An Ink Line Detection Method at Construction Site

Hengyu Yin[1,a], Hongyang Yu[1,b]

[1]Research Institute Electronic Science and Technology,
University of Electronic Science and Technology of China, Chengdu, China
a. getawayfrom@outlook.com, b. hyyu@uestc.edu.cn

Keywords: Lines detection, Image processing, Ink line detection, Plastering robot.

Abstract: When the plastering robot works in rooms, it needs to calculate its own precise position by recognizing the ink line on the ground, so that make the robot achieve the operation accuracy. Due to the special environment of the construction site, the traditional image processing algorithm cannot well filter out the interference of non-ink line on the ground, which makes the final detection result have some errors. Most of the noise in images can be filtered by identifying the color feature of ink lines, which not only improves the recognition accuracy but also declines the time of detection. Then a line detection called cankerworm-crawl algorithm (CCA) is used to identify ink lines segment and using the least squares method to fit the ground ink line in the image, so that make the differences of two walls which plastering by robot is in ±0.5mm.

1. Introduction

In the construction field, plastering operations are generally required for newly rooms to make its walls are flat. In order to make the walls which plastered by plastering robot in the same surface and the differences is in ±0.5mm. It is particularly important for the plastering robot to calculate its exact position. If laser sensors or image modeling methods are used, not only the computational complexity is high, but the accuracy cannot meet the requirements. Installing an RGB-D camera in front of the plastering robot is expensive and difficult to maintain. Therefore, this article adopts a method that install two distortion-free cameras (resolution 1920 * 1080) on the back of the plastering robot to recognize the ink line on the ground (Figure 1) to calculate its precise position. Because of the special operating environment of the robot, there are many disturbances such as wires, shadows, small pits, and scratches on the ground. Using traditional image filtering algorithms and line recognition algorithms cannot accurately identify the ground ink lines. Firstly, we use Gaussian filtering to blur the image to filter out Gaussian noises in the image. Secondly, the blurred image is filtered based on the ink line color features to filter out a lot of non-ink line noises. Thirdly, we use Sobel operator to get the gradient in the y direction of the image, and obtain the ink line edges. Fourthly, based on the edge information, a line detection called cankerworm-crawl algorithm (CCA)[1] is used to find a number of line segments and using the least squares method to fit these segments into a straight line. Finally, by comparing the line parameters of two camera recognized results determine whether it is valid. Thereby we can use the straight line's information to calculate the plastering robot's exact position from the wall.


(a) scratches (b) wires

(b) shadows

Figure 1: Disturbances of images.

2. Gaussian Filter

Gaussian filtering is the most common filtering algorithm in image processing, because most of the noises in the image is Gaussian noises, so using the Gaussian filter will filter a lot of interferences in the image. The two-dimensional distribution of the Gaussian formula is as follows:

$$G(x,y) = \frac{1}{\sqrt{2\pi}\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{1}$$

We often want to retaining the characteristics of the ink line as much as possible while filter out the noises in the image. Because the ink line has many horizontal features, so we should blur the image in y-direction more than x-direction. Therefore, this article uses a 3 * 3 Gaussian kernel for the x-direction and an 11 * 11 kernel for the y-direction. The core of Gaussian filtering is to calculate the Gaussian kernel and then perform the convolution on the image. When σ = 1.5, the kernel size is 3 * 3, blur radius is equal to 1, the Gaussian kernel is shown in formula 2.

$$G = \begin{pmatrix} 0.0453542 & 0.0566406 & 0.0453542 \\ 0.0566406 & 0.0707355 & 0.0566406 \\ 0.0453542 & 0.0566406 & 0.0453542 \end{pmatrix} \tag{2}$$

Finally, we need to perform normalization operations on the calculated Gaussian kernel 、

$$\text{S.t} \qquad \sum_{i=1}^{3}\sum_{j=1}^{3} G(x_i, y_j) = 1 \tag{3}$$

Therefore, the Gaussian kernel in the x direction used in this paper is shown as equation 4, and the calculation method of the Gaussian kernel in the y direction is the same as that in the x direction.

$$G' = \begin{pmatrix} 0.0947416 & 0.118318 & 0.0947416 \\ 0.118318 & 0.147761 & 0.118318 \\ 0.0947416 & 0.118318 & 0.0947416 \end{pmatrix} \tag{4}$$

After filtering by x-direction and y-direction, the image filtered out most of Gaussian noises and retain many horizontal features of the image. We can start the next filtering algorithm which based on the ink line color features.

