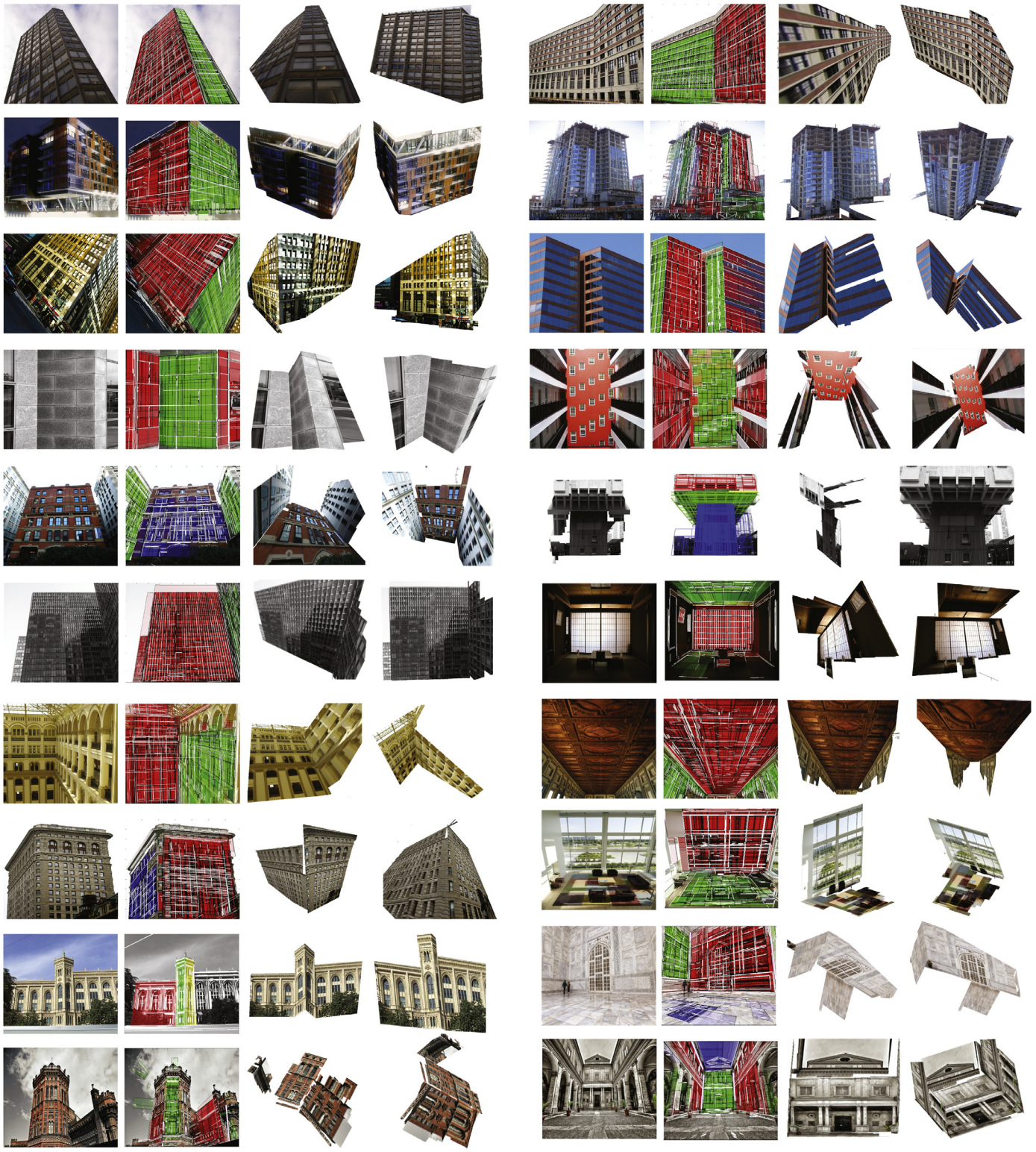


Fig. 16. End-to-end automatic reconstruction results: Each set shows an input image, segmentation and two novel viewpoints. The images include a typical set of indoor and outdoor urban environments. Since the proposed algorithm does not require visibility of floor/ceiling/ground planes, it is able to reconstruct the scenes even when such horizontal planes are not visible.

