Rectilinear Parsing of Architecture in Urban Environment

Peng Zhao[1], Tian Fang[1], Jianxiong Xiao[2], Honghui Zhang[1], Qinping Zhao[3], and Long Quan[1]
[1]CSE/HKUST [2]CSAIL/MIT [3]VRLab/BUAA

Abstract

We propose an approach that parses registered images captured at ground level into architectural units for large-scale city modeling. Each parsed unit has a regularized shape, which can be used for further modeling purposes. In our approach, we first parse the environment into buildings, the ground, and the sky using a joint 2D-3D segmentation method. Then, we partition buildings into individual façades. The partition problem is formulated as a dynamic programming optimization for a sequence of natural vertical separating lines. Each façade is regularized by a floor line and a roof line. The floor line is the intersection line of the vertical plane of buildings and the horizontal plane of the ground. The roof line links edge points of roof region. The parsed results provide a first geometric approximation to the city environment, and can be further analyzed if necessary. The approach is demonstrated and validated on several large-scale city datasets.

1. Introduction

In recent years, large scale 3D city modeling from images has been receiving more and more attention due to the popularity of digital earth applications, e.g. Google Earth and Microsoft Virtual Earth. Image parsing is a crucial step in image-based modeling [18, 19, 15]. The state-of-the-art successfully analyzed and modeled façades from images [11]. However, it is only applicable to single façade. To adapt such modeling techniques to large-scale city reconstruction, it is crucial to automate the process of detecting and extracting façades.

Unfortunately, parsing general scenes [3] from arbitrary images is still one of the most difficult problems in computer vision. Researchers are also trying to parse specific kinds of images, e.g. scenes with man-made objects [6] and with architectures [2], more accurately. Our work, which concentrates on urban environment parsing, falls into this category.

Without using Structure from Motion (SfM), Hoiem et al. [8] obtained promising results by considering several spatial cues found in single images. Berg et al. [2] shown a general model with carefully selected features that can parse the architectures into detailed components surprisingly well. These methods parse scenes to either coarse classes, such as ground and buildings, or fine details, such as roofs, windows and doors. In contrast, our system aims at parsing architectural units, i.e. façades, which are between coarse classes and fine details. Our system not only finds and reconstructs rectilinear regions but also groups them into architectural units semantically.

In our system, the 3D point cloud, 3D re-triangulated lines and camera parameters are first reconstructed as preprocessing results. Similar to some other image parsing methods [8, 2] designed for architectural scenes, we limit the types of objects and focus on three main categories: buildings, ground, and sky. Ground and buildings are assumed to be piecewise planar and provide strong geometry priors and constraints throughout our method. Therefore, inspired by the segmentation method using motion point clouds [4], we use 3D features to estimate the ground and buildings which in turn help to parse other objects. Knowing the building and ground region, the accurate sky region is detected. As for the building region, most of the existing methods cannot parse it into individual façades, we thus formulate this problem as to find the optimal partition of a 1D graph and solve it using dynamic programming. Furthermore, because buildings have regular structure, we introduce an algorithm to regularize roof lines of buildings. Towards the end, we discuss each component of the pipeline.

The major contributions of our work are as follows. (1) We utilize the knowledge of the street-side to obtain superior results. We select superior features to parse urban scenes. (2) To our best knowledge, we are the first to formulate and give a solution to the problem of automatically partitioning the building region into individual façades. This is an important subproblem in 3D city modelling, and its applications are quickly gaining importance. (3) By using the shape regularity of buildings, we develop an approach for floor line and roof line regularization. Compared with existing methods [4], the boundaries among buildings, sky, and ground are well preserved.

978-1-4244-6983-3/10/$26.00 ©2010 IEEE

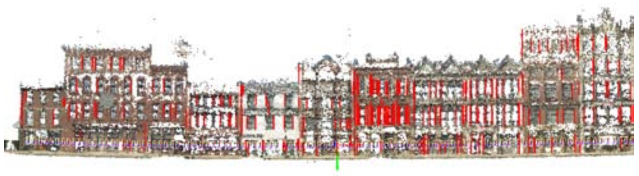


Figure 1. Points from SfM and re-triangulated vertical lines

2. Registration and Pre-processing

2.1. Points

A standard 3D reconstruction algorithm [7] is used to compute all camera poses and a 3D point cloud from a given sequence of images. The GPS data, if available, from street view imageries are also used to initialize the camera poses. To increase the density of the reconstructed 3D points, we compute a disparity map for each pair of images and use a re-sampling strategy inspired by the quasi-dense approach in [10, 9] to produce a semi-dense set of 3D points.

2.2. Lines

On each image, we perform the Canny edge detection [5] and link the detected edge points to form line segments. Then we check whether the line segments pass through the common vanishing point using the RANSAC method [14] and group the line segments according to three orthogonal vanishing points. Then, we use the quasi-dense pixel matching [10] between each pair of images to match line segments.

