# On the Uniqueness of Textureless Layers

**Qifa Ke, Simon Baker, and Takeo Kanade**

Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

**Abstract**

Without any assumptions, reconstructing the 3D shape of constant intensity regions is (almost always) inherently ambiguous. It is therefore natural to explore which priors can be used to break the ambiguity. One choice used in the "layers" literature is that the scene is (approximately) piecewise planar. In this paper we investigate whether the reconstruction of constant intensity regions is unique or not when we assume that the scene is piecewise planar. We assume that each of the constant intensity regions corresponds to a planar segment or "Textureless Layer" and show that generally, although not always, the problem in still ambiguous.

**Keywords:** Textureless layers, constant intensity regions, uniqueness, inherent ambiguities.

# Contact Information

**Contact Name:** Qifa Ke

**Contact Address:** Carnegie Mellon University, 5000 Forbes Avenue, Robotics Institute - NSH 4122, Pittsburgh, PA 15213, USA

**Contact Email:** qifa.ke@cs.cmu.edu

**Contact Phone:** +1 (412) 268-6569

**Contact Fax:** +1 (412) 268-5576

**Author Emails:** qifa.ke@cs.cmu.edu, simonb@cs.cmu.edu, tk@cs.cmu.edu

# 1 Introduction

Without prior (or silhouette) information, reconstructing the 3D shape of constant intensity regions is inherently ambiguous [1]. A natural question is then: "what priors can be used to break the ambiguity?" One common choice in the "layers" literature [3–5, 7–9] is that the scene is (approximately) piecewise planar. In most layers papers, however, it is assumed that there is enough texture in each layer to uniquely compute a homography or affine warp for that layer and to assign all the pixels to the correct layer. In man-made scenes, however, there are often many textureless regions. Walls are usually painted a single color and the tops of tables are usually textureless.

In this paper we address the theoretical question of whether the reconstruction of such scenes unique if we assume that the scene is planar in the constant intensity regions. In particular, we consider scenes consisting of a finite collection of constant intensity planar patches. In [1] a relatively simple mathematical criterion was given for when the reconstruction was unique and when it was ambiguous. In this paper, we propose a search algorithm that determines whether the reconstruction is unique or not, and if it is ambiguous, enumerates the ambiguities.

By applying this algorithm to appropriate scenes, we are then able to answer such questions as: "is the reconstruction ever ambiguous?" (yes it is), "is the reconstruction ever unique?" (yes it is), "is the reconstruction usually ambiguous?" (yes it is), and "roughly how many solutions are there for typical scenes?" (in the order of hundreds or more). The practical implication for computer vision is then that additional assumptions are needed to uniquely recover the shape of scenes with "Textureless Layers." We end by describing several possible alternatives.

# 2 Problem Statement and Notation

Assume that the scene is imaged simultaneously by a set of $m$ (stereo) cameras with projection matrices $\mathbf{P}^1, \ldots, \mathbf{P}^m$ which capture the images $I^1, \ldots, I^m$. As in [1] we assume that the image measurements are fully calibrated geometrically. If the system is not calibrated (as in many papers), the reconstruction will only be more ambiguous. See Figure 1 for an illustration.

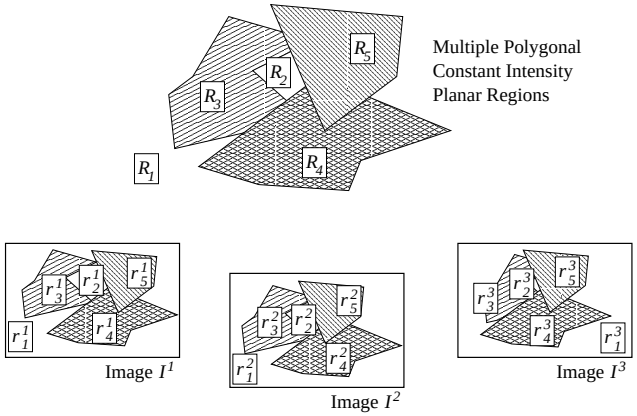



**Figure 1:** Problem Scenario. We assume that the scene consists of a collection of $n$ 3D constant intensity, polygonal planar regions $R_1, \ldots, R_n$, and is imaged by $m$ cameras $\mathbf{P}^1, \ldots, \mathbf{P}^m$ which capture the images $I^1, \ldots, I^m$. The $i^{\text{th}}$ constant intensity region in the $p^{\text{th}}$ image is denoted $r_i^p$.

Also assume that the scene consists of a collection of $n$ constant intensity, 3D polygonal[1] planar regions $R_1, \ldots, R_n$. Denote the 2D projections of these regions in the $p^{\text{th}}$ image $r_1^p, \ldots, r_n^p$. The boundaries of the 3D regions $R_1, \ldots, R_n$ are a set of 3D lines $L_i$, represented by $(\mathbf{F}_i, \mathbf{S}_i)$, where $\mathbf{F}_i$ and $\mathbf{S}_i$ are column vectors containing the 3D coordinates of two points on the line. Each 3D line $L_i$ borders two 3D regions $R_j$ and $R_k$. This information is recorded in the data-structure:

$$\text{Bordering}(L_i) \;=\; \{R_j, R_k\}. \tag{1}$$

Denote the 2D projection of the $i^{\text{th}}$ 3D line in the $p^{\text{th}}$ image $l_i^p = (\mathbf{f}_i^p, \mathbf{s}_i^p)$ where $\mathbf{f}_i^p$ is a column vector containing the 2D image coordinates of the first vertex, and $\mathbf{s}_i^p$ is a column vector containing the 2D image coordinates of the second vertex. Note that because of occlusions, the points, $\mathbf{f}_i^p$ and $\mathbf{f}_i^q$, and the points, $\mathbf{s}_i^p$ and $\mathbf{s}_i^q$, do not necessarily correspond to the same 3D points for $p \neq q$.

