Hindawi
Journal of Sensors
Volume 2017, Article ID 7051931, 8 pages
https://doi.org/10.1155/2017/7051931

Research Article

Automated Recognition of a Wall between Windows from a Single Image

Yaowen Zhang, Linsheng Huo, and Hongnan Li

State Key Laboratory of Coastal and Offshore Engineering, Dalian University of Technology, Dalian 116024, China

Correspondence should be addressed to Linsheng Huo; lshuo@dlut.edu.cn

Received 12 January 2017; Revised 8 April 2017; Accepted 12 April 2017; Published 3 May 2017

Academic Editor: Julio Rodriguez-Quiñonez

To avoid the time-consuming, costly, and expert-dependent traditional assessment of earthquake damaged structures, image-based automatic methods have been developed recently. Since automated recognition of structure elements is the basis by which these methods achieve automatic detection, this study proposes a method to recognize the wall between windows from a single image automatically. It begins from detection of line segments with further selection and linking to obtain longer line segments. The color features of the two sides of each long line segment are employed to pick out line segments as candidate window edges and then label them. Finally, the images are segmented into several subimages, window regions are located, and then the wall between the windows is located. Real images are tested to verify the method. The results indicate that walls between windows can be successfully recognized.

1. Introduction

An earthquake may result in thousands of structures suffering different levels of damage. The safety assessment of these structures is of great significance with regard to victim accommodation, intensity evaluation, and emergency aid provision. The traditional strategy is for certified engineers or structure experts to carry out the assessment. To analyze the defects carefully, the inspectors must have direct access to the structures and must move from one structure to another only on foot, due to destruction of the traffic system. Therefore, the traditional method is time-consuming, costly, and expert-dependent [1]. It is impossible to achieve fast and accurate assessment for all the damaged structures.

To develop a real-time and cost-effective method, researchers have applied computer vision technology for the assessment of damaged structures [2]. Recently, image-based methods have been developed to extract information related to defects such as cracks [3, 4] and spalling [5] and to recognize structural columns [6]. Although columns are the primary load-bearing elements, sometimes they may not be obvious to locate, particularly from the outside. In such conditions, the assessment of the walls could provide information relevant to the damage level of the structure. However, automated recognition of walls based on computer vision is still largely absent. Therefore, this study proposes a method to automatically recognize walls between windows.

Developing a recognition or detection method based on computer vision enables safety assessment to be achieved quickly. In the vista of the application of the methods, every person who can take pictures using cameras, mobile phones, or other devices can carry out the assessment task, with the images processed on local devices, on personal computers, or via cloud computing. With this intention, the method developed in this study is based on a single image, and there are no technical requirements for image acquisition.

Usually, there are few obvious and distinctive features on a wall itself, so this study describes how the windows next to the wall are used to locate it. Researchers have developed several methods for window detection from images [7–10]. These methods can detect windows in facades well, but they are based on the points that there are a couple of windows existing in one image and they have some same (or similar) features like shape, size, and so on, which means that the walls to be detected just occupy a small area in the image. In other words, the resolution of the wall is too low to provide enough information for the extraction detection. However, the automated recognition work is to be a basis for later

retrieval of defect information, so the wall in the image must be clear enough to ensure the accuracy of the defect information. In another aspect, when coming to an image only having two windows and a wall between them to assure the resolution of the wall, there are not sufficient same (or similar) features to be used. Therefore, the proposed method is for the image to have only one wall between windows.

The method first detects the line segments in the image by using a state-of-the-art line detection method. Then, the line segments are linked by a linking strategy, which is advanced by the authors. After selection of the long line segments, the color features are calculated to find and label the candidate window edges. The image is then segmented into subimages by the candidate line segments, and an assessment is conducted to identify a window region. Once the window regions are located, the wall between the windows is located.

2. Preconditions

The main purpose of the method described in this paper is to achieve automatic recognition. So, some preconditions are set to simplify the problem. (1) The image should be clear enough for people to observe it without much thinking. (2) The windows in the image should be easily recognizable and fully visible. (3) The window frame should be simple, common rectangles. (4) Only one target wall should exist, and its vertical edges should be vertical (or nearly vertical) in the image. It should be emphasized that condition (4) is not a natural constraint of the method proposed in this paper. As the recognition of the wall is just a prestep for the damage detection, condition (4) ensures that the wall region is clear enough during the following damage detection application. In addition, it could be easily extended to two or three target walls.