4. Results and discussion

We perform qualitative and quantitative evaluation of the complete algorithm as well as the important components of the pipeline. Additionally, we also compare the algorithm with state of the art in automatic segmentation.

4.1. Qualitative evaluation

First we show results of the complete Shape from angle regularity algorithm on a wide range of images which includes both indoor and outdoor scenes with widely varying viewpoints and structure regularity. It is followed by a demonstration of the 2D rectification

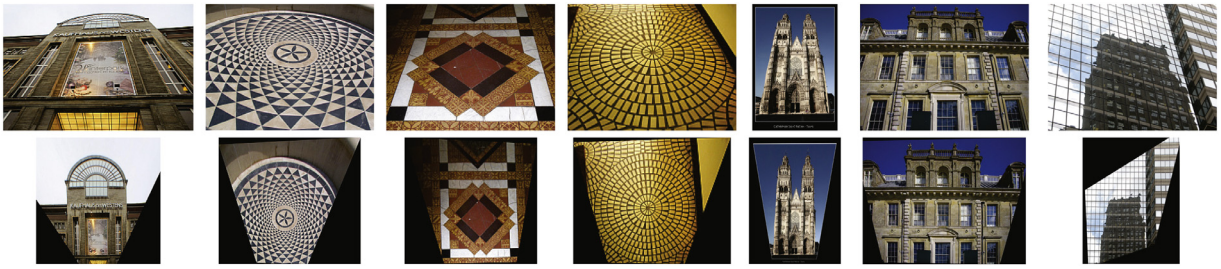


Fig. 17. Automatic rectification results (bottom row) on challenging images taken from the Internet (top row). EXIF data was available, hence two-parameter optimization was used. Note that even when lines do not align to common vanishing points, as in the circular tiled patterns, the algorithm works because it exploits orthogonalities locally. All these images contain line-pairs that meet at the same regular angle locally while the global structure varies widely between images.

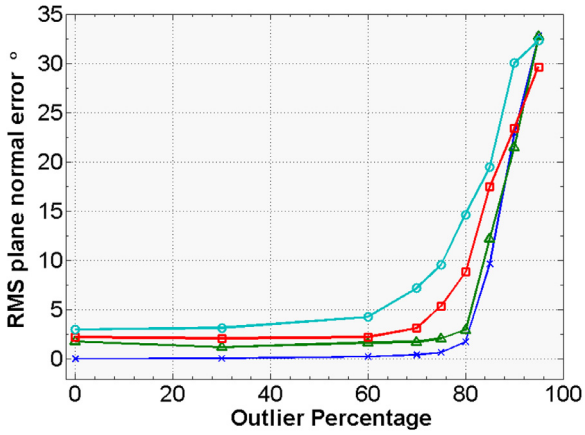


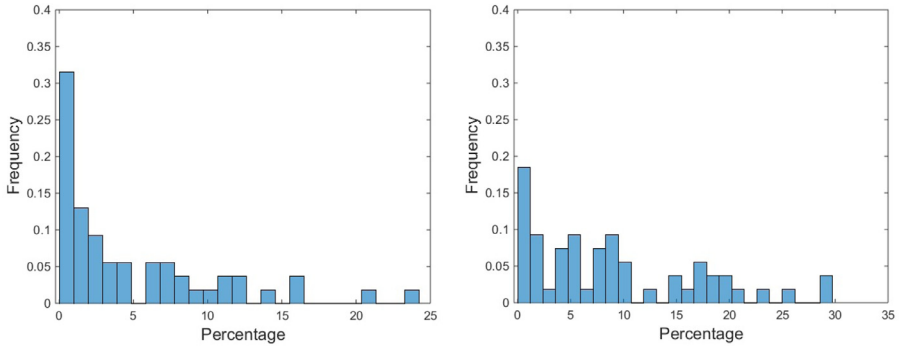
Fig. 18. Analysis of the rectification error in presence of noise using synthetic data. We plot the percentage of outliers on the horizontal axis while the vertical axis shows the error in plane normal. Outlier pairs were generated with uniformly random angles while Gaussian synthetic noise of various degrees (0° - blue, 1° - green, 2° - red, and 4° - cyan curve) was added in the inliers. The results remain stable and accurate with over 70% outliers. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

algorithm on similarly diverse images of planar scenes. For qualitative evaluation of the proposed method, we built a dataset of over 120 images from the Internet, mostly from FilckR. This dataset includes images from both indoor and outdoor scenes with 3D structures including Manhattan model, ground-vertical model and ground plane being visible or invisible. This is the first dataset with this level of variance in 3D structure and camera viewpoints used for Single View Reconstruction.