During the line extraction, lines are often broken into several segments. We post-process the extracted line segments to merge them into long line segments and extend their endpoints to the farmost point that they can achieve in multiple view [1].

The horizontal line segments in space usually lack of horizontal parallax. Therefore, we reconstruct only vertical lines if they are tracked over more than three views. Vertical lines are reconstructed reliably using RANSAC. Each line reconstruction hypothesis is generated by 3 line segments in different views. The hypothesis is evaluated by using the average reprojection error in all corresponding images. All the inliers of line segments are used to reconstruct the resulting line.

One example result is illustrated in Figure 1. Our following algorithm assumes that each sequence has one dominant plane as in the one shown in Figure 1. Thus, when the route along the street makes a turn at a corner, the sequence is broken into two different sequences by detecting such a turn.

3. Environment Parsing

In this section, we classify the scenes into three categories: buildings, ground and sky. First, we utilize the vertical lines and the 3D point cloud to fit the building plane and ground plane using RANSAC [14]. One floor line is detected in each image as the separator between buildings and ground. Then, we train a binary classifier by automatically obtaining the sky and non-sky samples for the sky segmentation.

3.1. Buildings and ground

After the line re-triangulation, most of the vertical lines are on the major building plane. Given the vertical lines and the 3D points which are higher than the lowest end point of all vertical line segments, we apply the plane fitting algorithm [14] to find the major building plane $\boldsymbol{\rho_b}$ and its inlier points set κ_b. Then we use the same method to detect the ground plane $\boldsymbol{\rho_g}$ by using 3D points around the plane passing the lowest end point of the vertical lines and orthogonal to the building plane. The boundary between buildings and ground is simplified as one ground line which indicates the real location of the buildings. We get the ground line $\boldsymbol{L}$ as the intersection of $\boldsymbol{\rho_b}$ and $\boldsymbol{\rho_g}$. In image I_p, the initial floor line $\boldsymbol{l_p}$ is the projection of $\boldsymbol{L}$. By adjusting $\boldsymbol{l_p}$ in I_p, we get the accurate floor line $\boldsymbol{l_{p_{new}}}$ as follows

$$\boldsymbol{l_{p_{new}}} = argmax\left(d_e\left(\boldsymbol{l_{p_{new}}}\right) - \mu d_m\left(\boldsymbol{l_p}, \boldsymbol{l_{p_{new}}}\right)\right)$$

where μ is the control factor, $d_e(\boldsymbol{l_{p_{new}}})$ is the sum of Sobel response on $\boldsymbol{l_{p_{new}}}$, and $d_m\left(\boldsymbol{l_p}, \boldsymbol{l_{p_{new}}}\right)$ is the distance between endpoints of $\boldsymbol{l_p}$ and $\boldsymbol{l_{p_{new}}}$. The accurate floor line $l_{p_{new}}$ is obtained by an search for its two endpoints on the left and right boundaries of I_p.

3.2. Sky

Having found the buildings and ground, we now try to detect the sky region. The state-of-the-art sky detector [17] achieved significant success by using a set of semantic attributes from half a million sky images. However, not only is the labeling tedious, the collecting of the sample images is also labour intensive. It requires enough diversity of the samples to train a robust detector. Instead of manually labeling so many samples, we automatically obtain the sky and non-sky samples from the original sequences for training. The ground region and the building region bounded by the detected lines are both non-sky regions. The non-sky samples are fetched from these non-sky regions. To identify the sky samples, we utilize two priors of urban environment: 1) very few 3D points can be reconstructed by SfM in sky regions as shown in Figure 2, since sky regions are almost textureless. Thus, the density of 3D points is useful in finding the sky regions. 2) sky regions usually appear at the

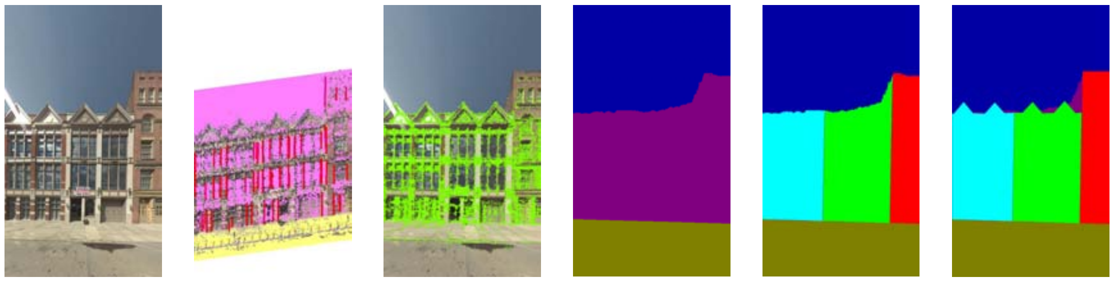


Figure 2. (a) The input image. (b) The vertical building plane and the horizontal ground plane (c) The projected 3D points (d) The enviroment is parsed into buildings, ground and sky (e) The buildings are parsed into individual architectural units. (f) The results after unit refinement. Color code: 🟦 sky, 🟪 buildings, 🟫 ground, (🟥🟩🟦) units