3. Framework of the Proposed Methodology

The main idea is to achieve automated recognition based on an obvious feature of the wall: that it is between windows. In other words, if there is a region for which both sides are windows, the region is recognized as a wall between windows. The method proposed in this paper mainly consists of three parts (Figure 1). The first is a line-detecting and linking part, aiming to detect the long line edges in the image. The second part is to design a classifier, which selects the candidate lines for window edges by using color features of the regions on both sides of the line. The third part involves segmenting the whole image by using all the candidate lines, after which the two window areas are located. The region between the two windows is then recognized as a wall between windows. Details of these three parts are explained in the following subsections.

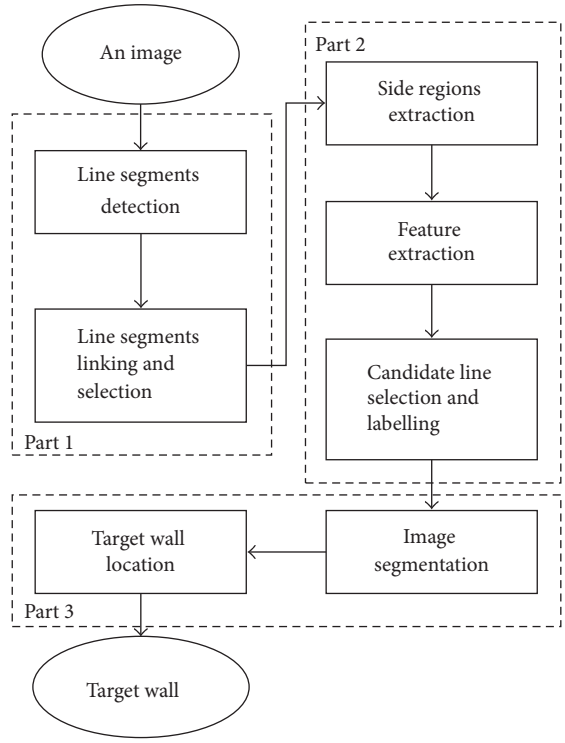


Figure 1: The framework of the proposed methodology.

4. Long Line Detection

4.1. Line Segments Detection. The first step to segment an image into regions is to find the edges of the regions. Considering the purpose of this paper, what we really need are lines to separate regions. Since line segment detection is an important and basic problem in many image processing-based applications, a number of methods have been proposed in the last few decades, from the typical traditional Hough transform [11] based to the newest CannyLines [12]. To detect line segments in images, there are usually two steps: edge detection and line segments detection. For the detection of edges, the Canny edge detection is a classical method and of better performance than others like Sobel, Roberts, and so forth. However, the newly Edge Drawing method can produce the similar quality edge map and runs up to 10% faster than the Canny edge detector [13]. For the detection of line segments, the result of the Hough transform was not very satisfied, due to too many fault detection with more computation time [14]. But the EDLines method, which makes use of the result of the Edge Drawing method, is 7 times faster with accurate results. Since this is not the main focus of the paper, just three methods state of the art, line segment detector (LSD) [15], EDLines [14], and CannyLines, are discussed here.

The LSD produces accurate line segments and controls the number of false detection instances at a low level by efficiently combining gradient orientations and line validation according to the Helmholtz principle [16].

The EDLines method [14] is a linear time line segment detector. It obtains edge segments (clean and continuous pixels) using the Edge Drawing algorithm [13]. According to the experiment of Akinlar and Topal [17], the parameter-free EDLines could run up to seven times faster than LSD with similar results.

The CannyLines method is developed to extract more meaningful line segments robustly from images. The Helmholtz principle, combining both gradient orientation and magnitude information, is used for verification. The line segments detected by CannyLines are longer than those of the other two methods. However, with its more complex steps, it is time-consuming.

Since this paper aims to achieve real-time detection, a method that is adaptive to different images without tuning of parameters and that has good accuracy and a high processing speed is the reasonable choice. Since the three methods are all parameter-free, and there is no large difference in their accuracy, this paper chooses the fastest method, EDLines, as the line segment detecting method.

4.2. Line Segments Linking. Due to image imperfection (e.g., weak contrast, clutter, blur, and noise), problems still persist when applying the line segment detector to practical civil infrastructure images [18]: (1) Edges of objects comprise small line segments with different orientations. (2) Missing line segments lead to edge discontinuity. To segment the image with windows, a long line, instead of many short line segments, is needed to indicate one edge of a window or wall. So before the segmentation, measures should be taken to obtain long lines from the line segments produced by EDLines.

