Virtual navigation of interior structures by lidar

Yongjian Xi[a], Xiaoling Li[a], Ye Duan[a], Norbert Maerz[b]

[a]University of Missouri at Columbia

[b]Missouri Univeristy of Science and Technology

ABSTRACT

In this project, we propose to develop a prototype system that can automatically reconstruct 3D scenes of the interior of a building, cave or other structure using ground-based LIDAR scanning technology. We develop a user-friendly real-time visualization software package that will allow the users to interactively visualize, navigate and walk through the room from different view angles, zoom in and out, etc.

Keywords: LIDAR, Virtual navigation, visualization

1. INTRODUCTION

In this project, we propose to develop a prototype system that can automatically reconstruct 3D scenes of the interior of a building, cave or other structure using ground-based LIDAR scanning technology. We develop a user-friendly real-time visualization software package that will allow the users to interactively visualize, navigate and walk through the room from different view angles, zoom in and out, etc.

To meet the challenges of occlusions, large gaps/holes, and significant noise in LIDAR data, we employ and develop robust, efficient algorithms for all major components of the system (Figure 1) including robust model estimation based on Random Sample Consensus (RANSAC) and Principal Component Analysis (PCA); fast and efficient interior object classification/recognition; and robust 3D reconstruction algorithm.

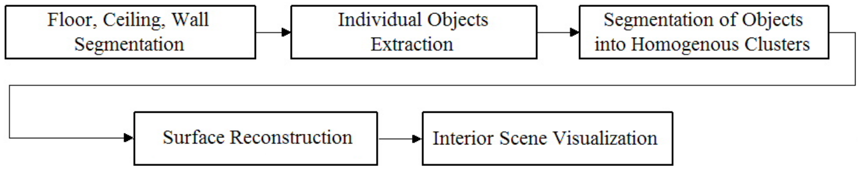
Floor, Ceiling, Wall Segmentation
Individual Objects Extraction
Segmentation of Objects into Homogenous Clusters
Surface Reconstruction
Interior Scene Visualization

Figure 1: System flow chart.

1.1 Hierarchical LIDAR data segmentation

Given the LIDAR data of an interior room (Figure 2 (a)), we will conduct hierarchical segmentation and grouping. More specifically, we will first identify all the major planar regions such as floor, ceiling and vertical walls of the room (Figure 2 (b)). Next, we will identify and extract individual objects such as chairs, tables, etc that are lying/attaching to floors/ceiling/walls (Figure 2 (c)). Finally, we will further segment each extracted object such as chair into homogenous clusters (Figure 2 (d)), which will then be used in the subsequent local shape primitive fitting and classification process.

1.1.1 Identification of floor, ceiling and walls

We will first compute the bounding box that encloses the LIDAR data. Assuming that the LIDAR data is scanned from one side of the bounding box such as the windows of the room, we will then search for major planar regions such as floor, ceiling and walls in the neighborhood of the other five sides of the bounding box. More specifically, for each side of the bounding box, we will first extract all the point clouds that are within a certain distance of the current side of bounding box. We will then use RANSAC [4] to find the best plane that fit these selected point clouds. RANSAC is a hypothesis generation and testing algorithm that is very robust for outliers. The main idea behind the technique is to use a minimal number of data points needed to estimate the model (i.e. fitted plane), and then count how many of the remaining data points are compatible with the estimated model, in the sense of falling within a chosen threshold.

The plane fitting will be based on the robust PCA algorithm which can compute local surface properties based on local neighborhoods of sample points. We will find the *k*-nearest neighbors of a sample point p, denoted by the index set N_p, The local surface properties of the point clouds can be efficiently estimated by the eigenanalysis of the covariance matrix C of a local neighborhood at sample point p:

$$\mathbf{C} = \begin{bmatrix} \mathbf{p}_{i_1} - \bar{\mathbf{p}} \\ \cdots \\ \mathbf{p}_{i_k} - \bar{\mathbf{p}} \end{bmatrix}^{\mathrm{T}} \cdot \begin{bmatrix} \mathbf{p}_{i_1} - \bar{\mathbf{p}} \\ \cdots \\ \mathbf{p}_{i_k} - \bar{\mathbf{p}} \end{bmatrix}, i_j \in N_p \ ,$$

where $\bar{p}$ is the centroid of the neighbors of p. Consider the eigenvector problem

$$\mathbf{C} \cdot \mathbf{v}_l = \lambda_l \cdot \mathbf{v}_l, \, l \in \{0, 1, 2\}$$

Since C is symmetric and positive semi-definite, all its three eigenvalues $\lambda 0 \leq \lambda 1 \leq \lambda 2$ are real-valued and the eigenvectors v_l form an orthogonal frame, corresponding to the principal components of the point set. Thus v_0 approximates the surface normal at p, or in other words, v_1 and v_2 span the tangent plane at p. Note that n is the size of the neighborhood, which serves as the scale-control parameter, and is dependent on the laser scanner resolution, i.e. how dense the point clouds data will be.

