Geometric Reasoning for Single Image Structure Recovery

David C. Lee
Dept. of Electrical and Computer Engineering
Carnegie Mellon University
dclee@cs.cmu.edu

Martial Hebert   Takeo Kanade
Robotics Institute
Carnegie Mellon University

Abstract

We study the problem of generating plausible interpretations of a scene from a collection of line segments automatically extracted from a single indoor image. We show that we can recognize the three dimensional structure of the interior of a building, even in the presence of occluding objects. Several physically valid structure hypotheses are proposed by geometric reasoning and verified to find the best fitting model to line segments, which is then converted to a full 3D model. Our experiments demonstrate that our structure recovery from line segments is comparable with methods using full image appearance. Our approach shows how a set of rules describing geometric constraints between groups of segments can be used to prune scene interpretation hypotheses and to generate the most plausible interpretation.

1. Introduction

It is easy for us to recognize the structure in Figure 1, as well as locate a few doors. However, automatic recognition of structure from a collection of line segments is challenging, as not all lines defining the building structure are perfectly detected by low level image processing. To further complicate the problem, extra edges may lie on surfaces of walls or even on objects that are not part of the target structure (Figure 2). We can still interpret the collection of line segments because 1) we perform geometric reasoning and only consider physically plausible interpretations, 2) we have the ability to look globally at the overall structure, and 3) we have prior knowledge on how the world, in our case the interior of a building, is structured.

As images are projections of the real world, it is desirable to interpret them only in ways which can be realized in the real world. Geometric inference, when jointly done with semantic labeling, may be more demanding, but it may significantly reduce the problem space and make the problem, in fact, easier.

In this paper, we tackle the problem of interpreting collection of line segments to recognize the structure of buildings. We search for building models that translate to physically plausible three dimensional building models. We perform geometric reasoning to generate many physically valid structure hypotheses from line segments. Each hypothesis is tested to find the one that best matches the collection of line segments. We have also done preliminary experiments to detect objects, using the recovered structure as a "scene frame", which provides geometric context to objects in the scene.

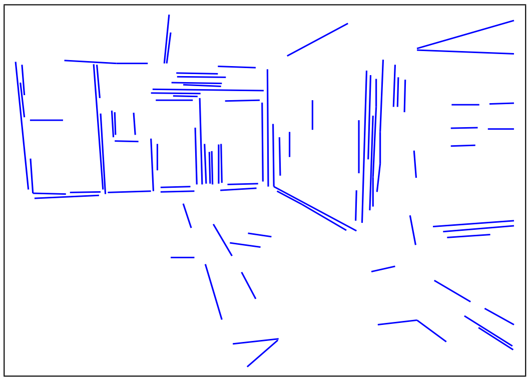


Figure 1. Line segments. Can you recognize the building structure? Can you find doors?

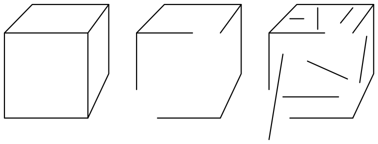


Figure 2. Levels of completeness of line drawings. Left: Complete. Middle: Missing. Not all structure edges in the real world are present in the image. Right: Missing and Spurious. Not all lines in the image are structure edges or even part of the target structure.

2. Prior Work

Line drawings have been studied from the early days of computer vision. Guzman [8] was the first to interpret line drawings to separate collection of polyhedral objects into parts. Huffman [12] and Clowes [1] came up with a formal scheme of labeling lines into convex, concave, and occluding for polyhedral objects, with which 3D description of objects can be recovered and impossible objects can be rejected. Mackworth [17] introduced the concept of gradient space and surface based constraints. Waltz [24] expanded the problem by allowing line drawings to include shadows, cracks, and missing edges (Figure 2). Kanade [13] dealt with "origami world", which includes hollow shells and planar sheets, and utilized heuristics, such as parallel lines in image are parallel in space. Sugihara [23] provided an algebraic optimization approach for interpreting line drawings. However, these approaches were limited to synthetic line drawings and were not applied to real images.

Kosecka's group have a number of papers on images of the Manhattan world by using information from line segments. Kosecka and Wei [14] developed a method to recover vanishing points and camera parameters from a single image by using line segments found in Manhattan structures. Using the recovered vanishing points, rectangular surfaces aligned with major orientations were detected by Wei and Kosecka [15] and more recently by Micusik *et al.* [18]. Han and Zhu [9] have also worked on finding rectangles aligned with vanishing points from line segments. They used top-down grammars, which helped finding rectangles forming regular patterns, such as grid or box patterns. However, these approaches operate directly in 2D image space (except when multiple images were used) and do not attempt to extract three dimensional information from a single image.