top boundary of the image. Therefore, we define a density map for each image. For each projection of the 3D points, we add a Gaussian kernel with size equal to 30 pixels on the density map. We accumulate the density for all points and obtain the density map by normalizing between 0 and 1. The sky region's density is almost zero and the sparsest region is the most likely sky region. Thus, we search for the minimal values around the top boundary of the density map and out of the non-sky region, and then regard them as sky samples. Finally, we train a binary classifier [13] to detect the sky region.

After the sky detection, the environment is parsed into buildings, ground and sky. The building region will be further parsed in Section 4. Thus, we need to extract and ignore the foreground objects. The foreground extraction includes three steps. First, the 3D points that are above the ground plane and in front of the building plane are segmented as foreground objects. Then, tensor voting [16] is used to remove isolated 3D points. The remaining 3D points of the foreground are clustered into several kernels by mean-shift. Finally, we adapt the idea from standard graph-based binary segmentation methods to segment the foreground object from the background. We use the minimum spanning tree (MST) of the 2D projection of 3D points to automatically label the foreground and background. Each foreground object is extracted individually.

4. Building Parsing

A façade is generally one side of the exterior of a building. Façades are distinguished by checking whether they belong to the same building. In this section, we propose a method to parse the building region into individual façades, i.e. architectural units. As shown in Figure 3, we first over-partition a sequence into sub-façades using the vertical line segments as separators, and then merge the sub-façades which belong to the same façade together by using dynamic programming that guarantees the optimal solution. In the partition process, we use four simple cues to measure the similarity and dissimilarity between sub-façades.

4.1. Formulation

For a sub-façade SF_i, let $R_{i,p}$ denote the reprojection region of SF_i in image I_p where I_p is in the set P^i_{SF} of images where SF_i is visible. We also denote the vertical line between two neighboring sub-façades SF_i and SF_j as separator l_{ij}. Similarly, the set of images where l_{ij} is visible is denoted as P^{ij}_{vl}.

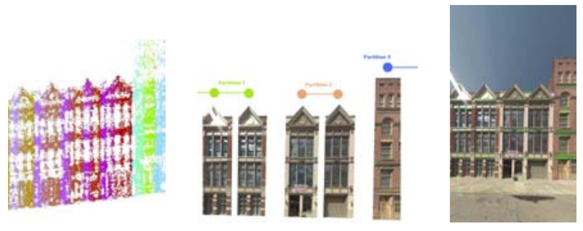
Suppose we have a total of N sub-façades. We define a weighted 1D graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E}, \mathcal{W} \rangle$, where $\mathcal{V}$ is the set of vertices representing sub-façades, $\mathcal{E}$ is the set of edges connecting neighbor sub-façades, and $\mathcal{W}$ is the set of edge weights. To merge the sub-façades is to find a partition of this 1D graph. The sub-façades are merged into a set of façades φ^X_N by retaining a subset X of all separators. Our goal is to partition a sequence into discernible façades, where façade should contain similar sub-façades. Thus, we define the energy of φ^X_N as

$$E\left(\varphi^X_N\right) = \sum_{s \in X} d_{inter}\left(s\right) + \sum_{\zeta \in \varphi^X_N} d_{intra}\left(\zeta\right)$$

where $d_{inter}\left(s\right)$ measures the dissimilarity between two façades with s as separator and $d_{intra}\left(\zeta\right)$ measures the overall similarity among all sub-façades inside the façade ζ. Our goal is to find the partition with maximum E. The brute-force method obtains optimal solution with a time complexity of $O(2^N)$. On the contrary, we use the dynamic programming optimization described in Section 4.3 to obtain the global optimal solution in $O(N^3)$.

4.2. Features

We define four simple features to make use of architectural properties such as the horizontal and vertical structure. The four features are integrated together by the boosting and used to measure the similarity between two sub-façades.



(a) (b) (c)

Figure 3. (a) Each sub-façade is color-coded in different color. (b) Suppose we have a 1D graph with five nodes, our algorithm aims to separate the graph into three partitions belonging to different buildings. (c) The vertical lines inside a façade have more intersections with horizontal line segments than the ones on the façade boundary.

