Think Globally, Cluster Locally:
A Unified Framework for Range Segmentation

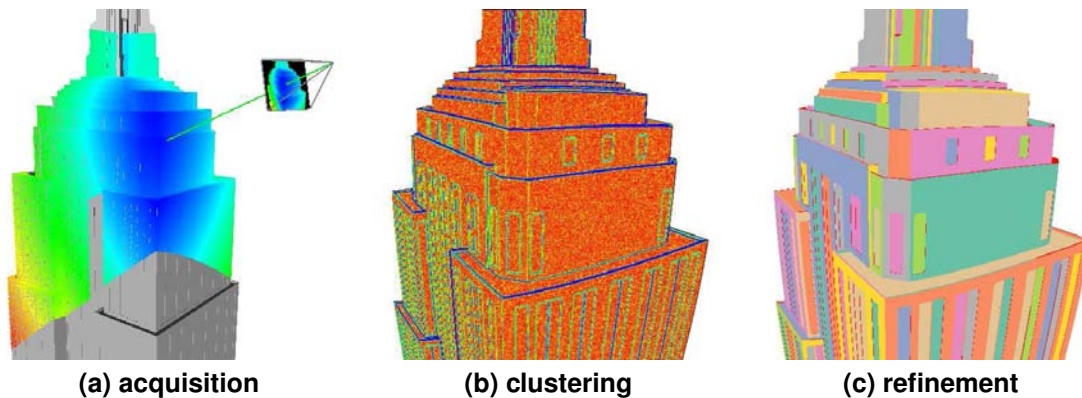Gene Yu   Michael Grossberg   George Wolberg   Ioannis Stamos[†][*]

Dept. of Computer Science
City College of New York / CUNY
New York, NY 10031

[†] Dept. of Computer Science
Hunter College / CUNY
New York, NY 10065

| (a) acquisition | (b) clustering | (c) refinement |

Figure 1. Segmentation pipeline. (a) In this simulated LIDAR setup, the frustum represents a scanner projecting a beam onto a 3D model. The beam strikes the nearest surface and measures the distance, rendered here in false color. (b) Similarity based on local plane fitting drives a hierarchical clustering process. (c) Planar components are refined and merged using a variant of the k-means algorithm.

Abstract

Modern range scanners can capture the geometry of large urban scenes on an unprecedented scale. While the volume of data is overwhelming, urban scenes can be approximated well by parametric surfaces such as planes. Piecewise planar representation can reduce the size of the data dramatically. Furthermore, it is ideal for rendering and other high-level applications. We present a segmentation algorithm that extracts a piecewise planar function from a large range image. Many existing algorithms for large datasets apply planar criteria locally to achieve efficient segmentations. Our novel framework combines local and global approximants to guarantee truly planar components in the output. To demonstrate the effectiveness of our approach, we present an evaluation method for piecewise planar segmentation results based on the minimum description tion length principle. We compare our method to region growing on simulated and actual data. Finally, we present results on large scale range images acquired at New York's Grand Central Terminal.

1. Introduction

Many papers in the range segmentation literature present algorithms that have advantages when applied to small meshes or to simple closed objects. Sophisticated approaches are possible when the number of points is limited or the topology is known, but the large, complex datasets produced by modern range scanners favor greedy local approaches such as region growing. The problem with such methods is that they do not produce a truly planar segmentation. Locally planar criteria drive the segmentation process, but they cannot guarantee output components that are verifiably planar in a global sense. We present a novel framework for enforcing planarity both locally and globally to produce a segmentation composed of output components that can be replaced by polygons with small error. Figure 1 illustrates our segmentation pipeline. First, a range image is acquired by a scanner. A local clustering phase groups points together if they are likely to belong to the same plane. Then, we refine the set of planes in a global setting using a variant of the k-means algorithm. Our approach shows that a

[*]email: {yu,grossberg,wolberg}@cs.ccny.cuny.edu, istamos@hunter.cuny.edu

combination of simple techniques can be used to produce high-fidelity segmentation results at a manageable cost.

We borrow liberally from the image segmentation literature to help us manage the cost of our algorithm. Image segmentation and range segmentation, while closely related, are not interchangeable. Image segmentation methods typically use the graph topology of the image grid. In current range segmentation work, such as moving least squares and other point-based methods, the image topology is often disregarded. The usual range data processing pipeline proceeds by performing registration and combining the scans first. Segmentation follows in unorganized point clouds. We feel strongly that segmentation should come first. Thus, we perform segmentation on individual range images and postpone the integration of multiple scans.

Since our goal is to replace planar components with polygonal approximants, we evaluate our results by measuring the fidelity of the output function to the original data. We develop a novel method for comparing piecewise planar segmentations based on the minimum description length principle. By encoding our output for compression, we can compare our results directly to the original dataset and to other segmentations by computing compression ratios. Tests on simulated range images show that a piecewise planar representation can approach the ground truth segmentation, while tests on actual range images show a corresponding empirical improvement over region growing.