A number of papers address the problem of recovering three dimensional structure from a single image. Three dimensional information can be extracted from a single image when there is a reference in the image [3]. A commonly used reference is the ground plane. Hoiem *et al.* [10] and Delage *et al.* [6] take a two-step approach for recovering 3D structure of outdoor images and indoor images respectively: 1) estimate image region orientation (e.g., ground, vertical) using statistical methods on image properties, such as color, texture, edge orientation, position in image, etc. 2) "pop-up" vertical regions by "folding" along the crease between ground and vertical regions. Saxena *et al.* have taken a different approach by estimating absolute depth directly from image properties [21], and smoothly connecting regions under weak assumptions, such as connectivity or coplanarity, without the explicit assumption of a ground plane [22].

An interesting observation was made by Nedovic *et al.* [19] that a typical scene can be categorized into a limited number of categories of 3D scene geometry, which they call "stages". Categories of stages include sky+ground, box, corner, and person+background, and the stage information can potentially serve as a guide for a more complete depth estimation or a more detailed scene understanding.

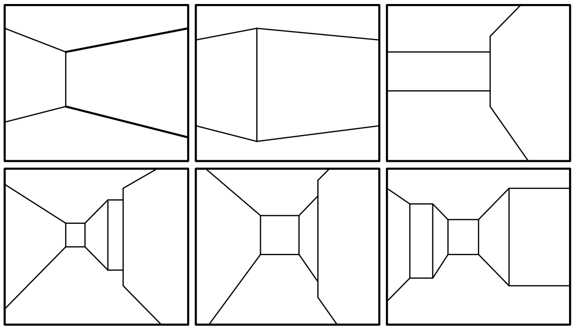


Figure 3. Examples of building models under Indoor World model. All building models are built by connecting three basic types of corners. Top left: concave(-) corner. Top middle: convex(+) corner. Top right: occluding(>) corner. Bottom row: combinations of corners.

3. Indoor World Model

Most indoor environments satisfy the Manhattan World assumption [2], i.e., most planes lie in one of three mutually orthogonal orientations. In addition, indoor environments usually have a single floor plane and a single ceiling plane with constant ceiling height. Combining the "Manhattan World" and "single-floor single-ceiling" models, we propose the "Indoor World" model as an useful approximation for indoor scenes.

This world model applies to most indoor environments and has a number of desirable properties. First of all, it is easy to represent a physically valid model of a scene in two dimensional image space, which can be effortlessly translated into a three dimensional model. By geometric reasoning on the configuration of edges, we can represent a scene structure in two dimensions that encodes a physically valid three dimensional structure. Examples of such representation of scenes are depicted in 3.

Another desirable property is the symmetry that it introduces between the shape of the ceiling and the floor. Building models under this assumption have symmetric floor and ceiling shape. Evidence to infer building structure from a single image mostly comes from the position of boundaries between planes, but floor-wall boundaries are often occluded by objects such as desks, chairs, and bookcases, as shown in Figure 14. Even in those cases, ceiling-wall boundaries are rarely occluded, so observing ceiling-wall boundaries and assuming symmetry between them allows us to infer the location of floor-wall boundaries.

4. Geometric Reasoning

As the world is made up of solid objects, projections of the world onto an image obey a set of rules. In particular, projections of buildings under the Indoor World assumption are geometrically constrained by a small set of rules defined on connection of walls, which we define as corners. An indoor scene can be fully represented by corners, so geometric constraits on corners will guarantee the entire structure to be valid.

There are three types of corners: convex(+), concave(-), and occluding($>$). A convex(+) or concave(-) corner is formed when two walls meet at one place in 3D space and an occluding($>$) corner is formed when one wall is in front of another wall but appears to be adjacent in the image. The type and position of a corner is constrained depending on where the corner is in the image.

The simplest constraint on a corner is that it should consist of two junctions, one above the horizon and one below the horizon. This rule holds because the camera itself is between the floor and the ceiling. Regions divided by vertical vanishing lines also create constraints. In each of the three regions divided by two vertical vanishing lines, only a total of four types of corners can exist, as illustrated in Figure 4. These rules are derived from facts about the physical world and geometry, such as, the camera must be in an empty quadrant of a wall in order for it to be able to observe the corner, and walls should have non-zero thickness.

These constraints are simple to adhere to, even at an early stage of inference when no consideration about the 3D coordinates are made. Also, they can be applied only to local and primitive corner structures, even when no consideration about the global structure of the scene has been made. Yet, performing geometric reasoning according to these constraints will guarantee that our entire building model encodes a valid model, which can be easily converted to a valid 3D model without ambiguity.

5. Finding Building Structure

Finding the building structure is done in three steps; line segments and vanishing points are found, many plausible building model hypotheses are created, and each hypothesis is tested against an orientation map, which is a map of local belief of region orientations, to find the best matching hypothesis. Each step will be explained in detail in the following sections.

