Research Article

Linear Correction and Matching Method for 3D Line Structure Reconstruction

Di Jia , Yuxiu Li, Si Wu, and Ying Liu

School of Electronic and Information Engineering, Liaoning Technical University, Huludao 125105, Liaoning, China

Correspondence should be addressed to Di Jia; lnu_jiadi@163.com

Received 1 March 2020; Revised 20 April 2020; Accepted 4 May 2020; Published 20 May 2020

Academic Editor: Raffaele Solimene

Copyright © 2020 Di Jia et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The 3D reconstruction technique using the straight-line segments as features has high precision and low computational cost. The method is especially suitable for large-scale urban datasets. However, the line matching step in the existing method has a mismatching problem. The two main reasons for this problem are the line detection result is not located at the true edge of the image and there is no consistency check of the matching pair. In order to solve this problem, a linear correction and matching method for 3D reconstruction of target line structure is proposed in this paper. Firstly, the edge features of the image are extracted to obtain a binarized edge map. Then, the extended gradient map is calculated using the edge map and the gradient to establish the gradient gravitational map. Secondly, the straight-line detection method is used to extract all the linear features used for the 3D reconstruction image, and the linear position is corrected by the gradient gravitational map. Finally, the point feature matching result is used to calculate the polar line, and the line matching results of the adjacent three images are used to determine the final partial check feature area. Then, random sampling is used to obtain the feature similarity check line matching result in the small neighborhood. The aforementioned steps can eliminate the mismatched lines. The experimental results demonstrate that the 3D model obtained using the proposed method has higher integrity and accuracy than the existing methods.

1. Introduction

Using the camera imaging model to recover the 3D structure of the object from the acquired 2D image sequence is one of the classic problems in the field of computer vision. The 3D reconstruction refers to the establishment of mathematical models suitable for computer representation and processing of 3D objects. The 3D reconstruction is the basis for processing, manipulating, and analyzing the properties of 3D objects in a computer environment. It is also a key technology for establishing virtual reality that expresses the objective world in a computer. How to make computers perceive the 3D environmental information has always been one of the goals in the field of computer vision. The development of computer vision and deep learning has provided significant enhancement in fields such as autonomous driving, biometrics, video recognition, and drones. However, if these areas want to further improve, 3D reconstruction may be a good breakthrough.

The existing 3D reconstruction technology generally uses structure-from-motion (SfM) [1] approaches and multiview stereo (MVS) [2] pipelines (e.g., PMVS [3] or SURE [4]). The former can obtain the sparse point cloud model of the scene and the camera pose information and apply it to MVS to develop a 3D dense point cloud model. However, because the feature point dataset is very large, the MVS algorithm has a slow processing speed, which often takes a large amount of time and computing memory. In addition, viewing in the point cloud viewer has become extremely difficult. Moreover, the image-based 3D reconstruction technology is affected by factors such as lighting and occlusion when extracting feature points. At the same time, it is sensitive to the accuracy of feature point matching and the accuracy of camera correction when calculating the camera projection matrix and solving space points. Correctly extracting and matching the feature points and accurately solving the 3D geometric relationships have always been difficult problems in the field of computer vision. Therefore, more complex

geometric primitives can be selected as the data representation, such as planes (e.g., [5–7]) or lines (e.g., [8–10]). By analyzing the pinhole camera model, epipolar geometry, and various line segment detection algorithms, it is found that 3D reconstruction based on line matching is feasible. In addition, the surrounding artificial buildings have prominent line segment geometric features. If the relevant 3D information is extracted and matched, the 3D reconstruction efficiency can be enhanced.

The unique 3D information can be used to extract and match the related 3D information using the unique geometrical features of the line segments. These line segments can be obtained by any line segment detector. The two most commonly used line detection algorithms are LSD [11] and EDL [12]. Both algorithms can provide accurate and almost abnormal detection in a very effective way. Figure 1 shows an example image with line segments obtained using the LSD algorithm.

The literature shows that the 3D reconstruction technology has evolved from a point-based motion recovery structure algorithm to a line-based multiview stereo vision algorithm, but each algorithm has its own advantages and disadvantages. Currently, how to obtain a high-precision 3D scene model is also the focus of 3D reconstruction research. Since the structure of the scene is usually complicated and of large scale, obtaining a high-precision 3D model is still a problem that deserves attention and requires significant amount of resources, energy investment, and technical research.

