Semantic Decomposition and Reconstruction of Residential Scenes from LiDAR Data *Supplementary Material*

Hui Lin*[1] Jizhou Gao*[1] Yu Zhou[2] Guiliang Lu[2] Mao Ye[1] Chenxi Zhang[1] Ligang Liu[3] Ruigang Yang[1]

[1]University of Kentucky [2]Nanjing University [3]University of Science and Technology of China

Abstract

This supplemental document provides more technical details of semantic segmentation, texture mapping, plant modeling and object replacement for other common subjects in landscape.

1 Semantic Segmentation

The first stage in our system is to segment the input point cloud into semantically distinctive groups. This is done in two steps. The first is to label the input into *categories*, such as houses or plants. Then points labelled as houses are further refined into more detailed structural *classes*, such as roofs or walls. We use supervised machine learning techniques to perform these tasks.

1.1 Categorization

We define the following categories in our current implementation: *houses*, *plants*, *mailboxes*, *street lights*, *waste bins*, *cars* and *ground*, which are common objects seen in a residential area. From our scanned data, we manually label a section as the training data set and adopt the semantic segmentation approach similar to Zhang et al. [2010]. It should be mentioned that other alternatives such as [Golovinskiy et al. 2009] and [Xiao and Quan 2009] can also be applied to perform this task.

In order to deal with our point cloud data, we have made a few changes to the original method. First, we group points into *superpoints*, analogues to the *superpixel* concept in image segmentation. We randomly start with a seed point and group its neighbors into a superpoint based on two thresholds: the maximum number of points and the maximum distance between points. This process repeats until all points are processed. In our experiments, these two numbers are set to 100 and 0.3m, respectively. Second, we use the Adaboost classifiers. Third, we introduce a new concept called *super-region* to differentiate objects with similar superpoint features but at different scales, such as houses and cars, cars and waste bins, etc. We group superpoints by a region growing algorithm with a threshold of the angle difference between superpoint normals; in this way, most coplanar superpoints are grouped into a large super-region and remaining small super-regions can be further combined using a similar grouping method based on distance as above-mentioned. Figure 1 shows an example. For each super-region, we compute the following features:

- Height: the maximum height of a super-region.
- Volume: the volume of the bounding box of a super-region.
- Area: the maximum area of the bounding box of a super-region.
- Planarity difference: the sum of difference of planarity among adjacent superpoints in one super-region.
- Length: the length between maximum and minimum superpoint height.

*The first two authors contributed equally to this work.

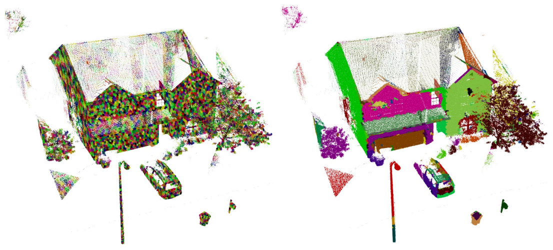

Figure 1: *Grouping point clouds into superpoints and super-regions. (left) Superpoints of a point cloud; (right) super-regions of the point cloud.*

These additional features are added to its superpoints; together with other superpoint features as defined in [Zhang et al. 2010], the augmented feature vectors are used for training. After the classification stage, the result is further grouped into connected components, each with its own semantic label.

1.2 Segmentation of House Point Cloud

For each point set labeled as a house, it will be further segmented into different *classes*, including *columns*, *roofs* and *walls*. The same segmentation method in Section 1.1 is adopted with one additional superpoint feature: surrounding emptiness, which measures the number of points within a neighboring bounding box around each superpoint center. The bounding box size is set to 0.5m, which is roughly twice the size of a typical column. This feature is mainly used to identify columns.

2 Texture Mapping

To compute texture maps for a reconstructed building model, we first automatically find the nearby camera views that capture the model, and then back-project the views to the planar surfaces of the model. However, images and point clouds may not be perfectly aligned, and a model is only approximately reconstructed. Thus, texture maps generated from the direct back projection would produce noticeable misalignment. In order to alleviate these errors, we propose an algorithm of content-preserving warps inspired by [Liu et al. 2009] based on 2D-3D line correspondences. We further fuse the multiple overlapping views using a multi-label MRF energy minimization framework similar to [Sinha et al. 2008]. Details are presented below.

2.1 Back-projection

As the reconstructed model M and images are geo-referenced and the scanning trajectory is also recorded, the images capturing M can be easily retrieved. In order to compute the texture map T_p of a plane P in M from an image I, as P might be entirely or partially occluded by other planes in M and/or nearby models (typically two along the driving path), we need to perform a visibility test on P

first. We use the standard two-pass z-buffer algorithm: given the projection matrix of I, we render M and nearby models and then render the plane P to get the visibility estimation and the texture map through back-projection.