Implicit in the above two paragraphs is the assumption that the distance between the cameras is small enough that the topology of the lines and regions does not change across the input images; i.e. no lines or regions appear or disappear across the images $I^1, I^2, \ldots, I^m$. We also assume that the depth ordering of the regions remains the same across the images. This is hardly an additional assumption since the ordering would only change if the camera moved from "in front

[1]It is possible to generalize the analysis in this paper to arbitrary shaped regions. Each region just needs to be bounded by a collection of bounding curves (some of which may be straight lines). The plane equation of a region in Equation (3) *may sometimes* be defined by a single curve rather than 2 lines. Otherwise, the analysis is very similar.

of" to "behind" some of the planar regions, in which case the topology would also likely change.

A practical "Textureless Layers" algorithm would need to segment the images $I^p$ into 2D polygonal regions $r_i^p$, find their 2D bounding lines $l_i^p$, match corresponding 2D lines and regions, determine the 3D lines $L_i = (\mathbf{F}_i, \mathbf{S}_i)$ using stereo, and then compute $\mathrm{Bordering}(L_i)$. In this paper, we ask (supposing all this can be done), whether the scene shape is still inherently ambiguous or not?

# 3 Determining Uniqueness

We now describe how to determine whether the "Textureless Layers" problem is unique or not. In particular, we present an algorithm to enumerate all the solutions. If there is just one solution, the problem is unique. If there are multiple solutions, the problem is ambiguous. Our approach parallels traditional layers algorithms [4, 5, 7–9] which consist of iterating two tasks: (1) assigning pixels to layers and (2) estimating the motion [3] or plane equation [2] of each layer. In the "Textureless Layers" problem there are two corresponding tasks: (1) assigning 3D lines to layers (instead of assigning pixels to layers) and (2) computing the plane equation of each layer.

## 3.1 Layer Assignment

Suppose that $\mathrm{Bordering}(L_i) = \{R_j, R_k\}$. There are three possible physical causes of the line $L_i$: (1) the region $R_j$ is in front of and occluding the region $R_k$ in which case $R_j$ contains $L_i$ but $R_k$ does not, (2) the region $R_k$ is in front of and occluding the region $R_j$ in which case $R_k$ contains $L_i$ but $R_j$ does not, and (3) the two regions $R_j$ and $R_k$ meet at and both contain the line $L_i$. There are therefore three ways to assign the 3D line $L_i$ to the regions $R_j$ and $R_k$:

$$\mathrm{Assign}(L_i) \;=\; \begin{cases} \{R_j\} & \text{if only } R_j \text{ contains } L_i \\ \{R_k\} & \text{if only } R_k \text{ contains } L_i \\ \{R_j, R_k\} & \text{if } R_j \;\&\; R_k \text{ meet at } L_i. \end{cases} \tag{2}$$

In Section 3.3 we describe how $\mathrm{Assign}(L_i)$ can be computed once the plane equations are known.

## 3.2 Layer Plane Equation

Ideally we would like to estimate a plane equation for each region $R_i$. Unfortunately this is not always possible. To compute a plane equation we need at least two lines assigned to $R_i$. If less than 2 lines are assigned to $R_i$ we cannot uniquely compute the plane equation. If one line is assigned to $R_i$ we can only constrain the plane of $R_i$ by that line. If no lines are assigned to $R_i$ the plane is unconstrained. Let $\boldsymbol{\pi}_i$ denote the plane equation of $R_i$ where:

$$\boldsymbol{\pi}_i \;=\; \begin{cases} (\mathbf{n}_i, d_i) & \text{if two lines assigned to } R_i \\ L_j & \text{if one line } L_j \text{ assigned to } R_i \\ \emptyset & \text{if zero lines assigned to } R_i \;. \end{cases} \tag{3}$$

In this definition $\mathbf{n}_i$ is a (column) vector normal to the plane and $d_i$ is the distance to the plane (both defined up to scale); i.e. the fully constrained plane equation is defined by $(x\ y\ z)\,\mathbf{n}_i + d_i = 0$. In Section 3.4 we describe how $\boldsymbol{\pi}_i$ can be computed once the layer assignment is known.

## 3.3 Assigning Lines to Layers

Suppose that $\text{Bordering}(L_i) = \{R_j, R_k\}$. If the plane equations of the two layers $R_j$ and $R_k$ are known, it is possible to estimate $\text{Assign}(L_i)$. If $L_i$ lies in the plane of $R_j$, then $R_j \in \text{Assign}(L_i)$, and similarly for $R_k$. What follows is a description for $R_j$. The same rules apply to $R_k$. If $\boldsymbol{\pi}_j$ is fully defined and equals $(\mathbf{n}_j, d_j)$, the layer assignment $\text{Assign}(L_i)$ can be computed using:

$$\text{if } \begin{pmatrix} \mathbf{F}_i^{\mathrm{T}} & 1 \\ \mathbf{S}_i^{\mathrm{T}} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{n}_j \\ d_j \end{pmatrix} \;=\; \mathbf{0} \;\text{ then } R_j \in \text{Assign}(L_i) \tag{4}$$

where $L_i = (\mathbf{F}_i, \mathbf{S}_i)$. If $\boldsymbol{\pi}_j$ is just defined by one line $L_l$, the assignment can be computed:

$$\text{if } L_i \;=\; L_l \;\text{ then } R_j \in \text{Assign}(L_i). \tag{5}$$

If the plane $\boldsymbol{\pi}_j$ is unconstrained and equals $\emptyset$, the assignment can be computed:

$$\text{if } \boldsymbol{\pi}_j = \emptyset \text{ then } R_j \notin \text{Assign}(L_i). \tag{6}$$