For generic interior scenes, the above six-sided rectangular cube assumption of the interior scene will not always hold. Thus in this case, we implemented a new region-growing based segmentation algorithm. The basic idea of the region-growing algorithm is: start with an unvisited point p, iteratively include its neighboring point q, if the distance between p and q is smaller than a threshold (e.g. the sampling density), and the difference between the normal at p and the normal at q is also smaller than a threshold (2 degrees in our case). This can be done in a breadth-first search, and stop when the above criteria is no longer hold. Then the algorithm move on to the next unvisited point, until all the points are visited.

To speed up the computation, we employed volumetric grid to store the points, thus the connectivity inference can be done instantly by grid indexing. The surface normal is estimated by the aforementioned Principal Component Analysis (PCA). After the region growing, we further merge clusters that are co-planar, i.e. we would like to merge all the clusters that belong to a bigger plane even though they are not connected. More specifically, a cluster is merged with another if the normal of both clusters are within 2 degrees of each other, AND the vector pointing from the first cluster's midpoint to the second cluster's midpoint is more than 85 degrees apart from the average of the two cluster's normals (by the first condition, these two normals will be already nearly identical).

1.1.2 Identification of individual objects and extraction

Once all the point clouds in these major planar regions such as floor, ceiling and walls are identified and removed, there are sufficient separation between points of individual objects such as chairs, tables, etc that are lying/attaching to the ground, ceiling, and walls. We then proceed to group all these points based on their proximity to each other by finding connected components such that each point in a connected component is within a given distance to at least one more point in that component. Hence, all the points belonging to an individual object such as a chair would lie in a single connected component. We employed efficient grid-based range-finding algorithms by using a volumetric grid to store the points to speed up the connected component analysis process.

1.1.3 Segmentation of individual objects into homogenous clusters

Once we segment the whole LIDAR data into individual connected components which may correspond to objects such as chairs, tables, fans, etc, we will proceed to further segment (i.e. divide) each objects into homogenous patches/clusters based on some similarity metrics (e.g. distance, normal, curvature, etc). The segmented homogenous clusters will then be used in the subsequent local shape primitive fitting process. In this project, we employed a region-growing based segmentation algorithm. This is the same method we used for user-guided semi-automatic planes identification for generic interior scenes, described in Section 1.1.1. The region-growing based segmentation algorithm works very well in our experiments.

1.1.4 Statistics based local shape primitive fitting

Once we segment the whole LIDAR data into individual connected components which may correspond to objects such as chairs, tables, fans, etc, if needed, we can proceed to further segment (i.e. divide) each objects into homogenous patches/clusters based on some similarity metrics (e.g. distance, normal, curvature, etc). We first segment the object into individual clusters based on the same aforementioned region growing algorithm for each individual objects. We then divide each segment into quadratic patches. More specifically, for each segment:

a. Fit quadratic to the current segment (use RANSAC method to only fit inliers)
b. Calculate the error per point in the segment (distance from point to quadratic)
c. If the average error for the entire segment is greater than +/-1%
    i. Split the segment into 8 equal segments
    ii. Repeat step a.

The following are the detailed implementation:

1. Given a segmented cluster, we compute its local coordinate system with local origin *c* by conducting Principal Component Analysis (PCA) of all the points in this cluster.
2. Transform all the points in this cluster into the local coordinate system.
3. Within the current local coordinate system, fit a local bivariate quadratic polynomial function *Q(u, v, w) = w – (Au*u + Bu*v + Cv*v + Du + Ev + F)* of all the points in the cluster, by conducting weighted least square optimization to minimize: $\sum_{i=1}^{n} Q(p_i)\theta(\| p_i - c \|)$, where p_i are points in the cluster, and θ is a non-negative, smooth, radial, monotonously decreasing function, usually with finite support, e.g. a truncated Gaussian function.

Since the points may contain outliers such as noise or artifacts, we further employed Random Sample Consensus (RANSAC) algorithm to integrate with the above quadratic fitting, so that the algorithm will be robust to noise. RANSAC is a hypothesis generation and testing algorithm that is very robust for outliers. The main idea behind the technique is to use a minimal number of data points needed to estimate the model (i.e. quadratic function), and then count how many of the remaining data points are compatible with the estimated model, in the sense of falling within a chosen threshold (we use the average fitting error evaluated from the quadratic function Q).