2. Related Work

Region growing is commonly applied to large range datasets because it is fast and easy to implement [2, 3, 9]. Several methods to augment basic region growing use global techniques in a similar spirit to our work. Gotardo *et al.* [8] employ robust statistics to extract seed regions, while Bab-Hadiashar and Gheissari [1] group inliers by analyzing residuals from parametric fitting. We also perform statistical clustering, but instead of shrinking clusters to seed regions, we use the clusters directly, similar to the graph-based method of Felzenszwalb and Huttenlocher [7]. Their greedy similarity measure is based on minimum spanning trees, while our iterative refinement, though not as fast, gives us greater control over the output.

Our refinement phase, like all iterative methods, converges more quickly the closer the initial guess is to the final result. Therefore, we use an anisotropic filter to improve the local plane fit that drives the clustering process. The bilateral filter introduced by Tomasi and Manduchi [13] combines weighting functions in orthogonal domains to create an edge-aware convolution kernel. Choudhury and Tumblin [4] use geometric information in the form of piecewise linear approximation to improve the bilateral filter near edge discontinuities. In our purely geometric context, we apply

piecewise linear approximation to range images using a local plane fit based on moving least squares [12].

After clustering the points into connected components, we refine the components using a variant of the k-means algorithm. A form of k-means iteration was used by Cohen-Steiner *et al.* [5] to simplify piecewise planar manifolds. Because the topology is known, they are able to choose the number of components k arbitrarily. Our refinement phase expands the applicability of such an approach to the topologically complex surfaces present in raw range images by fixing k automatically during the clustering process.

To evaluate the final result, we measure the space savings and the fidelity of the segmentation using a minimum description length (MDL) criterion. Darrell *et al.* [6] used MDL locally to allow neighboring components to compete for boundary points. Our method also allows competition between neighboring components, but using global plane fitting directly ensures that a point belongs to a given component only if it fits the output representation. Once we know that the output truly matches the desired type, we apply MDL globally to evaluate the final representation.

3. Segmentation Algorithm

Our algorithm proceeds in two distinct phases. The first is based on local surface fitting, while the second is based on global surface fitting. A block diagram illustrating our approach is presented in figure 2. By allowing local and global
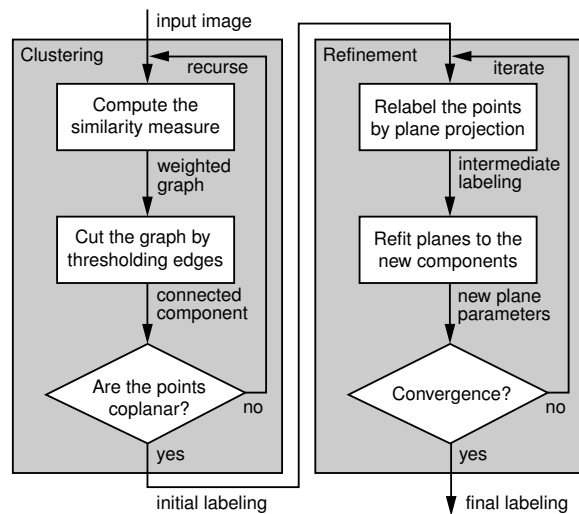
Figure 2. Block diagram. On the left, graph-based clustering based on local surface fitting generates a hierarchy of connected components. Then, on the right, we traverse the hierarchy bottom-up, refining the subtree at each internal node by fitting global surfaces to the individual components.

notions of planarity to communicate, we expand the range of evidence available at each point to determine its membership in a component. Using only local relations, boundaries cannot be resolved cleanly due to ambiguities at corners and depth discontinuities. In addition, local variations in scale and sampling rate make suitable thresholds difficult to determine, since no single set of parameters can be optimal in all areas of the image. By contrast, purely global methods require additional information beyond the input data points, such as the number of components we expect to find. We must bootstrap global methods by aggregating local surface features or by applying some form of domain knowledge. Our method is motivated by the observation that local methods can achieve a partial solution to the planar segmentation problem while providing exactly the information needed to initiate a global process of refinement.

The purpose of the first phase of our algorithm is to build a hierarchical subdivision of the input image such that the leaf nodes of the resulting tree represent planar connected components. The second phase then traverses the tree bottom-up, refining the plane parameters for each component and merging components where possible. Afterward, the set of discovered planes is removed from the dataset, and the process is repeated on the remaining points. Since the distribution of similarity weights changes at each iteration, repeated passes target surfaces with different error characteristics. Thus, we can distinguish planes at different depths and sampling rates, and made of different materials.