## 3.4 Computing Layer Plane Equations

Consider the region $R_i$ and the set of lines $L_{j_1}, \ldots, L_{j_q}$ assigned to $R_i$:

$$R_i \in \text{Assign}(L_{j_k}) \text{ for } k = 1, \ldots q. \tag{7}$$

The plane equation $\boldsymbol{\pi}_i$ of $R_i$ can then be computed. First it is checked whether $q = 0$. If $q = 0$ then $\boldsymbol{\pi}_i = \emptyset$. Second it is checked if all of the lines $L_{j_k}$ are the same line; i.e. co-linear. If all of the lines $L_{j_k}$ are co-linear then $\boldsymbol{\pi}_i = L_{j_1}$. Finally, if there are more than two distinct lines, $\boldsymbol{\pi}_i = (\mathbf{n}_i, d_i)$ is computed as follows. The $q$ lines $L_{j_1}, \ldots, L_{j_q}$ are defined by the $2 \times q$ 3D points, $\mathbf{F}_{j_1}, \mathbf{S}_{j_1}, \ldots, \mathbf{F}_{j_q}$, $\mathbf{S}_{j_q}$. Since all of these points must lie in $\boldsymbol{\pi}_i$ it follows that:

$$\begin{pmatrix} \mathbf{F}_{j_1}^{\mathrm{T}} & 1 \\ \mathbf{S}_{j_1}^{\mathrm{T}} & 1 \\ \vdots & \vdots \\ \mathbf{F}_{j_q}^{\mathrm{T}} & 1 \\ \mathbf{S}_{j_q}^{\mathrm{T}} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{n}_i \\ d_i \end{pmatrix} \equiv \mathbf{A}_i \begin{pmatrix} \mathbf{n}_i \\ d_i \end{pmatrix} = \mathbf{0}. \tag{8}$$

The plane equation $\boldsymbol{\pi}_i = (\mathbf{n}_i, d_i)$ is then computed with a Singular Value Decomposition on $\mathbf{A}_i$.

## 3.5 Layer Consistency

Given a potential solution to the "Textureless Layers" problem (i.e. a set of layer assignments and plane equations) we need a layer consistency function that determines whether this solution is valid or not. There are two components to layer consistency: (1) local consistency, i.e. for each region, the plane equation is consistent with the assignment of lines to that region, and (2) depth ordering,

i.e. the depth ordering implied by the plane equations is consistent with the occlusion ordering implied by the assignment of lines to regions. We now discuss each in turn.

### 3.5.1 Local Consistency

Local consistency means that the plane equation for each region $R_i$ is consistent with the 3D line equations of all of the lines $L_j$ assigned to $R_i$. Checking local consistency means checking:

- For all $j$: $\text{Assign}(L_j) \neq \emptyset$ and $\text{Assign}(L_j) \subseteq \text{Bordering}(L_j)$.
- For all $i$, $j$, $R_i \in \text{Assign}(L_j)$, $L_j = (\mathbf{F}_j, \mathbf{S}_j)$:
  - If $\boldsymbol{\pi}_i = (\mathbf{n}_i, d_i)$ is defined by two lines then

$$\begin{pmatrix} \mathbf{F}_j^{\mathrm{T}} & 1 \\ \mathbf{S}_j^{\mathrm{T}} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{n}_i \\ d_i \end{pmatrix} = \mathbf{0}. \tag{9}$$

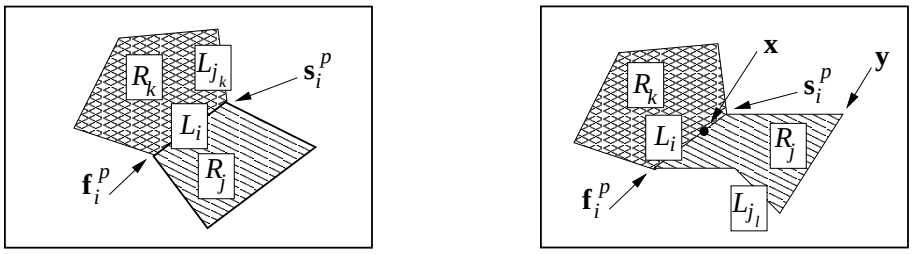
  - If $\boldsymbol{\pi}_i = L_{j_i}$ is defined by one line then $L_{j_i} = L_j$.
  - If $\boldsymbol{\pi}_i = \emptyset$ is defined by zero lines then the solution is locally inconsistent.

### 3.5.2 Depth Ordering

Suppose $L_i \in \text{Assign}(R_j)$ where $\text{Bordering}(L_i) = \{R_j, R_k\}$. For the depth ordering to be correct, the plane of $R_j$ must be closer to (or at the same distance from) the camera than the plane of $R_k$ along the line $L_i$. It is easiest to check this condition for each region $R_j$ in turn. Depending whether the plane equation $\boldsymbol{\pi}_j$ of $R_j$ is defined by two, one, or zero lines, there is a different condition.

