Low-Computational-Cost Algorithm for Inclination Correction of Independent Handwritten Digits on Microcontrollers

H. Waruna H. Premachandra, Maika Yamada, Chinthaka Premachandra and Hiroharu Kawanaka

Topic
Human–Machine Interaction

Edited by
Prof. Dr. Enrico Vezzetti, Dr. Andrea Luigi Guerra, Dr. Gabriele Baronio, Dr. Domenico Speranza and Dr. Luca Ulrich

https://doi.org/10.3390/electronics11071073
Article

Low-Computational-Cost Algorithm for Inclination Correction of Independent Handwritten Digits on Microcontrollers

H. Waruna H. Premachandra 1, Maika Yamada 2, Chinthaka Premachandra 2,* and Hiroharu Kawanaka 3

1 ICT Center (Information Communication Technology Center), Wayamba University of Sri Lanka, Makandura, Gonawila (NWP) 60170, Sri Lanka; waruna@wyb.ac.lk
2 Department of Electronic Engineering, School of Engineering, Shibaura Institute of Technology, Tokyo 135-8548, Japan; ag17111@shibaura-it.ac.jp
3 Department of Electrical and Electronic Engineering, Mie University, Mie 514-8507, Japan; kawanaka@elec.mie-u.ac.jp
* Correspondence: chintaka@sic.shibaura-it.ac.jp

Abstract: In recent years, the digitization of documents has progressed, and opportunities for handwritten document creation have decreased. However, handwritten notes are still taken for memorizing data, and automated digitalization is needed in some cases, such as making Excel sheets. When digitizing handwritten notes, manual input is required. Therefore, the automatic recognition and input of characters using a character recognition system is useful. However, if the characters are inclined, the recognition rate will be low. Therefore, we focus on the inclination correction problem of characters. The conventional method corrects the inclination and estimates the character line inclination. However, these methods do not work when characters exist in independent positions. Therefore, in this study, we propose a new method for estimating and correcting the tilt of independent handwritten digits by analyzing a circumscribed rectangle and other digital features. The proposed method is not based on an AI-based learning model or a complicated mathematical model. It is developed following a comparatively simple mathematical calculation that can be implemented on a microcontroller. Based on the results of the experiments using digits written in independent positions, the proposed method can correct the inclination with high accuracy. Furthermore, the proposed algorithm is low-computational cost and can be implemented in real-time on a microcontroller.

Keywords: document image processing; character inclination estimation; circumscribed rectangle; digit feature analysis

1. Introduction

Owing to document digitalization, handwritten document usage is decreasing. However, there are many situations in which handwriting is needed. For example, writing down experimental data in notes during scientific experiments using apparatus is still common. Sometimes, automatically converting memorized data to electronic documents, such as Excel sheets, sheets by OCR platforms (ex: Nanonets-OCR platform), is important because manual insertion is time-consuming. The automatic conversion of written documents to electronic documents does not work well when written data are inclined. This inclination correction problem is solved when the written data exists as character lines. However, they do not work well when characters exist as independent characters.

Many studies have been conducted on document image analysis [1–7], including mathematical expression recognition [8], character recognition, etc. Character tilt correction is related to document image analysis. In conventional research, some methods are used to correct the inclination of character lines, such as the inclination correction of the character string using a wavelet transform and statistical-based methods [8–21]. When it is a character string (line) with two or more characters, the slope of the character string is estimated, and the slope is corrected. However, these methods cannot be applied to
independent character tilt correction because the slope cannot be estimated as a character line. Therefore, in this study, we examined the tilt estimation and correction of handwritten numbers because there are many situations where independent digits are written compared to independent character writing (Figure 1). Furthermore, the conventional methods cannot be implemented on small controllers since the algorithms include comparatively complex computing.

Figure 1. Example of a handwritten document.