2.2 Content-preserving Warps based on 2D-3D line correspondences

The texture maps generated from the direct back projection would produce noticeable misalignment due to the registration error and imperfect model fitting. We propose an algorithm of content-preserving warps to generate a distorted image I' where 2D edges are better aligned with 3D lines in M. Our method is inspired by [Liu et al. 2009], however, different from point correspondences used in their method, we establish 2D-3D line correspondences between a model M and an image I because line features are much more stable than point features across different modalities.

We extract a set of line segments from an image I using the LSD algorithm [von Gioi et al. 2010]. We denote a projected 3D edge segment of M onto an image I as l^M. An image edge segment l^I is matched to the model edge segment l^M if they have short distance (less than 20 pixels), small angle (less than 15°), similar length (the longer one is no more than twice the length of the shorter one), and sufficient overlapping (at least half of the shorter one can be projected onto the other segment). In order to prune the wrong line matches, one observation is that the larger the model plane is, the more likely line matches can be found. Therefore, we sort the model planes according to plane size in a descending order, and iteratively apply robust planar homography fitting algorithm based on RANSAC to prune wrong line matches for each model plane, so that model edges from the large planes find their matches first, and model edges from the small planes can be adjusted accordingly.

To refine a texture map, we synthesize a new image I' from an image I where matched 2D image edges and projected 3D model edges are co-aligned. We use the content preserving warps similar to [Liu et al. 2009], which divides an image into a quad mesh and compute a distorted quad mesh minimizing an energy functional consisting of a data term and a smoothness term. Different from their method, we encode 2D-3D line matches into the data term instead of point correspondences. Our new data term ensures a line l^I to align l^M by minimizing the total square of point-to-line distance between sample points on line l^I and line l^M. The warped image I' can then be generated by a standard texture mapping algorithm according to the distorted quad mesh. We then compute a texture map for each model plane through the back-projection algorithm based on the warped image I'. As shown in Figure 2, the misalignment artifacts can be reduced by our approach.

2.3 Multiple Texture Fusion

To fuse the texture maps for a model plane P from overlapping camera views, we apply a multi-label MRF energy minimization framework similar to [Sinha et al. 2008], where the data term is a weighted sum of two terms: preference for a frontal view and preference for a camera view that has a large number of visible pixels of P's projection; and the smoothness term encourages neighboring pixels should have same labels.

3 Landscape Modeling

Given the semantic labels for the remaining points, we have developed simple and effective methods to reconstruct other objects. These objects enrich the visual realism of the final model.

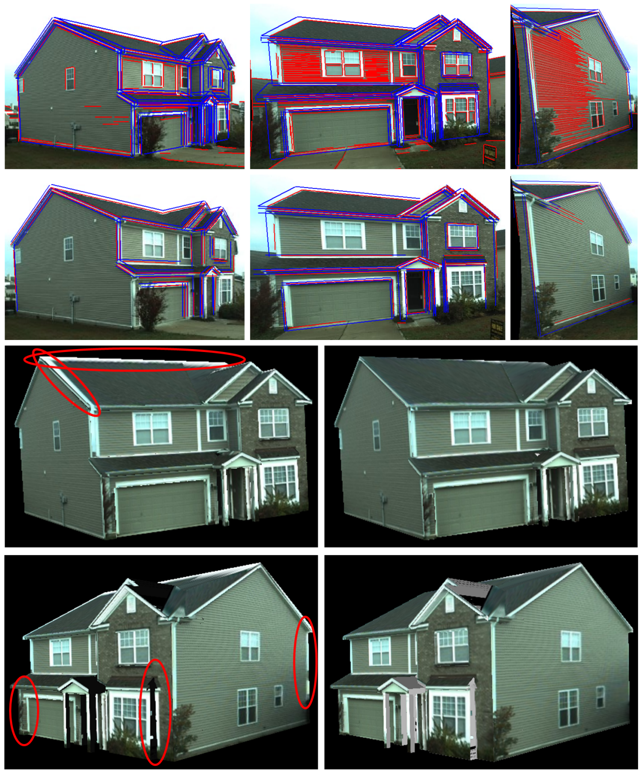

Figure 2: *Comparison on two texture mapping algorithms. 1st row shows the detected image edges (red) and projected model edges (blue); the misalignment is obvious due to the registration error. 2nd row shows the automatically extracted correspondences between image edges (red) and model edges (blue). In 3rd row and 4th row, the left part shows the results from the original back-projection texture mapping where the imperfection marked by red circles are improved by our method shown in the right part.*

