Extraction and Regularization of Various Building Boundaries with Complex Shapes Utilizing Distribution Characteristics of Airborne LIDAR Points

Jeongho Lee, Soohee Han, Younggi Byun, and Yongil Kim

This study presents an approach for extracting boundaries of various buildings, which have concave boundaries, inner yards, non-right-angled corners, and nonlinear edges. The approach comprises four steps: building point segmentation, boundary tracing, boundary grouping, and regularization. In the second and third steps, conventional algorithms are improved for more accurate boundary extraction, and in the final step, a new algorithm is presented to extract nonlinear edges. The unique characteristics of airborne light detection and ranging (LIDAR) data are considered in some steps. The performance and practicality of the presented algorithm were evaluated for buildings of various shapes, and the average omission and commission error of building polygon areas were 0.038 and 0.033, respectively.

Keywords: Building boundary, building extraction, boundary regularization, LIDAR.

I. Introduction

In recent years, airborne light detection and ranging (LIDAR) technology has been widely used in many fields owing to its high positioning accuracy, high surveying efficiency, and active sensing property. Building extraction is one of the most popular applications of LIDAR. Numbers of strategies have been presented to classify or filter LIDAR points to ground and nonground along with buildings [1]-[6], which aim at digital elevation model (DEM) generation or building extraction. Image processing techniques have been applied to extract buildings from raster grid data which is interpolated from LIDAR points [7]-[9]. Some researchers have constructed triangulated irregular network (TIN) models from LIDAR point data and then segmented buildings and extracted boundaries by some processes on the TIN model [10], [11]. A weakness of applying LIDAR data is that building boundaries cannot be clearly defined because the data relate to discrete points that are not necessarily on exact edges. Many techniques have been used to extract buildings by fusing LIDAR data and other data, such as satellite images, aerial images, or building plans [12], [13]. It is true that fusion of LIDAR and aerial images provides the most reliable building information if both of them are fully usable. However, occlusion or shadow in aerial images of densely urban areas makes it difficult to extract all edges in aerial images. In this context, regularization of building boundary derived from LIDAR is necessary in complex urban areas.

Generalization or regularization has been employed to refine the building boundaries, and many methods are based

Manuscript received Dec. 31, 2010; revised June 3, 2011; accepted June 10, 2011.

This study was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MEST) (No. 20100027762).

Jeongho Lee (phone: +82 2 880 7371, email: ilome79@snu.ac.kr) and Yongil Kim (email: yik@snu.ac.kr) are with the Department of Civil & Environmental Engineering, Seoul National University, Seoul, Rep. of Korea.

Soohee Han (email: scivile2@snu.ac.kr) is with the School of Civil & Environmental Engineering, Yonsei University, Seoul, Rep. of Korea.

Younggi Byun (email: ygbyun@kari.re.kr) was with the Department of Civil and Environmental Engineering, Seoul National University, Seoul, Rep. of Korea, and is now with the Satellite information Research Group, Korea Aerospace Research Institute, Daejeon, Rep. of Korea.

doi:10.4218/etrij.11.1610.0022

on rectangularity and orthogonality conditions. Fu and Shan [14] used three primitive models with a horizontal rectangular base to describe buildings with right-angled corners. Alharthy and Bethel [15] presented a polygon extraction algorithm that estimates the dominant directions of boundaries using a rotating template and angle histogram. Weidner and Förstner [16] utilized the minimum description length method to regularize ragged building boundaries. Ma [8] classified line segments into two perpendicular groups and carried out a weighted adjustment to calculate the azimuth of the two classes. Sampath and Shan [17] utilized a hierarchical least-squares method under the condition that the slopes of parallel lines are equal and the product of the slopes of perpendicular lines is –1. Lach and Kerekes [18] presented an algorithm that forces boundary line segments to be parallel or perpendicular to the dominant building orientation when appropriate and fits the data without constraint elsewhere. Sohn and others [19] developed a compass line filter to extract straight lines from irregularly distributed LIDAR points. Shen [20] utilized the alpha-shape algorithm to trace boundary points and applied a circumcircle regularization method to rectangular buildings.

In summary, previous approaches start from the assumption that a building boundary is composed of straight line segments, and many of them also assume that the segments are in only two dominant directions. In addition, there have been few attempts to trace a building boundary with inner yards from LIDAR points. This restricts wider application, necessitating the development of more delicate methods to deal with buildings with complex boundaries.

The study aims to detect and regularize various types of building boundaries including concave, non-right-angled, and nonlinear edges along with inner yards. The approach consists of four steps: building segmentation from a point cloud, boundary tracing, boundary grouping, and boundary regularization. In the first step, raw LIDAR points are segmented into homogeneous groups sharing similar altitudes, and various building groups are selected. Next, the convex hull algorithm is modified such that an elliptical neighborhood search is applied to trace boundary points more compactly. It is extended to trace the inner yard boundary by finding a seed point of the inner yard boundary. In the third step, critical points are detected through a progressive sleeve-fitting algorithm followed by angle-based generalization, and the points between two adjacent critical points are grouped to produce a line or an arc segment. The segments of a building are then classified into four types, and building boundaries are refined by deriving final parameters under different constraints according to the type of each segment. In each step, the characteristics of the airborne LIDAR scan profile are considered for more accurate boundary extraction. These steps, shown in Fig. 1, are explained in more detail from sections II to V. Section VI presents the experimental results obtained by applying the proposed method to several buildings of different shapes. Finally, conclusions and future research directions are discussed in section VII.