2. Related Work

The point-based motion recovery structure algorithm relies heavily on the unique textures in the scene and appears weak when facing some monotonous environments. Although, the SFM algorithm strives to create sufficient feature matches in order to successfully calculate the correct camera pose, the 3D models generated are usually very sparse. Since the linear characteristics of the artificial building environment are very obvious and the line segment is the most common geometric feature in the artificial building environment, it would be a good choice to complete the feature extraction and matching using the straight-line segment feature.

Bay et al. [13] used line segments from two uncalibrated images to determine the relative camera poses and to compute a piecewise planar 3D model. However, this method is not suitable for processing more than two images, is not robust when dealing with unstable lighting conditions, and is unable to handle some outdoor scenes.

Further, Schindler et al. [14] incorporated the Manhattan-world assumption into the reconstruction procedure to decrease the computational complexity and to reconstruct buildings from two views.

In 2010, Jain et al. [15] proposed a method of reconstructing lines from multiple different stereo images. The method does not require the correspondence of line segments in different images. The method independently reconstructs the line segments using connectivity constraints

and then calculates the final 3D model by merging. Although this method has achieved good visualization, it is not suitable for large-scale datasets.

In 2014, Micusik and Wildenauer [16] proposed a SLAM-like system with line matching through narrow baselines and showed impressive results, especially for indoor scenes. However, this method only attempts to estimate the camera pose estimation and 3D reconstruction through line segments is extremely difficult.

Hofer et al. [17] proposed a public line-based 3D reconstruction tool, called Line3D++. The method first establishes a large set of potential line correspondences between images through weak epipolar constraints and uses a scoring formulation based on mutual support to separate correct matches from incorrect matches for each segment. The final line-based 3D model is obtained by clustering the 2D segments from different views using effective graph-clustering formulas. However, the 3D reconstruction results of Line3D++ method will lose part of the line structure mainly because the line detection result of the line matching step is not located at the true edge of the image, and there is no consistency check of the matching line pair.

In order to solve this problem, this paper first corrects the LSD line detection results produced by Line3D++ and then uses the epipolar constraint principle to eliminate the mismatched lines. Finally, an accurate and complete 3D reconstruction result is obtained.

3. Correct Line Position

Let the image be I . Use the Canny operator [18] to perform edge detection on I and obtain the edge map E . Solve the gradient map G by E , using the form of first-order difference:

$$\begin{aligned} d(i, j) &= \sqrt{d_x^2 + d_y^2}, \\ d_x(i, j) &= I(i+1, j) - I(i, j), \\ d_y(i, j) &= I(i, j+1) - I(i, j), \end{aligned} \quad (1)$$

where (i, j) are the coordinates of the pixel, while $d_x(i, j)$ and $d_y(i, j)$ are the first-order partial derivatives of x and y directions, respectively.

Figure 2 shows the construction process of the extended gradient map, where Figure 2(a) shows the edge graph and Figure 2(b) shows the result of Figure 2(a) solved by the gradient formula (1). The gradient direction of the surrounding pixels of the edge pixels is cleared and then all of the surrounding pixels are regarded as the edge pixels. At this time, the gradient is calculated again as shown in Figure 2(c). The above process is repeated until all the pixels are traversed, and the final result is obtained as shown in Figure 2(d).

The gradient gravitational maps are created by extending the gradient map to correct the line detection results. Figure 3 shows the process of constructing the gradient gravitational map. Let matrix A be an extended gradient map generated from the edge map. Figure 3(a) shows the representation of extended gradient map.

FIGURE 1: Line segment detection using the LSD algorithm.

FIGURE 2: Sketch of extended gradient map construction process: (a) edge graph; (b) result of (a) using formula (1); (c) recalculated gradient; (d) final result.

