Planar segmentation from point clouds via graph Laplacian regularized K-planes

Wei Sui  
NLPR  
CASIA  
Email: wsui@nlpr.ia.ac.cn

Lingfeng Wang  
NLPR  
CASIA  
Email: lfwang@nlpr.ia.ac.cn

Huaiyu Wu  
NLPR  
CASIA  
Email: hywu@nlpr.ia.ac.cn

Chunhong Pan  
NLPR  
CASIA  
Email: chpan@nlpr.ia.ac.cn

Abstract—Extracting planar surfaces from 3D point clouds is an important and challenging step for generating building models as the obtained data are always noisy, missing and unorganised. In this paper, we present a novel graph Laplacian regularized K-planes method for segmenting piece-wise planar surfaces of urban building point clouds. The core ideas behind our model are from two aspects: 1) a linear projection model is utilized to fit planar surfaces globally; 2) a graph Laplacian regularization is applied to preserve smoothness of each plane locally. The two terms are combined as an objective function, which is minimized via an iterative updating algorithm. Comparative experiments on both synthetic and real data sets are performed. The results demonstrate the effectiveness and efficiency of our method.

Keywords—segmentation, piece-wise planar surfaces, graph Laplacian, point clouds

I. INTRODUCTION

3D reconstruction for urban buildings has increasingly attracted attention for its wide applications such as navigation [1], model design [2] and 3D printing. Many multi-view stereo algorithms [3], [4], [5] have been developed to recover the geometry of cameras as well as obtain 3D point clouds of the scene. Segmentation of 3D point clouds is an important step for generating models [6], [7]. In urban environment, the most significant features of buildings are the planar surfaces. Such planarity assumption can be utilized as prior constraints to help generate models of architectures. Unfortunately, planar segmentation on 3D point clouds is still a difficult task, as the 3D points obtained are always missing, noisy and unorganized.

Recently, many researchers are attracted to planar segmentation problem. Existing methods can be roughly categorized into two broad classes: clustering based methods [8], [9], [10], [11] and fitting based methods [12], [13].

Clustering based methods utilize criterions related to the coplanarity to cluster points into different planes. Stamos et al. [8] proposed a method similar to region growing. They first estimated the normal direction of each point, then distinguished planar regions from non-planar regions in the point clouds and finally merged points that are coplanar. The performance is determined by the accuracy of the normal estimation. In [9], [10], [11], MRF based methods were proposed to assign piece-wise depth to each image pixel. The core ideas behind their work were quite similar. Hypothesis planes were generated from 3D points obtained from a kind of Structure from Motion, and then an optimal label assignment was achieved via minimizing the energy function. The main difference between their work lies in the generation of hypothesis planes. Furukawa et al. [9] searched point density peaks along each axis; Gallup et al. [10] repeatedly performed RANSAC and removed inliers; Sinha et al. [11] used vanishing points and 3D reconstructed lines. However, completeness is still the main problem in the MRF based stereo methods.

Another large group of methods solve planar segmentation problem as a fitting problem. RANSAC has been commonly used to fit planes to noisy data [12], but it is unreliable for detecting data with many planar structures. Zuliani et al. [14] proposed a multi-RANSAC algorithm in which RANSAC was extended to deal with multiple planes simultaneously. The method works well when the planes do not insect each other, and the number of planes must be specified by the user in advance. A robust fitting method called J-linkage was proposed by Toldo et al. [13], which can be used to fit multiple planes to data heavily corrupted by noise and outliers. This method requires neither prior specification of the number of planes, nor much parameter tuning. However, the method cannot detect points belonging to more than one plane. Besides, it is not suitable to process large numbers of points as it’s quite time-consuming.

In this paper, we propose an effective and efficient graph
Laplacian regularized K-planes algorithm which can segment architecture point clouds into planes. We construct an energy function that optimizes global residual and local smoothness simultaneously. In each local region, an undirected neighbor graph is constructed which is finally used to form a graph Laplacian regularization. The energy function is minimized via a non-increasing iterative updating algorithm. The main contributions of our work are:

1) A naive K-planes model is first proposed to fit multiple planar surfaces in 3D point clouds globally, so that planar surfaces can be well recovered.
2) A graph Laplacian regularization is then applied to make the recovered planes locally smooth. Experiments show that each plane can be well preserved by the regularized K-planes (A pair of close-ups are shown in Figure 1).
3) Our method is optimized via a fast iterative updating algorithm. Compared with J-linkage, our method is hundreds of times faster, and can be applied to large scale problem.