5.1. Line Segment Detection and Vanishing Point Estimation

We extract line segments using the Matlab toolbox by Kovesi [16], which runs Canny edge detector, links edge pixels, and fits line segments. We then recover vanishing points from these line segments.

From the three vanishing points, we can recover the orientation of the three axes of the building in the camera coordinate by formulas in Appendix. This allows us to reconstruct an accurate 3D model, even when none of the camera axes are aligned with world coordinates.

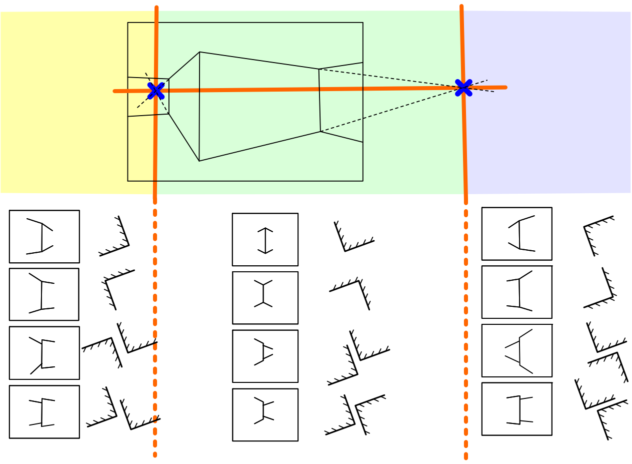


Figure 4. Regions divided by vanishing lines and restrictions on types of corners. Top: Line drawing, vanishing points, and vanishing lines. Bottom: Types of possible corners in each of the three regions. Enclosed in small boxes are depictions of corners as they would appear in the image, and next to it are the top-down view of each corners. In each of the three regions, four types of corners can exist: one convex(+), one concave(-), and two occluding($>$) corners.

We loosely follow Rother [20] to find three orthogonal vanishing points. Two pairs of lines are randomly sampled in RANSAC fashion and the intersection of each pair of lines generates a candidate vanishing point. Orthogonality of the two vanishing points is verified using formulas in Appendix and the third vanishing point is computed to be orthogonal to the two vanishing ponts. Then the three candidates are evaluated using the cost function proposed in [20]. Finally, the x, y coordinates of the best RANSAC solution are fine tuned using non-linear optimization (Matlab fminsearch) with the same cost function. To ensure orthogonality under optimization, vanishing points are translated into a rotation matrix, which can then be parameterized with three unbounded parameters using Rodrigues' formula [7]. The highly non-convex nature of the cost function is not a big issue, as the RANSAC solution was already close to the true solution.

For uncalibrated images with no available camera intrinsic parameters, three pairs of lines are sampled to create a proposal, and orthogonality is loosely enforced by constraining three vanishing points to be apart from each other. Once three vanishing points are found in image space, the focal length of the camera can be recovered by finding a focal length that makes the angles exactly 90 degrees.

In practice, this method returned vanishing points within a few pixels of the true vanishing points for all 102 test images when camera parameters were available, and 40 out of 44 images when camera parameters were not available. It failed when there were no lines in one of the three direction, or when many lines were not in the principal directions.

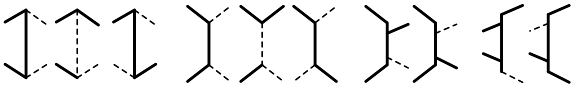


Figure 5. Solid lines are the minimal set of lines needed to define a corner. Three lines are needed for convex(+) and concave(-) corners. Four lines are needed for occluding(>) corners.

5.2. Generating Building Hypotheses

For this and the following section, we define "orientation of a line segment" to be the orientation of the line in the world, which can be estimated by the vanishing point that lies on the extension of the line segment in the image. Similarly, "parallel" line segments means parallel in the world. "Orientation of a surface' is defined as the normal orientation of the surface in the world and "pixel orientation" as the orientation of the surface projected to the pixel.

Building models can be generated by connecting line segments to create corners, and connecting corners to create building models. A corner consists of five lines, but not all five lines need to be present to define a corner. Concave(-) and convex(+) corners need three lines, and occluding(>) corners need four lines to be defined (Figure 5). A new corner is proposed when a minimal set of lines defines a corner, while obeying the constraints on corners described in Section 4.

The process of generating hypotheses is illustrated in Figure 6. We start by creating building hypotheses with zero corners, i.e., scenes with just one wall. Two parallel line segments, one above the horizon and one below the horizon, are extended until the image boundaries to define the floor-wall and ceiling-wall boundary of a wall. Next, we search for line segments that can be extended to "attach" to existing walls to propose a new corner. Note that an existing wall already defines two lines, so only one additional line need to be added to propose a concave(-) or a convex(+) corner, and two for an occluding(>) corner. By repeatedly attaching more corners to an existing structure, we can create a scene with many corners. This process is described in Algorithm 1.