#### Planes Defined by Two Lines

Suppose that the plane equation of $R_j$ is defined by two lines; $\boldsymbol{\pi}_j = (\mathbf{n}_j, d_j)$. We consider each line $L_i$ for which $R_j \in \text{Bordering}(L_i)$. If $R_k \in \text{Bordering}(L_i)$ is the other region that borders $L_i$ the situation is as in Figure 2(a). The two points $\mathbf{f}_i^p$ and $\mathbf{s}_i^p$ are the end points of the corresponding 2D line $l_i^p$ in image $I^p$. Although there are three possibilities for $\text{Assign}(L_i)$, we only need to check:




(a) Depth Ordering "Two Lines" (b) Depth Ordering "One Line"

**Figure 2:** Depth ordering consistency checks for planes defined by (a) two lines and (b) one line.

1. $R_j \in \text{Assign}(L_i)$: The plane of $R_j$ must lie in front of the plane of $R_k$ along $L_i$.

2. $R_k \in \text{Assign}(L_i)$: The plane of $R_k$ must lies in front of the plane of $R_j$ along $L_i$.

The third case that both $R_j, R_k \in \text{Assign}(L_i)$ is taken care of by checking both conditions. How these conditions are checked depends on how $\boldsymbol{\pi}_k$, the plane equation of $R_k$, is defined:

1. If $\boldsymbol{\pi}_k = (\mathbf{n}_k, d_k)$ is defined by two lines, the depth ordering of the planes is determined by checking the depth ordering along the rays defined by the two points $\mathbf{f}_i^p$ and $\mathbf{s}_i^p$. See [6] for the details of how to do this.

2. Suppose $\boldsymbol{\pi}_k = L_{j_k}$ is defined by one line. If $\mathbf{f}_i^p$ lies on $l_{j_k}^p$ the projection of $L_{j_k}$ into $I^p$ we check the depth ordering of the plane $\boldsymbol{\pi}_j$ and the line $L_{j_k}$ along the ray defined by $\mathbf{f}_i^p$. Similarly if $\mathbf{s}_i^p$ lies on $l_{j_k}^p$ we check the depth ordering of the plane $\boldsymbol{\pi}_j$ and the line $L_{j_k}$ along the ray defined by $\mathbf{s}_i^p$. If $L_{j_k}$ equals $L_i$ then both of these conditions are satisfied and so the depth must be checked for both $\mathbf{f}_i^p$ and $\mathbf{s}_i^p$. If neither $\mathbf{f}_i^p$ nor $\mathbf{s}_i^p$ lie on $L_{j_k}$ then nothing needs to be checked. Again, see [6] for a description of how to check the depth along a given ray. Figure 2(a) illustrates the case that only $\mathbf{s}_i^p$ lies on the projection of $L_{j_k}$ into $I^p$.

3. If $\boldsymbol{\pi}_k = \emptyset$ is defined by zero lines, there is nothing to do to check that $R_j$ lies in front of $R_k$.

**Planes Defined by One Line**

Suppose that the plane equation $\boldsymbol{\pi}_j = L_{j_l}$ of $R_j$ is defined by one line. We then consider each line $L_i$ for which $R_j \in \text{Bordering}(L_i)$. If $R_k \in \text{Bordering}(L_i)$ is the other region that borders $L_i$ the

situation is as in Figure 2(b). The 2D lines $l_i^p$ and $l_{j_l}^p$ are then intersected to give the pixel $\mathbf{x}$. If $\mathbf{x}$ lies between the two end points $\mathbf{f}_i^p$ and $\mathbf{s}_i^p$ of the 2D line $l_i^p$, the depth ordering check for "planes defined by two lines" is performed for the point $\mathbf{x}$ (rather than $\mathbf{f}_i^p$ and $\mathbf{s}_i^p$.) See [6] for the details. If $L_i = L_{j_l}$ the depth ordering check should be made for both $\mathbf{f}_i^p$ and $\mathbf{s}_i^p$.

The depth ordering check described above checks that every point on the 3D line $L_{j_l}$ is correctly ordered with respect to the neighboring planes. Since the plane $\boldsymbol{\pi}_j = L_{j_l}$ is just defined by one line, it could be any plane "rotated" about that line. We also need to check that there is a "rotation" that is consistent with the depth ordering implied by the layer assignment. In particular, consider Figure 2(b) where the plane equation of $R_j$ is defined by the one line $\boldsymbol{\pi}_j = L_{j_l}$. Consider the point $\mathbf{s}_i^p$. Since $L_i \neq L_{j_l}$ then $\text{Assign}(L_i) = \{R_k\}$. We therefore know that the region $R_k$ is in front of $R_j$ at the point $\mathbf{s}_i^p$. This puts a constraint on the rotation of the plane of $R_j$ about $\boldsymbol{\pi}_j = L_{j_l}$. We determine whether all of these constraints can be simultaneously satisfied as follows.