3.1. Clustering

Computing the similarity measure. Local surface fitting in actual range data is typically performed using smoothing operators to combat noise. We use a form of moving least squares (MLS) approximation to fit planes in small neighborhoods around each point. Because MLS approximants are weighted operators that decay with distance, we can achieve a degree of anisotropy that improves the initial clustering over simpler isotropic convolution operators. In our formulation, a local frame is represented by a tuple $\phi = (\overline{\mathbf{x}}, \mathbf{n})$ with origin $\overline{\mathbf{x}}$ and normal vector $\mathbf{n}$. MLS projection maps any point $\mathbf{x}$ within a radius h of the input data to a point $\overline{\mathbf{x}}$ on the MLS surface (fig. 3a). To determine the radius h, we choose a window size w and use the points in a $w \times w$ neighborhood in the image grid. This choice has two practical advantages:

1. There is no need to build either explicit connectivity or search data structures such as kd-trees.
2. Neighborhoods scale automatically to the sampling rate of the scanner.

The global threshold w is more robust than choosing h directly would be, because the derived radius scales gracefully to adjust for the spread of the scanner beam at different

(a) MLS projection.

(b) Point-to-point distance.
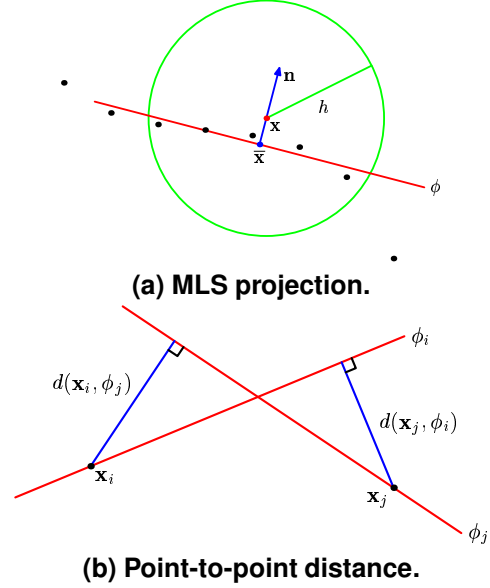
Figure 3. Similarity. (a) Project $\mathbf{x}$ to the local frame $\phi = (\overline{\mathbf{x}}, \mathbf{n})$ induced by the neighborhood of radius h around $\mathbf{x}$. The origin is $\overline{\mathbf{x}}$, and the normal is $\mathbf{n}$. (b) Distance d_{ij} between $\mathbf{x}_i$ and $\mathbf{x}_j$ is the sum of point-to-plane distances $d(\mathbf{x}_i, \phi_j)$ and $d(\mathbf{x}_j, \phi_i)$. Similarity is the probability, given d_{ij}, that $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same plane.

depths. It is only necessary to determine the general level of noise in the image, which is related to the error characteristics of the scanner. For the Leica HDS 2500 [11] scanner used in our study, a small 5×5 window is generally satisfactory, with 7×7 or 9×9 windows being useful sometimes if the surfaces in the image are bumpy or if they are made of absorptive materials that partially degrade the signal.

The local frame at $\mathbf{x}$ is determined by minimizing the weighted least squares functional

$$\mathcal{J} = \sum_{i=1}^{w \times w} \big[(\mathbf{x}_i - \mathbf{x}) \cdot \mathbf{n} \big]^2 \theta(\|\mathbf{x}_i - \mathbf{x}\|), \qquad (1)$$

where θ is the weighting function. We define θ using the tail probability of the normal distribution

$$\theta(t) = 1 - P(t), \qquad (2)$$

where $P(t) = (\sigma\sqrt{2\pi})^{-1} \int_{-\infty}^{t} \exp\big[-(u-\mu)^2/(2\sigma^2)\big]\, du$ is the cumulative distribution function with mean μ and standard deviation σ. The tail probability is the probability that a normally distributed random variable exceeds t. We prefer the tail probability to simpler weighting functions because it never decays so rapidly that all of the weights vanish. Parameters μ and θ are determined by fitting a normal

3

distribution to the distance values $\|\mathbf{x}_i - \mathbf{x}\|$. We minimize the MLS functional (eq. 1) at $\mathbf{x}$ by computing the singular value decomposition (SVD) of the weighted covariance matrix of the points $\mathbf{x}_i$ in the neighborhood of $\mathbf{x}$, centered at their mean position $\overline{\mathbf{x}}$. The SVD produces the matrix $\mathbf{V}^T$ of right eigenvectors containing the normal vector $\mathbf{n}$ corresponding to the smallest eigenvalue. In addition, since $\mathbf{V}^T$ encodes the rotation from world coordinates to the local frame, it is convenient to compute the perpendicular distance from a point $\mathbf{x}$ to a plane ϕ as

$$d(\mathbf{x}, \phi) = \mathbf{V}^T (\mathbf{x} - \overline{\mathbf{x}}) \cdot \mathbf{n}. \qquad (3)$$

To compute the distance between two points $\mathbf{x}_i$ and $\mathbf{x}_j$ (fig. 3b), we sum two perpendicular projections

$$d_{ij} = |d(\mathbf{x}_i, \phi_j)| + |d(\mathbf{x}_j, \phi_i)|. \qquad (4)$$

Cut the graph by thresholding edges. The search for planar components begins by representing the image as a weighted graph, then removing edges from the graph by histogram thresholding. We compute point-to-point distance (eq. 4) for each edge between a pair of 4-connected vertices in the image. Then we generate similarity weights by fitting a normal distribution to the distance values and scoring edges using the tail probability (eq. 2). The changing distribution of weights makes it possible for repeated application of the clustering method to find different sets of planes. By contrast, region growing would produce the same components on each pass no matter how many points were removed from the dataset, unless some mechanism for automatically adjusting the parameters were devised.