5.3. Evaluating Building Hypotheses

We test all building hypotheses to find the best fitting hypothesis to a given collection of line segments. This is done by evaluating the fitness of hypotheses to an orientation map (Figure 7), which is a map that expresses the local belief of region orientations computed from line segments. The fitness of a hypothesis to an orientation map is defined as the total number of pixels which the orientation agrees between that encoded by the hypothesis and that given by the orientation map. The hypothesis with the largest fitness is chosen as the best fitting hypothesis.

Two line segments having different orientation supporting a pixel is a strong indication of the pixel orientation to be perpendicular to the orientation of the two lines. For example, we, as human, believe pixel **(1)** in Figure 7(a) is on a horizontal surface because a green line above it and a blue line to the right supports pixel **(1)** to be perpendicular to the orientation of both lines. Pixel **(2)** seems to be on a vertical surface because green lines above and below and red lines to the left support it. Notice that, although there is a blue line below pixel **(2)**, its support is blocked by the green line between the blue line and the pixel. The support of a line extends until it hits a line which has the same orientation as the normal orientation of the surface it is supporting. This is because a line can not be on a plane that is perpendicular to it. This logic usually produces accurate orientation map, except around occluding boundaries.

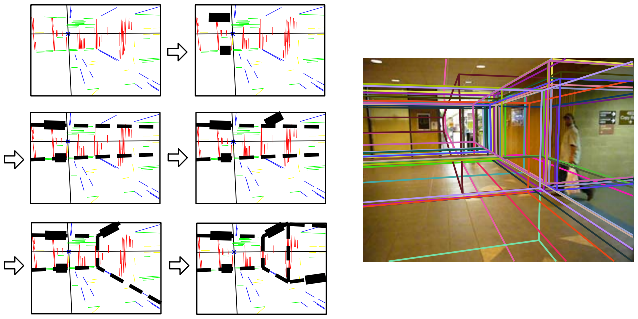


Figure 6. Generating hypotheses. Left: The process of a hypothesis being generated by four line segments. Right: A sample of generated building hypotheses.

Algorithm 1 Generating building hypotheses

```
Set $H_0 \leftarrow \emptyset$, where $H_0$ is the set of hypotheses with zero corners.
for all pair of line segments $(l_i, l_j)$ do
    if $l_i$ above horizon $\wedge$ $l_j$ below horizon $\wedge$ $l_i$ and $l_j$ have overlap then
        Add scene with no corner $(l_i, l_j)$ to $H_0$
    end if
end for
for $k = 1$ to $n$, where $n$ is maximum number of corners in scene do
    Set $H_k \leftarrow \emptyset$, where $H_k$ is the set of hypotheses with $k$ corners.
    for all $h \in H_{k-1}$ do
        Find sets of lines that create corners that attaches to $h$ and satisfies geometric constraints.
        $H' \leftarrow$ Set of all scenes with a new corner attached to $h$
        $H_k \leftarrow H_k \cup H'$
    end for
end for
return $H \leftarrow H_0 \cup H_1 \cup \cdots \cup H_n$
```

More formally, let $L_x = \{l_{x,1}, l_{x,2}, \cdots, l_{x,n_x}\}$ be the set of line segments of orientation x, where $x \in \{1,2,3\}$ denotes the one of the three orientations. A "sweep" $S(l_{x,i}, v_y, \alpha)$ of a line $l_{x,i}$ towards vanishing point v_y by amount α is the set of pixels that is supported by line $l_{x,i}$ to be orientation z (Figure 8). x, y, and z take values in $\{1,2,3\}$ and all three should be different ($x \neq y$, $x \neq z$, and $y \neq z$).

Given a line segment $l_{x,i}$ with end points p_1 and p_2, $S(l, v_y, \alpha)$ is the convex hull created by p_1, p_2, p_1', and p_2', where p_1' and p_2' is given by

$$p_1' = p_1 + \alpha(v_y - p_1),$$

$$p_2' = \mathit{intersection}(\mathit{line}(v_x, p_1'), \mathit{line}(v_y, p_2)),$$

where $\mathit{line}(\cdot,\cdot)$ denotes a line passing through two points and $\mathit{intersection}(\cdot,\cdot)$ denotes the point of intersection of two lines.

The sweep extends until the sweep region contains a line that "blocks" the sweep. The amount of sweep $\widehat{\alpha}_{x,i}$ and $-\widehat{\beta}_{x,i}$, towards and away from its sweep direction is:

$$\widehat{\alpha}_{x,i} = \mathit{max}(\alpha), \widehat{\beta}_{x,i} = \mathit{max}(\beta),$$

such that $\alpha \geq 0$, $\beta \geq 0$, and no lines in L_z intersect $S(l_{x,i}, v_y, \alpha)$ and $S(l_{x,i}, v_y, -\beta)$.