3. Ink Filter

As we know, an image is stored in computer is a two-dimensional matrix. The numerical value represents the color information of pixel at that point, and multiple pixels can represent the texture characteristics of the image in this area. Although the image has been Gaussian filtered, it cannot completely filter the non-ink noise in the image. If other filtering algorithms are used again, the ink line features will be blurred, that may make us cannot recognized the line or get an inaccurate detection result. We need to extract the features of ink line from the image. For a grayscale image, it is enough to use 8bits memory save one pixel in this image, and the value range is 0~255. But for color image (three-channel, non-transparency), computer usually needs 24bits to represent B, G, R values of one pixel as P(x, y).

$$P(x,y) = (b,g,r)\{0 \leq b,g,r \leq 255 \text{ and } b,g,r \in Z\} \quad (5)$$

Among them: b, g, r respectively represented the three primary colors: blue, green and red. The ink line on the ground of construction site is a black straight line. There are many ink dots may splatter near the ink line when workers draw it on the ground (Fig.1 a) and there will be other interferences too, such as electrical wires (Figure.1 b), ground scratches (Figure.1 a), shadows (Figure.1 c) and so on. For these reasons, this paper proposes an ink-line-based filtering algorithm (Ink-filter) to filter out non-ink line features in the image as much as possible. Due to lighting and environmental influences, the color of the ink line cannot be pure black $P(x,y) = (0, 0, 0)$. Under certain lighting conditions, the values of the ink lines B, G and R will change within a certain range. We only need to set the threshold to determine the B, G and R values of each pixel whether it is belonged to ink line. The specific process is:

1) Create a single channel image I_d and set the values for each pixel is 0.
2) For each pixel of image I_s,the value $P_{I_s(x,y)}$ is:

If $P_{I_s(x,y)} \leq \lambda,\kappa,\mu$, then $P_{I_d}(x,y) = P_{I_s(x,y)}$,
otherwise $P_{I_d}(x,y) = 0$.

Among them, I_d represent a filtered image and I_s is a three-channel color image. λ,κ, μ are the thresholds of b, g, r.

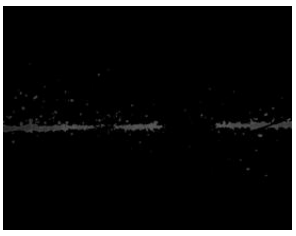


Figure 2: After filtered.

As shown as Figure 2, after filtered, the main features of the ink line can be obtained and most of the noises in the picture already be filtered out.

4. Sobel Operator

Generally, we usually calculate the derivative of a continuous function to represent its rate of change. Since the image is stored as discrete numerical values in the computer, we use first-order differences to represent its change.

$$f'(x) = f(x+1) - f(x-1) \quad (6)$$

Into:

$$f'(x) = (f(x+1), f(x), f(x-1)) \cdot (-1,0,1)^T \quad (7)$$

In the image, we can obtain the change rate of the image in the x direction and y direction by calculating the differences of two directions respectively. The edge information is often related to the gradient. In this article, the ink line generally is a horizontal straight line, so we can obtain the information about the edge of the ink line by performing a differential operation on the image in the y direction. According to equation 7, define an operator to convolve with the image to get the gradient in the y direction. This operator is the Sobel operator. As shown in Figure 3, G_x and G_y represent the operators in the x direction and y direction.

+1	+2	+1
0	0	0
-1	-2	-1

G_x

-1	0	+1
-2	0	+2
-1	0	+1

G_y

Figure 3: Sobel operators.

During the convolution, we can separate the rows and columns to improve the running speed [2]. Suppose image I has M rows N columns, and the size of the convolution kernel S is k * k. We can infer from the convolution formula:

$$I(x,y) = \sum_{m=0}^{M} \sum_{n=0}^{N} I(m,n) \cdot S(s-m, t-n) \quad (8)$$

Moreover:

$$\{(s,t) | 0 \le s < M + K - 1, 0 \le t < N + K - 1\} \quad (9)$$

If we record $a + b$ or $a * b$ as a single operation, when we convolve the image, we can know from equation 8 that the average operation time (Average Time) AT is:

$$AT_1 = \frac{M * N * 2k^2}{M * N} = 2k^2 \quad (10)$$

We decompose the Sobel operator on rows and columns as:

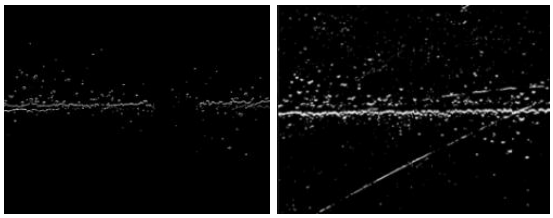
$$G_x = (1,2,1)^T \cdot (-1,0,1) \quad (11)$$

After doing that, when we perform Sobel convolution on the image in the x direction, we can decompose it to use $(1,2,1)^T$ to perform column convolution on the image, and then use $(-1,0,1)$ for row convolution. At this time, the average time AT_2 is:

$$AT_2 = \frac{M * N * 2k + M * N * 2k}{M * N} = 4k \quad (12)$$

From the above, we know that $AT_2 < AT_1$, using decomposition can greatly reduce the time complexity of Sobel operation.