By generating a histogram, it is easy to determine a suitable threshold automatically by the midpoint method. First, we initialize the threshold to the mean weight. Then, we partition the weights into two groups above and below the threshold, and compute the mean of each group. The mean of the two means becomes the new threshold, and the process iterates until convergence. Implicitly, we are positing that the distribution is bimodal. One mode corresponds to all edges connecting coplanar points, while the other mode corresponds to all edges that cross discontinuities.

Once the threshold is chosen, removing edges with weight below the threshold cuts the graph into connected components. Then, we filter the components, searching for planar pieces. A component is rejected immediately if it has fewer points than some minimum number m. Otherwise, we treat each component as a candidate plane by fitting plane parameters to its constituent points. To estimate the parameters, we construct an unweighted covariance matrix in homogeneous coordinates, and solve for the eigenvector corresponding to the smallest eigenvalue using SVD. The eigenvector is a plane equation in point-normal form $\Phi = (\mathbf{n}, p)$, where $\mathbf{n}$ is the surface normal and p is the perpendicular distance to the origin. The distance between a
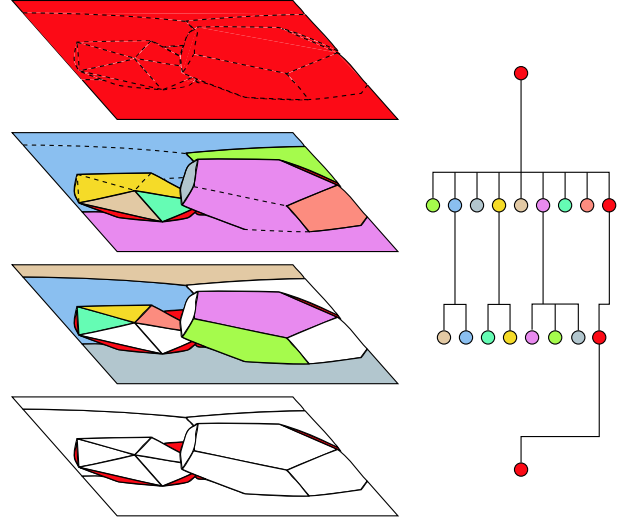
Figure 4. Three levels of hierarchical clustering from top to bottom. Left: Initially, all points are nonplanar (red) and connected by weighted edges. Removing low-weight edges cuts the image into components. Planar components (white) are removed, while the remaining components are processed recursively. Right: hierarchical subdivision of regions shown as a tree structure.

point $\mathbf{x}$ in space and a global plane Φ is

$$D(\mathbf{x}, \Phi) = \mathbf{n} \cdot \mathbf{x} + p. \qquad (5)$$

If the standard error of the residuals is smaller than a predetermined tolerance τ, then the candidate is planar. We recurse on the nonplanar candidates, creating a hierarchy of nested components with the planar components stored in the leaves (fig. 4). The global threshold m limits the depth of the component hierarchy. It can be determined by cursory inspection of the data, such that m corresponds to the smallest planar component desired. The threshold τ can be set according to the manufacturer's error estimate for the scanner used to create the image. For the HDS 2500, we used $\tau = 6$mm for all of the examples shown, based on the positional accuracy provided by Leica.

3.2. Refinement

After constructing the hierarchy, refinement proceeds bottom-up by the k-means method using objective function

$$J = \min_{\{C_i\}_{i=1}^k} \sum_{i=1}^k \sum_{\mathbf{x} \in C_i} D(\mathbf{x}, \Phi_i), \qquad (6)$$

where the C_i are disjoint components. The children of each node of the component hierarchy partition its points into

connected subspaces, each containing a set of plane parameters that are fitted to the constituent points. By alternately relabeling the points, then updating the parameters, we iteratively converge to the largest and most accurate planar components we can find.

Relabel the points by plane projection. During bottom-up traversal, we update membership in the current node by projecting each point to the set of child planes and allowing the point to join the closest child. Most points rejoin the same child, but adjacent components compete along their boundary, which makes our algorithm particulary effective at corners. Planarity is enforced using a threshold $\tau' = \min(\tau, 3\sigma_i)$ where σ_i is the standard error of the ith component. Using a multiple of the standard error tends to clip peaks, which we repair in a post-processing step by dilating components subject the threshold $\tau'' = (n + 1)\sigma_i$ where n is the number of neighbors that belong to component i. This constraint fills small holes liberally, but restricts growth along the boundaries.