Consider any vertex $\mathbf{y}$ of $R_j$ that does not lie on the 2D line $l_{j_l}^p$ corresponding to $L_{j_l}$. Then consider every line $L_i$ that borders $R_j$ that does not equal $L_{j_l}$, as in Figure 2(b). It follows that, $\text{Assign}(L_i) = \{R_k\}$ where $R_k$ is the other region bordering $L_i$. Then consider the two points $\mathbf{s}_i^p$ and $\mathbf{f}_i^p$. Since $\text{Assign}(L_i) = \{R_k\}$ we can compute the 3D location of the points on $R_k$ that project to these two points. For each point in turn we compute the plane through the line $L_{j_l}$ and the point on $R_k$ corresponding to $\mathbf{s}_i^p$ or $\mathbf{f}_i^p$. We then compute the depth of the intersection of this plane with the ray through $\mathbf{y}$. See [6] for the details. If $\mathbf{f}_i^p$ (or $\mathbf{s}_i^p$) is on the same side of $L_{j_l}$ as $\mathbf{y}$ this distance is a lower bound on the distance to the point $\mathbf{y}$ (which implicitly constrains the "rotation"). Similarly, if $\mathbf{f}_i^p$ (or $\mathbf{s}_i^p$) is on the other side of $L_{j_l}$ from $\mathbf{y}$ this distance is an upper bound on the distance to the point $\mathbf{y}$. If all of these constraints on the depth of $\mathbf{y}$ (over $\mathbf{f}_i^p$ and $\mathbf{s}_i^p$ for each $L_i \neq L_{j_l}$) cannot be simultaneously satisfied then there is a depth ordering inconsistency.

**Planes Defined by Zero Lines**

If the plane equation $\boldsymbol{\pi}_j = \emptyset$ of $R_j$ is defined by zero lines there is nothing to check for the depth ordering. Since the plane is defined by zero lines, all of the lines $L_i$ that border $R_j$ are assigned to

other planes. The plane for $R_j$ can always be placed behind those of the other regions it borders.

### 3.6 Enumerating the Solutions

In the previous section we described the layer consistency function; an algorithm to compute whether a set of layer assignments and plane equations is a correct solution to the "Textureless Layers" problem. We enumerate all the solutions by: (1) generating every possible set of plane equations, (2) computing the implied layer assignment using Section 3.3, and (3) checking the potential solution is consistent using Section 3.5. The details are as follows:

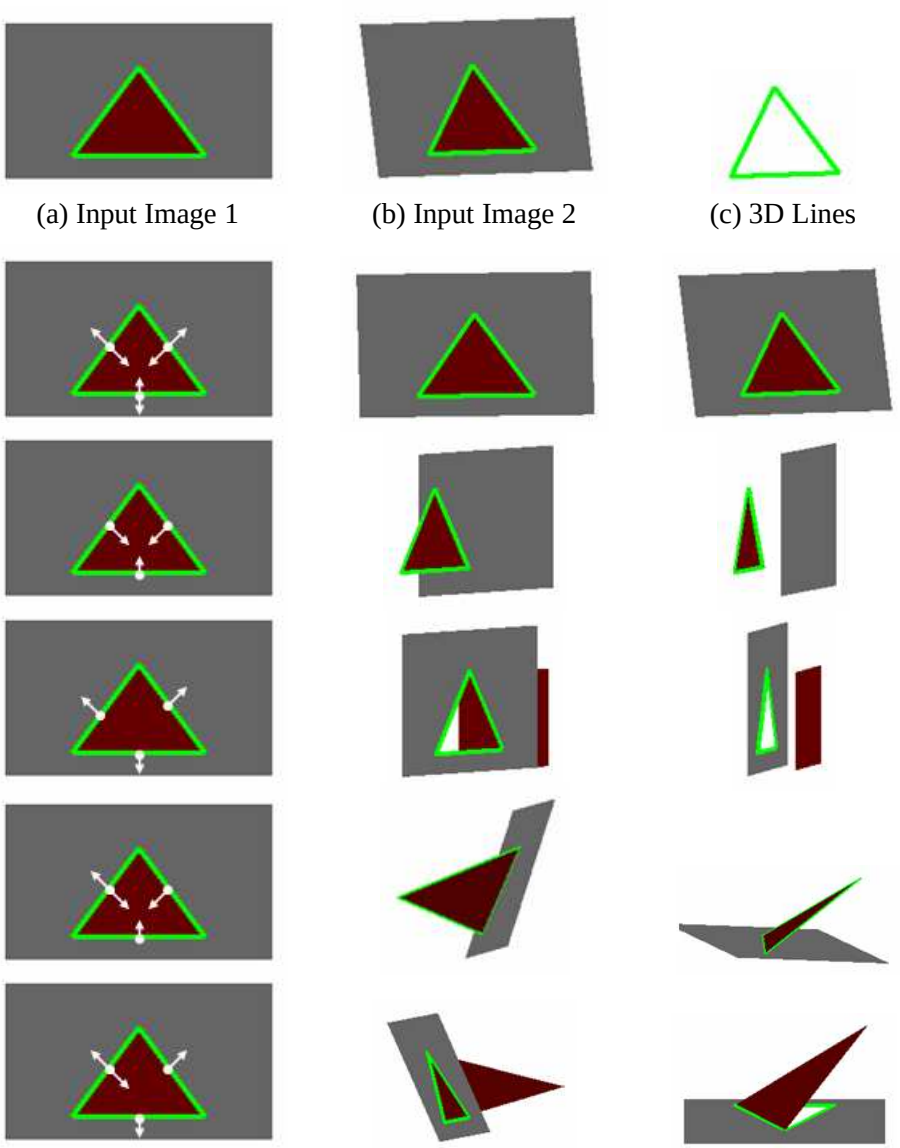
**Generating Plane Equations:** For each region $R_j$ consider every line $L_{j_1}$ that borders $R_j$. Each such line defines one possible plane equation $\boldsymbol{\pi}_j = L_{j_i}$ defined by that line. Secondly, consider every pair of lines $L_{j_1}$ and $L_{j_2}$ that border $R_j$. These lines are checked to see whether they are coplanar (using Section 3.4). If they are coplanar, the plane equation defined by those two lines is computed using Section 3.4 and added to the list of plane equation candidates. Finally, the plane defined by zero lines $\boldsymbol{\pi}_j = \emptyset$ is added to the list of candidates.