Height The height of a sub-façade is an important cue for partition. Two buildings of different heights can be easily distinguished by humans. Reconstructed 3D features give an estimation of the upper boundary of the building region. We compute the average height of each sub-façade H_i in 3D. The distance using the heights between two sub-façades is defined as

$$d_h(SF_i, SF_j) = |H_i - H_j|$$

Strip histogram In one façade, the textures in a horizontal strip region across two sub-façades usually have similar color distribution. In order to measure both the global difference and the horizontal structural consistency, we divide the texture of a sub-façade SF_i into t horizontal strips $S_k^i(i = 1, 2, .., t)$ before computing the texture difference of two sub-façades. h_k^i denotes the normalized histogram of S_k^i. For two sub-façades, the distance of color histogram is defined as

$$d_c\left(SF_i, SF_j\right) = \frac{1}{t}\sum_{k=1}^{t} d\left(h_k^i, h_k^j\right)$$

where $d(\cdot,\cdot)$ is the Kullback Leibler Distance. In our implementation, the texture of the sub-façade is composed by projecting $R_{i,p}$ onto the local building plane estimated in Section 3.1. When two sub-façades are of different heights, the lower height is taken as the upper bound to compute d_c.

The number of intersections According to our observations, the vertical boundaries of a façade intersect fewer horizontal line segments than the vertical lines inside the façade as shown in Figure 3. On the same façade plane, the unit separator should not be a vertical line segment whose extension intersects many horizontal line segments. In practice, we compute a score for each separator s_i by counting the number of intersections it makes with all horizontal line segments in the building region as

$$\nu(l_{ij}) = -\frac{1}{\|P_{vl}^{ij}\|}\sum_{p\in P_{vl}^{ij}} \tau_{ij}^{p}$$

where τ_{ij}^p is the number of intersections of l_{ij} in image I_p where l_{ij} is visible. Thus, a distance between two sub-façades is defined by using the intersections of separators between them as

$$d_s\left(R_i, R_j\right) = \max_{s_k\in S_{ij}} \nu(s_k)$$

where S_{ij} is the set of all separators between two sub-façades.

Edge response The unit separators are usually vertical lines whose projections in the images have strong edges. The strength of line segment l_{ij} is computed as

$$\xi\left(l_{ij}\right) = \frac{1}{\|P_{vl}^{ij}\|}\sum_{p\in P_{vl}^{ij}} \frac{d_e(l_{ij})}{\mathcal{L}_p^{ij}}$$

where $d_e(l_{ij})$ is the sum of Sobel responses on l_{ij}, and $\mathcal{L}_p^{ij}$ is the length of the projection of l_{ij} in image I_p. We define the distance between two sub-façades using the edge feature as

$$d_r\left(R_i, R_j\right) = \max_{s_k\in S_{ij}} \xi(s_k)$$

The combined measurement Boosting [13] is used to combine the four features and learn the weighted similarity measure. For each feature, two distributions describe the positive and negative samples. Positive samples come from the distances of sub-façades in two neighbour façades. Negative samples come from the distances of sub-façades in the same façade. After the training, given the distances between two sub-façades, we compute the weighted sum of positive probabilities P_p and the weighted sum of negative probabilities P_n. P_p measures the dissimilarity between two sub-façades which is the d_{inter}. P_n measures the similarity between two sub-façades. In a façade which contains M sub-façades, d_{intra} is defined as the sum of all P_n divided by $M/2$.

4.3. Dynamic Programming Optimization

In this section, the partition problem is reformulated in terms of dynamic programming. We define $\mathcal{G}_i$ as the subgraph that only contains the first i nodes of $\mathcal{G}$. When $\mathcal{G}_i$ can only contain j partitions, the maximum energy is $\phi_{i,j}$ and the optimal solution is $\varphi_{i,j}$, where $\varphi_{i,j}[w]$ stores the nodes in the w-th partition. First of all, it is easy to directly compute the energy $\phi_{i,1}(i = 1, 2, ..., N)$ and $\phi_{1,1}$.

Then, for $\mathcal{G}_k(k > 1)$, we compute the $\phi_{k,j}(j = 2, ..., k)$ by reusing the optimal solutions of $\mathcal{G}_{k-1}$. Since $\mathcal{G}_k$ is the union of $\mathcal{G}_{k-1}$ and the k-th node of $\mathcal{G}$, given the solutions of $\mathcal{G}_{k-1}$, there are two ways to compute $\varphi_{k,j}$ by adding the k-th node to $\mathcal{G}_{k-1}$: 1) Based on $\varphi_{k-1,j}$, the k-th node is added to $\varphi_{i-1,j}[j]$. 2) The k-th node forms a new partition. Thus, we compute the $\phi_{k,j}$ as

$$\phi_{k,j} = \max \left\{\phi_{k-1,j} + \Delta d_{intra}(k), \phi_{k-1,j-1} + d_{inter}(l_{k-1,k})\right\}$$

where $\Delta d_{intra}(k)$ is the energy change based on $\phi_{k-1,j}$ and $d_{inter}(l_{k-1,k})$ is the energy change based on $\phi_{k-1,j-1}$.