Refit planes to the new components. After relabeling the points, we recompute the plane parameters for each child to reflect the change in membership, repeating the process until the membership stabilizes. Then, we remove the parent, promote the child segments one level, and continue traversing upward to the root. As we travel up the tree, we fit our candidate planes to larger and larger sets of points, and our confidence in the accuracy of the fit grows.

Overfitting occurs when adjacent, coplanar components are created in different branches of the hierarchy. Redundant planes are also removed during post-processing by testing the fit of each plane to the points in adjacent components. We use global point-to-plane distance (5) to project the points from one component into the plane of the other, merging components if the error distributions agree.

3.3. Complexity Analysis

In the best case, our algorithm is a typical recursive divide-and-conquer algorithm, which runs in $\Theta(n \lg n)$ time. Unfortunately, we usually cannot label all of the points on the first pass. Consider two planes meeting at a corner. Local neighborhoods degrade in quality as they approach the corner because outliers cross the edge discontinuity. However, if one of the surfaces can be extracted, its points no longer pollute the distribution of the other. Therefore, repeated passes often reveal finer levels of detail.

In the worst case, one component containing the minimum number of points is extracted on each pass, and total the running time is $O(n^2)$. To investigate the average case, we segmented 25 actual range scans from Grand Central Terminal at their full size of 999×999 pixels. We also tiled the images into 1×2, 2×2, 3×3, 4×4, 5×5, and 8×8 regions. Results appear in figure 5. For tiled images, we recorded only the worst running time encountered. Error
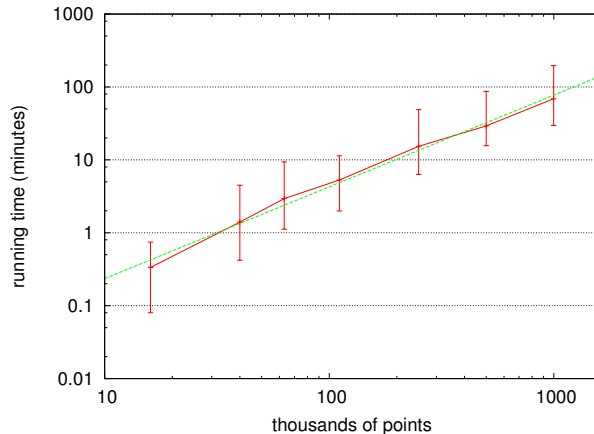
Figure 5. Timing. Log-log plot of number of points vs. mean running time for 25 scans on a 3GHz Intel Xeon 5160 processor with $w = 5$, $\tau = 6$mm and m ranging from 400 points at full size down to 40 for the smallest tile. Error bars indicate minimum and maximum times. The best-fitting line has a slope of 1.3.

bars indicate minimum and maximum times—the spread reflects dependence on image content. A straight line through the mean values in the log-log graph suggests an average case power law relationship with a slope of 1.3. The average case running time appears to fall closer to the best case.

Our algorithm is implemented in C++ on a Linux system with data structures and procedures from the Standard Template Library and Boost. For linear algebra, we use the hardware-optimized ATLAS library integrated with CLA-PACK. We compute the right tail probability (eq. 2) using the GNU Scientific Library.

Finding the major planes in 25 range images of Grand Central Terminal took an average of 69 minutes on a 3GHz Intel Xeon 5160 processor. The complex example in figure 6 took three hours to find major planes covering 60% in the image (see inset). On subsequent runs, we increased the window size to approximate the curved ceiling, and we decreased the window size to capture fragments formed by partial occlusion. A large $\tau = 2$cm captured the fenestrated panels at either end of the Main Concourse. Since m affects the depth of the component hierarchy, it can have a great impact on running time. Nevertheless, the Grand Central Terminal dataset is highly complex, so we included $m = 50$, which took over 7 hours to run on average. Even with such a small number of points per component, however, our method was able to avoid nonplanar areas such as the hanging lamps in figure 6 robustly.
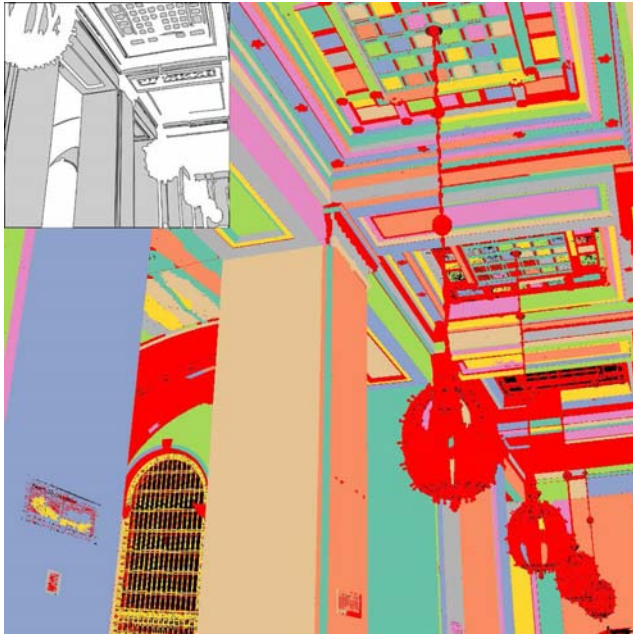
5

Figure 6. Segmentation results for a 999 x 999 range image of Grand Central Terminal. Black denotes holes where the scanner failed to return a value. Red denotes points classified as nonplanar by our algorithm. Eight other colors are assigned randomly to differentiate the planar regions. The inset shows the major planes found in the first of four runs ($w = 5$, $\tau = 6$mm, and $m = 400$).