The remainder of the paper is organized as follows. In Section 2, we introduce our graph Laplacian regularized K-planes algorithm and then present the simplification and optimization for our method. Comparative experiments are conducted on both synthetic and real data sets, and results are reported in Section 3. We end our paper with conclusions and a discussion of future work in Section 4.

II. THE PROPOSED METHOD

A. Problem formulation

Let \( P = \{p_i\}_{i=1}^N \) be \( N \) points in the scene. Let \( L = \{l_i\}_{i=1}^N \), denote the corresponding plane labels with \( l_i \in \{1, ..., K\} \), where \( K \) is the number of planes. In the Bayesian framework, the inference of \( L \) can be treated as a Maximum a Posteriori (MAP) estimate of the probability given the observations \( P \), that is

\[
L = \arg \max_L P(L|P) \\
\propto \arg \max_L P(P|L)P(L).
\]

The MAP estimate of \( P(L|P) \) is equivalent to minimizing the following function with respect to \( L \):

\[
E(L) = E_{data}(L) + \lambda E_{smooth}(L),
\]

where \( \lambda \) is a positive trade-off; \( E_{data}(\cdot) \) is the energy for data term derived from the likelihood \( P(P|L) \); \( E_{smooth}(\cdot) \) is the energy for smoothness term derived from the prior \( P(L) \).

**Data term.** The data term denotes the cost of assigning labels \( L \) to \( P \). It computes global residual via collecting the projection distances from points to their corresponding planes.

\[
E_{data}(L) = \sum_{k=1}^K \sum_{i=1}^N \| \pi_k^T X_i \|_2^2 \delta(l_i = k),
\]

where \( X_i = [x_{i1}, x_{i2}, x_{i3}]^T \) is the homogeneous coordinate of \( p_i \); \( \pi_k = [\pi_{k1}, \pi_{k2}, \pi_{k3}, \pi_{k4}]^T \) determines a plane in the scene with its unit normal direction being \( \{\pi_{k1}, \pi_{k2}, \pi_{k3}\}^T \); \( \delta(\cdot) \) is Dirac function such that \( \delta(l_i = k) \) equals 1 if \( l_i = k \) and 0 otherwise. The data term is fitted by a model which we call K-planes. The proposed K-planes algorithm describes the coplanarity of points. We can see that \( \| \pi_k^T X_i \|_2 \) is the projection distance from \( p_i \) to a plane \( \pi_k \), and \( \delta(l_i = k) \) selects projections of the points belonging to the plane \( \pi_k \).

**Smoothness term:** The K-planes algorithm can not keep local smoothness of the planes, so we need to add a smoothness term which enforces the spatial consistency by penalizing inconsistent neighbours of 3D points. We adopt the pair-wise consistency for saving computation cost, and then the smoothness term is given as follows:

\[
E_{smooth}(L) = -\sum_{i=1}^N \sum_{j \in N(i)} w(i,j) \delta(l_i \neq l_j),
\]

where \( N(i) \) denotes pairs of neighbors around \( p_i \); \( w(i,j) \) measures the similarity between \( p_i \) and \( p_j \).

The local neighbor structure is built as follows. We regard each point as a node, and add a weight to each edge. Then an undirected weighted graph is constructed on the points in the scene. Let \( R \) denote the corresponding plane labels with \( l_i \in \{1, ..., K\} \), where \( K \) is the number of planes. In the Bayesian framework, the inference of \( L \) can be treated as a Maximum a Posteriori (MAP) estimate of the probability given the observations \( P \), that is

\[
L = \arg \max_L P(L|P) \\
\propto \arg \max_L P(P|L)P(L).
\]

The MAP estimate of \( P(L|P) \) is equivalent to minimizing the following function with respect to \( L \):

\[
E(L) = E_{data}(L) + \lambda E_{smooth}(L),
\]

where \( \lambda \) is a positive trade-off; \( E_{data}(\cdot) \) is the energy for data term derived from the likelihood \( P(P|L) \); \( E_{smooth}(\cdot) \) is the energy for smoothness term derived from the prior \( P(L) \).