First, create a matrix $\mathbf{B}$ based on the red region in matrix $\mathbf{A}$ and assign (2, 5), (3, 5), (6, 3), (6, 4), and (6, 5) to the elements located at positions $\mathbf{B}[2, 5]$, $\mathbf{B}[3, 5]$, $\mathbf{B}[6, 3]$, $\mathbf{B}[6, 4]$, and $\mathbf{B}[6, 5]$, respectively. Then, traverse the green area in matrix $\mathbf{A}$, fill in the corresponding positions in $\mathbf{B}$ with the same element values as the pointing positions, and then calculate the purple area in similar manner. Note that if there are edge pixels around it but it points to other pixels, such as

$\mathbf{A}[4, 3]$, the circle is drawn around $\mathbf{B}[4, 3]$, and the edge position at which it intersects first is $\mathbf{B}[6, 3]$. At this time, $\mathbf{B}[4, 3]$ is assigned the same element value as $\mathbf{B}[6, 3]$. The process is repeated until all the elements in $\mathbf{A}$ are traversed and matrix $\mathbf{B}$ is filled, as shown in Figure 3(b). At this time, each element in matrix $\mathbf{B}$ is assigned, thereby obtaining a gradient gravitation diagram $\mathbf{B}$. Figure 3(c) shows the representation of matrix $\mathbf{B}$.

FIGURE 3: Sketch of gradient gravitational map construction process: (a) extended gradient map; (b) traversed matrix; (c) representation of (b).

The line detected by the line detection algorithm usually has a certain deviation from the actual edge. Figure 4 shows an example of linear correction problem. Figure 4(a) shows the line detection result, where the yellow dotted lines and the red solid lines represent the actual edges and the detected lines, respectively. Figure 4(b) shows the corresponding gradient gravitational map of Figure 4(a). Let the point on the line and the distance from the point to the line be ϵ and D , respectively. Correcting the linear position by the gradient gravitational map may have the following two problems:

- (1) The first problem exists in the local area of the corner. If $D(p_1, e_1) < D(p_1, e_2)$, the correction result is $p_1 \in e_1$ and $p_2 \in e_2$. In this case, the gradient gravitational map correction will obtain the incorrect correction result. In fact, p_1 should satisfy $p_1 \in e_2$.
- (2) The second problem is at the abutting edge. If $D(p_3, e_2) < D(p_3, e_3)$, the correction result is $p_3 \in e_2$ and $p_4 \in e_3$. In this case, incorrect correction result will be obtained by the gradient gravitational map correction. In fact, p_3 should satisfy $p_3 \in e_3$.

In order to address the aforementioned problems, this paper proposes the following straight-line correction method.

In Figure 5, the blue line is the correct edge position. Take the yellow straight-line segment l as an example. The two endpoints of yellow line are P and Q . The yellow line is divided into n equal parts, and the equal points are recorded as $E_1, E_2, \dots, E_{n-1}$. The corrected positions P' and Q' of the endpoints P and Q , respectively, and the corrected positions $E'_1, E'_2, \dots, E'_{n-1}$ of the bisector are calculated by the gradient gravitational map.

Let K be the slope of the line segment. Calculate the slope $K(P'E'_1), \dots, K(E'_{n-1}Q')$ of each small line segment after correction. Then, sort the sequence in ascending order to obtain the new sequence $K_1 < K_2 < \dots < K_n$. Take out the middle w consecutive slopes $K_{[(n-w/2)+1]}, K_{[(n-w/2)+2]}, \dots, K_{[(n-w/2)+w]}$ and calculate the standard deviation as

$$\sigma = \sqrt{\frac{1}{w-1} \sum_i^w (K_{[(n-w/2)+i]} - \bar{K})^2}, \quad (2)$$

$$\text{where } \bar{K} = \left(\sum_{i=1}^w K_{[(n-w/2)+i]} / w \right).$$

Set the threshold ϵ ; if $\sigma < \epsilon$, then add $K_{[(n-w/2)]}, \dots, K_1$ from left to right to the aforementioned w consecutive slopes and calculate a new standard deviation σ' for each addition and check the condition $\sigma' < \epsilon$. If condition is true, continue to join the next one and repeat the above operation. Otherwise, stop joining and remove the current added slope. Perform same operations on $K_{[(n-w/2)+w+1]}, \dots, K_n$.