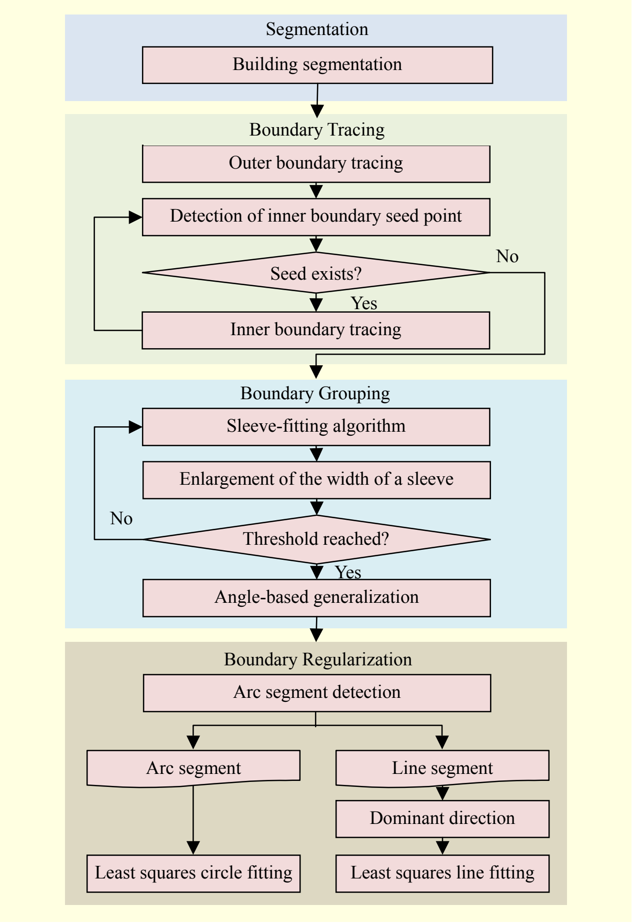

Fig. 1. Flowchart of presented approach.

II. Segmentation

Raw LIDAR points have to be segmented in advance to extract building boundaries. Generally, ground points and nonground points are first separated from raw LIDAR data by using filtering or classification algorithms, and then segmentation is conducted to classify nonground points into different buildings [1]-[6]. In our previous research [4], an algorithm was proposed to classify raw LIDAR points into points belonging to the same building. The algorithm is partly similar to region growing and unsupervised classification. The first input point becomes the seed point of the first group. The next input point is then compared with the points belonging to the first group. If they are adjacent and have similar height, the input point is classified in the first group; otherwise, it becomes the seed point of a new group. In this way, input points become members of existing groups or the seed points of new groups. Using this segmentation algorithm, it is possible that in only one process, raw LIDAR points are separated into different buildings, and moreover, points in one building are grouped into several parts with different heights.

III. Boundary Tracing

After building points have been segmented, the next step is boundary tracing, which is done to find the outermost and innermost points of a building. The presence of concavity in irregularly distributed points makes it difficult to trace all the boundary points without missing the concave corner. To deal with this problem, Jarvis [21] modified the convex hull algorithm to limit the searching space. Sampath and Shan [17] modified the convex hull algorithm to limit the searching space to a rectangle, considering uneven LIDAR point spacing in the along-scan and across-scan directions. Shen [20] traced a boundary from LIDAR points by applying the alpha-shape algorithm proposed by [22], where a circle of certain radius rolls around the points and forms the boundary of the point set on the plane. In this step, we modify the convex hull algorithm to trace a more compact boundary, and we extend it to trace an inner yard boundary while excluding inner holes due to occlusion.

1. Convex Hull Algorithm with Elliptic Neighborhood

The convex hull algorithm determines the smallest convex set containing discrete points. Among a few convex hull algorithms, we utilize the Jarvis march algorithm (also called the gift wrapping algorithm), which is the simplest algorithm in the case of a plane [23]. First, we determine the first boundary point, which has a minimum *x*-coordinate among all the points in a building. We then draw line segments to connect the last found boundary point with the other points in the building. The point that gives the smallest clockwise angle from the negative *y*-axis is chosen as the next boundary point. We repeat above processes using the previously formed boundary edge instead of the *y*-axis until the first boundary point is found again.

To trace a more compact boundary, we modify the convex hull algorithm by restricting the search space to an elliptical neighborhood. The major axis of the ellipse corresponds to the across scan line direction, and the dimensions of the ellipse are set to about one and a half of the point spacing in the across-scan and along-scan directions. The equation describing the elliptical neighborhood is

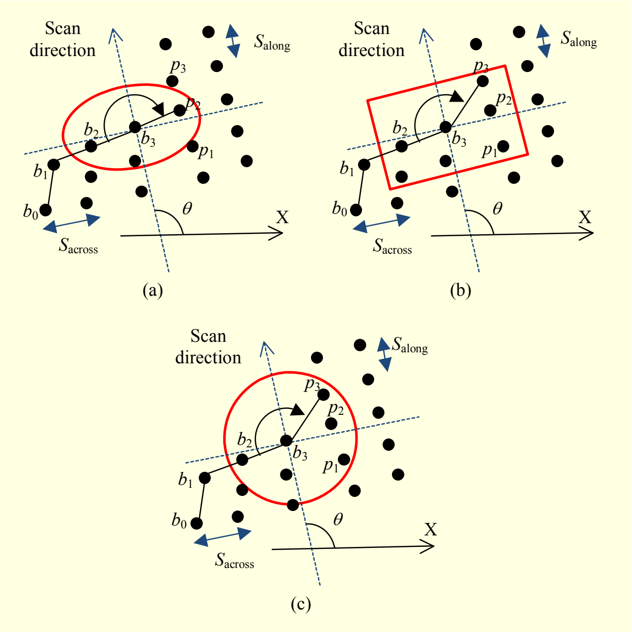

Fig. 2. Illustration of modified convex hull algorithm with (a) elliptical neighborhood (the presented method), (b) rectangular neighborhood, and (c) circular neighborhood.

$$\left(\frac{(x-x_{\mathrm{c}})\times\cos\theta+(y-y_{\mathrm{c}})\times\sin\theta}{k\times S_{\text{along}}}\right)^2 +\left(\frac{-(x-x_{\mathrm{c}})\times\sin\theta+(y-y_{\mathrm{c}})\times\cos\theta}{k\times S_{\text{across}}}\right)^2=1, \qquad (1)$$

where θ is the counterclockwise angle from the positive *x*-axis to the LIDAR scan profile, $(x_{\mathrm{c}}, y_{\mathrm{c}})$ are the coordinates of the boundary point found just previously, S_{along} and S_{across} are the average point spacings in the along-scan and across-scan directions, respectively, and k is a coefficient denoting the ellipse dimension.

The data in Fig. 2 shows larger point spacing in the across-scan direction than in the along-scan direction, which is considered only in the case of using an ellipse or rectangle. The red circle or rectangle represents a searching space from the previously detected boundary point b_3. In detecting the next boundary point, the convex hull algorithm using a rectangle or circle may miss the point p_2 and detect the next point p_3. Alternatively, the presented method using an elliptical neighborhood detects the exact corner point p_2 at the concave corner. In this way, a more compact boundary can be traced with one ground spacing of LIDAR points and fewer points are missed at the concave corner.