Then we calculate the gradient in the y direction of the image which filtered by ink filter. The result is shown in Figure 4.



(a)after ink filter (b)before ink filter

Figure 4: Edge of image.

The valid detection points obtained before and after filtering based on the ink line color features are shown in Table 1:

Table 1: Valid points in image.

	Before ink filter	**After ink filter**
Valid points	37617	7603

5. Cankerworm-crawl Algorithm

In general, the traditional algorithm of line detection is Hough line detection and its improved algorithms [3][4][5], However, the Hough detection algorithm not only has large time and space complexity, but also sensitive to noises. It is easy to find interference straight lines, which affects the final recognition result. Li Chao [6] used the Freeman chain code idea for straight line detection. But these detection algorithms are too sensitive to other interference straight lines. Because this article only needs to detect the position of the ink lines in the image, if there are too many interference lines, it will cause a large error in the final result. The plastering robot needs to recognize the ink line on the ground to adjust the position in real time and finally reach the working position, so the Hough change is not suit for this. The ink line can be regarded as a parallel line composed of two edges, so we just need to detect the edge of the ink line. Therefore, this paper uses cankerworm-crawl algorithm to detect lines.

CCA is based on the characteristics of the creeping movement of the ruler. According to the bionics idea, the process of finding a straight line is analogous to the creeping process of the ruler on a branch to find food, and the edge information in the image is "food". In order to eat food, the track of ruler crawling is a detected line. The search process is：

1) Use a fan-shaped window to simulate the range that a ruler can crawl. The max angle of the window is 45°, and the length of the window is the ruler length L.
2) Make the point (X_0,Y_0) as start point to begin to detect straight line segment. Check the window of step 1) whether it has some points which can forms a line with start point (X_0,Y_0). If it has, select the point (X_e,Y_e) that furthest from the starting point in the window and make that point as new starting point for the next search. Mark all points on this window, indicating that these points are on a same line. Repeat step 2), otherwise go to step 3).
3) If there are no points to form a line segment, it proves these are not satisfied in the detection window. We show that a straight line has been found. Continue to find other points which not marked by the detection window. When we search all point sets, this algorithm already found all line segments.

Generally, the edge information of an image is usually perpendicular to its gradient direction, so we find the gradient direction of the image and classify it according to the gradient direction. Since the ink lines we detect are close to horizontal straight lines, we only need to consider four gradient

points sets in the interval are sufficient. This not only greatly reduces the complexity of the operation, but also resolves the ambiguity of the search direction that occurs during the detection process. According to the above algorithm idea, the specific process of ink line detection processes as follows:

1) Use Sobel operator to perform a convolution operation on the filtered image I in the x-direction and the y-direction to get the gradient dx and dy of each pixel.
2) According to step 1. We can use formula (13) to calculate the angle θ of gradient direction:

$$\theta = arctan\frac{dy}{dx} \tag{13}$$

3) Base on known dx, dy. Expand the range of θ from $\left(-\frac{\pi}{2},\frac{\pi}{2}\right)$ to $(0, 2\pi)$.
4) Create four sets which classified with 60°~82.5°, 82.5°~105°,240°~262.5°, 262.5°~285°. According the θ of each pixel, put them into different point sets.
5) Using CCA algorithm to detect straight line until all points have been searched.

Because the detection process is completed in different gradient sets, the straight-line detection algorithms in different gradient intervals do not interfere with each other, so the parallel computing can greatly improve the operation efficiency of the program. Currently, CUDA and OpenCL are most popular in parallel computing methods. If you want to use CUDA it requires NVIDIA hardware support. Owing to OpenCL is a general-purpose parallel computing language, so this article uses it to perform a parallel computing on the detection process. Under windows10-64bit operating system, CPU: intel-i5 7200U, graphics card: intel (R) HD Graphics 620, the running time is shown in Table 2:

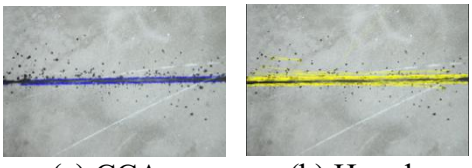
Table 2: The run time of two detection algorithms.

Image	Hough(ms)	CCA(ms)
Before ink filter	73.412	8.211
After ink filter	14.266	1.432

The results of detecting straight line segments are shown in Table 3. Figure 5 shows the visualization results. It can be seen that many interference straight lines have been detected when using the Hough detection algorithm. Because OpenCL library is used in this article, so the runtime of detected straight line greatly declined. The image detection time after using the ink filter is less, which is about 10 times that using the Hough transform, which can fully meet the real-time positioning requirements of the plastering robot.