After selecting vertical or horizontal line segments, an algorithm, a modified version of that proposed by [19], to link line segments is implemented. From computational geometry, two line segments that meet the following two criteria can be linked: (1) The two line segments are nearly collinear. (2) The two line segments are adjacent. For the collinearity problem, Kou et al. [19] transform it into an angle determination problem. For two line segments, there are four endpoints, named A, B, C, and D (Figure 2).

If every two endpoints are connected as a line segment, all these line segments can be matched to three pairs (AB and CD; AC and BD; AD and BC), with no consideration of their orientation and orders. While the three angles (α, β, and γ) constructed by the three pairs are all less than a specified tolerance, the original two line segments (AB and CD) are considered as "collinear." The first criterion could be written as follows:

$$\max(\alpha, \beta, \gamma) < \tau_1, \tag{1}$$

where τ_1 is the specified tolerance.

The second criterion can be written as follows:

$$\min(d_1, d_2, d_3, d_4) < \delta, \tag{2}$$

where d_1, d_2, d_3, and d_4 are the Euclidean distances of the end points AC, AD, BC, and BD, respectively, and δ is the specified tolerance.

This means that if the smallest of the distances between every two end points of the two line segments is smaller than the tolerance, the two line segments (AB and CD) are considered to be "adjacent." According to Kou et al. [19], if two line segments meet both (1) and (2), the two line segments that are mostly parallel, are close, and have no overlap (Figure 3), which commonly occurs in edge detection and line detection, and then in most cases they should be linked as one line.

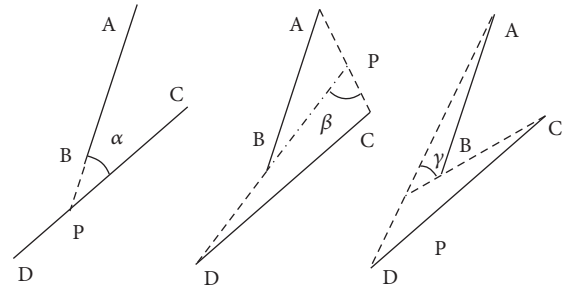
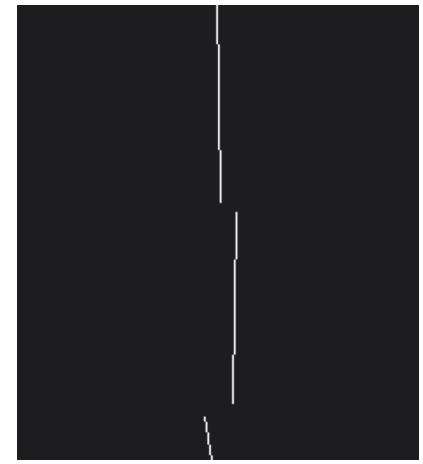
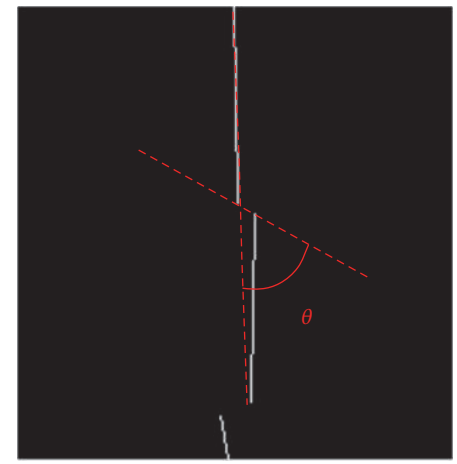


Figure 2: Illustration of constructed angles by the three pairs of endpoints.



(a)



(b)

Figure 3: Illustration of two line segments being mostly parallel, close, and with no overlap but not meeting criteria (1).

Input: line segments set L
Output: the new line segments set T
(1) $T \leftarrow \emptyset$
(2) **for each** line segment $l \in L$ **do**
(3) $S \leftarrow \emptyset$
(4) **for each** line segment $s \in L$ $(s \neq l)$ **do**
(5) **if** s is collinear with l **then**
(6) $S \leftarrow S \cup s$
(7) $T \leftarrow$ newline (l, S)
(8) **function** newline (l, S)
(9) **if** $S \neq \emptyset$ **then**
(10) find the most close line segment $s \in S$
(11) **if** s is adjacent to l **then**
(12) $l \leftarrow$ link (l, s)
(13) delete s from S
(14) newline (l, S)
(15) **return** l
(16) **function** link (l, s)
(17) find the top most vertex u and bottom most vertex v of l and s
(18) draw a line segment between u and v

Algorithm 1: The algorithm of line segments linking.