At this time, the line segment that holds the slope is saved with a total of s . The total number of endpoints is $2s$, and the connectivity is determined by the number of occurrences of the same endpoint. Let $N(x)$ denote the number of occurrences of endpoint x . If there are two endpoints satisfying $N(x) = 1$ and there are $s-1$ endpoints satisfying $N(x) = 2$, then all the saved small line segments are connected, and the two points satisfying $N(x) = 1$ are taken as new endpoint pairs. It is extended to maintain the same length as the original straight-line segment.

The extension method is as follows.

Assume that the distance between the two ends p and q of the line segment l in Figure 6 is shortened by d_1 and d_2 , respectively, and p_{11} and q_{11} are used as the starting nodes. Then, the extended line vector is calculated as

$$\begin{cases} V_{p_{11}} = Vt \left(p_{11}, d_1 \cdot \frac{V(q_{11}, p_{11})}{|V(q_{11}, p_{11})|} \right), \\ V_{q_{11}} = Vt \left(q_{11}, d_2 \cdot \frac{V(p_{11}, q_{11})}{|V(p_{11}, q_{11})|} \right), \end{cases} \quad (3)$$

where Vt is a calculation vector function, while $V_{p_{11}}$ and $V_{q_{11}}$ are determined to obtain a new pair of endpoints of the corrected straight-line segment.

The line matching results are refined. The polar line in the adjacent three images is calculated using the point feature matching result. Then, the matching results are combined to determine the final verified local feature area and random sampling is used to verify the feature similarity in the small neighborhood. Thus, the incorrect matching line features are eliminated. If the matching line exists only in two adjacent images, the above method is only performed in

FIGURE 4: Example of linear correction problem: (a) detected lines and actual edges; (b) gradient gravitational map of (a).

FIGURE 5: Sketch map of line correction.

FIGURE 6: Line extension diagram.

two adjacent images. Figure 7 shows the process of determining small neighborhoods by combining epipolar lines.

In Figure 7, take three adjacent images I_1, I_2 , and I_3 as examples. The blue lines are the polar lines, while the yellow and the black lines represent one of the polar lines and the corresponding matching line, respectively. In this paper, the polar and the corresponding matching lines are used as reference to obtain the small neighborhood $A, B, C, A', B', C, A'', B'', C''$, around the line, that is, the areas surrounded by the green lines. The specific solution method is as follows.

Firstly, the bisection points $\{a', b', c', \dots\}$ are obtained using the bisection of line segment L' . Then, according to epipolar constraint, the corresponding polar lines of $\{a', b', c', \dots\}$ in Figure 7(a) and 7(c) are obtained, and the corresponding points $\{a, b, c, \dots\}$ and $\{a'', b'', c'', \dots\}$ are obtained according to the intersection of the polar and L, L'' . The process determines the corresponding points on the matching lines L, L' , and L'' and calculates the similarity of the local regions of the lines by determining the small neighborhood features of $\{ab, bc, \dots\}$, $\{a'b', b'c', \dots\}$, and $\{a''b'', b''c'', \dots\}$, respectively. In order to speed up the calculation, the regional similarity can be calculated by random sampling.

Secondly, the size of each small neighborhood is determined. In the experiment, let the radius threshold be R ,

and save the pixels smaller than the radius threshold in a new image I_m . Finally, the gradient direction determines whether the neighborhood is the left or the right neighborhood to which the pixel belonging to I_m is pointing. In order to maintain the consistency of the direction, the area pointed by the straight-line gradient represents the right neighborhood and the other side is the left neighborhood.

The similarity of the line neighborhood is determined by calculating the similarity of the pixel colors within the region. Let the corresponding matching areas of L, L' , and L'' be NE, NE' , and NE'' , respectively. The number of pixels in areas NE, NE' , and NE'' is m, m' , and m'' , respectively. The neighborhood similarities of $NE'-NE$, $NE'-NE''$, and $NE-NE''$ are calculated. The neighborhood similarity between NE' and NE is

$$\text{Sim}(NE', NE) = \left| \frac{\sum_{k=1}^m I_k}{m'} - \frac{\sum_{k'=1}^m I_{k'}}{m} \right|. \quad (4)$$

The neighborhood similarities between $NE'-NE''$ and $NE-NE''$ are calculated in similar manner as equation (4).