**Generating Layer Assignments:** For every possible way to initialize the plane equations $\boldsymbol{\pi}_j$, compute the line assignment $\mathrm{Assign}(L_i)$ for every line $L_i$ using the procedure in Section 3.3.

**Consistency Checking:** First the local consistency in Section 3.5.1 is checked, then the depth ordering consistency in Section 3.5.2 is checked. To save computation time, processing should be terminated for any candidate solution as soon as any of the checks fail.

## 4 Experimental Results

We are now ready to answer the questions posed in the introduction: "is a Textureless Layers reconstruction ever ambiguous?", "is it ever unique?", "is it usually ambiguous?", and "roughly how many ambiguities are there for typical scenes?". To answer the first two questions we applied the algorithm described in Section 3 to the synthetic inputs in Figures 3 and 4.

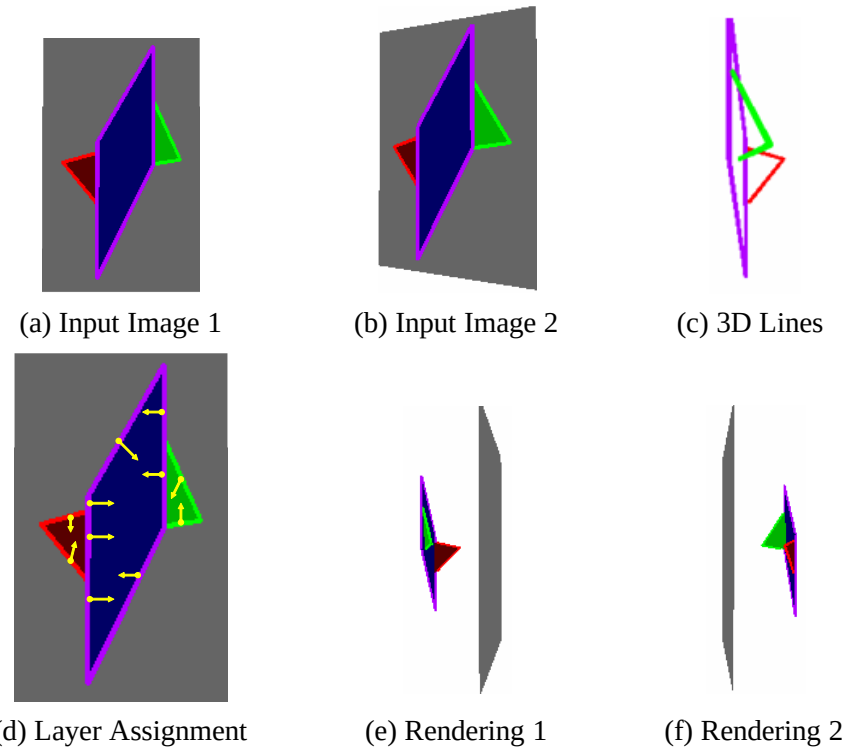


(a) Input Image 1 (b) Input Image 2 (c) 3D Lines

(d) Layer Assignment (e) Rendering 1 (f) Rendering 2

**Figure 3:** An example (a–c) with multiple consistent solutions (d–f). The 9 solutions can be grouped into 5 types, each displayed in a separate row. (d) The layer assignment. (e–f) Renderings of the planes. See the supplemental material `textureless_layers.html` for fly-by movies of the consistent solutions.

The results in Figure 3 show that the "Textureless Layers" problem can be ambiguous. Two input images and the 3D lines are shown in Figures 3(a), (b), and (c). The scene consists of a triangle, with two textureless regions, one inside and one outside the triangle. Our algorithm finds 9 consistent solutions. The solutions can be grouped into 5 types. One solution of each type is shown in rows 2–6 of Figure 3. In the first column of each row we illustrate the layer assignment by drawing arrows on one of the input images to show which region(s) each line is assigned to. In the other



(a) Input Image 1 (b) Input Image 2 (c) 3D Lines

(d) Layer Assignment (e) Rendering 1 (f) Rendering 2

**Figure 4:** An example (a–c) with a unique solution (d–f). See also `textureless_layers.html`.

2 columns we present renderings of the computed plane equations from 2 different viewpoints. (See also the supplemental material `textureless_layers.html` for fly-by movies.) In the first type of solution (row 2, 1 solution) there is a single plane with a triangular region "painted" on it. In the second type of solution (row 3, 1 solution) the triangle is a plane "floating" in front of a background plane (which is unconstrained.) In the third type of solution (row 4, 1 solution) the triangle is a "hole" in a plane (with an unconstrained background plane.) In the fourth type of solution (row 5, 3 solutions) the triangle is plane in front of a background plane which is joined to the triangle along one edge. In the fifth type of solution (row 6, 3 solutions) the triangle is "hole" in front of a background plane which is joined to the "hole" along one edge.