To obtain the optimal solution, we search the maximum value from $\phi_{N,j}(j = 1, 2.., N)$ and adapt the corresponding partition solution. Distances among sub-façades are precomputed within $O(N^2)$. The time complexities to compute all $\Delta d_{intra}(i)$ and $d_{inter}(l_{k-1,k})$ are all $O(N^3)$. Therefore, the time complexities of the algorithm is $O(N^3)$.

Since most buildings are of regular size, if the span between two neighbouring separators are too large or too small, the space between them will contain several façades or partial façade. Thus, these two separators is incompatible with each other and many search states that contain incompatible separators are pruned. Additionally, given the range of the unit width, the minimum and maximum number of units in a sequence are computed to improve the performance.

5. Unit Refinement

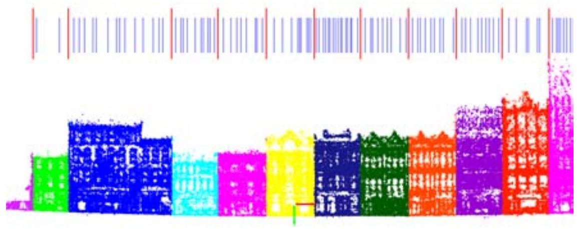


Figure 4. The partition result is shown in 3D. The locations of final separators are in red and the other candidate separators are in blue.

After the building partition, buildings are parsed into individual architectural units. In this section, the shape of each unit is further refined by regularizing the roof line. We define the roof line as the boundary between the building roof and the background region such as far buildings and sky, etc. Since rectilinear roofs are widely used in architectures, especially in the urban environment, we assume the roof line is piecewise linear.

In a unit U_k, an initial roof line R_{rough} is first estimated by linking the upper boundary of κ_b and then smoothed using tenser voting [16]. r^p_{rough} is the projection of R_{rough} in image I_p. Many line segments are detected around r^p_{rough} within a distance θ. We extend these line segments, let them intersect each other and also with two vertical boundaries of U_k, and then take all the intersections as the set $\mathcal{Q}_p$. Then, for all I_p which is visible to U_k, we sum $\mathcal{Q}_p$ to get $\mathcal{Q}_r$.

Without losing generality, all elements in $\mathcal{Q}_r$ are sorted from the left to the right. Thus, a direct edge $e_{x,y}$ is built from a point q_x to each point q_y on the right of q_x. We define the weighted graph $\mathcal{G}_\nabla = \langle \mathcal{V}_\nabla, \mathcal{E}_\nabla, \mathcal{W}_\nabla \rangle$, where $\mathcal{V}_\nabla$ is the set of vertices in $\mathcal{Q}_r$, $\mathcal{E}_\nabla$ is the set of direct edges, and $\mathcal{W}_\nabla$ is the set of edge weights. The weight of an edge $e_{x,y}$ is

$$\Gamma\left(e_{x,y}\right) = \alpha_e d_e\left(e_{x,y}\right) + \alpha_a d_a\left(e_{x,y}\right) + \alpha_d d_d\left(e_{x,y}\right)$$

where $d_a(e_{x,y})$ and $d_d(e_{x,y})$ are the differences in appearance and density of points between two sides of $e_{x,y}$, respectively. α_e, α_a and α_d are coefficients.

We define $\mathcal{V}_\int$ and $\mathcal{V}_\rceil$ as the set of vertices on the left and the right boundaries of U_k, respectively. The cost ς_r of a path L_r from $\mathcal{V}_\int$ to $\mathcal{V}_\rceil$ is defined as $\sum_{e \in L_r} \Gamma(e)$. Therefore, the problem is formulated as to find a cut with the maximum ς_r. The global optimal solution is found by using dynamic programming in $O(|\mathcal{V}_\nabla|^2)$ time. In image I_p, if both the upper boundaries of κ_b and r^p_{rough} are close to or higher than the image top boundary t_p, we snap this roof line to t_p.

The crossroads are the units with very low density of building points and usually at the two ends of a sequence. We re-classify them into ground and buildings by training a binary classifier using the method in Section 3.2. If the whole roof is occluded by the foreground segmentation mask in most views, our algorithm takes the boundary between the buildings and the sky as the roof line.