2. Inner Boundary Tracing

Segmented LIDAR building points may contain empty

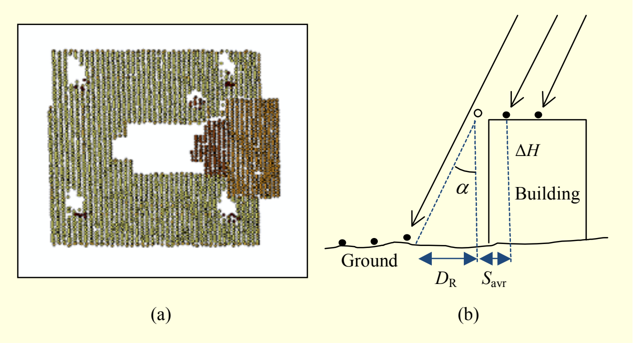

Fig. 3. Inner boundary tracing: (a) building with several holes and (b) allowed point spacing.

spaces. Only some empty spaces represent inner yards to be detected, but others result from occlusion. The building in Fig. 3(a) not only has an inner yard but also five small holes that are due to occlusion by more highly elevated parts expressed by a darker color. Hence, it is not a trivial problem to trace only the inner yard boundary among the empty spaces. The convex hull algorithm traces only the outer boundary not the inner boundary, while the alpha-shape algorithm can trace the inner boundary and also detect small inner holes due to occlusion.

To extend the presented boundary tracing algorithm to trace an inner yard boundary, we develop the algorithm to find the seed point of an inner yard boundary. If we know only the seed point of an inner yard boundary, the inner boundary can be traced in the same way as the outer boundary is traced.

Figure 3(b) indicates that it is very likely that two points are inner yard boundary points if they are further apart than the allowed point spacing, which is defined as the average point spacing added to the displacement D_R due to the height difference between the two points. Given the maximum scan angle α_{max}, the allowed point spacing can be written as

$$S_{allowed} = S_{avr} + D_R = S_{avr} + \Delta H \times \tan \alpha_{max}, \qquad (2)$$

where S_{avr} is the average point spacing in the along-scan direction, and ΔH is the height difference of two adjacent points on the same scan line. If the spacing between two adjacent points is at least 30% greater than $S_{allowed}$, the two points are regarded as points of an inner yard boundary, and they become members of a set P_S. The point-spacing irregularity method would fail for near vertical laserbeams. Two points, therefore, become members of P_S only if the spacing is greater than 10% of the dimension of a building. The dimension of each building can be approximated by the first and last points on a scan line. The set $P_1(\subset P_S)$ of candidates for the seed point of an inner yard boundary satisfies the condition $P_1 \cap B_O = \varphi$, where B_O is a set of outer boundary points. The minimum x-coordinate point in P_1 is selected as the seed point of an inner boundary. Boundary points of an inner yard are then detected in the same way as the outer boundary is traced, and they make up a set B_1.

To trace another inner boundary, we determine a set $P_2(\subset P_1)$, which satisfies the conditions $P_2 \cap B_O = \varphi$ and $P_2 \cap B_1 = \phi$. The minimum x-coordinate point in P_2 is selected as the seed point of another inner boundary, and the inner yard boundary is traced. The above processes are repeated until a set P_i is an empty set.

IV. Grouping of Boundary Points

Once the boundary points of a building are traced, we apply simplification or generalization to detect critical points that show an obvious turning in shape. If the critical points are determined through simplification or generalization, boundary points can be grouped by the critical points, which become the starting point or ending point of each segment. The Douglas-Peucker algorithm [24], Jenks algorithm [25], and sleeve-fitting algorithm [26] are good examples of generalization, but there are limitations in applying such conventional generalization algorithms directly to building boundaries. Therefore, we apply the sleeve-fitting algorithm progressively followed by angle-based generalization.

In the sleeve-fitting algorithm, a sleeve (rectangular strip in two dimensions) deletes points that are contained within, and the width of the sleeve determines how much the points in a segment can deviate. We see from Fig. 4(a) that p_3 is determined as a noncritical point because points p_2 and p_3 lie inside the sleeve. Whether a point is inside a sleeve is determined by using sector bound, because it is far easier to work with angle (sector bounds) than ε-buffering (epsilon bounds) and the two are geometrically equivalent as shown in [26]. In Fig. 4(b), point p_2 is deleted if and only if point p_3 lies in the dashed area. The dashed area (sector bounds) is defined as a point set given by

$$\left\{ q \in R^2 \,\middle|\, \alpha_{11} \le \alpha(p_0, q) \le \alpha_{22} \right\}, \qquad (3)$$
$$\alpha_{11} = \alpha(p_0, p_1) - \sin^{-1}(\varepsilon / d(p_0, p_1)),$$
$$\alpha_{22} = \alpha(p_0, p_2) + \sin^{-1}(\varepsilon / d(p_0, p_2)),$$

where $\alpha(p_0, q)$ is the angle measured counter-clockwise from the positive x-axis to the line $\overline{p_0 q}$, and $d(p_0, p_1)$ is the distance from point p_0 to point p_1.

The only parameter of the sleeve-fitting algorithm is the width of the sleeve, which has much influence on the generalization result. While a narrow sleeve will result in too many redundant critical points, a wide sleeve may miss some critical points. Because it is not easy to find an appropriate width of a sleeve, we perform the sleeve-fitting algorithm multiple times from a narrower sleeve to progressively wider

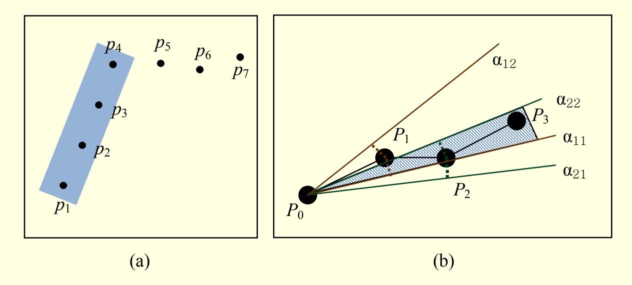

Fig. 4. Concept of the sleeve-fitting algorithm.

sleeves. The width of a sleeve is expressed as

$$D = (k_1 + k_2 \times i) \times \min(S_{\text{along}}, S_{\text{across}}), \tag{4}$$

where i is the number of iterations. The width of a sleeve in the first run is set by k_1, and how much the width increases in the next run is determined by k_2. We apply the sleeve-fitting algorithm in the forward direction and reverse direction in each iteration to compensate for a potential angular bias as in [16].