The calculation is performed using the above method. If the corresponding regions in the three images are randomly selected to have high similarity, it is determined that the matching straight line is correct. In this way, the straight line is refined, and all three adjacent images in the dataset are refined in order to obtain the line matching results.

Construction of gradient gravitational map is shown in Algorithm 1.

The method of refining the line matching result is shown in Algorithm 2.

4. Experiments

For the experiments, 3.2 GHz CPU, 16G RAM, and a Nvidia GeForce GTX 1060 6GB GPU were used. The proposed algorithm was implemented in C++ and (optionally) also in CUDA. The Herz-Jesu-P25, Castle-P10 from [19], and Brickstone were used as ground truth test sequences.

In the experiment, the local neighborhood radius R was set to 5 pixel values, the number of samples K was 30% of all

FIGURE 7: Determining small neighborhoods by combining epipolar lines: (a) image I_1 ; (b) image I_2 ; (c) image I_3 .

Input: image I .

Output: gradient gravitational map $\mathbf{GM}$

(1) Calculate the edge map E of image I and calculate the size of E

$[M\ N] = \text{sizeof}(E)$

(2) Solve the extended gradient map $\mathbf{EGM}$ of E

(3) Construct a 3D matrix $\mathbf{GM} = \text{Rect}[2\ M\ N]$ called Rect, where $\mathbf{GM}(1, :, :)$ and $\mathbf{GM}(2, :, :)$ are X and Y coordinates, respectively.

Fill in $\mathbf{GM}$ with E and $\mathbf{EGM}$ according to the above method.

(4) Calculate the gravitational value at each location:

$mExit = \text{true}$

while ($mExit$)

for $i = 1:M$

for $j = 1:N$

{

$mExit = \text{false};$

if ($\mathbf{GM}(1, i, j) \neq 0$)

{

$mExit = \text{true};$

for $kx = -1:1$

for $ky = -1:1$

if ($\mathbf{GM}(1, i + kx, j + ky) = 0$)

{

if gradient direction == 45°

draw a circle centered on the current position,
and according to the edge pixels that intersect
first to fill in

$\mathbf{GM}(1, i + kx, j + ky)$ and $\mathbf{GM}(2, i + kx, j + ky)$;

else

according to the $\mathbf{EGM}(i + kx + ky)$; }

}

}

(5) Return $\mathbf{GM}$

ALGORITHM 1: Steps of constructing gradient gravitational map.

small neighborhoods, the similarity threshold T was set to 10, and the proportional threshold P was set to 90%.

In this paper, the effectiveness of the proposed method is illustrated by three parts of comparative experiments. The first shows results of line matching and purification. The second compares the sparse 3D point cloud model with the experimental results of the proposed method. The third part compares the results of Line3D++ with the experimental results of the proposed method. Figures 8–13 show the experiment results.

Figures 8 and 9 show results of line matching and purification. Figures 8(a) and 9(a) show the direct match results with LSD. Figures 8(b) and 9(b) show the results of the

straight line correction. Figures 8(c) and 9(c) show the results of the purification matching.