Fig. 1 shows segmentation and reconstruction on some uniquely challenging images that demonstrate the key difference between our approach and earlier work. The images in first and third rows do not follow the geometric models of any of the state-of-the-art automatic SVR algorithms. The first image contains a non-traditional structure whose details have been correctly segmented and reconstructed by our approach. The third example violates ground-vertical assumption required by earlier SVR algorithms, and also contains sky and clutter which has been correctly filtered out. The second example is that of a line sketch. In this case, three parameter search was carried out to recover focal length, and correct structure recovery is illustrated. While the line sketch almost follows a Manhattan model with only one plane not being parallel to the principal planes, it does not have any texture and therefore a texture based approach like Photo Popup, TILT or Barinova et al. is not the solution. The last row shows result on a Rubik's Cube which would not be reconstructed by any SVR algorithm that imposes an indoor or outdoor layout simply because of the top plane. While the Rubik's cube is a simple Manhattan structure but the outdoor algorithms assume that ground plane is the only horizontal plane visible while the indoor algorithms assume that floor and ceiling are the only horizontal planes visible and the camera is placed between them. These examples demonstrate the flexibility of the angle regularity model as well as the wide applicability of the proposed algorithm.

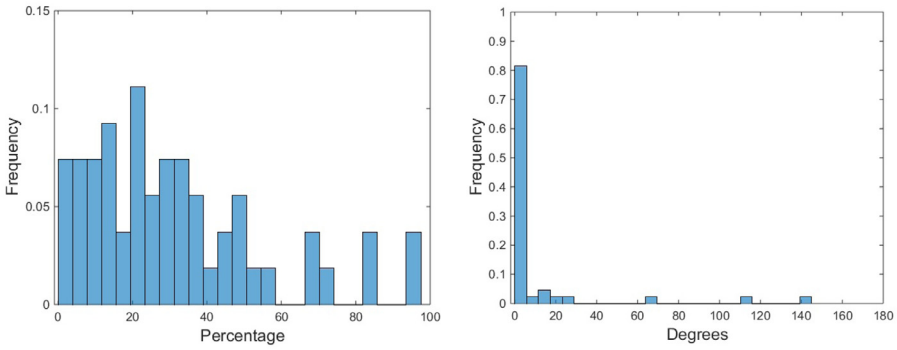
Some typical test cases are shown in Fig. 16 with the input image, multi-planar segmentation and two novel views of the reconstructed model for each image. It contains several indoor and outdoor scenarios with various amounts of clutter. Many of them do not contain a visible ground plane, floor or ceiling as required by other SVR approaches. Typically, the proposed algorithm can reconstruct scenes reasonably well as long as enough orthogonal line-pairs are available on each plane that needs to be reconstructed.

Finally, we show some results of 2D rectification on a very small subset of interesting images containing planar scenes in Fig. 17. It includes three images that follow a regular grid scenario, on the right, that can be handled by orthogonal vanishing points approach. However, rest of the images would not be rectified by the two orthogonal



(a) Bad rectangles not removed (b) Good rectangles removed

Fig. 19. Behavior Analysis of Rectangle Consensus Algorithm 2: Each image contains hundreds of rectangle hypotheses and the Rectangle Consensus algorithm tries to remove most of the bad hypotheses at the cost of removing some good hypotheses. The histogram on the left shows that majority of the images have less than 3% remaining bad rectangle hypotheses when compared with the ground truth. However, a larger percentage of good rectangle hypotheses is removed making the distribution on the right more even.