The set of pixels that is supported by all lines in L_x swept towards v_y to be orientation z is:

$$P_{x,y,z} = \bigcup_{l_{x,i} \in L_x} S(l_{x,i}, v_y, \widehat{\alpha}_{x,i}) \cup S(l_{x,i}, v_y, \widehat{\beta}_{x,i}).$$

A pixel is believed to have orientation z when two lines of different orientation x and y support the pixel, and only when it is exclusively supported to be z. The final orientation map O_z for orientation z is given by:

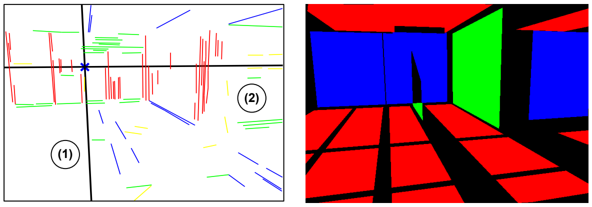
$$R_z = P_{x,y,z} \cap P_{y,x,z}$$

$$O_z = R_z \cap R_x^c \cap R_y^c.$$

Figure 7(b) shows O_1, O_2, and O_3 colored in red, green, and blue.

5.4. Converting Building Models to 3D

Two dimensional building model hypotheses always encode valid 3D models, so computing 3D coordinates can be done easily without ambiguity. 3D coordinates can be computed sequentially for floor, then walls using the constraint that floor and walls are connected, and finally the ceiling, using formulas in Appendix. However, small errors can accumulate during this sequential process, and we follow Delage *et al.* [5] to globally minimize the distances between connected planes using linear programming. Recovered 3D models are visualized in Figure 9.



(a) (b)

Figure 7. Line segments and Orientation map. (a) Line segments, vanishing points, and vanishing lines. (b) Orientation map. Lines segments and regions are colored according to their orientation. (Best viewed in color)

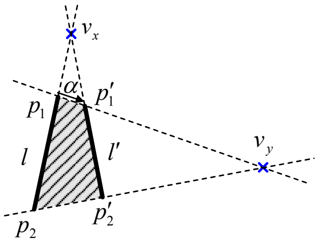


Figure 8. The shaded area denotes the sweep $S(l, v_y, \alpha)$ of line l towards vanishing point v_y by amount α, and it potentially supports the region to be orthogonal to v_x and v_y.

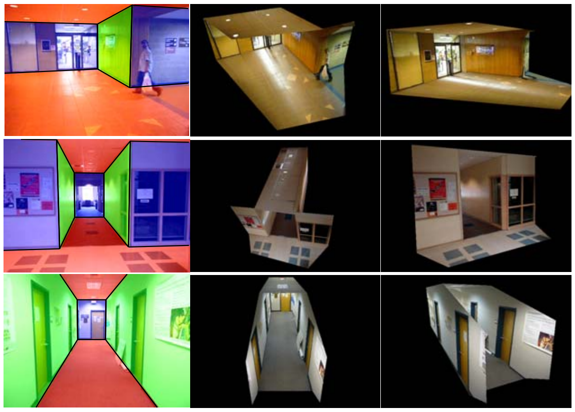


Figure 9. 3D models

6. Experiments

We have collected 54 images of indoor scenes. We have also included objects in the image that obstruct the view of the scene frame. We have manually labeled the ground truth orientation for every pixel, ignoring the occluding objects. The percentage of pixels that have the correct orientation for each image is reported in Figure 10. On average, 81% of the pixels were classified correctly. 76% of the images had less than 30% misclassified pixels, and 44% had less than 10% misclassified pixels. Qualitatively, around 70%

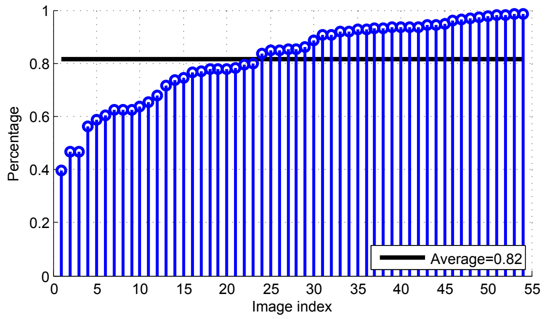


Figure 10. Percentage of pixels with correct orientation.

of the images returned acceptable 3D models. Notice that even when objects occlude the floor-wall boundary, the underlying building structure could be recovered (Figure 14). In these cases, the unobstructed view of the ceiling-wall boundary have helped finding the underlying building structure. Typical failure cases are: hallways being cut off early when there are no lines supporting down the hallway, missing corners, or misaligned boundaries (Figure 15).