To discard redundant critical points further, we conduct angle-based generalization. For all remaining critical points, we calculate the angles between two adjacent lines formed by connecting two adjacent critical points. If the largest angle in a building is greater than a threshold, the point is regarded as a redundant point and is discarded. In this study, 160° is used as the angle threshold. Angle-based generalization discards one critical point at a time, and it is repeated until no more critical points are removed.

Because boundary points of a building are stored in good order (clockwise), critical points obtained through generalization are used as the first and last points of each group. The points between two adjacent critical points are grouped into segments, and each group corresponds to a line segment or an arc segment.

V. Boundary Regularization

In reality, some buildings have line segments with more than two directions, and furthermore, some buildings have nonlinear segments. Therefore, we classify the boundary segments into four types and use different methods for each type: line segments with dominant direction, line segments with perpendicular direction, line segments with other directions, and arc segments. The first work in boundary regularization is to determine whether a segment is an arc segment.

1. Arc Segment Detection

If a set of points is distributed with even spacing and near continuity, it is possible to derive an arc easily from the points on the basis of curvature. However, the points on an arc segment obtained from LIDAR data have a discontinuous and uneven distribution. We develop two measures in order to detect arc segments. The first measure is based on the theorem that the center of a circle or an arc should lie on a perpendicular projection through the midpoint of a chord that connects any two points on it (Fig. 5(a)). If there are n points on a segment, there are $n(n–1)/2$ lines connecting two points, and at most, the lines have $n(n–1)(n^2–n–2)/8$ points of intersection of two perpendicular projections passing through the midpoint of each line. The largest number of intersections is obtained for points on an ideal arc, in which case, all the perpendicular projections intersect only at one point, that is, the center of the arc. For points on a perfectly straight line, any two perpendicular projections do not intersect.

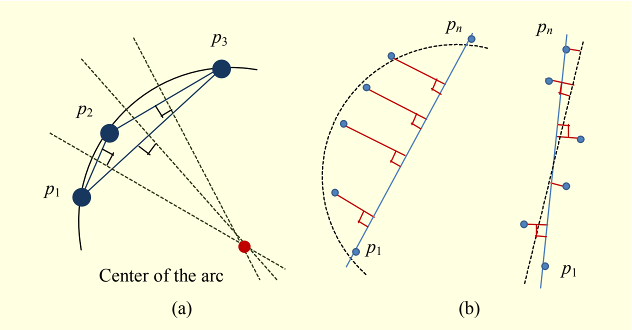

Fig. 5. Two components for arc detection: (a) theorem of perpendicular projection through midpoint of chord and (b) point distributions of arc and straight line segment.

Whether a set of points on a segment makes up an arc or a straight line is determined by how many projections intersect at the same point. A 1 m × 1 m grid is generated around the points of the segment, and each cell is attributed the count of how many pairs of two projections intersect inside it. The cell that has the maximum attributed value is the potential location for the center of the arc. To accomplish this, we use the ratio of the maximum cell value to the total number of possible intersections:

$$R_{\text{int}} = \frac{8 \times \max(A(i))}{n(n-1)(n^2-n-2)}, \tag{5}$$

where $A(i)$ is the number of intersections in the i-th cell, and n is the number of points on the segment.

In the case that points of a segment deviate greatly from the fitting line, a straight line segment may be mistaken as an arc segment owing to its large ratio R_{int}. To avoid such error, we additionally use the absolute value of the sum of the signed distance (ASSD). The plane in which the points of a segment exist can be divided into two parts by a line $\overline{p_1 p_n}$, which connects the first and last points of the segment. We see from Fig. 5 that all the points of an arc segment are likely to lie on the same side of the line $\overline{p_1 p_n}$, but points of a line segment lie on both sides of the line $\overline{p_1 p_n}$. On the basis of this property, *ASSD* is expressed by

$$ASSD = \left| \sum_{i=2}^{n-1} \left(d_i \times \frac{ccw(p_1, p_n, p_i)}{\left| ccw(p_1, p_n, p_i) \right|} \right) \right|,$$
$$ccw(p_1, p_n, p_i) = (x_1 y_n + x_n y_i + x_i y_1) - (y_1 x_n + y_n x_i + y_i x_1), \tag{6}$$

where d_i is the distance from point p_i to line $\overline{p_1 p_n}$, and $ccw(p_1, p_n, p_i)$ is a function which returns a positive value for a point p_i lying on the left (counterclockwise) of the line $\overline{p_1 p_n}$ and negative on the right. In this way, an arc segment has a larger *ASSD* value than a straight line segment.

If both the ratio R_{int} and *ASSD* of a set of points on a segment are higher than thresholds, the segment is classified as an arc segment. If not, it is classified as a line segment.

2. Determination of the Dominant Direction

After arc segments and line segments are discriminated, line segments are further classified into one of three types: those of the dominant direction, perpendicular direction, and other directions. If the dominant direction of a building is determined, the three types can be defined easily. The dominant direction of a building is determined by the length-weighted average angle.

In determining the dominant direction of a building, the characteristics of the LIDAR data are taken into consideration to obtain building boundaries that are more accurate. There is some spacing between LIDAR scan lines in most cases and the data used in this study have spacing of about 1 m. Therefore, in the case where the directions of building boundary segments are somewhat different from the LIDAR scan direction, there may be skewing errors in calculating the dominant direction of a building. To avoid this, we calculate the dominant direction of a building by excluding segments whose directions are within ε_θ, a buffer of the scan direction.