(a) Plane Segmentation Error (b) Plane normal error

Fig. 20. Error in plane segmentation was computed by comparing the final segmentation to ground truth segmentation marked manually for each image and the error histogram is shown in (a). We manually marked some mutually orthogonal and parallel plane-pairs and used them as ground truth for plane normals. The histogram of error in inter-plane angles in final reconstruction (after nonlinear optimization) is shown in (b).

vanishing points approach. The first image on left does contain two orthogonal vanishing points but many lines converge on a finite point which would be detected as a vanishing point too. The second and fourth images from the left have many orthogonal line-pairs that are not arranged according to principal directions. Similarly some of the images, when rectified, show no repetition/symmetry about a principal direction which is required by TILT based rectification to work.

4.2. Quantitative evaluation

We begin with a robustness analysis of plane pose estimation against presence of non-orthogonal line-pairs as well as line detection errors which add noise to the angle between line-pairs. We synthetically generate 50 line-pairs which include inliers with noise and outliers, that is, non-orthogonal line-pairs. Zero mean Gaussian noise with standard deviation ranging between 0° to 4° is added in the inliers, while outliers are taken to be at uniform random angles. The whole set of lines was projected to a randomly oriented image plane and rectified using our plane-pose estimation algorithm. The results were averaged over 200 random trials of the described setup. Fig. 18 shows the resulting error in the plane normals. The algorithm works exceptionally well and can successfully rectify data with even 70% outliers in the presence of Gaussian noise in the inliers. This robust plane pose estimation algorithm forms the core component of our Single View Reconstruction approach.

For real data, we tested the algorithm on a diverse set of images collected from the Internet but their actual 3D structures are not available to us. Therefore, we use some proxy tests instead of the direct comparison of reconstructed depths with a ground truth depth. For this purpose, we manually mark the segmentation on each image and compute errors in our automatic segmentation. Similarly, we manually figure out some known angles between plane-pairs and then report the error observed in inter-planar angles in our reconstructions. Since the 3D structure is completely defined using multi-planar segmentation and plane normals, therefore the stability these two components provides ample evidence for the stability of the reconstruction itself.

In order to test the accuracy of segmentation, we automatically compute the good rectangles using our conflict removal method and compare the results with the ground truth segmentation of the image. If any rectangle has at least 10% overlap with a conflicting ground truth orientation, it is considered to be a bad rectangle, and vice versa. This way we compute all the ground truth good rectangles that have been removed incorrectly by our algorithm as well as the ground truth bad rectangles not removed. The percentage of these errors is then computed for every image. A histogram of these percentages per image is shown in Fig. 19. Our approximate algorithm understandably removes some good rectangles but generally avoids almost all bad rectangles. Additionally, some planes may not have been detected at all if they did not have enough orthogonal line-pairs. Therefore, we have small false positives - where a wrong plane membership has been assigned to a pixel - but may have a large number of false negatives - where a pixel is not assigned any plane membership - resulting in an accurate but incomplete segmentation. To quantify the overall segmentation error, we compare our final automatic segmentation to the ground truth segmentation which shows higher error due to incompleteness of the segmentation. The results are summarized in Fig. 20(a).

The key strength of the proposed method is the ability to reconstruct widely varying scenes in both indoor and outdoor scenarios, arbitrary plane orientations and atypical camera viewpoints. It depends on a robust plane pose estimation algorithm which generates plane orientation hypotheses for each plane, independently of other planes in the scene. In order to evaluate our pose estimation on real data, we manually mark out known orthogonal and parallel plane-pairs in the dataset. These ground truth angles are then compared to the reconstructed inter-planar angles after nonlinear optimization and angle error distribution is shown in Fig. 20(b). Over 80% of the plane-pairs have an error less than 5° showing stable reconstruction across all different scenarios.

4.3. Comparisons

Fig. 1 shows some scenarios where the 3D scene does not follow any restrictive models of previous algorithms. It may be unfair to compare the previous algorithms on the kind of scenes that they do not claim to reconstruct. The main characteristics of such scenes include the 3D structure not following the Manhattan or ground-vertical assumption; invisibility of floor/ceiling/ground plane; unavailability of texture in line sketches; and atypical camera viewpoints. Instead, we also chose image subsets that do follow the assumptions in previous literature and compare all the algorithms on each of those subsets.