4. Experimental Results

Hoover *et al*. [9] first compared planar segmentations using a collection of heuristic counts. More recently, Jiang *et al*. [10] examined statistical distance measures, including an information-theoretic distance derived from mutual information. In the same vein, we designed a practical information-theoretic evaluation procedure based on MDL, using the byte coder bzip2 as a proxy for entropy. We parameterize each component in a coordinate frame that minimizes the variance in z. If the constituent points are truly coplanar, the z-coordinates of all such points will be close to zero in parameter space. After quantization, the greater proportion of zero bytes in the data stream leads to a better compression ratio. For each component,

1. Fit a least squares plane.
2. Transform each point to the local coordinate frame.
3. Quantize the transformed values.

After transforming each component, we write the quantized values to a flat file, and compress the file using bzip2.
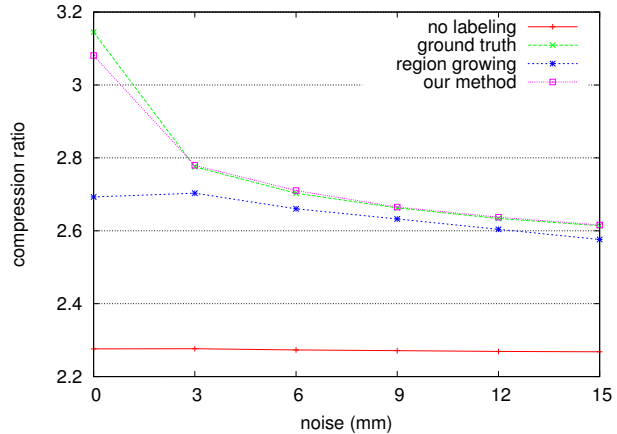

Figure 7. Evaluation. Minimum description length comparison between region growing and our method on simulated LIDAR images with fixed amounts of white noise added.
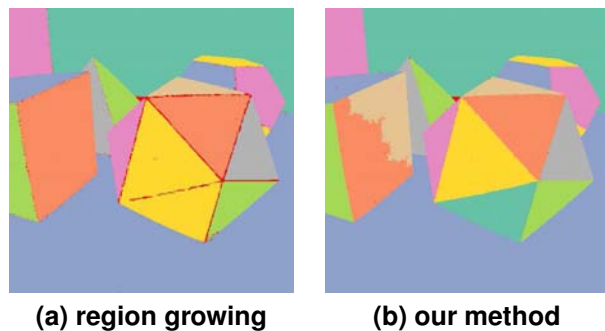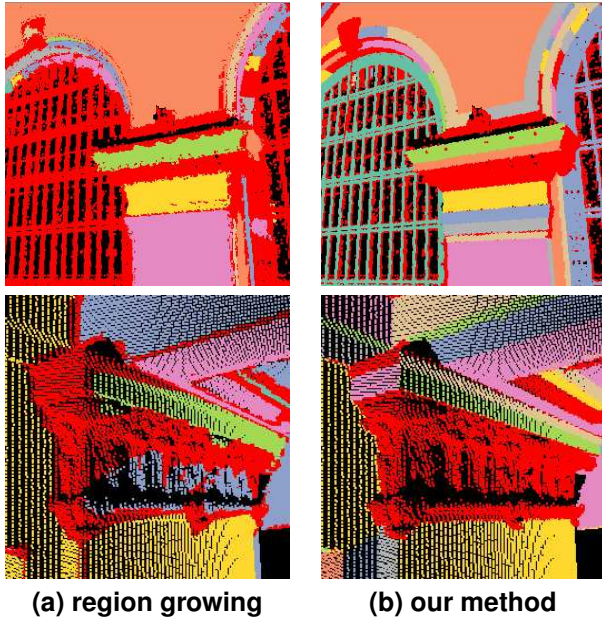

(a) region growing   **(b) our method**

Figure 8. Failure modes. Region growing tends to underfit, while our method is more likely to overfit. (a) One component spans two faces of the icosahedron. (b) One face of the cube is described by two components.

We compared our results to the region growing method of Chen and Stamos [3]. Figure 7 shows the results of direct comparison on simulated LIDAR images. Since the simulated examples are constructed to match our assumptions about the surfaces in a range image, our algorithm predictably approaches the ground truth compression ratio. Cases in which our method seem to beat ground truth correspond to overfitting. Region growing, on the other hand, consistently undersegments. A comparison of failure modes is presented in figure 8.