3.1 Plant Reconstruction

Plants, including trees and shrubberies, are integral parts of our living environment. A number of reconstruction methods have been developed to model trees from LiDAR data (e.g., [Livny et al. 2011], or even images (e.g., [Tan et al. 2007]). While these methods can generate very high quality geometric tree models, we choose to develop a fully automatic method suitable for large-scale reconstruction. Toward this goal, we choose to adopt the light-weight billboard representation for visually-plausible plant models. Our plant model consists of two orthogonal planes and one billboard image. The key is to extract from the input the billboard image and find the tree trunk (the axis of rotation). First, we project each plant point set onto the nearest corresponding image. A shape mask is extracted as the billboard image. The point set is also projected onto the ground plane normal direction (i.e., the z axis) and the point density along the z axis is measured. A sharp increase (3 times) in density means a transition from trunk to foliage. If no trunk is detected, the center z axis of 3D bounding box is used as the axis of rotation. If the ratio of x and y axes of the 3D bounding box is too big or too small, it is likely to be a shrubbery, therefore we segment it into parts with equal length of x and y axes. We usually use a 2:1 ratio as the threshold for cutting. Another problem we have to deal with is misalignment of images. We segment the billboard images into superpixels and cluster the superpixels into three groups (trunk, leaf, and error) according to average chroma value of pixels in the image. Then the error superpixels will be automatically replaced by randomly selected leaf superpixels. Figure 3 illustrates the entire process.

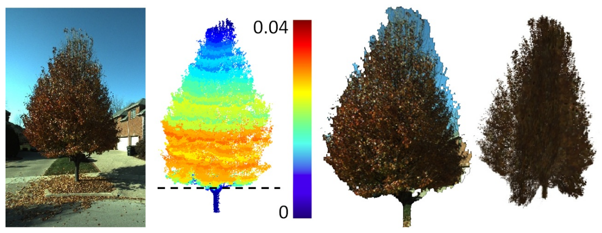

Figure 3: *Tree Reconstruction. From left to right: (a) corresponding color image of point clouds; (b) point clouds with color coded horizontal density, the dotted line shows the trunk/leaf boundary; (c) the texture and point clouds are off by some pixels; (d) rendered billboard tree with the corrected texture.*

3.2 Model Replacement

Other frequently occurring static objects in residential landscape such as mailboxes and street lights are often hard to directly reconstruct using simplified geometric models from the cluttered, incomplete and noisy data. Inspired by the recent success of model replacement applied to indoor scene modeling (e.g., [Shao et al. 2012]), we download the similar models from Google 3D Warehouse, and use PCA to estimate global scaling and an initial pose between the models and the recognized raw points and further align them using iterative closest point (ICP) method; in this way, points from a categorized object (e.g., a mailbox) are replaced by the corresponding model.

References

GOLOVINSKIY, A., KIM, V., AND FUNKHOUSER, T. 2009. Shape-based recognition of 3d point clouds in urban environments. In *ICCV*.

LIU, F., GLEICHER, M., JIN, H., AND AGARWALA, A. 2009. Content-preserving warps for 3d video stabilization. *ACM Trans. Graph. 28*, 3 (July), 44:1–44:9.

LIVNY, Y., PIRK, S., CHENG, Z., YAN, F., DEUSSEN, O., COHEN-OR, D., AND CHEN, B. 2011. Texture-lobes for tree modelling. In *ACM Transactions on Graphics*, vol. 30, ACM, 53.

SHAO, T., XU, W., ZHOU, K., WANG, J., LI, D., AND GUO, B. 2012. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph.*.

SINHA, S., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. 2008. Interactive 3d architectural modeling from unordered photo collections. In *SIGGRAPH Asia*, 159:1–159:10.

TAN, P., ZENG, G., WANG, J., BING, S., AND QUAN, K. L. 2007. Image-based tree modeling. *ACM Trans. Graph 26*, 43.

VON GIOI, R. G., JAKUBOWICZ, J., MOREL, J.-M., AND RANDALL, G. 2010. Lsd: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis and Machine Intelligence 32*, 722–732.

XIAO, J., AND QUAN, L. 2009. Multiple view semantic segmentation for street view images. In *ICCV*.

ZHANG, C., WANG, L., AND YANG, R. 2010. Semantic segmentation of urban scenes using dense depth maps. In *ECCV*.