In Fig. 21, each row contains two images from “Inside the Box”, “Manhattan Indoor” and “Ground-Vertical” models respectively. We also added two images of vertical walls where the ground plane is not visible to showcase the over-dependence of some algorithms on the ground plane visibility. Each image follows ground truth segmentation and results of the compared algorithms. We used publicly available implementations of the corresponding papers by their authors. For Lee et al. (2009) implementation of their segmentation algorithm is only available with their subsequent work by Gupta et al. (2010) so we used that code instead.

4.4. Limitations

Shape from angle regularity is the first SVR algorithm to work across a wide set of scenarios because it does not assume restricted plane orientations or multi-planar layout. However, this also introduces two key limitations. 1) *More data:* In case of Manhattan world assumption, recovering any one plane orientation is enough to recover other two plane orientations even in the absence of any cues on those planes. However, the proposed algorithm requires line-pairs on each

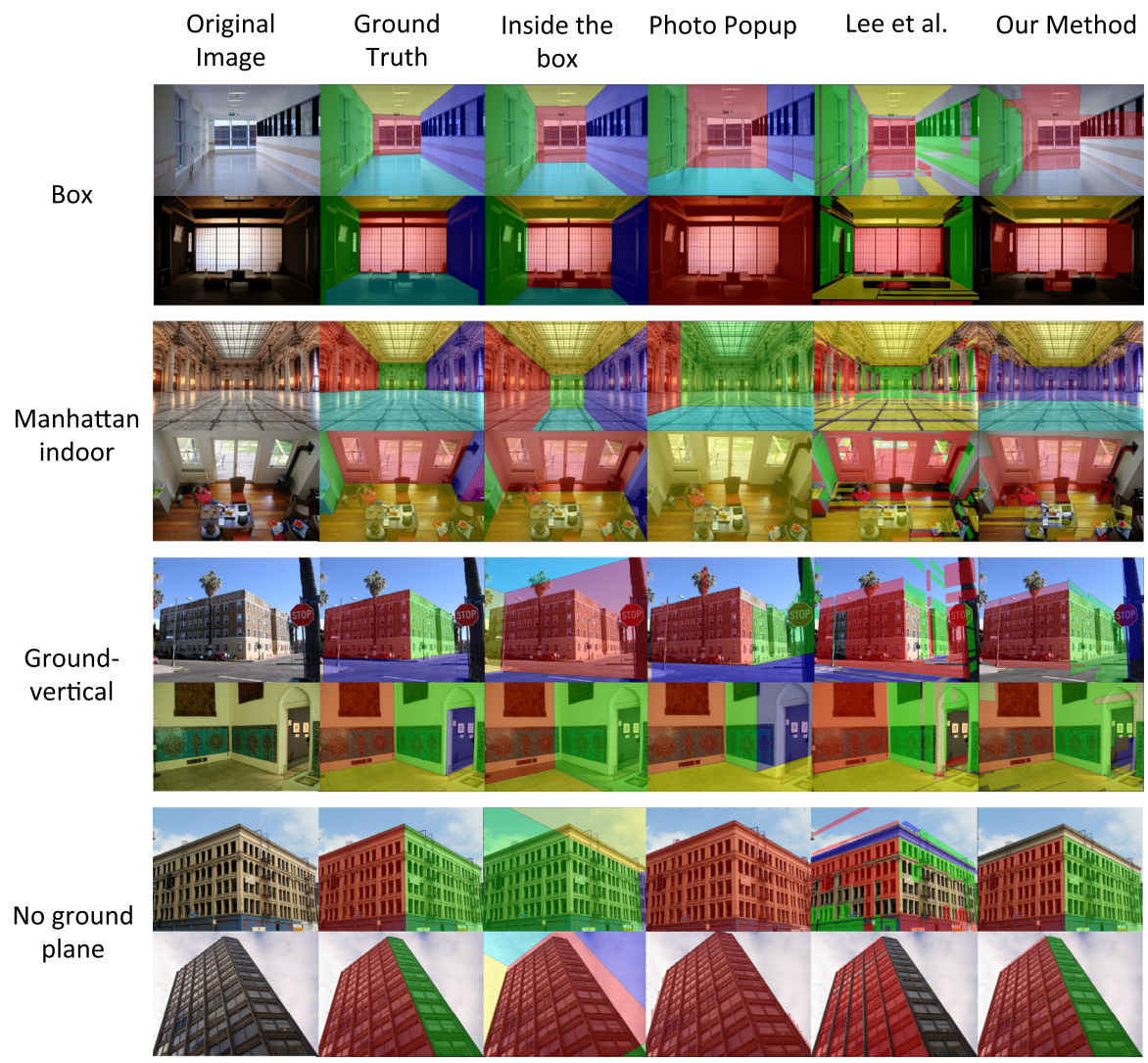


Fig. 21. Segmentation comparison with various algorithms and scenarios: Each row represents an image in restricted scenario where previous algorithms should perform better. The columns show input image and ground truth followed by results obtained from each algorithm i.e. *Inside the Box* (Hedau et al., 2010), *Photo Popup* (Hoiem et al., 2005), Lee et al. (2009), and our method. Pixels are color coded according to their plane membership.

plane to recover the plane orientation as it does not assume a dependence of orientations between different planes. 2) *Segmentation completeness:* In addition to missing plane orientations, the proposed algorithm discards some correct rectangle hypotheses. Since it does not assume a restrictive multi-planar layout such as "Box", indoor, or outdoor, it cannot "complete" the segmentation by explicitly joining planar segments where they may or may not meet. Therefore, not all pixels in the image are assigned plane memberships.

4.5. Conclusion