1.2 Progressive surface reconstruction

Since the original LIDAR data can be quite large, thus in this project, we developed a progressive surface reconstruction algorithm. More specifically, we will utilize the result from the previous hierarchical segmentation, and progressively transmit the data into the surface reconstruction algorithm cluster by cluster. There will be two major types of data, one is the planar data, one is the non-planar object data. To handle extreme large data, we further extend the aforementioned data segmentation algorithm into two hierarchical steps. In the first step, we conduct grid-based data segmentation using a subset of the original data. For example, we sequentially subsample the data (e.g. 10%), and conduct clustering as described before. We then label the cells in the embedding grid by the clusters. In the next step, we load in the whole set of the original data sequentially, and label each data points based on the label of the grid cell it belongs to. The reconstruction results (triangle meshes) are then put together in the final model. The reconstruction is based on a non-parametric anisotropic kernel-based method we developed for 3D surface reconstruction from noisy LIDAR data. In particular, we proposed an anisotropic kernel based nonparametric density estimation method for outlier removal, and an anisotropic density and normal propagation techniques for saliency field extraction, and a non-maximum suppression for 3D surface reconstruction, which we will describe in the following.

1.2.1 Anisotropic kernel density estimation

Points outside the object surface are outliers that have to be removed. Since the real object surface is unknown, it is hard to specify a general criterion to detect outliers. We propose to employ parzen-window based nonparametric density estimation method for outlier removal. Parzen window based kernel density estimation is the most popular nonparametric density estimation method [4]. Given *n* data points x_i, *i* = 1, … , *n* in the *d*-dimensional Euclidean space

R^d, the multivariate kernel density estimate obtained with kernel $K(x)$ and window radius h, computed in the point x is defined as:

$$f(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \qquad (1)$$

Without loss of generality, let's assume h = 1 from now on, so we could simplify Equation (1) as:

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} K(x - x_i) \qquad (2)$$

$K(x)$ is generally a spherically symmetric kernel function satisfying:

$$K(x) = C_{k,d}\, k(\| x \|^2), \qquad (3)$$

where k(x) is the profile function of the kernel $K(x)$. $C_{k,d}$ is the normalizing constant such that

$$K(x) \geq 0, \qquad (4)$$

and

$$\int_{R^d} K(x)dx = 1, \qquad (5)$$

$\|x\|$ is the L_2 norm (i.e. Euclidean distance metric) of the d-dimensional vector x. Employing the profile function notation, Equation (2) can be further rewritten as

$$f(x) = \frac{C_{k,d}}{n} \sum_{i=1}^{n} k(\| x - x_i \|^2) \qquad (6)$$

There are three types of commonly used spherical kernel functions $K(x)$: the Epanechnikov kernel, the uniform kernel, and the Gaussian kernel. The Epanechnikov kernel is defined by the profile function $k_E(x)$:

$$k_E(x) = \begin{cases} 1 - x & 0 \leq x \leq 1 \\ 0 & x > 1 \end{cases} \qquad (7)$$

The uniform kernel is defined by the profile function $k_U(x)$:

$$k_E(x) = \begin{cases} 1 & 0 \leq x \leq 1 \\ 0 & x > 1 \end{cases} \qquad (8)$$

The Gaussian kernel is defined by the profile function $k_N(x)$:

$$k_N(x) = \exp(-\frac{1}{2}x) \qquad x \geq 0 \qquad (9)$$

1.2.2 Outlier removal by anisotropic ellipsoidal kernel

For 3D point cloud obtained by laser scan, the outliers tend to spread in the space randomly, while "real" (we use a quotation here to emphasize the fact that the real surface is unknown) surface points will spread along a thin shell which encloses the real surface object. In other words, the distribution of the outliers is relatively isotropic, while the distribution of the real surface points is rather anisotropic. Hence we propose to employ an anisotropic ellipsoidal kernel based density estimation method for outlier removal.