We have compared our results with other works on recovering indoor structure from a single image. We had comparable results as Delage *et al*. [6], with their experimental setup and dataset, which had 48 images of indoor campus scenes. RMS error between the estimated and ground truth floor boundary was measured in pixel space, and is plotted as a function of the position of the true floor boundary (Figure 11). Comparing with Hoiem *et al*. [11], using their classifier trained for indoor images, we have a higher percentage of correctly classified pixel orientation on 20 out of 48 images, and a mean percentage of 80% versus 87%. In both cases, our results are comparable, while relying only on line segments and not on image properties such as colors and image gradients, which can be scene specific.

We have also tested on the 44 images downloaded from the web, also collected by Delage *et al*. Qualitatively, around 20 of them returned acceptable 3D models. Failures were due to many objects that cluttered the scene, and scenes that do not match our building model. Sample results are shown in Figure 16.

7. Populating the Scene Frame with Objects

Now that we have the scene structure, we would like to use it as a "frame" that defines the scene, and populate it with objects in the scene. Recovering the "scene frame" is a stepping stone toward a more complete scene understanding, as it provides a global geometric context of the scene. Our ultimate goal is to recognize all the objects in a scene. Most objects of interest fall into one of the two categories: objects that lie on the floor, and objects that are attached to a wall. Objects that lie on the floor interacts with the scene frame by being supported at the point it contacts the floor of the frame, which determines its 3D location. These objects need to be in an empty space of the frame, and not inside walls. Locations of objects attached to walls are also constrained by the scene frame. We have done preliminary experiments on one object for each category, namely people and doors. Doors were found by finding rectangles generated by line segments which have the correct size and are attached to walls. People were found by running a publically available detector by Dalal and Triggs [4] and pruning out ones with incorrect size, and ones inside walls. Two examples are shown in Figure 12. Eventually, we would like to tie structure recovery and object recognition into a unified framework for a complete scene interpretation.

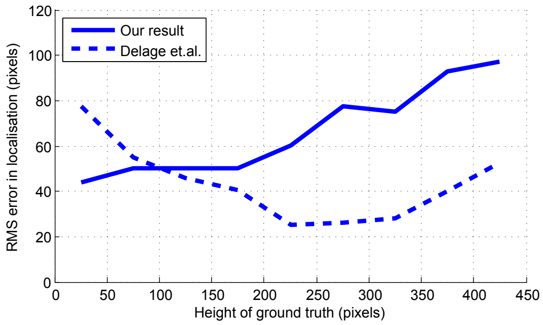


Figure 11. Comparison of floor boundary error

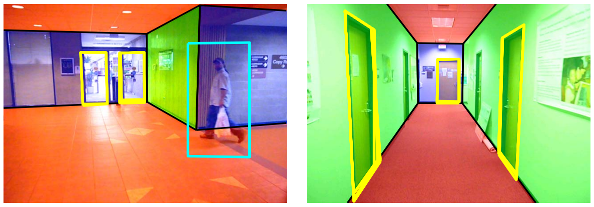


Figure 12. Examples of doors and people in a scene frame.

8. Conclusion

We have proposed a framework to interpret collection of line segments to recover three dimensional building structure. We have shown that, by geometric reasoning, and by using the prior knowledge of indoor environments, we can recover the structure of a building, using only line segments. An interesting future problem would be to use our recovered structure as a "scene frame" to recognize more components in the scene and step towards the grand goal of complete scene interpretation.

9. Acknowledgements

This work is based on work supported by the National Science Foundation under Grant No. EEEC-0540865.

References

[1] M. B. Clowes. On seeing things. In *Artificial Intelligence*, 1971.

[2] J. Coughlan and A. Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *Proceedings ICCV*, 1999.

[3] A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. In *Proc. International Conference on Computer Vision (ICCV)*, 1999.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of IEEE Conference Computer Vision and Pattern Recognition*, 2005.

[5] E. Delage, H. Lee, and A. Y. Ng. Automatic single-image 3d reconstructions of indoor manhattan world scenes. In *ISRR*, 2005.

[6] E. Delage, H. Lee, and A. Y. Ng. A dynamic bayesian network model for autonomous 3d reconstruction from a single indoor image. In *CVPR*, 2006.

[7] O. Faugeras and Q.-T. Luong. *The Geometry of Multiple Images*. The MIT Press, 2001.

[8] A. Guzman. Decomposition of a visual scene into three-dimensional bodies. In *Proceedings of Fall Joint Computer Conference*, 1968.