Results on actual data are more difficult to interpret, because region-growing treats smoothly curving surfaces such as the ceiling of the Main Concourse as one connected component, while our algorithm cuts curved surfaces into planar pieces. Nevertheless, our results were consistent with the

(a) region growing   **(b) our method**

Figure 9. Actual data. 3D point rendering of two detail comparisons between region growing and our method on range images from Grand Central Terminal. Our method produces tight corners, and it correctly distinguishes parallel planes that are separated by small depth discontinuities.

simulated results in 12 trials on actual data. On simulated data with 6mm of noise, our method achieved an average compression ratio of 2.71 vs. 2.66 for region-growing. On actual data, our method achieved a ratio of 2.72 vs. 2.47 for region-growing. Two examples from Grand Central Terminal are rendered using 3D points in figure 9. These areas contain both planes and sculptural detail. Our method extracts planes with great precision, especially in corners, and it is able to distinguish difficult cases such as parallel planes that are separated by small depth discontinuities. Additional examples on full-size scans are presented in figure 10.

5. Conclusion

The main contribution of our work is a unified framework for range segmentation utilizing local and global methods driven by a single model. In our novel framework, a simple hypothesis test negotiates with a simple iterative refinement step to narrow down a huge range of possible scales, orientations, and noise levels.

Many previous segmentation methods are not directly comparable to our algorithm because they do not yield a planar segmentation. They do not distinguish between flat and curved surfaces, they group non-coplanar regions, and they perform poorly on edges. The remedy we propose is

to evaluate the output components dynamically. By using a single model both to select components locally, and to evaluate those components globally, we can resolve highly ambiguous configurations of points. It is this unification of local and global methods around a single model that allows us to produce a truly planar segmentation.

Acknowledgements

References

[1] A. Bab-Hadiashar and N. Gheissari. Range image segmentation using surface selection criterion. *IEEE Trans. Image Process.*, 15(7):2006–2018, 2006.

[2] P. J. Besl and R. C. Jain. Segmentation through variable-order surface fitting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(2):167–192, 1988.

[3] C. Chen and I. Stamos. Range image segmentation for modeling and object detection in urban scenes. In *Intl. Conf. 3-D Digital Imaging and Modeling (3DIM)*, pages 185–192, Montreal, Canada, 2007.

[4] P. Choudhury and J. Tumblin. The trilateral filter for high contrast images and meshes. In *Proc. Eurographics Symp. Rendering*, pages 186–196, Leuven, Belgium, 2003.

[5] D. Cohen-Steiner, P. Alliez, and M. Desbrun. Variational shape approximation. In *Proc. SIGGRAPH*, pages 905–914, Los Angeles, CA, 2004.

[6] T. Darrell, S. Sclaroff, and A. Pentland. Segmentation by minimal description. In *Proc. Intl. Conf. Computer Vision (ICCV)*, pages 112–116, Osaka, Japan, 1990.

[7] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *Intl. J. Computer Vision*, 59(2):167–181, 2004.

[8] P. F. U. Gotardo, K. L. Boyer, O. R. P. Bellon, and L. Silva. Robust extraction of planar and quadric surfaces from range images. In *Proc. Intl. Conf. Pattern Recognition (ICPR)*, pages 216–219, Cambridge, England, 2004.

[9] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon, and R. B. Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(7):673–689, 1996.

[10] X. Jiang, C. Marti, C. Irniger, and H. Bunke. Distance measures for image segmentation evaluation. *EURASIP J. Applied Signal Processing*, 2006:1–10, 2006.

[11] Leica Geosystems. http://www.leica-geosystems.com.

[12] D. Levin. Mesh-independent surface interpolation. In G. Brunnett, B. Hamann, and K. Mueller, editors, *Geometric Modeling for Scientific Visualization*, pages 37–49. Springer-Verlag, Berlin, 2003.