To begin, we first find the parameters of the line fitting each line segment S_i by general least squares, and compare the angle from the *x*-coordinate to S_i with the LIDAR scan direction. The longest segment that satisfies the condition of (7) is selected as the seed segment S_{seed} for calculation of the dominant direction:

$$S_{\text{seed}} = S_k \ ; \ l_k = \max(\ l_i)\ (i = 1, 2, \cdots, n),\ \left| \theta_i - \theta_s \right| > \varepsilon_\theta, \tag{7}$$

where θ_s is scan direction and l_i and θ_i mean the length and the angle from the *x*-coordinate of a segment S_i, respectively. Now, the dominant direction θ_d is equal to the angle of segment S_{seed}. It is then determined whether each segment is involved in the calculation of the dominant direction one by one:

$$\begin{aligned} &\text{if } \left| \theta_n - \theta_s \right| > \varepsilon_\theta,\ \left| \theta_n + \delta - \theta_d \right| < \varepsilon_1, \\ &\quad \text{then } \theta = \theta_n + \delta,\ l = l_n\ (\delta = 0°,\ 90°,\ -90°), \\ &\text{otherwise}, \quad \theta = 0,\ l = 0, \end{aligned} \tag{8}$$

where l_n and θ_n are the length and angle of a new checking segment, respectively. δ=0° means that the checking segment is nearly parallel with the dominant direction, and δ=90°, –90° means the segment is nearly perpendicular to the dominant direction. If a new checking segment satisfies the conditions, the dominant direction is updated including the length and angle of that segment:

$$\theta_d\,' = (\theta_d \times \sum_i l_i + \theta) / (\sum_i l_i + l)\ . \tag{9}$$

If the conditions of (8) are not satisfied, the angle and length of that segment are input as zero into (9), which results in no updating of the dominant direction. This computation is repeated for all line segments, and thus, the dominant direction is obtained. We now define the direction perpendicular to the dominant direction as $\theta_p = \theta_d \pm 90°$.

3. Final Building Boundary

The least-squares method has been widely used to find the unknown parameters of a line or circle from a set of points. Now that we have computed the dominant direction and perpendicular direction for each building, line segments are classified into three types on the basis of their directions. A parametric line is derived employing the least-squares method under different slope conditions according to the type of line segment.

If a segment has a direction similar to the dominant direction θ_d, the segment is forced to be parallel with the dominant direction. The unknown parameters of the line segment are found by employing least-squares method under the constraint that the line has a slope of θ_d. If a segment has a direction similar to the perpendicular direction θ_p, the unknown parameters of the line segment are found in the same way as for a segment with a dominant direction. In this case, θ_p is used for θ_d in the adjustment:

$$\begin{aligned} &\theta_d - \varepsilon_1 < \theta_i < \theta_d + \varepsilon_1 :\ A_i x_{ij} + B_i y_{ij} = 1,\ \tan^{-1}\left(-\frac{A_i}{B_i}\right) = \theta_d, \\ &\theta_p - \varepsilon_1 < \theta_i < \theta_p + \varepsilon_1 :\ A_i x_{ij} + B_i y_{ij} = 1,\ \tan^{-1}\left(-\frac{A_i}{B_i}\right) = \theta_p, \end{aligned} \tag{10}$$

where x_{ij} and y_{ij} are the *x*-coordinates and *y*-coordinates of the *j*-th point of the *i*-th line segment, respectively, and θ_i is the counterclockwise angle from the *x*-coordinate to the *i*-th line segment. For other line segments whose directions are much different from the dominant direction or perpendicular direction, the unknown parameters are found employing the general least-squares method without any constraints of slopes.

For a set of points on an arc segment, the unknown parameters are found by least-squares circle fitting, and the equations for center point (x_c , y_c) and radius R are:

$$x_c = \frac{\sum x_i}{n} + \frac{\sum v_i^2 (\sum u_i^3 + \sum u_i v_i^2) - (\sum u_i v_i)(\sum v_i^3 + \sum u_i^2 v_i)}{2(\sum u_i^2 \sum v_i^2 - \sum u_i v_i \sum u_i v_i)}$$

$$y_c = \frac{\sum y_i}{n} + \frac{-\sum u_i v_i (\sum u_i^3 + \sum u_i v_i^2) + (\sum u_i^2)(\sum v_i^3 + \sum u_i^2 v_i)}{2(\sum u_i^2 \sum v_i^2 - \sum u_i v_i \sum u_i v_i)},$$

$$R = (x_c - \frac{\sum x_i}{n})^2 + (y_c - \frac{\sum y_i}{n})^2 + \frac{\sum u_i^2 + \sum v_i^2}{n}, \tag{11}$$

where $u_i = x_i - \sum x_i / n$ and $v_i = y_i - \sum y_i / n$.

Now that we have found all the parameters of the line segments and arc segments, the intersection points of two adjacent segments make up the final building boundary. If a line segment has the same slope as an adjacent line segment, they are parallel and never intersect. In this case, the two segments are merged as one segment, and the parametric line is newly derived using all the points in it.

Table 1. Specifications of LIDAR data.

System	ALTM 3070 (Optech, Inc.)
Point spacing	0.8 m (along), 1.3 m (across)
Scan angle	0 to ± 25°
Flying height	1,500 m
Beam divergence	0.3 mrad

Table 2. List of threshold values.

Threshold	Value	Description
k	1.5	Denote ellipse dimension
$\alpha_{\max}$	25°	Maximum scan angle
k_1, k_2	0.8, 0.3	Decide width of a sleeve
R_{int}	0.08	Ratio of the maximum cell value to the total number
$ASSD$	1.2 m	The absolute value of the sum of the signed distance
ε_θ	10°	Buffer for the scan line direction
ε_1	15°	Buffer for line segment direction

VI. Experimental Results and Discussion

1. Evaluation and Inspection of Results

To evaluate the proposed approach, we use the first returns of LIDAR data for the city of Daejeon, Korea collected with Optech LIDAR equipment. The specifications of the utilized LIDAR data are given in Table 1. After segmentation, we select a large variety of buildings, including rectangular buildings, rectilinear buildings with more than two dominant directions, buildings with inner holes, and buildings also having nonlinear segments. Threshold values utilized for experiments are listed in Table 2.

In Fig. 6, 6(a) in each row shows segmented LIDAR points overlaid atop the TIN surface model, 6(b) shows critical points after progressive sleeve fitting with traced boundary points, and 6(c) shows the final regularized building boundaries overlaid atop segmented building points. The values in 6(d) show the dimension of each building, and the omission and commission error of the area.