[9] F. Han and S. Zhu. Bottom-up/top-down image parsing by attribute graph grammar. In *Proc. Int'l Conf. on Computer Vision (ICCV)*, 2005.

[10] D. Hoiem, A. Efros, and M. Hebert. Geometric context from a single image. In *Proceedings of IEEE Conference Computer Vision and Pattern Recognition*, 2005.

[11] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. In *CVPR*, 2006.

[12] D. A. Huffman. Impossible objects as nonsense sentences. In *Machine Intelligence*, 1971.

[13] T. Kanade. A theory of origami world. In *Artificial Intelligence*, 1980.

[14] J. Kosecka and W. Zhang. Video compass. In *Proceedings of European Conference on Computer Vision*, pages 657 – 673, 2002.

[15] J. Kosecka and W. Zhang. Extraction, matching and pose recovery based on dominant rectangular structures. *Computer Vision Image Understanding*, 2005.

[16] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia. Available from: <http://www.csse.uwa.edu.au/∼pk/research/matlabfns/>.

[17] A. K. Mackworth. Interpreting pictures of polyhedral scenes. In *Artificial Intelligence*, 1973.

[18] B. Micusik, H. Wildenauer, and J. Kosecka. Detection and matching of rectilinear structures. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[19] V. Nedovic, A. W. Smeulders, and A. Redert. Depth information by stage classification. In *Proc. International Conference on Computer Vision*, 2007.

[20] C. Rother. A new approach for vanishing point detection in architectural environments. In *BMVC*, pages 382–391, 2000.

[21] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *In Neural Information Processing Systems (NIPS)*, 2005.

[22] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Transactions of Pattern Analysis and Machine Intelligence (PAMI)*, 2008.

[23] K. Sugihara. A necessary and sufficient condition for a picture to represent a polyhedral scene. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI*, 1984.

[24] D. A. Waltz. Generating semantic descriptions from line drawings of scenes with shadows. Technical report, MIT, 1972.

Appendix: Formulas for 2D to 3D conversion

All units of metrics are in camera height, i.e., the distance between the floor and the camera measured perpendicular to the floor equals 1, since absolute distances can not be measured from images. Lower case: 2D homogeneous coordinates. Upper case: 3D coordinates. Vanishing points with subscript 1 (v_1, V_1) indicates the vertical vanishing point. K: camera intrinsic parameter matrix

- Ray
$$P = \lambda K^{-1} p, \qquad \lambda > 0$$
- Normal direction of the three major axes given coordinates of three vanishing points (x_k, y_k) in image.
$$v_k = (x_k, y_k, 1)^T \quad \Leftrightarrow \quad V_k = \frac{K^{-1} v_k}{\|K^{-1} v_k\|_2}$$
- 3D coordinate of a point on the floor. Note that the height is normalized to 1.
$$P = \frac{K^{-1} p}{V_1^T K^{-1} p}$$
- Height h between two points p_1 and p_2, with p_1 being a point on the floor. p_1, p_2, and v_1 should roughly be in line when applying this formula, as we assume P_1 and P_2 are vertically aligned in 3D.
$$\begin{aligned} P_2 &= \lambda K^{-1} p_2 \\ &= P_1 + h V_1 \\ &= \frac{K^{-1} p_1}{V_1^T K^{-1} p_1} + h V_1 \end{aligned}$$
$$\begin{bmatrix} -V_1 & K^{-1} p_2 \end{bmatrix} \begin{bmatrix} h \\ \lambda \end{bmatrix} = \frac{K^{-1} p_1}{V_1^T K^{-1} p_1}$$
Solving least-squares gives h.