Table 3: The line segments of two detection algorithms.

Image	Hough(lines)	CCA(lines)
Before ink filter	156	74
After ink filter	82	21



(a) CCA (b) Hough

Figure 5: Results of two algorithms.

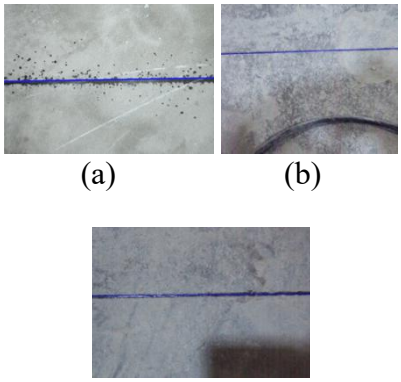
6. Error Elimination

After the above steps, we have detected many short line segments at the edges of the ink line. Because we used the ink filter when filtering the image, this may cause the CCA too sensitive to recognized the ink line when meeting the shadow of the ground or the black wires. It will also be retained in the filtered image as an ink line feature, so we need to check the results to minimize the detected errors. For these results, this paper uses Random Sample Consensus (RANSAC) try to eliminate segments that not contains the edge of ink line. In the end we use the least squares method to fit these segments to a straight line that in the range of the ink line on the ground. Thus, we completed all processes of detected the ink line.

But when using the above method, if the ink line features are obvious and the interferences are less, the accuracy of recognition can reach more than 98%. On the contrary, the RANSAC algorithm will be used to weaken the ink line features, and the interference lines will be misidentified as ink lines, which will result in a great deviation. However, the plastering robot has two cameras at the rear and the positions are relatively parallel, so we can examine the detected result of two cameras and determined whether it is valid. Now, we supposing the detected result of left camera is L_1, the right is L_2. We will discard this recognized result when the difference of slope between L_1 and L_2 is too large. Otherwise we compare the vertical coordinates of the intersections of L_1 and L_2 with X = 0 in the image as Y_{l1}, Y_{l2}. If $Y_{l1} - Y_{l2}$ less than η, The ink line detection is valid, otherwise we will re-identification. η is the difference of Y-axis between left camera and right camera when they are installed. The final results are shown in Figure 6. The coordinates of two points in the detected result is shown as Table 4.

Table 4: The coordinates of results.

	Image(a)	Image(b)	Image(c)
Real result	(0,963), (1920,872)	(0,323), (1920,271)	(0,834), (1920,812)
Detected result	(0,958), (1920,869)	(0,323), (1920,268)	(0,827), (1920,815)



(a) (b)

(c)

Figure 6: The detected results of Figure 1.

7. Conclusions

Compared with the traditional line detection algorithm, this paper proposes a method more suitable for the detection of the ground ink line on the construction site. First of all, we filter an image with Gaussian filter and ink filter, and use Sobel operator to convoluted with that image to

get gradient information of edges. Second, this paper discard traditional Hough transform and use an algorithm which called CCA to detect a straight line. In CCA, we also use OpenCL to reduce the runtime. In the end, we try to eliminate the detected error by the result of left camera and right to get an accurate recognized result. This result makes the differences of two walls which plastering by robot is in ±0.5mm. The recognition algorithm in this paper only has a good recognition effect in the case of relatively constant light. If the light is too dark or too strong, it is easy to filter the ink line features. In addition, if the ground has interference similar to the color of the ink line, it cannot be filtered well. Experiments and research are needed for these cases.

References

[1] Guan B Q, Yu X R, Wang S G. Cankerworm-crawl algorithm for line detection[J]. Chinese Journal of Optical Technique 2005(2):182-186.(in Chinese).

[2] Andrew Lavin, Scott Gray. Fast Algorithms for Convolutional Neural Networks[C]// 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2016.

[3] Kiryati N, Eldar Y, Bruckstein A M. A probabilistic Hough transform[J]. Pattern Recognition, 1991, 24(4):303-316.

[4] Duan R J, Zhao W, Huang S L, Chen J Y. Fast line detection algorithm based on improved Hough transformation[J]. Chinese Journal of Scientific Instrument (12):2774-2780. (in Chinese) (12):2774-2780.

[5] Wang Q, Song W D, Wang J X. The Line Extraction Method and Improvement Based on Hough Transform[J]. Chinese Journal of Geomatics and Spatial Information Technology 2019(6) (in Chinese).

[6] Chao L, Zhong W, Lin L. An Improved HT Algorithm on Straight Line Detection Based on Freeman Chain Code[C]// Image and Signal Processing, 2009. CISP '09. 2nd International Congress on. IEEE, 2009.