The first set of LIDAR data is for rectangular buildings, which have different sizes and various numbers of line segments. The line segments have only two directions, which are perpendicular to each other, and each building has several concave corners. Figure 6 shows that most of the building edges are determined quite well. The regularized building boundaries show good delineation of the real buildings at most concave corners except for the upper right concave corner of the second building and the middle corner of the third building. The former is due to missed points in boundary tracing, and the latter is due to the protruding part being quite small compared with the building size.

The second set contains buildings whose boundary segments have more than two directions. Line segments with directions similar to the dominant direction or perpendicular direction are forced rightly to be parallel or perpendicular, whereas the segments with other directions are modeled well, and their original directions are preserved.

The third set of LIDAR data is for three buildings, each having one or two arc segments as well as several straight line segments. In differentiating between arc segments and line segments, the best results are acquired when 0.04 and 1.2 are used for the thresholds of R_{int} and $ASSD$, respectively. Four arc segments of the three buildings are identified properly as arcs, and one arc segment, which is at the lower right corner of the sixth building, is mistaken for line segment. The threshold for R_{int} is small and sensitive to the noise or point distribution. If raw LIDAR points are acquired more densely, the identification of arcs will be less sensitive and the proposed method will be more robust. Also, fusing airborne LIDAR and aerial images will be a better solution for the problem because an aerial image represents clear edge information of circular arcs as well as straight lines.

In the case that a line segment and arc segment meet, it is rarely possible to detect the exact critical points in boundary grouping. Nevertheless, the final building boundary is modeled with an equation of a line or circle, and it is well adjusted

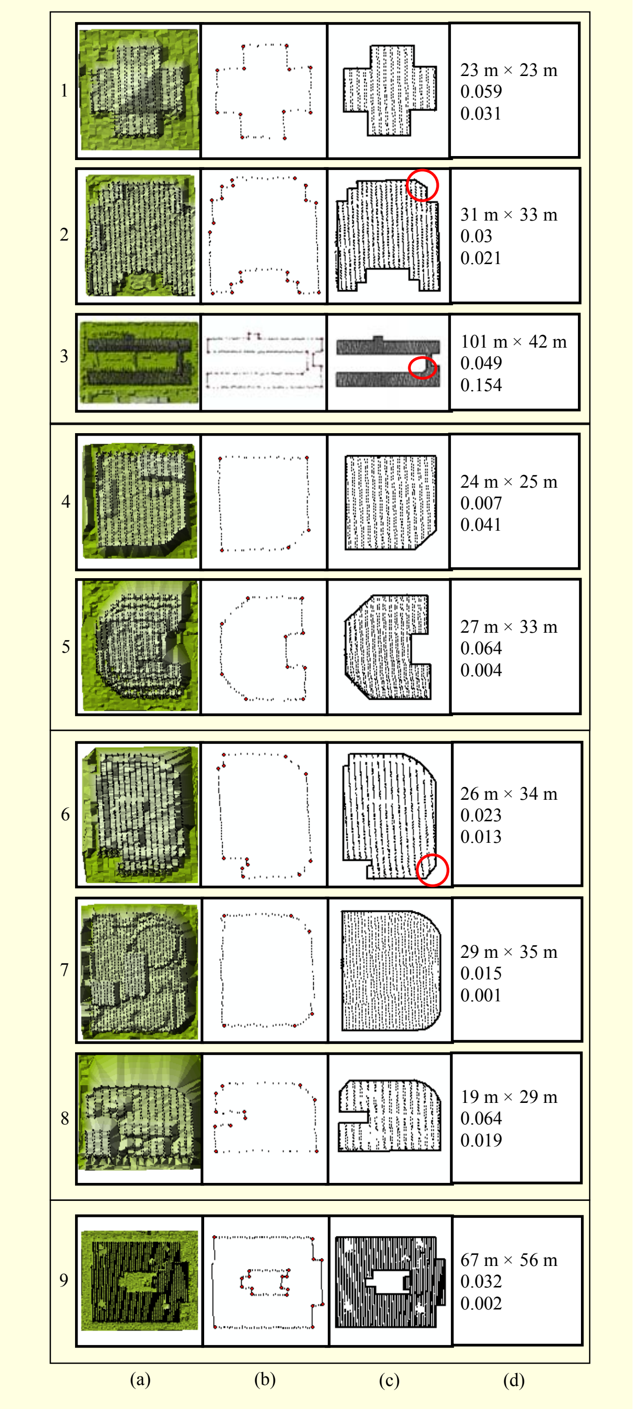

Fig. 6. Building boundary extraction results: (a) segmented building points, (b) building boundary points and critical points, (c) regularized building boundary, and (d) building dimensions, omission and commission error of each polygon area.

by intersection points. More accurate delineation of complex buildings can be achieved by modeling nonlinear segments with arcs. Moreover, the storage space required for a final building model is reduced if we store an arc segment as three points: the starting point, ending point, and midpoint.

The final set shows linear boundaries for two orthogonal directions and some empty interior space. The building not only has an inner yard but also five small holes, which have to be distinguished from the inner yard to extract building boundaries that are more exact. The boundary of the inner yard is detected well using the presented approach, with the five small inner holes being excluded. This can be achieved by determining the seed point of the inner yard boundary using the allowed point spacing.

For quantitative analysis, we compared the areas of building polygons obtained from our method with that of reference polygons, which were generated from digital maps and aerial photos. We calculated omission and commission error for each building polygon. It is given in Fig. 6, and the average omission and commission errors are 0.038 and 0.033, respectively

2. Comparison with Other Methods

To determine the performance of the presented approach, the results obtained using the presented method are compared with those obtained employing other methods. First, boundary tracing results obtained with the presented method (Fig. 7(a)) are compared with those obtained with the alpha-shape algorithm, the modified convex hull algorithm by [11], and the original convex hull algorithm. It is known that the original convex hull algorithm never detects points at a concave corner, and when applying the algorithm, the shapes of buildings are greatly distorted.