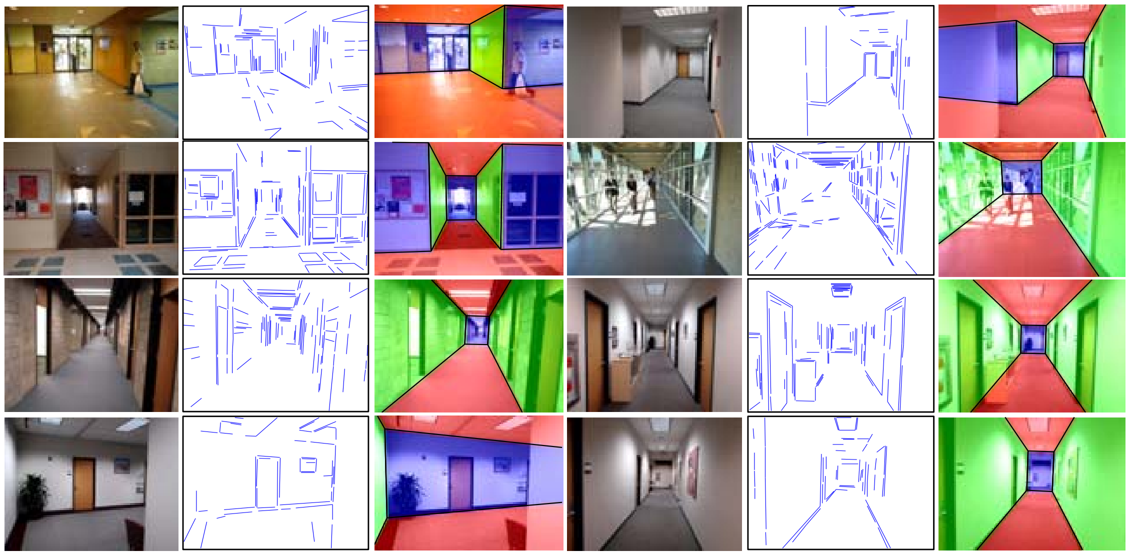


Figure 13. Examples (Best viewed in color)

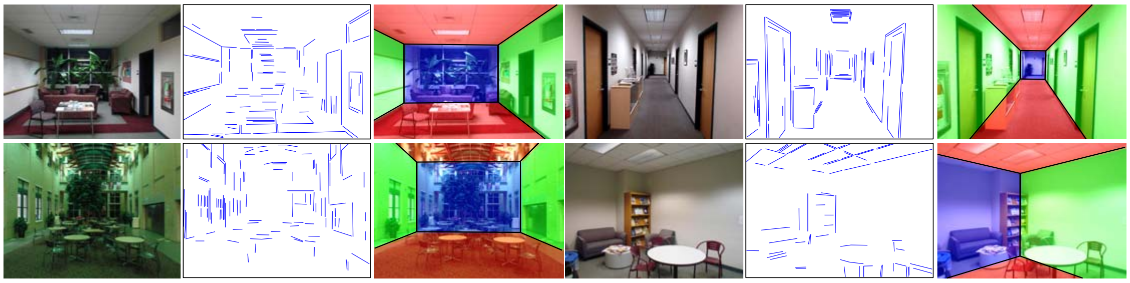


Figure 14. Examples with occluding objects. Unobstructed view of the ceiling-wall boundary helps finding the underlying building structure. (Best viewed in color)

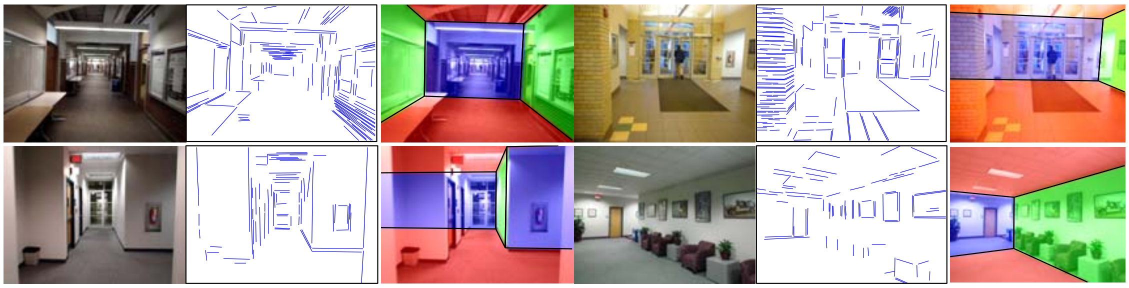


Figure 15. Failure examples. (Best viewed in color)

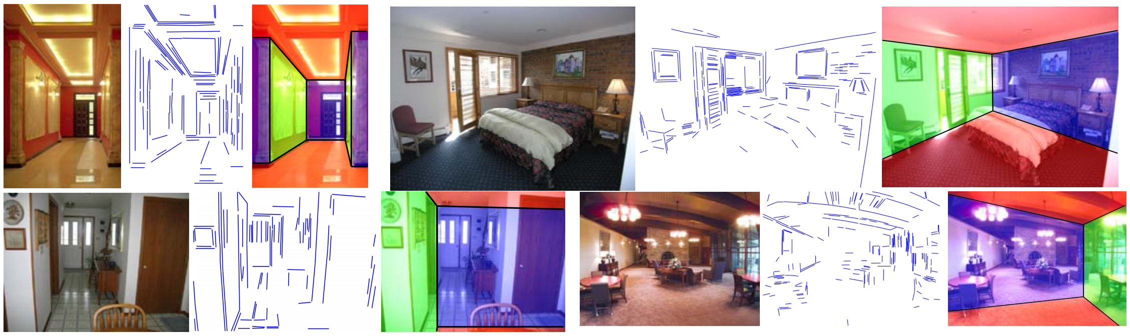


Figure 16. Examples of images downloaded from the web. Top row: Success. Bottom row: Failure. (Best viewed in color)