6. Experiments

In this section, we discuss every single component of the whole pipeline. We run our method on three datasets: 1) Pittsburgh. 8915 images of Pittsburgh at a resolution of 800 pixels by 1380 pixels. 2) Minneapolis. 2297 images of Minneapolis at a resolution of 1024 pixels by 1360 pixels. 3) Canton. 613 images of Canton at a resolution of 1024 pixels by 1360 pixels. The first two datasets were taken by a camera mounted on a moving automobile and the Canton dataset was captured by a handheld camera. The distance between neighboring images is about 0.9-5 meters. The experiments were run on a small cluster with 15 nodes. The results are produced automatically in 44 hours, including approximately 5 hours for SfM, 28 hours for environment segmentation, and 11 hours for partition and regularization. The ground truth is labeled by experienced human after training.

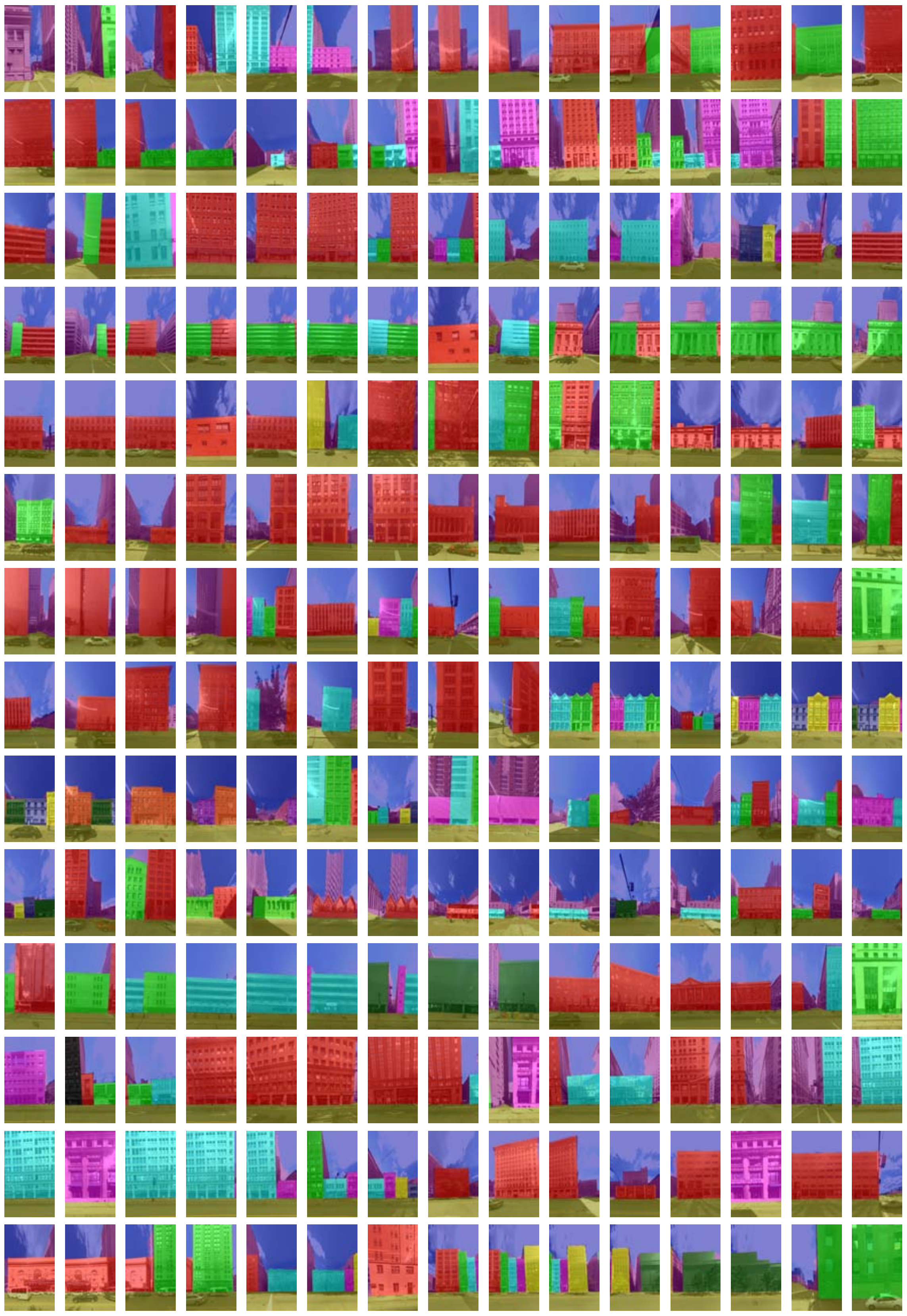


Figure 5. Results sequentially selected from the Pittsburgh dataset. Color code: 🟦 sky, 🟪 buildings, 🟫 ground, (🟥🟩🟦🟪🟨🟦🟩⬛) units

	Normal case	No point	Overall
Pittsburgh	(3.04, 1.22)	(5.01, 1.42)	(3.10, 1.23)
Minneapolis	(3.29, 2.18)	N/A	(3.29, 2.18)
Canton	(2.23, 1.16)	N/A	(2.23, 1.16)
All	(3.05, 1.40)	(5.01, 1.42)	(3.09, 1.41)

Table 1. The average distance from estimated floor lines to the ground truth is 3.09 pixels (mid-point distance [12]) and 1.41 degrees. The algorithm achieved stable performance in different datasets.