Algorithm 1: Graph Laplacian regularized K-planes algorithm for planar segmentation.

**Input:** A point cloud \( P = \{p_i\}_{i=1}^N \); the number of planes to be segmented \( (i.e., K) \); parameters \( \lambda, \mu \) in Eqn. (10) and threshold \( \tau \).

**Output:** The labels \( L \) for all the points.

begin
Initialize \( R \) by K-means, and fit each cluster a plane;
Construct the adjacent matrix \( W \) and the diagonal matrix \( D \) according to Eqns. (5), (6), (7) and (8);
while \( \| S_{new} - |S_{old} |_1 > \tau \) (\( S \) defined in Eqn. (10)) do
Update \( S \) by computing the distance from each point to each plane;
Update \( R \) according to Eqn. (15);
Get the label for each point through \( l_i = \arg \max_k R_{ik} \);
Fit a plane to each cluster;
end
Output \( L \) as the final clustering result.
end
tions, colors and normal directions, given by
\[
w_{3d}(i, j) = \begin{cases} 
\exp\left(-\frac{\|X_i - X_j\|^2}{\sigma_{3d}^2}\right), & \text{if } p_j \in \mathcal{N}(p_i) \text{ or } p_i \in \mathcal{N}(p_j), \\
0, & \text{otherwise},
\end{cases}
\]
\[
w_c(i, j) = \begin{cases} 
\exp\left(-\frac{|c_i - c_j|^2}{\sigma_c^2}\right), & \text{if } p_j \in \mathcal{N}(p_i) \text{ or } p_i \in \mathcal{N}(p_j), \\
0, & \text{otherwise},
\end{cases}
\]
\[
w_n(i, j) = \begin{cases} 
\exp\left(-\frac{|n_i - n_j|^2}{\sigma_n^2}\right), & \text{if } p_j \in \mathcal{N}(p_i) \text{ or } p_i \in \mathcal{N}(p_j), \\
0, & \text{otherwise},
\end{cases}
\]
where \(c_i\) and \(c_j\) are the colors of points \(p_i\) and \(p_j\); \(n_i\) and \(n_j\) are the corresponding unit normals. The parameters \(\sigma_{3d}\), \(\sigma_c\) and \(\sigma_n\) are adaptively estimated from input data, namely, \(\sigma_{3d} = \max_{i,j} \|X_i - X_j\|^2\), \(\sigma_c = \max_{i,j} |c_i - c_j|^2\) and \(\sigma_n = \max_{i,j} \|n_i - n_j\|^2\). In Eqsns. (4), (6), (7) and (8), \(\mathcal{N}(i)\) is obtained by the k-nearest neighbor (K-nn) technique with the spatial distance as a criterion.

Combining Eqsns. (3) and (4), we obtain the objective function for our graph Laplacian regularized K-planes as follows:
\[
E(L) = \sum_{k=1}^{K} \sum_{i=1}^{N} \|\pi_k^T X_i\|^2_2 \delta(l_i = k) - \lambda \sum_{i=1}^{N} \sum_{j \in \mathcal{N}(i)} w(i, j) \delta(l_i = l_j).
\]

B. Model simplification

A responsible matrix \(R \in \mathbb{R}^{N \times K}\) is introduced, where \(R_{ik} = 1\) if \(p_i\) is labeled as \(k\), otherwise \(R_{ik} = 0\). In practice, the constraint is relaxed to \(l_i = \arg \max_k R_{ik}\). We assume that \(l_i = l_j\) when \(\mathcal{N}(i)\), where \(\mathcal{N}(i)\) is the \(i\)-th row of \(\mathcal{R}\). Notice that the sum of each row in \(\mathcal{R}\) equals 1, and then the objective function can be reformulated as
\[
E(\mathcal{R}) = \sum_{k=1}^{K} \sum_{i=1}^{N} \|\pi_k^T X_i\|^2_2 R_{ik} - \lambda \sum_{i=1}^{N} \sum_{j \in \mathcal{N}(i)} w(i, j) \delta(\mathcal{N}(i)) + \mu \sum_{i=1}^{N} (K R_{ik} - 1)^2
\]
\[
\geq Tr(\mathcal{S} \mathcal{R}) + \lambda Tr(\mathcal{R}(\mathcal{D} - \mathcal{W}) \mathcal{R}) + \mu \|\mathcal{R} 1_k - 1_N\|^2_2,
\]
where \(\mu\) is a trade-off; \(1_k\) and \(1_N\) are \(K\)-dimensional and \(N\)-dimensional all-one vectors respectively. Here, we define a matrix \(\mathcal{S} \in \mathbb{R}^{N \times K}\) with \(S_{ik} = \|\pi_k^T X_i\|^2_2\); \(\mathcal{W}\) is a symmetric matrix containing manifold structures underlying in this point cloud, and \(\mathcal{W}_{ij} = w(i, j)\). \(\mathcal{D}\) is a diagonal matrix with \(D_{ii} = \sum_{j=1}^{N} w(i, j)\). Obviously, \(\mathcal{D} - \mathcal{W}\) is a graph Laplacian.