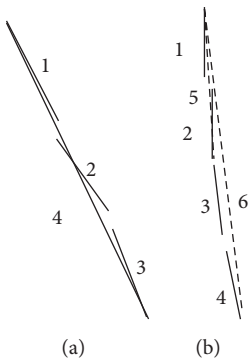


Figure 4: Diagram of line linking. (a) Three short line segments 1, 2, and 3 are linked as a long line segment 4. (b) 1 and 2 are linked as 5 (short dashed), if collinearity is tested between 5 and 3, and then 4, the linking result, would be 6 (long dashed).

But, obviously, this situation does not meet (1). Therefore, based on the two formulas, the authors add a condition to (1):

$$\max(\alpha, \beta, \gamma) < \tau_1 \quad \text{or } \alpha < \tau_2, \tag{3}$$

where α is the angle of the original two line segment. τ_2 is a specified tolerance, which is smaller than τ_1. Therefore, the authors propose that if two line segments meet both (2) and (3), they are linkable.

When two line segments are linkable, the top-most and bottom-most endpoints of the segments are used to construct a new line segment (Figure 4(a)). At the same time, the original line segments are eliminated.

At this time, it is important that any new line segment should not be tested with other line segments to see if they are collinear. Or that would lead to a serious fault (Figure 4(b)). Fortunately, this is naturally avoided in the algorithm (Algorithm 1).

After linking, the long line segments, whose length is larger than a threshold, are picked out to be classified.

5. Classifier Designing

5.1. Subimage Extraction. To determine which long line segment should be the candidate for the window edge, it is necessary to inspect the regions on both sides of the line segment. Therefore, the subimages of both regions are extracted from the image, whose shape and size are chosen experimentally. Since only rectangular windows are considered in this paper, a rectangle is used as the shape of the side region, with its longer side parallel to the line segment and having the same length. The side ratio of the rectangle is set as 0.5.

5.2. Features Extraction. According to human intuition, when a window is seen from outside, the edge of the window usually has two obvious features. (1) The color of the regions on the two sides of the edge has sharp contrast. (2) The gray scale of the window side is lower than that of the wall side. This paper uses feature (1) to select the candidate line segments as the edges of window and then feature (2) to label the window side.

The digital image is specified by a RGB color model, which is a mixture of three primary colors: red, green, and blue. After extracting the side subimages of the line segment, the histogram of each primary image is calculated by dividing the range (0–255) into several bins N_{bins}. Using $\mathbf{H}_{r1}$, $\mathbf{H}_{g1}$, $\mathbf{H}_{b1}$, and $\mathbf{H}_{r2}$, $\mathbf{H}_{g2}$, $\mathbf{H}_{b2}$ represent the histogram vectors of red, green, and blue primary images of each subimage. The subscript 1 represents the left or upper side of the line segment

Table 1: The window position related to the point.

Point label	Window position
1	Upper, left
2	Upper, right
3	Below, left
4	Below, right

and subscript 2 the right or lower side. The same meanings of subscripts are used in the following contexts. Then

$$\mathbf{D} = \left\|\mathbf{H}_{r1} - \mathbf{H}_{r2}\right\|_2 + \left\|\mathbf{H}_{g1} - \mathbf{H}_{g2}\right\|_2 + \left\|\mathbf{H}_{b1} - \mathbf{H}_{b2}\right\|_2, \quad (4)$$

where $\mathbf{D}$ represents the color difference between the two sides. The larger the value of $\mathbf{D}$ is, the more obvious one side contrasts with the other.

Taking g_{m1} and g_{m2} as the average gray scale of both subimages, if $g_{m1} > g_{m2}$, then the line segment is labeled with 1, or else labeled with 0. For the edge of the image, the image side is assumed as the window side. So the left and top edges are labeled with 0, while the bottom and right edges are labeled with 1. The reason for including the edge of image is that sometimes windows may not be completely included in the image.