For anisotropic kernel, the L_2 norm $\|x-x_i\|$ in Equation (6), which measures the Euclidean distance metric between two points x and x_i will be replaced by the Mahalanobis distance metric $\|x-x_i\|_M$:

$$\| x - x_i \|_M = ((x - x_i)^t H^{-1} (x - x_i))^{1/2}, \quad (10)$$

here H is the covariance matrix defined as:

$$H = DD^T, \qquad (11)$$

and

D = (x1 - x, x2 - x, … , xn - x). (12)

Geometrically, $(x-x_i)^t H^{-1}(x-x_i)=1$ is a three-dimensional ellipsoid centered at *x*, with its shape and orientation defined by *H*. Using Single Value Decomposition (SVD), the covariance matrix *H* can be further decomposed as:

$$H = UAU^T, \tag{13}$$

With

$$A=\begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \tag{14}$$

$\lambda_1 \geq \lambda_2 \geq \lambda_3$ are the three eigenvalues of the matrix *H*, and *U* is an orthonormal matrix whose columns are the eigenvectors of matrix *H*.

To compute the anisotropic kernel based density, we apply an ellipsoidal kernel *E* of equal size and shape on all the data points. The orientation of the ellipsoidal kernel *E* will be determined locally. More specifically, given a point *x*, we calculate its covariance matrix *H* by points located in its local spherical neighborhood of a fixed radius. Without loss of generality, we assume the radius is 1 (which can be done by normalizing the data by the radius). The *U* matrix of Equation (13) calculated by the covariance analysis is kept unchanged to maintain the orientation of the ellipsoid. The size and shape of the ellipsoid will be modified to be the same as the ellipsoidal kernel *E* by modifying the diagonal matrix *A* as:

$$A=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & r \end{bmatrix}, \tag{15}$$

r is half of the length of the minimum axis of the ellipsoidal kernel *E*. After the density value is estimated, we remove all the points whose estimated density value is smaller than a user-defined threshold.

1.2.3 Orientation propagation by minimum spanning tree

The normal estimated by the PCA algorithm might not be consistently oriented as there is an ambiguity of 180 degree of the obtained eigenvector. Thus we need to conduct an orientation propagation algorithm based on minimum spanning tree. This is done in two steps. First, within each volumetric grid cell, we conduct clustering based on the normal information, and choose the orientation of the largest cluster as the orientation within the volumetric grid cell. Then we compute the average normal within each grid cell and build a graph with each grid cell (containing points) as a node, and the weights of edges between the adjacent grid cell are defined as 1- ||n1 * n2||, where n1 and n2 are the average normal of the two adjacent grid cells.

The minimum spanning tree is implemented based on Kruskal's algorithm, which is an algorithm in graph theory that finds a minimum spanning tree for a connected weighted graph. This means it finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized. If the graph is not connected, then it finds a minimum spanning forest (a minimum spanning tree for each connected component). Kruskal's algorithm is an example of a greedy algorithm. It works as follows:

```
1. create a forest F (a set of trees), where each vertex in the graph is a separate tree
2. create a set S containing all the edges in the graph
3. while S is nonempty
    a. remove an edge with minimum weight from S
    b. if that edge connects two different trees, then add it to the forest, combining two trees into a single tree
    c. otherwise discard that edge.
```

At the termination of the algorithm, the forest has only one component and forms a minimum spanning tree of the graph. This algorithm first appeared in 1956 in the Proceedings of the American Mathematical Society (page 48–50), written by Joseph Kruskal [5].

1.2.4 Density and normal propagation

Once we obtain a consistent orientation of the normal, we propagate the normal and density to the whole volumetric grid. More specifically, we first embed the LiDAR data by a volumetric grid, and conduct the aforementioned Parzen-window based kernel density estimation for each LiDAR point based on its Mahalanobis distance to its neighboring LIDAR data (Equation (10)). We then propagate the density at individual points into the volumetric grids based on the same anisotropic formula. In other words, we create a volumetric density field, with the density value corresponding to the probability of belonging to the iso-surface of the object.

1.2.5 Non-maximum suppression based 3D surface reconstruction

The unsigned density field is then converted into a signed indicator function by calculating the inner product of the normal vector at the grid node with the gradient of the density field at the grid node. Finally, we employ iso-surface extraction algorithm such as Marching Cubes to extract the iso-surface, indicated by the zero-value of the signed indicator function.

1.3 3-D visualization

We developed a prototype software system based on OpenGL® and Visual C++® that can automatically reconstruct the 3D scene of the interior of a building or other structure from point clouds acquired by the ground-based LIDAR scanner. We developed a user-friendly Graphical User Interface (GUI) that allows the users to interactively visualize, navigate and walk through the room from different view angles, zoom in and out, etc. The reconstructed 3D scene can be exported in the "OBJ" data format that is fully compatible and exportable to other commercial visualization software. Moreover, we also developed an additional functionality in the prototype software system: "automatically generating the floor plan of the building". This additional functionality can be very useful for the user to have a high level understanding of the interior of the building and is complementary to the interactive 3D visualization provided by our software system.