C. Optimization

The objective function is non-convex for \(\mathcal{R}\), therefore an iterative updating algorithm is introduced. Taking the non-negative constraint on \(\mathcal{R}\), the objective function can be reformulated as
\[
\min_{\mathcal{R}} Tr(\mathcal{S} \mathcal{R}) + \lambda Tr(\mathcal{R}(\mathcal{D} - \mathcal{W}) \mathcal{R}) + \mu \|\mathcal{R} 1_k - 1_N\|^2_2, \quad s.t. \quad \mathcal{R} \geq 0.
\]

Let \(\theta_{ik}\) be a Lagrange Multiplier for constraint \(R_{ik} \geq 0\). The Lagrange function \(L(\mathcal{R})\) is given by
\[
L(\mathcal{R}) = Tr(\mathcal{S} \mathcal{R}^T) + \lambda Tr(\mathcal{R}^T(\mathcal{D} - \mathcal{W}) \mathcal{R}) + \mu \|\mathcal{R} 1_k - 1_N\|^2_2 + Tr(\Theta \mathcal{R}^T),
\]
where \(\Theta = [\theta_{ik}] \in \mathbb{R}^{N \times K}\) is a Lagrange Multiplier Matrix. The partial derivative of \(L(\mathcal{R})\) with respect to \(\mathcal{R}\) can be written as
\[
\frac{\partial L(\mathcal{R})}{\partial \mathcal{R}} = S + 2\lambda(\mathcal{D} - \mathcal{W}) \mathcal{R} + 2\mu \mathcal{R} 1_k \mathcal{1}_K - 2\mu 1_N 1_K^T + \Theta.
\]

According to the Karush-Kuhn-Tucker conditions \(\theta_{ik} \mathcal{R}_{ik} = 0\), we can get
\[
\mathcal{S}_{ik} \mathcal{R}_{ik} + 2\lambda(\mathcal{D} - \mathcal{W}) \mathcal{R}_{ik} + 2\mu \mathcal{R} 1_k \mathcal{1}_K - 2\mu 1_N 1_K^T = 0.
\]
Therefore Eqn. (11) can be solved by the iterative updating rule proposed in [15] given by
\[
\mathcal{R}_{ik} = \frac{2\lambda(\mathcal{W}) \mathcal{R}_{ik} + 2\lambda \mu 1_N 1_K^T}{\mathcal{S}_{ik} + 2\lambda(\mathcal{D}) \mathcal{R}_{ik} + 2\mu \mathcal{R} 1_k \mathcal{1}_K^T}.
\]

This iterative update algorithm has been proved non-increasing by Lee et al. in [15]. The algorithm of graph Laplacian regularized K-planes for planar segmentation is summarized in Alg.1.

III. EXPERIMENTS AND ANALYSIS

In this section, we conducted experiments on both synthetic and real data sets to evaluate the effectiveness and efficiency of our method. To further demonstrate the capability of our method, we compare our method with the state-of-the-art J-linkage\(^1\) algorithm. In all experiments we use K-means results as the initialization, and we keep \(\mu = 1 \times 10^4\), \(\alpha_1 = 0.8\), \(\alpha_2 = 0.1\), \(\alpha_3 = 0.1\) fixed. The only parameters in our method are \(K\) and \(\lambda\).