Environment parsing We use the midpoint-angle distance [12] from the results to the ground truth to evaluate the floor line estimation. As shown in Table 1, the overall performance is satisfactory. However, in some sequences, very few ground points are reconstructed. In such case, we use the ground plane of its neighbour sequence to initialize the search. This is reasonable since all the images are captured sequentially.

To evaluate sky detection, we manually labeled 5% of the images in the original sequences for comparison. By training the same sky classifier, the automatic label and manual label obtained 86.3% and 90.5% in accuracy [17], respectively.

Building parsing For the training, we have labeled 16 sequences with 1105 images from Pittsburgh dataset and 9 sequences with 693 images from Minneapolis dataset as training data. To evaluate the performance of the proposed partition method, we define the percentage accuracy as the number of correct separators over the number of all selected separators. The overall accuracy on the evaluated sequences is 85.1%. We also tested the appearance feature and Haar-like feature and got 72.9% and 61.3% in accuracy, respectively. We observed the superior results of our combined feature.

We discuss the performance of features by dividing the sequences into three classes: 1) the sequences that contain large units; 2) the sequences that contain normal units; 3) the sequences that contain repeated units. If a sequence contains more than one class, each separator is counted individually. In our dataset, there are 117 units, 215 units and 24 units in class 1, 2, and 3, respectively. From Table 2, we observed that the overall performance is best on class 2, since the proposed features are distinguishable when the heights, textures and structures are varying. While the performance on repeated units is relatively weak as the heights and appearances of such units are very similar. The four selected features are complementary that the efficiency and robustness of the method is enhanced.

We compared our method with a filtering method similar to [19] which filers the separators using pre-defined constraints and the hierarchical clustering algorithm. As shown

	Large	Normal	Repeated	Overall
All	83.0%	**87.2%**	**76.2%**	**85.1%**
Height	83.7%	85.3%	48.0%	82.2%
Strip Histogram	**84.1%**	83.5%	66.2%	79.8%
Intersections	69.2%	71.9%	75.5%	71.3%
Edge Response	67.5%	71.2%	68.1%	69.8%

Table 2. 1.'Large' is short for the sequences that contain large units. 2.'Normal' is short for the sequences that contain normal units. 3.'Repeated' is short for the sequences that contain repeated units. Each feature was evaluated individually. Our method work well in class 1 and 2. The height and strip histogram achieved higher accuracy in class 1 and 2, while the boundary features performed better in class 3 since edge and intersections can still be detected as usual in this case.

Training-Test dataset	Filtering	HC	Ours
Pittsburgh-Minneapolis	60.1%	71.9%	**83.7%**
Minneapolis-Pittsburgh	61.7%	69.2%	**80.5%**
All-All	61.3%	69.7%	**85.1%**

Table 3. We compared our partition algorithm with the filtering and the hierarchical clustering using average-link distance. By using training data from the other city, we evaluated the case invariance of our combined feature. We observed significant improvement of the proposed method over the other two methods and good generalization of our learned distance function for new city datasets.

Input	Roof Line Offset
Segmentation Ground Truth	3.78
Partition Ground Truth	3.89
All Ground Truth	**3.72**
Our result	4.02

Table 4. We tested our regularization algorithm in several cases with various segmentation and partition results. We observed that the regularization is not sensitive to the segmentation and partition results.

in Table 3, the proposed method is better since it is global optimized and is not sensitive to the local minima. We also evaluated it using the training data not in the same city and achieved similar accuracy. Failure cases include losing correct separators and selecting incorrect separators. As shown in Figure 6, the former is caused by missing line segments, large occlusion and highly similar features. The latter is caused by significantly changed lighting condition or height.

Unit refinement The average vertical offset from the roof line to the ground truth is 4.02 pixels. From Table 4, we observed that our roof line regularization does not heavily rely upon the former steps. This method performed well to parse buildings with rectilinear shape but not only the ones with non-rectangular shape. Human eyes can also easily evaluate the quality of the results shown in Figure 5 by inspection.