When all the long line segments in one image are processed as above, the average of all their contrast differences $\mathbf{D}$ is calculated as follows:

$$\mathbf{D}_m = \frac{1}{n}\sum_{i=1}^{n}\mathbf{D}_i, \quad (5)$$

where n is the number of long line segments.

$\mathbf{D}_m$ is chosen as the threshold to classify the long line segments. The line segment whose contrast difference $\mathbf{D}$ is larger than $\mathbf{D}_m$ is selected as the candidate edge line segment.

6. Wall between Windows Recognition

6.1. Image Segmentation. After establishing the candidate edge line segments, the image would be segmented into several areas by the line segments.

As the line segments are always shorter or longer than the window edges, all the candidate line segments are extended to the edge of the image. Then, the intersection point of each pair of extended line segments and each line segment and image edge are calculated. Since all the line segments have been labeled above, each intersection point is also labeled (Table 1) to indicate the window position relative to the point (Figure 5).

6.2. Wall Recognition. As mentioned above, the strategy to recognize a wall between windows is to find a region of which both sides are window regions. So firstly, the window regions should be located. To achieve that, all the subimages segmented during the image segmentation part are inspected with two constraints. (1) The area of the subimage should be larger than a threshold T_a. (2) The clockwise order of the vertexes of the subimage should be 4, 3, 1, 2 starting from the upper left vertex (Figure 5).

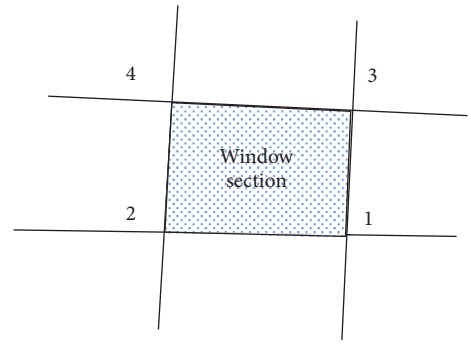


Figure 5: Diagram of the point label.

Table 2: Threshold setting.

Threshold	Value
τ_1	$Ls/120$
δ	$7\pi/180$
τ_2	$\pi/180$
N_{bins}	11
T_a	0.05

Note: Ls is the length of the short edge of the image.

During the inspection, the point where the subimage meets the two constraints would be recognized as a window region, and the four vertexes are recorded. Usually, there should be only two window regions. Vertexes 3 and 1 from the left region and vertexes 4 and 2 from the right are then used as the vertexes of the wall between windows regions.

7. Implementation and Results

In order to evaluate the overall performance of the proposed method, the algorithm is developed in Matlab 2014a. During the line segment detection step, the code provided by Topal and Akinlar [13] is used. All the programs are implemented on a laptop (Lenovo E531 with Intel i5-3230M and 8-GB memory).

The thresholds of the method are set experimentally as shown in Table 2. An example is shown in Figure 6, where (a) is the original image, (b) shows the horizontal and vertical candidate line segments detected, (c) shows the regions segmented by the candidate line segments after extension, (d) illustrates the window sections located, and (e) is the detection result, from which the wall between the windows is extracted.

Figure 7 shows another example of the detection result.

Precision and recall ratios are used to measure the detection performance of the method. They are calculated as follows:

$$\text{precision} = \frac{\text{TP}}{(\text{FP} + \text{TP})}$$
$$\text{recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})}, \quad (6)$$