In Figure 8, Figure 8(b) eliminates lines 17, 20, 18, 13, 15, 27, 9, and 21 in the matching result compared with Figure 8(a). While compared with Figure 8(b), Figure 8(c) eliminates lines 20, 1, and 13. The matching error in Figure 8(a) of this set of experiments is not obvious. For example, although lines 20 and 18 are on the same side plane, they are located in the middle of the side plane in row 1 and at the edge of the side plane in row 2 of Figure 8(a). This situation can easily lead to incorrect matching results. The main reason is that the line of the edge position detected by the LSD is offset to the middle of the side plane. However, the proposed method can solve this problem.

```

Input: Bisection point  $X^1 = \{x_1^1, x_2^1, \dots, x_m^1\}$  of line  $L^1$ , line  $L^0$ ,  $L^2$ , local neighborhood radius  $R$ , sampling number  $K$ , similarity threshold  $T$ , and proportional threshold  $P$ 
Output: Flag bit Flag ( if Flag = true, it is the correct match )
(1) Calculate the polar  $l_i^0 = F_1 x_i^1 (x_i^1 \in X^1)$ ,  $l_i^2 = F_2 x_i^1 (x_i^1 \in X^1)$  according to  $X^1$  and find the intersection sets  $X^0 = \{x_1^0, x_2^0, \dots, x_m^0\}$  and  $X^2 = \{x_1^2, x_2^2, \dots, x_m^2\}$  formed with  $L^0$ , and  $L^2$ , respectively.
(2) Determine the segmented local neighborhood:
First, find two lines  $ll^c$  and  $lr^c$  with distance  $R$  on both sides of  $L^c$ .
for  $c = 0 : 2$ 
   $D(ll^c, L^c) = R$ ;
   $D(lr^c, L^c) = R$ ;
end
Save the determined neighborhood
for  $i = 1 : m-1$ 
  for  $c = 0 : 2$ 
    store the neighborhood around  $x_i^c x_{i+1}^c$  in ImTemp[c]
  end
end
(3) Distinguish left and right neighborhoods:
for  $c = 0 : 2$ 
  From ImTemp[i], the left neighborhood  $A_l^i$  and the right neighborhood  $A_r^i$  of  $L^i$  are determined.
end
(4) Calculate the similarity of lines:
Random sampling of  $K$  local neighborhoods to obtain a new set  $SA_l^0, SA_r^0, SA_l^1, SA_r^1, SA_l^2, SA_r^2$ 
num = 0;
for  $i = 1 : K$ 
  if  $\text{Sim}(SA_l^1(i), SA_l^0(i)) < T \& \& \text{Sim}(SA_l^1(i), SA_r^2(i)) < T \& \&$ 
     $\text{Sim}(SA_r^0(i), SA_r^2(i)) < T \& \& \text{Sim}(SA_r^1(i), SA_l^0(i)) < T \& \&$ 
     $\text{Sim}(SA_r^1(i), SA_r^2(i)) < T \& \& \text{Sim}(SA_l^0(i), SA_r^2(i)) < T$ 
    num++
  if (num/k > P)
    Flag = true
  else
    Flag = false
  end
end
end
(5) Return Flag

```

ALGORITHM 2: Steps of refining line matching result.

In Figure 9, Figure 9(b) eliminates lines 23, 5, 4, 3, 1, 8, and 26 in the matching result compared with Figure 9(a). Compared with Figure 9(b), Figure 9(c) removes line 1. Among all the matched line pairs that are removed, the line pairs with obvious errors are lines 2 and 26. Since they are not located at the true edges of the image, matching errors are caused. When the corrected result is used as an input for matching, a matching result with higher accuracy can be obtained. In addition, lines 1 and 5 in Figure 9(a) are obviously located at different edges. After being processed by the method proposed in this paper, this kind of error can be effectively reduced.

Figures 10 and 11 show a comparison between the sparse and the line-based 3D models. Figures 10 and 11 show the sparse 3D point cloud models of the scene obtained by processing the image set using the SFM algorithm and the line-based 3D reconstruction results, respectively.

It can be seen from the comparison that the 3D sparse point cloud model is able to represent the characteristics of

the building, but the overall structure is not very obvious. On the other hand, although the 3D line segment model is very vague at some curve features, the lines of the building are very obvious. Using the 3D line segment model can more prominently represent the geometric structure of the building, especially for buildings with straight lines and few curves. In addition, the 3D point cloud model has poor reconstruction accuracy in the absence of textures, and the obtained 3D models often appear hollow. The 3D line segment model can provide more sufficient structural information and better reflect the geometric topology of the scene compared with the 3D point cloud model. Thus, the 3D line segment model provides a highly meaningful semantic 3D information for reconstruction.

Figures 12 and 13 compare the Line3D++ results with the experimental results of the proposed method. Figures 12(a), 12(b), 13(a), and 13(b) show the results of Line3D++ and the experimental results of the proposed method, respectively. Figures 12(c), 12(e), 13(c), and 13(e)

FIGURE 8: Results of line matching and purification for viewpoint change image pair.

FIGURE 9: Results of line matching and purification for wide baseline image pair.

FIGURE 10: Comparison of two kinds of 3D representations of Brickstone.