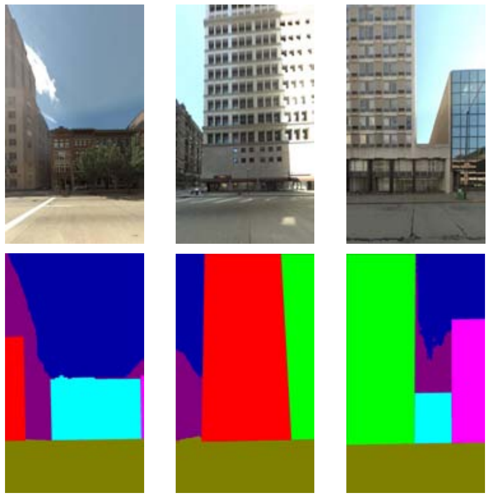


Figure 6. (a)Occlusion, similar appearances and heights (b) Shadow (c) Sudden change in height of a single building

7. Conclusion

In this paper, we present an automatic method to first parse the urban scenes into buildings, ground and sky and further parse the building region into individual architectural units with regular boundaries. Our approach is validated on large-scale datasets on a city scale. The selected features achieved significant improvements and the building partition method demonstrated satisfactory performance in this novel problem. It is true that urban environments in Europe and even in US suburbs are different from those in typical US downtowns. However, our assumptions throughout the paper are quite general. In the future, we will try to collect those data, improve the algorithm and evaluate the system on them.

Acknowledgement This work was partially supported by the Hong Kong RGC GRF 618908 and RGC GRF 619409, and the National Natural Science Foundation of China (60933006). We thank Google for the image data and a research gift. We also thank Lei Yang for the proofreading and Hao Hu for useful discussions.

References

[1] C. Baillard, C. Schmid, A. Zisserman, A. Fitzgibbon, and O. O. England. Automatic line matching and 3D reconstruction of buildings from multiple views. In *Proc. of ISPRS Conf. on Automatic Extraction of GIS Objects from Digital Imagery*, pages 69–80, 1999.

[2] A. Berg, F. Grabler, and J. Malik. Parsing images of architectural scenes. In *Proc. of Int'l Conf. on Computer Vision*, pages 1–8, 2007.

[3] M. R. Boutell, J. Luo, and C. M. Brown. Scene parsing using Region-Based generative models. *IEEE Trans. on Multimedia*, 9(1):136–146, 2007.

[4] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla. Segmentation and recognition using structure from motion point clouds. In *Proc. of European Conf. on Computer Vision*, pages 44–57, 2008.

[5] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[6] F. Han and S. Zhu. Bottom-up/top-down image parsing by attribute graph grammar. In *Proc. of Int'l Conf. on Computer Vision*, pages 1778–1785, 2005.

[7] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. 2003.

[8] D. Hoiem, A. Efros, and M. Hebert. Putting objects in perspective. *Int'l Journal of Computer Vision*, 80(1):3–15, 2008.

[9] M. Lhuillier and L. Quan. Robust dense matching using local and global geometric constraints. In *Proc. of Int'l Conf. on Pattern Recognition*, pages 968–972, 2000.

[10] M. Lhuillier and L. Quan. A Quasi-Dense approach to surface reconstruction from uncalibrated images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 27(3):418–433, 2005.

[11] P. Müller, G. Zeng, P. Wonka, and L. V. Gool. Image-based procedural modeling of façades. *ACM Trans. on Graphics*, 26(3):#85, 1–9, 2007.

[12] C. Rother. A new approach for vanishing point detection in architectural environments. In *Proc. of British Machine Vision Conference*, pages 382–391, 2002.

[13] J. Shotton, J. Winn, C. Rother, and A. Criminisi. TextonBoost for image understanding: Multi-Class object recognition and segmentation by jointly modeling texture, layout, and context. *Int'l Journal of Computer Vision*, 81(1):2–23, 2009.

[14] S. N. Sinha, D. Steedly, R. Szeliski, M. Agrawala, and M. Pollefeys. Interactive 3D architectural modeling from unordered photo collections. *ACM Trans. on Graphics*, 27(5):#27, 1–10, 2008.

[15] P. Tan, G. Zeng, J. Wang, S. B. Kang, and L. Quan. Image-based tree modeling. *ACM Trans. on Graphics*, 26(3):#87, 1–7, 2007.

[16] C. Tang, G. Medioni, and M. Lee. N-dimensional tensor voting and application to epipolar geometry estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 23(8):829–844, 2001.

[17] L. Tao, L. Yuan, and J. Sun. Skyfinder: attribute-based sky image search. *ACM Trans. on Graphics*, 28(3):#68, 1–5, 2009.

[18] J. Xiao, T. Fang, P. Tan, P. Zhao, E. Ofek, and L. Quan. Image-based façade modeling. *ACM Trans. on Graphics*, 27(5):#161, 1–10, 2008.

[19] J. Xiao, T. Fang, P. Zhao, M. Lhuillier, and L. Quan. Image-based street-side city modeling. *ACM Trans. on Graphics*, 28(5):#114, 1–12, 2009.