Naturally, the next question is whether the "Textureless Layers" problem is ever unique. The example in Figure 4 shows that it can be. Our algorithm finds just a single consistent solution. (Note that there are actually 24 locally consistent solutions, but only one of these obeys the depth ordering constraint.) The scene consists of 3 layers, a quadrilateral and 2 triangles. Two input images and the 3D lines are shown in Figures 4(a), (b), and (c), the assignment of lines to layers in

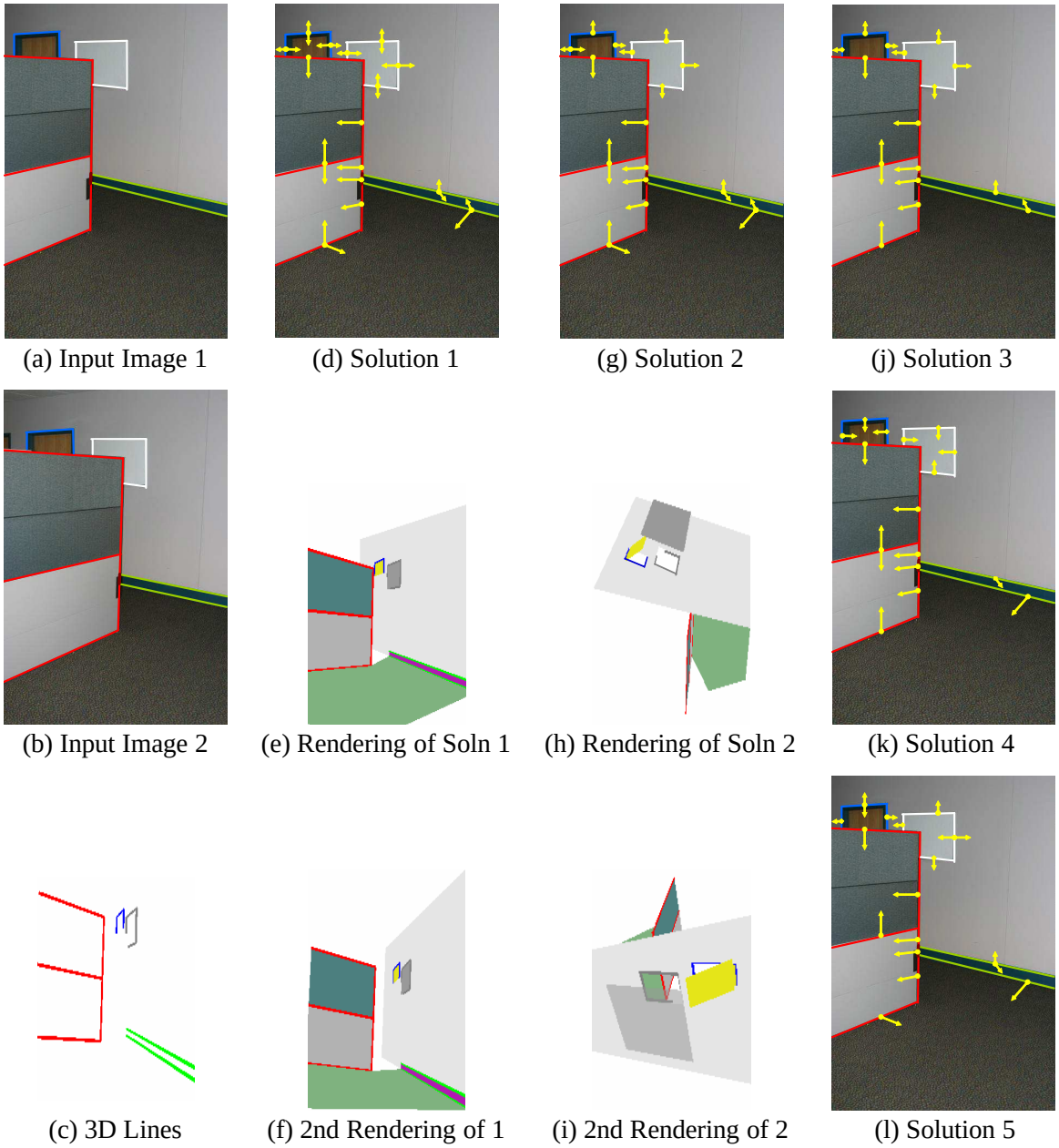
Figure 4(d), and 2 renderings of the recovered planes in Figures 4(e) and (f). See the supplemental material `textureless_layers.html` for a fly-by movie of the one unique solution.

The results in Figures 3 and 4 contain extreme cases that show that the "Textureless Layers" problem can sometimes be ambiguous and sometimes unique. We now address the question of "roughly how many ambiguities are there for typical scenes?" We applied our algorithm to a set of images of a real scene of a "corner walkway." Two input images, with the 2D lines overlayed on them, are included in Figures 5(a) and (b). The scene is typical of many encountered in 3D reconstruction and robot navigation tasks. Because the scene is largely man-made, there is little, if any, texture in any of the regions. The recovered 3D lines are shown in Figure 5(c). Our algorithm finds 448 consistent solutions. The solution that is "most plausible" to us as humans is shown in Figures 5(d–f). The line assignment is shown in Figure 5(d). Figures 5(e) and (f) contain renderings of the reconstructed 3D planes. See `textureless_layers.html` for a fly-by movie.