[13] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *Proc. Intl. Conf. Computer Vision (ICCV)*, pages 839–846, 1998.
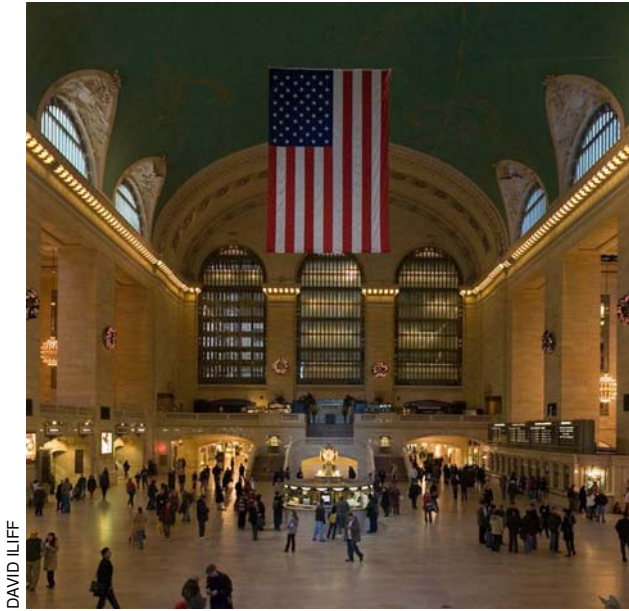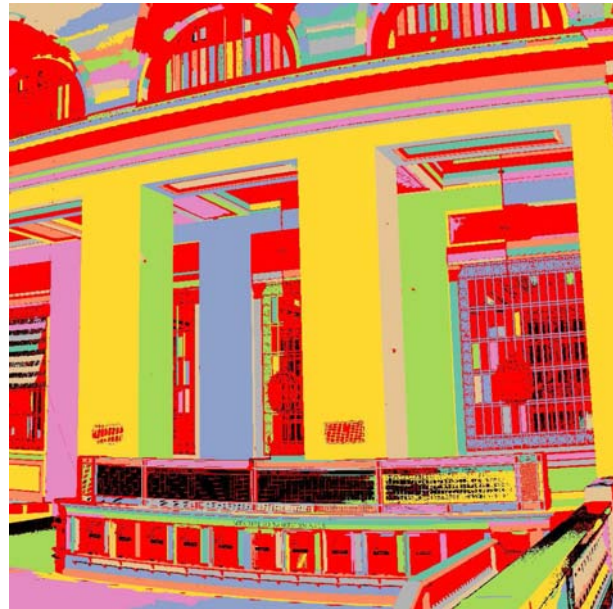
(a) Photograph of the Grand Concourse.
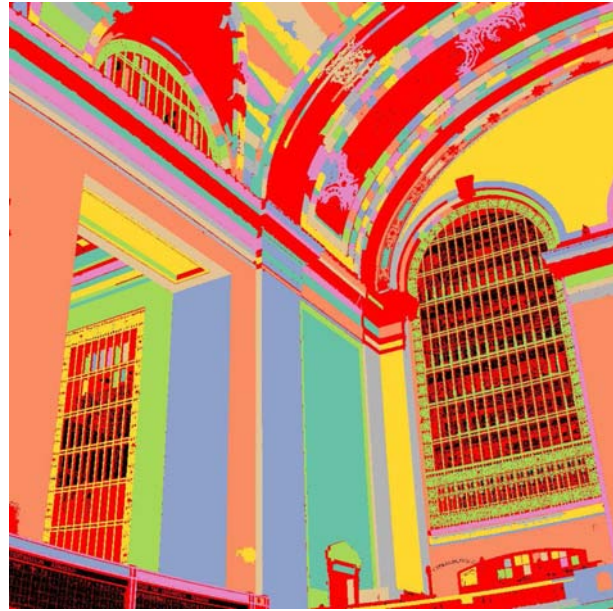
(b) Southwest corner.
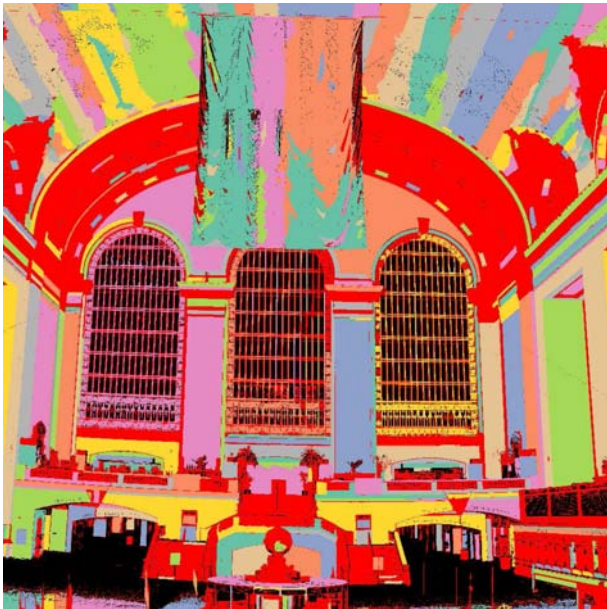
(d) East end / Entrance to 42nd St. passage.

(c) South side / Ticket booths.

Figure 10. Segmentation results for 999 x 999 range images taken inside Grand Central Terminal. Black denotes holes where the scanner failed to return a value. Red denotes points classified as nonplanar by our algorithm. Eight other colors are assigned randomly to the planar components to differentiate the regions. (a) A photograph of the Grand Concourse, facing east. (b) Planar areas yield high fidelity components with sharp corners. Our method can also handle noisy components with complex topology, such as the large fenestrated panels. (c) Piecewise planar approximation of smooth curved surfaces such as the vaulted ceiling are easy to generate. (d) Highly nonplanar areas such as ticket windows that cannot be approximated well by piecewise planar components are correctly identified and excluded.