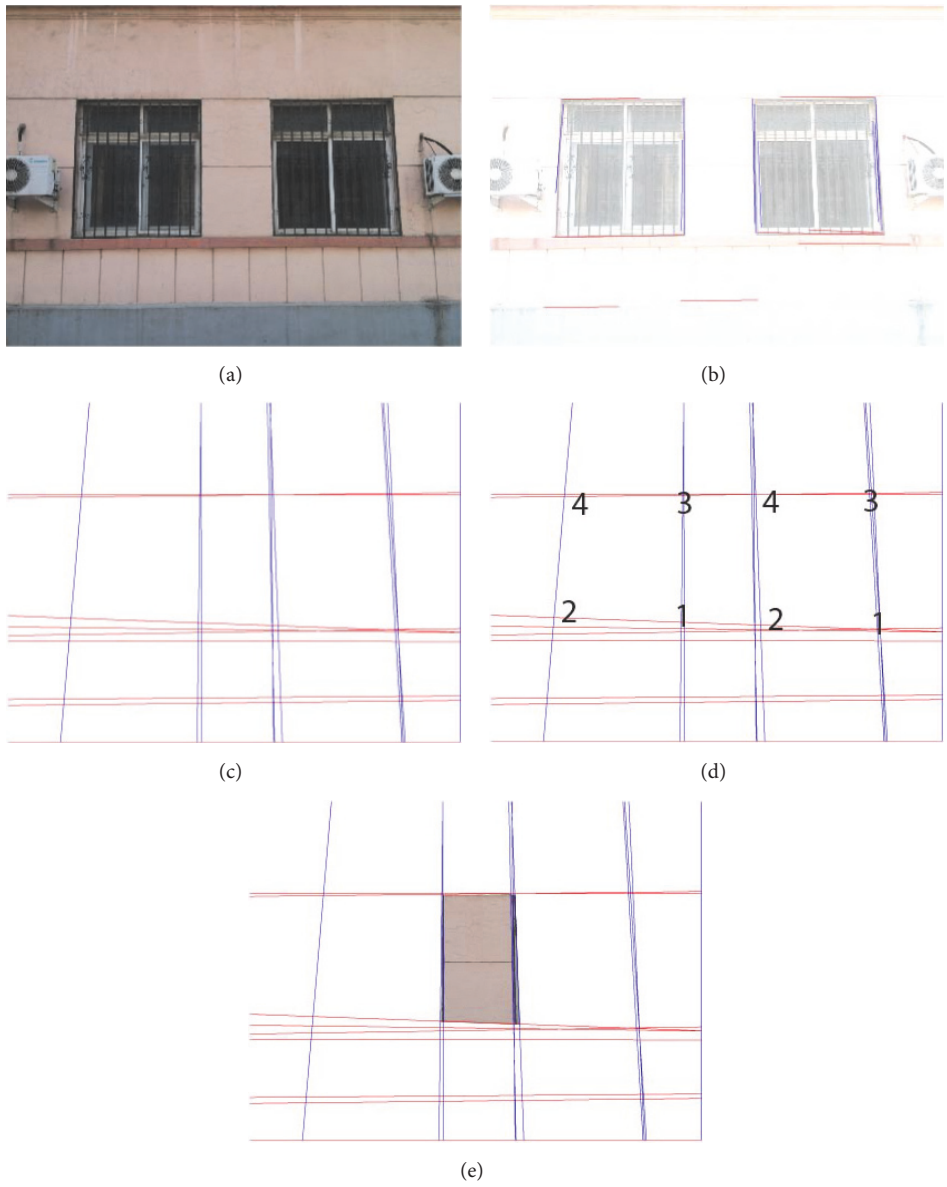


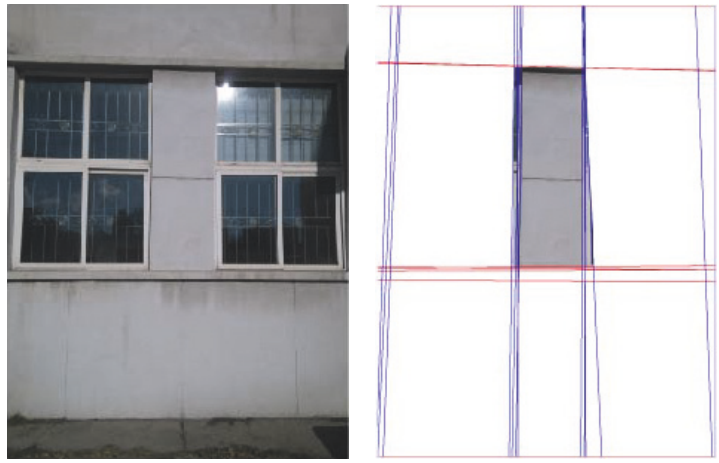
Figure 6: An example of the detection result. (a) The original image. (b) Candidate line segments. (c) Regions segmented. (d) Window section located. (e) Wall between windows extraction.

where TP is the number of walls between windows that are correctly detected as walls between windows. FP is the number of other areas incorrectly detected as walls between windows. FN is the number of walls between windows incorrectly detected as other areas. The incomplete result is also treated as incorrect, because it cannot be used in future damage detection.

High precision means that many detected walls between windows are actual walls between windows, whereas low precision means that few detected walls are actual walls. Similarly, high recall means many actual walls between windows are correctly detected, whereas low recall means that few actual walls are correctly detected. Both sets of results represent the quality of the detection result of the method. A set of images captured by a mobile phone (Xiaomi 2) around the campus is used to test the method. Before testing, the images that do not meet the specified preconditions of the paper are removed manually. So, 20 images are actually tested. The resolution of the images is $2448 * 3264$. The precision and recall ratio of the test results are shown in Table 3.