Figures 5(g–i) show another solution. In this solution, the "door" is ajar; i.e. it is a plane defined by the one line where the door is connected to the wall. This solution is also a valid solution (assuming we cannot see the small "crack" at the top of the door if it is ajar.) The degree to which the door is ajar is not uniquely determined by the input images (although there are constraints on it.) We just know that the door is connected to the wall along the appropriate line. Also in this solution, the "white-board" has becomes a hole (an opening into the room behind.) Although less likely in practice, walls can have holes in the them and so this solution is a valid 3D interpretation.

Figures 5(j), (k) and (l) show the line assignments for three more solutions. In Figure 5(j), the "floor" is a unconstrained plane (i.e., a hole). This is unlikely in practice, however, this reasoning is based on the high-level knowledge that there is normally a ground plane. In Figure 5(k), the "wall" is an unconstrained plane. This leaves the "white-board" floating in mid air. This interpretation is also unlikely, however, this reasoning is based on the knowledge that objects rarely float in mid air. As with all 448 solutions, the layers are a valid 3D interpretation of the images,

The results in Figure 5 are typical in the sense that there are generally a large number of consistent solutions. The main cause is the large number of solutions with planes constrained by



(a) Input Image 1 (d) Solution 1 (g) Solution 2 (j) Solution 3

(b) Input Image 2 (e) Rendering of Soln 1 (h) Rendering of Soln 2 (k) Solution 4

(c) 3D Lines (f) 2nd Rendering of 1 (i) 2nd Rendering of 2 (l) Solution 5

**Figure 5:** A real example (a–c) with 448 consistent solutions. (d) The layer assignment of the "most plausible" solution. (e–f) Renderings of the planes. (g–i) Another solution. (j–l) The layer assignment for three more solutions. See the supplemental material `textureless_layers.html` for fly-by movies.

1 line, which can explode combinatorially. Of the 448 solutions, 434 contain regions with one or more plane equations constrained by a single line. Of the remaining 14 solutions, 13 solutions contain layers with unconstrained plane equations. The final solution, where every layer is defined by two or more lines, is the one shown in Figure 5(d–f), i.e. the "most plausible" one.

# 5 Conclusion

We have posed and analyzed the "Textureless Layers" problem; i.e. the 3D reconstruction of constant intensity regions assuming that the scene is piecewise planar. In [1], it was shown that not only are constant intensity regions (almost) always ambiguous, there is generally a continuum of solutions. In this paper, we have shown that, although the planarity assumption does substantially reduce the ambiguity to give only finitely many solutions, the problem generally still does not have a unique solution. Moreover, the number of solutions is usually very large. Note that the ambiguity is inherent in the visual information in the images. It is not due to a limitation in our algorithm. All of the 448 solutions to the inputs in Figures 5(a) and (b) are valid 3D reconstructions.

## 5.1 Towards Practical Textureless Layers Algorithms

This paper does not describe a practical "Textureless Layers" algorithm. Building such an algorithm is not easy. Seemingly constant intensity regions such as man-made walls and ceilings rarely appear completely constant in images. Segmenting man-made scenes into constant intensity regions and their bounding lines is difficult. The main contributions of this paper are: (1) to pose the "Textureless Layers" problem, and (2) to show that, even if the segmentation into constant intensity regions can be performed, the resulting 3D reconstruction will still generally be ambiguous.

The difficulty of segmenting typical man-made scenes raises an interesting question: "when should 3D reconstruction algorithms use (point or line) features and when should they use brightness constancy based techniques?" One area for future work is to develop an algorithm to determine whether: (1) there is enough texture for brightness constancy algorithms to work, or (2) the regions are so constant that a feature-based "Textureless Layers" reconstruction should be used.

## 5.2 Practical Implications

The main practical implication of this theoretical paper is that, because the piecewise planarity assumption is insufficient to uniquely specify the 3D structure of constant intensity regions, ad-

ditional constraints are needed. As hinted towards to the end of Section 4, one possibility is to require that the scene be "physically realizable"; i.e. to enforce constraints that there has to be a ground-plane, and that planes cannot float in the air without a visible support, etc. (Note that valid 3D reconstructions are not necessarily physically realizable.) Another possibility it to choose the solution with the fewest unconstrained, or partially constrained, plane equations. Note that this is a heuristic because if the door were ajar or the whiteboard were a "hole" in Figure 5 then this solution would not be the "correct" one (although still a valid reconstruction of the images.)

# References


[1] S. Baker, T. Sim, and T. Kanade. When is the shape of a scene unique given its light-field: A fundamental theorem of 3D vision? *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 25(1):100 – 109, 2003.

[2] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 434–441, 1998.

[3] J.R. Bergen, P. Anandan, K.J. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Proceedings of the European Conference on Computer Vision*, pages 237–252, 1992.

[4] T. Darrell and A.P. Pentland. Cooperative robust estimation using layers of support. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):474–487, 1995.

[5] S. Hsu, P. Anandan, and S. Peleg. Accurate computation of optical flow by using layered motion representations. In *Proceedings of the International Conference on Pattern Recognition*, pages 743–746, 1994.

[6] Q. Ke, S. Baker, and T. Kanade. Textureless layers. Technical Report CMU-RI-TR-04-17, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, March 2004.

[7] H. Sawhney and S. Ayer. Compact representations of videos through dominant and multiple motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):814–830, 1996.

[8] J. Wang and E.H. Adelson. Layered representation for motion analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 361–366, 1993.

[9] Y. Weiss and E.H. Adelson. A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 321–326, 1996.