2. CONCLUSION

We developed a prototype system that can automatically reconstruct 3D scenes of the interior of a building, cave or other structure using ground-based LIDAR scanning technology. We develop a user-friendly real-time visualization software package that will allow the users to interactively visualize, navigate and walk through the room from different view angles, zoom in and out, etc.

REFERENCES

[1] M. A. Fischler and R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". Comm. of the ACM 24: 381-395, 1981.

[2] D. Comaniciu and P. Meer, " Mean Shift: A Robust Approach Toward Feature Space Analysis". IEEE Transanctions on Pattern Analysis and Machine Learning Vol. 24, No. 5, pp. 603-619, May 2002.

[3] Ye Duan, Liu Yang, Hong Qin, and Dimitris Samaras, "Shape Reconstruction from 3D and 2D Data Using PDE-Based Deformable Surfaces". Proceedings of The 8th European Conference on Computer Vision (Computer Vision - ECCV 2004), Part III, May 11-14, 2004, Prague, Czech Republic, pages 238 -- 251 (Lecture Notes in Computer Science 3023).

[4] Richard O. Duda, Peter E. Hart, and David G. Stork, "Pattern Classification", Wiley-Interscience, 2nd edition, October 2000.

[5] Joseph. B. Kruskal, "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem". Proceedings of the American Mathematical Society, Vol 7, No. 1, pp. 48–50, Feburary 1956.

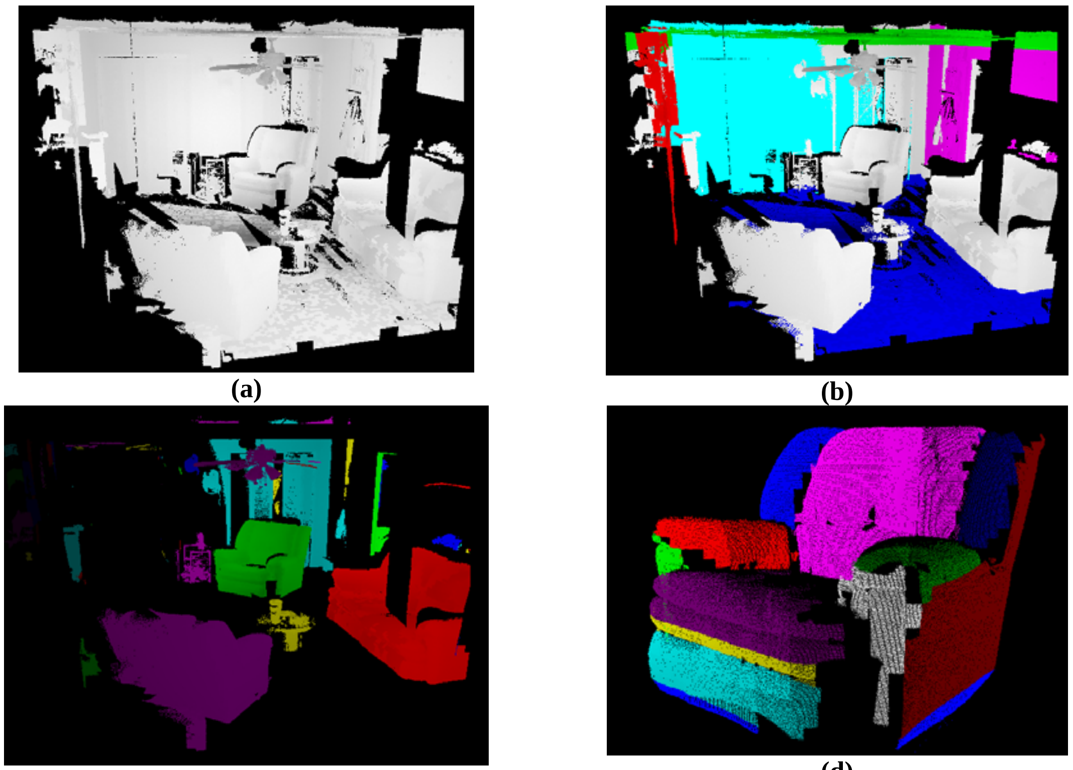
(a) (b) (c) (d)

Figure 2: Hierarchical LIDAR data segmentation. (a) Original LIDAR data; (b) Floor (blue color), ceiling (green color), and vertical walls (red, cyan and magenta colors) identified; (c) Individual objects extracted (shown in different colors); (d) Each object is segmented into homogenous clusters (shown in different colors).