Although two of the compared methods, the alpha-shape algorithm and modified convex hull algorithm of Sampath, detect compact boundary points, they miss a few crucial points at a concave corner, which is shown by the red circles in Fig. 7. It is possible to trace more compact points at the concave

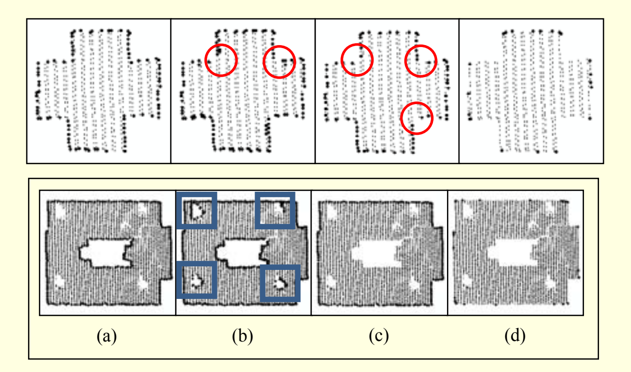

Fig. 7. Traced building boundary points obtained employing (a) proposed method (elliptical neighborhood), (b) alpha-shape algorithm, (c) modified convex hull algorithm of Sampath (rectangular neighborhood), and (d) original convex hull algorithm.

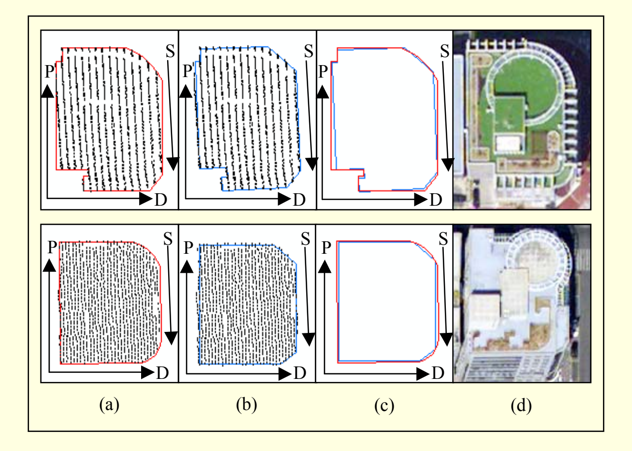

Fig. 8. Regularized building boundaries obtained employing (a) proposed method considering scan direction, (b) compared least-squares method without consideration of scan line direction, (c) superimposition of them, and (d) aerial photo (D and P are dominant and perpendicular directions acquired from aerial photo, and S is scan direction).

corner using the presented method, which restricts the search space to an elliptical neighborhood.

The convex hull algorithm and its modified version trace only the outer boundary but fails to trace the inner boundary (see Figs. 7(c) and (d)). The alpha-shape algorithm traces the inner boundary as well as the outer boundary but also extracts some small inner holes due to occlusion, which are shown by the blue rectangles in Fig. 7. The presented boundary tracing method traces the inner yard boundary, excluding the inner holes. This is achieved by distinguishing the inner yard from inner holes utilizing the allowed LIDAR point spacing based on the height difference between adjacent points.

Second, the results obtained employing the presented regularization method are compared with those obtained using another method in which the dominant direction is calculated without consideration of the LIDAR scan direction and all segments are assumed to be straight lines. The presented approach calculates the dominant direction, excluding segments whose directions are similar to the along-scan direction, and discriminates arc segments from line segments.

In the case of the LIDAR data used in this study, the directions of some line segments are similar to the along-scan direction. Figure 8(a) shows that the regularized building boundary obtained by employing our method is correctly determined without skewing, and there is good agreement with the reference directions (D and P) acquired manually from the aerial photo. In Fig. 8(b), the final building boundary segments are rotated globally counterclockwise and are nearly parallel or perpendicular to the scan direction, which is more obvious for the upper building.

The compared method describes a nonlinear edge as a straight line, and the regularized boundary is distorted. On the other hand, the presented method describes a nonlinear edge as an arc segment, and the regularized boundary is more similar to the real building boundary. If the raw LIDAR data is acquired more densely, the boundary determined employing our approach will better approximate the real building.

3. Discussion

The proposed procedure focuses on reconstructing 2D building boundaries from single strip LIDAR data. Some threshold values utilized in the proposed method are based on distribution characteristics of single strip LIDAR points. When overlapping strips are concerned, point spacing will be smaller and the related threshold values can be adjusted. If they are not parallel, some conditions including scan direction cannot be utilized, which should be remarked as a limitation of the method.

The proposed procedure is capable of producing a 2D building polygon or 3D prismatic building models, but not 3D building models with a tilted roof or a superstructure. However, prismatic building models or 2D building polygons are used in most of web-based maps and 3D navigation maps. In this context, the proposed approach will be useful to update building information of maps.

VII. Conclusion

An approach for detecting and regularizing various building boundaries from airborne LIDAR points was presented. The approach has four steps: building point segmentation, boundary tracing, boundary grouping, and boundary regularization. The experimental results show that the approach works well for complexly shaped buildings and produces quite robust boundaries. A concave boundary is traced more compactly by applying an elliptical search space based on uneven LIDAR point spacing for convex hull generation, and application of the allowed point spacing makes it possible to trace only the inner yard boundary and not holes. In cases where buildings have multiple directions and nonlinear edges, the boundaries are effectively regularized by differentiating nonlinear edges from linear edges with R_{int} and *ASSD*. As a result, building boundaries represented as arc segments and line segments delineate the real buildings well. In addition, considering the LIDAR scan direction in calculating the dominant direction prevents the building boundary from skewing. In future work, the approach will be extended to extract 3D building models from terrestrial LIDAR data, and components with more complex shapes will be detected and regularized.

References

[1] G. Sithole, "Filtering of Laser Altimetry Data Using a Slope Adaptive Filter," *Int. Archives Photogrammetry Remote Sens.*, vol. 34, part 3W4, 2001, pp. 203-210.

[2] K. Zhang et al., "A Progressive Morphological Filter for Removing Nonground Measurements from Airbone LIDAR Data," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 4, 2003, pp. 872-882.

[3] J. Shan and A. Sampath, "Urban DEM Generation from Raw LIDAR Data: A Labeling Algorithm and Its Performance," *Photogrammetric Eng. Remote Sens.*, vol. 71, no. 2, 2005, pp. 217-226.

[4] S. Han, J. Lee, and K. Yu, "An Approach for Segmentation of Airborne Laser Point Clouds Utilizing Scan-Line Characteristics," *ETRI J.*, vol. 29, no. 5, Oct. 2007, pp. 641-648.