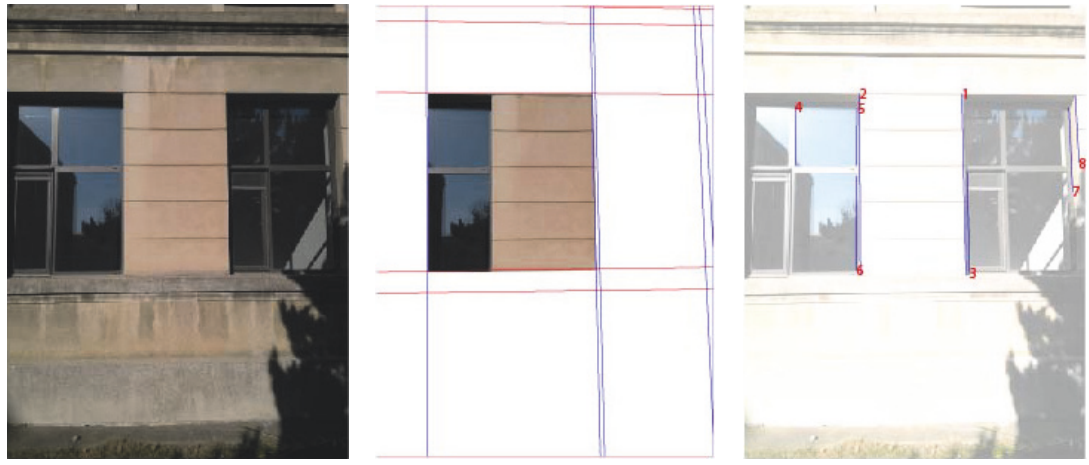
8. Discussion

According to the results, the quality of the method is acceptable. Although the precision is not very high, the recall reaches 85.71%. This means that only a few of the actual walls would be detected as other objects. However, the FP is a little high because some lines that are not window edges also have sharp contrast between their two sides. Take one typical image (Figure 8) as an example. Due to the reflection



(a) (b)

Figure 7: An example of the detection result. (a) The original image. (b) Wall between windows extraction.



(a) (b) (c)

Figure 8: Example for incorrect detection: (a) original image. (b) The incorrect detection result. (c) The candidate edge line segments of the image.

Table 3: Testing results.

Image number	TP	FP	FN	Precision	Recall
20	12	6	2	66.67%	85.71%

of neighboring structures, the contrast of line segment 4 is sufficiently large to lead to an incorrect selection as a window edge. Since its label is the same as that of line segments 2, 5, and 6, the latter three actual window edge line segments have no opportunity to play a part in the correct detection by the proposed method.

The average time cost of the images is 8.3 s. Considering the purpose of the paper, this is acceptable. The time is heavily dependent on the complexity of the image. The length of time required increases with the number of line features and complex textures in the image. For the image in Figure 6 the time is 8.225 s, and for the image in Figure 7 the time is 8.716 s, as there are many line features within the guardrails.

9. Conclusion and Future Work

The traditional method to assess the damage of a structure element after an earthquake is time-consuming, costly, and expert-dependent. Incorporating image technology into the assessment can make the task simpler and faster and is especially feasible when there are many people who can take pictures of the damaged sites. Thanks to the preresearchers who have advanced image processing methods, we can now get more information from images. With the intention of simplifying the assessment, this paper tries to develop a method based on a single image.

As there are few studies on structure element recognition, especially recognition of walls, the paper develops a method to automatically recognize walls between windows from a

single image. The method first detects the line segment in the image and then picks out the horizontal (near horizontal) and vertical (near vertical) line segments and links them to get longer line segments using several principles. The color features of the two sides of each line segment are calculated and used for selecting and labeling candidate window edges. While the image is segmented by the candidate line segments, window regions are located, and then the wall between the windows is located.

Real images taken by mobile phone cameras are used to test the validity of the method. The results show that the method can detect the wall between windows when there is only one target in the single image.

Although the method is for images that have only one wall between windows, it can be easily adapted to apply to two or more targets in a single image. Moreover, with only simple adaption of the region location strategy, other types of walls can be detected.