In this paper, we proposed a line-based SVR method for the urban environments that have local angle regularity and consist of a 2.5D connected set of planes with arbitrary plane orientations. We have shown that the profusion of orthogonal angles between adjacent line-pairs in man-made scenes is an important structure cue, which we have exploited in all the steps of the SVR pipeline. Not only are orthogonal angles a robust feature for plane rectification, they are also a significant cue for multi-planar segmentation. The complete SVR system presented here makes less restrictive assumptions about scene geometry and camera orientation than existing papers, and is able to robustly reconstruct multi-planar man-made scenes under 2.5D assumption.

Acknowledgments

This research was supported by Higher Education Commission, Pakistan, through Indigenous PhD Scholarships program, and Lahore University of Management Sciences through PhD student research funding.

We thank Mr. Khurram Amin, Mr. Ali Hassan and Dr. Yaser Sheikh for reviewing earlier versions of this paper and sharing their invaluable insights. We are also indebted to the journal reviewers for helpful suggestions in improving the manuscript.

References

Almansa, A., Desolneux, A., Vamech, S., 2003. Vanishing point detection without any a priori information. IEEE Trans. Pattern Anal. Mach. Intell. 25 (4), 502–507.
Barinova, O., Konushin, V., Yakubenko, A., Lee, K., Lim, H., Konushin, A., 2008. Fast automatic single-view 3-d reconstruction of urban scenes. European Conference on Computer Vision. Springer, pp. 100–113.
Barinova, O., Lempitsky, V., Tretiak, E., Kohli, P., 2010. Geometric image parsing in man-made environments. European Conference on Computer Vision. pp. 57–70.
Bartoli, A., Sturm, P., 2003. Constrained structure and motion from multiple uncalibrated views of a piecewise planar scene. Int. J. Comput. Vis. 52 (1), 45–64.
Delage, E., Lee, H., Ng, A.Y., 2006. A dynamic Bayesian network model for autonomous 3d reconstruction from a single indoor image. Computer Vision and Pattern Recognition. Vol. 2. IEEE, pp. 2418–2428.
Doersch, C., Singh, S., Gupta, A., Sivic, J., Efros, A., 2012. What makes paris look like