\(^1\)http://profs.sci.univr.it/ fusiello/demo/jlk/.
Results on synthetic data are reported in Figure 2. This data contains 6 planes and 2400 points, and points in the same planes share the same color and normal direction. We can see that both our regularized K-planes algorithm and J-linkage algorithm accurately extract planes from 3D points, while the naive K-planes algorithm fails and its results are quite stochastic. Moreover, we can see that our regularized K-planes algorithm provides a better performance than J-linkage algorithm when handling points belonging to more than one plane (See the close-ups in Figure 2). These points of intersection are hard to distinguish in spatial position, but easily distinguishable when colors and normal directions are taken into account. Any local inconsistency will be penalized by our method. Running time and accuracy can be found in Table I.

The real data sets, Hall-dense (H-dense) and Building-dense (B-dense), are produced by PMVS3, with which accurate colors and normal directions are obtained as well. Different from the synthetic data set, the real data sets are quite noisy and missing as shown in Figure 3. There are many unreasonable planes appeared in the results of the naive K-planes algorithm which contain points that are far away in spatial positions and inconsistent in colors or normal directions as well. Our regularized K-planes algorithm can rectify these unreasonable planes via penalizing such inconsistency. As illustrated in Figure 1, with graph Laplacian regularization, the results of our method are smoother and cleaner which means local structures of planes are well preserved.

We compare our method with J-linkage algorithm in both performance and speed on Hall-sparse (H-sparse) and Building-sparse (B-sparse) which are produced by down sampling from H-dense and B-dense respectively. Figure 4 and Figure 5 illustrate the results. Both our regularized K-planes algorithm and J-linkage algorithm provide satisfying results. However, our method is hundreds of times faster than J-linkage, and can be applied to handling large numbers of points. Running time of each experiment is reported in Table II. Experiments are conducted in MATLAB on platform Intel(R) Core(TM) i3-2100 CPU @ 3.10GHz with 4GB RAM.

<table>
<thead>
<tr>
<th></th>
<th>K-planes</th>
<th>Regularized K-planes</th>
<th>J-linkage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>0.06s</td>
<td>0.90s</td>
<td>54.33s</td>
</tr>
<tr>
<td>Accuracy</td>
<td>53.75%</td>
<td>99.4%</td>
<td>96.7%</td>
</tr>
</tbody>
</table>

Table I. Running time (seconds) and clustering accuracy on synthetic data sets (2400 points).

---

2. http://vision.ia.ac.cn/data/index.html
4. The present MATLAB code version published on the web can not handle large numbers of points on our machine for lack of memory, so we just run J-linkage on sparse data sets.
<table>
<thead>
<tr>
<th></th>
<th>K-planes</th>
<th>Regularized K-planes</th>
<th>J-linkage</th>
</tr>
</thead>
<tbody>
<tr>
<td>H-dense (83910 points)</td>
<td>23.51s</td>
<td>47.68s</td>
<td>#7</td>
</tr>
<tr>
<td>B-dense (96722 points)</td>
<td>15.68s</td>
<td>44.26s</td>
<td>#7</td>
</tr>
<tr>
<td>H-sparse (16782 points)</td>
<td>3.12s</td>
<td>12.71s</td>
<td>4433.84s</td>
</tr>
<tr>
<td>B-sparse (16121 points)</td>
<td>0.95s</td>
<td>1.19s</td>
<td>4619.37s</td>
</tr>
</tbody>
</table>

TABLE II. RUNNING TIME (SECONDS) OF GRAPH LAPLACIAN REGULARIZED K-PLANES AND J-linkage ON REAL DATA SETS.

IV. CONCLUSION

In this paper, we have proposed a graph Laplacian regularized K-planes algorithm for planar segmentation from urban building point clouds. Due to the graph Laplacian regularization, our method can well preserve local smoothness of planes. Experiments show that our method can be applied to large scale point clouds.

The data term in our model is represented by the linear projection. With some modification, it can be easily extended to fitting complex surfaces, such as the cylindrical surface. In the future, we mainly focus on this extension, and make our model be applied to general urban buildings.

ACKNOWLEDGMENT

We would first thank the anonymous reviewers for their valuable feedback and guiding comments. Also we want to thank Yasutaka Furukawa for the PMVS algorithm through we get our data. This work is supported by the National Natural Science Foundation of China under Grants 61075016, 61203277, 61272049, and 61272331.

REFERENCES