[5] Q. Chen et al., "Filtering Airborne Laser Scanning Data with Morphological Methods," *Photogrammetric Eng. Remote Sens.*, vol. 73, no. 2, 2007, pp. 175-185.

[6] X. Meng et al., "A Multi-directional Ground Filtering Algorithm for Airborne LIDAR," *ISPRS J. Photogrammetry Remote Sens.*, vol. 64, no. 1, 2009, pp. 117-124.

[7] F. Rottensteiner and C. Briese, "A New Method for Building Extraction in Urban Areas from High-Resolution LIDAR Data," *Proc. ISPRS Commission III Symp.*, Graz, Austria, unpaginated CD-ROM, 2002.

[8] R. Ma, "DEM Generation and Building Detection from LIDAR Data," *Photogrammetric Eng. Remote Sens.*, vol. 71, no. 7, 2005, pp. 847-854.

[9] X. Meng and N. Currit, "Morphology-Based Building Detection from Airborne LIDAR Data," *Photogrammetric Eng. Remote Sens.*, vol. 75, no. 4, 2009, pp. 437-442.

[10] Z. Wang, and T. Schenk, "Building Extraction and Reconstruction from LIDAR Data," *Int. Archives Photogrammetry Remote Sens.*, vol. 33, part B3, Amsterdam, The Netherlands, unpaginated CD-ROM, 2000.

[11] M. Morgan, and A. Habib, "Interpolation of LIDAR Data and Automatic Building Extraction," *ACSM-ASPRS Annual Conf. Proc.*, unpaginated CD-ROM, 2002.

[12] F. Rottensteiner et al., "Fusing Airborne Laser Scanner Data and Aerial Imagery for the Automatic Extraction of Buildings in Densely Built-Up Areas," *Proc. 20th ISPRS Congress, Commission III*, Istanbul, Turkey, 2004, pp. 512-517.

[13] L. Matikainen, J. Hyyppä, and H. Kaartinen, "Comparison between First Pulse and Last Pulse Laser Scanner Data in the Automatic Detection of Buildings," *Photogrammetric Eng. Remote Sens.*, vol. 75, no. 2, 2009, pp. 133-146.

[14] C.S. Fu and J. Shan, "3-D Building Reconstruction from Unstructured Distinct Points," *Int. Archives Photogrammetry Remote Sens.*, vol. 35, part B3, CD-ROM, 2004.

[15] A. Alharthy and J. Bethel, "Heuristic Filtering and 3D Feature Extraction from LIDAR Data," *Proc. ISPRS Congress, Commission III Symp.*, Graz, Austria, 2002.

[16] U. Weidner and W. Förstner, "Towards Automatic Building Extraction from High-Resolution Digital Elevation Models," *ISPRS J. Photogrammetry Remote Sens.*, vol. 50, no. 4, 1995, pp. 38-49.

[17] A. Sampath and J. Shan, "Building Boundary Tracing and Regularization from Airborne LIDAR Point Clouds," *Photogrammetric Eng. Remote Sens.*, vol. 73, no. 7, 2007, pp. 805-812.

[18] S. Lach and J. Kerekes, "Robust Extraction of Exterior Building Boundaries from Topographic Lidar Data," *IEEE Proc. IGARSS*, Boston, USA, 2008, pp. 85-88.

[19] G. Sohn, X. Huang, and V. Tao, "Using a Binary Space Partitioning Tree for Reconstructing Polyhedral Building Models from Airborne LIDAR Data," *Photogrammetric Eng. Remote Sens.*, vol. 74, no. 11, 2008, pp. 1425-1438.

[20] W. Shen "Building Boundary Extraction Based on LIDAR Point Clouds Data," *Int. Archives Photogrammetry, Remote Sens., Spatial Info. Sci.*, Beijing, vol. 37, part B3b, 2008, pp. 157-161.

[21] R.A. Jarvis, "On the Identification of the Convex Hull of a Finite Set of Points in the Plane," *Info. Process. Lett.*, vol. 2, no. 1, 1973, pp. 18-21.

[22] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the Shapes of a Set of Points in the Plane," *IEEE Trans. Info. Theory*, vol. 29, no. 4, 1983, pp. 551-559.

[23] R.A. Jarvis, "Computing the Shape Hull of Points in the Plane," *Proc. IEEE Comput. Soc. Conf. Pattern Recognition Image Process.*, 1977, pp. 231-241.

[24] D. Douglas and T. Peucker, "Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature," *The Canadian Cartographer*, vol. 10, no. 2, 1973, pp. 112-122.

[25] G.F. Jenks, "Geographic Logic in Line Generalization," *Cartographica*, vol. 26, no. 1, 1989, pp. 27-42.

[26] Z. Zhao and A. Saalfeld, "Linear-Time Sleeve-Fitting Polyline Simplification Algorithms," *AutoCarto 13*, Seattle, Washington, 1997, pp. 214-223.

Jeongho Lee received the BS and MS in GIS and remote sensing from Seoul National University, Seoul, Rep. of Korea, in 2002 and 2004, respectively. He is currently working toward a PhD in remote sensing in Seoul National University, Seoul, Rep. of Korea.

Soohee Han received the BS, MS, and PhD in GIS and remote sensing from Seoul National University, Seoul, Rep. of Korea, in 2000, 2002, and 2008, respectively. He is now a postdoctoral researcher of the School of Civil & Environmental Engineering, Yonsei University, Seoul, Rep. of Korea.

Younggi Byun received the BS in energy resources engineering from Chonnam National University, Rep. of Korea, and the MS and PhD in civil, urban, and geosystem engineering from Seoul National University, Seoul, Rep. of Korea, in 1995, 2004, and 2011, respectively. He is currently a senior researcher in satellite information research group at Korea Aerospace Research Institute, Rep. of Korea.

Yongil Kim received the BS in urban engineering in 1986, and the MS and PhD in remote sensing from Seoul National University, Seoul, Rep. of Korea, in 1988 and 1991, respectively. He is currently working as a professor in the Department of Civil & Environmental Engineering, Seoul National University, Seoul, Rep. of Korea. His major research interests include remote sensing, global positioning systems, and geographic information systems.