Paris? ACM Trans. Graph. 31 (4).
Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24 (6), 381–395.
Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R., 2009. Reconstructing building interiors from images. 12th International Conference on Computer Vision. IEEE, pp. 80–87.
Gupta, A., Hebert, M., Kanade, T., Blei, D.M., 2010. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. Advances in Neural Information Processing Systems. pp. 1288–1296.
Hartley, R.I., Zisserman, A., 2004. Multiple View Geometry in Computer Vision, 2nd. Cambridge University Press ISBN: 0521540518.
Hedau, V., Hoiem, D., Forsyth, D., 2009. Recovering the spatial layout of cluttered rooms. 12th International Conference on Computer Vision. IEEE, pp. 1849–1856.
Hedau, V., Hoiem, D., Forsyth, D., 2010. Thinking inside the box: using appearance models and context based on room geometry. European Conference on Computer Vision. pp. 224–237.
Hoiem, D., Efros, A.A., Hebert, M., 2005. Automatic photo pop-up. ACM Trans. Graph. 24 (3), 577–584.
Horry, Y., Anjyo, K.-I., Arai, K., 1997. Tour into the picture: using a spidery mesh interface to make animation from a single image. Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques. ACM Press/Addison-Wesley Publishing Co., pp. 225–232.
Kanade, T., 1980. A theory of origami world. Artif. Intell. 13 (3), 279–311.
Kang, H.W., Pyo, S.H., Anjyo, K.-i., Shin, S.Y., 2001. Tour Into the Picture Using a Vanishing Line and its Extension to Panoramic Images. In: Computer Graphics Forum, Vol. 20 no. (3). Wiley Online Library, 132–141.
Lee, D.C., Hebert, M., Kanade, T., 2009. Geometric reasoning for single image structure recovery. Computer Vision and Pattern Recognition. IEEE, pp. 2136–2143.
Liebowitz, D., Criminisi, A., Zisserman, A., 1999. Creating architectural models from images. Computer Graphics Forum. 18. Wiley Online Library, pp. 39–50.
Liebowitz, D., Zisserman, A., 1998. Metric rectification for perspective images of planes. Computer Vision and Pattern Recognition. IEEE, pp. 482–488.
Pan, J., Hebert, M., Kanade, T., 2015. Inferring 3d layout of building facades from a single image. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2918–2926.
Park, S., Lee, H., Lee, S., Yang, H.S., 2015. Line-based single view 3d reconstruction in manhattan world for augmented reality. Proceedings of the 14th ACM SIGGRAPH International Conference on Virtual Reality Continuum and its Applications in Industry. ACM, pp. 89–92.
Ramalingam, S., Brand, M., 2013. Lifting 3d manhattan lines from a single image. Proceedings of the IEEE International Conference on Computer Vision. pp. 497–504.
Saxena, A., Sun, M., Ng, A.Y., 2009. Make3d: learning 3d scene structure from a single still image. IEEE Trans. Pattern Anal. Mach. Intell. 31 (5), 824–840.
Stella, X.Y., Zhang, H., Malik, J., 2008. Inferring spatial layout from a single image via depth-ordered grouping. Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE, pp. 1–7.
Sturm, P., Maybank, S., 1999. A method for interactive 3d reconstruction of piecewise planar objects from single images. The 10th British Machine Vision Conference (BMVC'99). The British Machine Vision Association (BMVA), pp. 265–274.
Von Gioi, R.G., Jakubowicz, J., Morel, J.-M., Randall, G., 2010. Lsd: a fast line segment detector with a false detection control. EEE Trans. Pattern Anal. Mach. Intell. 32 (4), 722–732.
Wang, X., Fouhey, D., Gupta, A., 2015. Designing deep networks for surface normal estimation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 539–547.
Wilczkowiak, M., Sturm, P., Boyer, E., 2005. Using geometric constraints through parallelepipeds for calibration and 3d modeling. IEEE Trans. Pattern Anal. Mach. Intell. 27 (2), 194–207.
Wildenauer, H., Vincze, M., 2007. Vanishing point detection in complex man-made worlds. International Conference on Image Analysis and Processing. IEEE, pp. 615–622.
Zaheer, A., Khan, S., 2014. 3d metric rectification using angle regularity. Winter Conference on Applications of Computer Vision. IEEE.
Zaheer, A., Rashid, M., Khan, S., 2012. Shape from angle regularity. European Conference on Computer Vision. Springer.
Zhang, Z., Ganesh, A., Liang, X., Ma, Y., 2012. Tilt: transform invariant low-rank textures. Int. J. Comput. Vis. 99 (1), 1–24.