FIGURE 11: Comparison of two kinds of 3D representations of Herz-Jesu.

FIGURE 12: Comparison of Castle reconstruction.

show the two enlarged local areas in Figures 12(a) and 13(a). Figures 12(d), 12(f), 13(d), and 13(f) show the corresponding local enlarged areas in Figures 12(b) and 13(b).

It can be seen from the figures that the 3D reconstruction results of Line3D++ lack many lines. Compared with Line3D++, the proposed improved method has more lines and more local structures, which significantly improve the integrity of the object. The experimental results can be analyzed from different angles and different structures. For example, Figure 12(c) shows the position of the door, and the reconstructed line segment is very sparse. The line segment reconstructed in Figure 12(d) is relatively dense, and many important lines are restored, making the structure of the door clearly visible.

Figure 12(e) shows the position of the brick wall, but most of the reconstructed lines are vertical lines. There are only few horizontal lines, and it is impossible to see what

structure is reconstructed. In contrast, Figure 12(f) restores relatively large number of horizontal lines, making the line segment and the outline of the brick clearer.

Figure 13(c) shows a partial enlargement of the stepped portion. It can be seen that only a few lines have been reconstructed, while Figure 13(d) shows that the step is reconstructed properly due to the extremely high degree of reconstruction integrity of the proposed method. Figure 13(e) shows a partial area of a part of the wall. Obviously, Figure 13(f) has more lines and better results.

Table 1 shows the number of feature lines of the Line3D++ and the method proposed in this paper for two datasets. For the Castle dataset, the number of feature lines in Line3D++ and the proposed method is 1,590 and 1,676, respectively. For the Herz-Jesu dataset, the number of feature lines in Line3D++ and the proposed method is 1704 and 2394, respectively. The number of feature lines in both datasets

FIGURE 13: Comparison of Herz-Jesu reconstruction.

TABLE 1: Number of feature lines on Line3D++ and proposed method.

Dataset	Number of feature lines (Line3D++)	Number of feature lines (proposed method)
Castle	1590	1676
Herz-Jesu	1704	2394

TABLE 2: RMSE on Line3D++ and proposed method.

Dataset	RMSE (cm) (Line3D++)	RMSE (cm) (proposed method)
Castle-P10	10.08	9.16
Herz-Jesu-P25	5.48	5.28

increased significantly. Table 2 shows the RMSE of the Line3D++ and the method proposed in this paper for two datasets. As we can see, the proposed method has a slightly higher accuracy.

Reconstruction of the 3D line segment model was conducted for two sets of classical datasets. By comparing the experimental results and the experimental data in Table 1 and 2 before and after using the proposed method, it can be seen that the proposed method effectively solves the defects of insufficient accuracy and visual effects, for example, there are too many stray lines, and some areas cannot restore the characteristics of buildings. Moreover, the proposed method can improve the matching accuracy, produce detailed model outline, and provide high 3D reconstruction efficiency.

5. Conclusion

This paper presents a linear correction and matching method for 3D reconstruction of target line structure and resolves the

mismatching problem in the line matching step in the Line3D++ algorithm. The gradient map is extended to construct the gradient gravitational map in order to correct the position of the straight-line segment detected by the straight-line extraction method. The epipolar constraint is used to eliminate the mismatched straight lines in order to improve the quality of the 3D reconstruction. The experimental results demonstrate and validate that the 3D reconstruction results obtained by the proposed method are more accurate and complete than the Line3D++.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This project was supported financially by the National Natural Science Foundation of China (no. 61601213), Department of Education of Liaoning Province (China) (no. LR2016045), and Liaoning Provincial Natural Science Foundation of China (no. 2019-ZD-0038).