Since the precision of the method is not very high and it is mostly based on the feature classifier strategy, future work will be conducted regarding this issue. As the work of this paper aims to achieve automatic assessment of structural elements, automatic retrieval of defect information on the wall will also be attempted in future work.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

Innovative research group project (Grant no. 51421064) and general project (Grant no. 51578114) of Natural Science Foundation of China and the Fundamental Research Funds for the Central Universities (DUT16TD03) support this work. The authors would like to thank their sponsor.

References

[1] B. Sun and G. Zhang, "Statistical analysis of the seismic vulnerability of various types of building structures," *China Civil Engineering Journal*, vol. 45, no. 6, pp. 26–30, 2012 (Chinese).

[2] C. Koch, K. Georgieva, V. Kasireddy, B. Akinci, and P. Fieguth, "A review on computer vision based defect detection and condition assessment of concrete and asphalt civil infrastructure," *Advanced Engineering Informatics*, vol. 29, no. 2, pp. 196–210, 2015.

[3] T. Yamaguchi and S. Hashimoto, "Fast crack detection method for large-size concrete surface images using percolation-based image processing," *Machine Vision and Applications*, vol. 21, no. 5, pp. 797–809, 2010.

[4] Z. Zhu, *Column Recognition and Defects (or) Damage Properties Retrieval for Rapid Infrastructure Assessment and Rehabilitation Using Machine Vision*, Department of Civil and Environmental Engineering, Georgia Institute of Technology, Atlanta, USA, 2011.

[5] S. German, I. Brilakis, and R. Desroches, "Rapid entropy-based detection and properties measurement of concrete spalling with machine vision for post-earthquake safety assessments," *Advanced Engineering Informatics*, vol. 26, no. 4, pp. 846–858, 2012.

[6] Z. Zhu and I. Brilakis, "Concrete column recognition in images and videos," *Journal of Computing in Civil Engineering*, vol. 24, no. 6, pp. 478–487, 2010.

[7] S. C. Lee and R. Nevatia, "Extraction and integration of window in a 3D building model from ground view images," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. II113–II120, USA, July 2004.

[8] M. Recky and F. Leberl, "Windows detection using K-means in CIE-Lab color space," in *Proceedings of 20th International Conference on Pattern Recognition (ICPR)*, pp. 356–359, Turkey, August 2010.

[9] M. Miljanovic, T. Eiter, and U. Egly, "Detection of windows in facades using image processing algorithms," *Indian Journal of Computer Science Engineering*, vol. 3, no. 4, pp. 539–547, 2012.

[10] J. Miao, J. Chu, and G. Zhang, "Window detection based on constraints of image edges and glass attributes," *Journal of graphics*, vol. 36, no. 5, pp. 776–782, 2015 (Chinese).

[11] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, no. 1, pp. 11–15, 1972.

[12] X. Lu, J. Yao, K. Li, and L. Li, "CannyLines: A parameter-free line segment detector," in *Proceedings of IEEE International Conference on Image Processing (ICIP)*, pp. 507–511, Canada, September 2015.

[13] C. Topal and C. Akinlar, "Edge drawing: a combined real-time edge and segment detector," *Journal of Visual Communication and Image Representation*, vol. 23, no. 6, pp. 862–872, 2012.

[14] C. Akinlar and C. Topal, "EDLines: A real-time line segment detector with a false detection control," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1633–1642, 2011.

[15] G. V. G. Rafael, J. Jérémie, M. Jean-Michel, and R. Gregory, "LSD: a fast line segment detector with a false detection control," *Pattern Analysis Machine Intelligence IEEE Transactions online*, vol. 32, no. 4, pp. 722–732, 2010.

[16] A. Desolneux, L. Moisan, and J. M. Morel, *From Gestalt Theory to Image Analysis: A Probabilistic Approach*, Springer, 2008.

[17] C. Akinlar and C. Topal, "Edpf: a real-time parameter-free edge segment detector with a false detection control," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 26, no. 1, Article ID 1255002, 22 pages, 2012.

[18] F. Dai and Z. Zhu, "Line Segment Grouping and Linking: A Key Step Toward Automated Photogrammetry for Non-Contact Site Surveying," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 79, no. 3-4, pp. 371–384, 2015.

[19] Z. Kou, Z. Shi, and L. Liu, "Airport detection based on Line Segment Detector," in *Proceedings of International Conference on Computer Vision in Remote Sensing (CVRS)*, pp. 72–77, China, December 2012.