References

- [1] K. Häming and G. Peters, "The structure-from-motion reconstruction pipeline—a survey with focus on short image sequences," *Kybernetika Praha*, vol. 46, no. 5, pp. 926–937, 2010.
- [2] S. M. Seitz, B. Curless, and J. Diebel, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Proceedings of the 2006 Computer Vision and Pattern Recognition*, IEEE, New York, NY, USA, pp. 17–22, June 2006.
- [3] Y. Furukawa, B. Curless, S. Seitz, and R. Szeliski, "Towards internet-scale multi-view stereo," in *Proceedings of the 2010 Computer Vision and Pattern Recognition*, IEEE, San Francisco, CA, USA, pp. 1434–1441, June 2010.
- [4] M. Rothermel, K. Wenzel, D. Fritsch, and N. Haala, "SURE: photogrammetric surface reconstruction from imagery," in *Proceedings of the LC3D Workshop*, no. 2, Berlin, Germany, December 2012.
- [5] C. Kim and R. Manduchi, "Planar structures from line correspondences in a manhattan world," in *Proceedings of the 2014 Asian Conference on Computer Vision*, pp. 509–524, Shanghai, China, December 2014.
- [6] S. N. Sinha, D. Steedly, and R. Szeliski, "Piecewise planar stereo for image-based rendering," in *Proceedings of the 2009 IEEE International Conference on Computer Vision*, Kyoto, Japan, October 2009.
- [7] C. Raposo, M. Antunes, and J. P. Barreto, "Piecewise-planar stereoscan: structure and motion from plane primitives," in *Proceedings of the 2014 European Conference on Computer Vision*, pp. 48–63, Zurich, Switzerland, September 2014.
- [8] M. Hofer, M. Maurer, and H. Bischof, "Line3D: Efficient 3D scene abstraction for the built environment," in *Proceedings of the 2015 German Conference on Pattern Recognition*, pp. 237–248, Aachen, Germany, October 2015.
- [9] B. Micusik and H. Wildenauer, "Structure from motion with line segments under relaxed endpoint constraints," in *Proceedings of the 2016 International Journal of Computer Vision*, pp. 65–79, Italy, 2017.
- [10] L. Zhang and R. Koch, "Structure from motion from line correspondences: representation, projection, initialization and sparse bundle adjustment," in *Proceedings of the 2014 Journal of Visual Communication and Image Representation*, pp. 904–915, Kyoto, Japan, March 2014.
- [11] L. Wang, S. Y. You, and U. Neumann, "Supporting range and segment-based hysteresis thresholding in edge detection," in *Proceedings of the 2008 International Conference on Image Processing*, pp. 12–15, San Diego, CA, USA, October 2008.
- [12] C. Akinlar and C. Topal, "Edlines: real-time line segment detection by edge drawing," in *Proceedings of the 2011 International Conference on Image Processing*, pp. 2837–2840, Chennai, India, December 2011.
- [13] H. Bay, A. Ess, and A. Neubeck, "3D from line segments in two poorly-textured, uncalibrated images," in *Proceedings of the 2006 International Symposium on 3d Data Processing, Visualization, and Transmission*, pp. 496–503, Chapel Hill, NC, USA, January 2006.
- [14] G. Schindler, P. Krishnamurthy, and F. Dellaert, "Line-based structure from motion for urban environments," in *Proceedings of the 2006 International Symposium on 3d Data Processing, Visualization, and Transmission*, pp. 846–853, Chapel Hill, NC, USA, January 2006.
- [15] A. Jain, C. Kurz, and T. Thormählen, "Exploiting global connectivity constraints for reconstruction of 3D line segments from images," in *Proceedings of the 2010 Computer Vision and Pattern Recognition*, IEEE, San Francisco, CA, USA, pp. 1586–1593, June 2010.
- [16] B. Micusik and H. Wildenauer, "Structure from motion with line segments under relaxed endpoint constraints," in *Proceedings of the 2014 International Conference on 3D Vision*, pp. 13–19, Tokyo, Japan, December 2014.
- [17] M. Hofer, M. Maurer, and H. Bischof, "Efficient 3D scene abstraction using line segments," in *Proceedings of the 2016 Computer Vision and Image Understanding*, pp. 167–178, September 2016.
- [18] W. Rong, Z. Li, and W. Zhang, "An improved Canny edge detection algorithm," in *Proceedings of the 2014 International Conference on Mechatronics and Automation*, Beijing, China, October 2014.
- [19] C. Strecha, W. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen, "On bench marking camera calibration and multi-view stereo for high resolution imagery," in *Proceedings of the 2008 Computer Vision and Pattern Recognition*, IEEE, Las Vegas, NV, USA, December 2008.