The purpose of this study is to develop a low-computational cost algorithm to estimate the slope of an independent handwritten number. Furthermore, the algorithm should be implemented on a microcontroller. Focusing on the fact that the number is vertically long, we obtained the minimum external rectangle surrounding the digit structure. Accordingly, the tilt was corrected by rotating the image area of the external rectangle. However, we found that tilt correction is not possible with external rectangles in the case of digit “4”. Thus, we perform separated processing to achieve a tilt correction of digit “4” as described below. The number “4” was identified from other numbers, and the tilt was estimated using only the “4” feature without relying on the circumscribed rectangle. Furthermore, to distinguish between the number “4” and other numbers, we pay attention to the unique characteristics of the number “4” and propose a new identification method. In this work, we try to handle a handwritten digit inclination correction problem only using the image data at hand, without using any datasets for training, since the training stage is not included in the proposal. This time the inclination of the digits is measured by calculating its circumscribing rectangle and analyzing other unique features. The overall process comprises trigonometry calculation and the Hough Transform (HT). The HT is comparatively time-consuming. But, in this study, we have reduced the computation cost of HT by reducing the number of pixels for the HT process, and the proposed algorithm can be implemented on microcontrollers.

Based on the experimental results of the proposed method, the inclination of many numbers can be corrected with good accuracy, and the method is very effective for numbers inclined in a certain range. Furthermore, the proposed method has a short processing time, and it can be implemented in real-time with a small computer, such as a Raspberry Pi. Therefore, the proposed method is of an applicable level.

2. Tilt Estimation and Correction

2.1. Tilt Estimation with the Circumscribed Rectangle

A sample input image is displayed on the left side of Figure 2. As shown in the figure, some digits are inclined at a certain level. In this study, we first estimated the tilt angle and then performed tilt correction following the angle. The main processing flow for tilt estimation and correction is shown in Figure 3.
To determine the circumscribed rectangle, the image was first binarized based on discriminant analysis [22]. This time, digits written on a consistent-color paper were used. The binarization following discriminant analysis can separate the digits and background when the digits are written with a colored pen or pencil. Subsequently, the lines related to the digits were further extracted using the Canny edge detector [23–25]. For character line components, they were sandwiched between two close contours (edges). Through this feature, the lines that correspond only to the digits were determined, and objects other
than the digits were removed. Figure 4 shows an example image obtained following these processes. Subsequently, we calculated the slope of the long side of the circumscribed rectangle. In this work, a vertical circumscribed rectangle was not considered, and the minimum circumscribed rectangle surrounding the digit was estimated, as shown in Figure 5.

![Sample image of digit extraction.](image1)

Figure 4. Sample image of digit extraction.

![Estimation of the minimum circumscribed rectangle.](image2)

Figure 5. Estimation of the minimum circumscribed rectangle.

Figure 6 shows how to find the slope of the long side of the estimated circumscribed rectangle. Because the coordinates of the four corners of the rectangle can be obtained from the circumscribed rectangle, the length was calculated using those coordinates. As shown in Figure 6, with the corner points of the rectangle $(ax, ay), (bx, by)$, the side length of the axis $l$ can be expressed by Equation (1). Consequently, the lengths of the long and short sides of the rectangle are calculated. Then, the long side length is used. In Figure 6, the rectangle $\theta$, which represents the angle of the long side against the horizontal direction, can be expressed by Equation (2). Conversely, $\theta'$, which represents the angle of the long
side against the vertical direction, can be expressed by Equation (3). $\theta'$ becomes the tilt and considered the tilt of the digit.

\[
l = \sqrt{(ax - bx)^2 + (ay - by)^2}
\]  \hspace{1cm} (1)

\[
\theta = \cos^{-1}\left(\frac{bx - ax}{l}\right)
\]  \hspace{1cm} (2)

\[
\theta' = 90 - \cos^{-1}\left(\frac{bx - ax}{l}\right)
\]  \hspace{1cm} (3)

Figure 6. Calculation of the side length of the long axis.

2.2. Separation of “4” and other Digits

Figure 7 shows the minimum circumscribed rectangle surrounding the digit “4”. As shown in Figure 4, the circumscribed rectangle considering the rotation of “4” looks like a rhombus with respect to the number, and the slope of the circumscribed rectangle does not coincide with the slope of the digit. Therefore, we first separated “4” and other digits. In the case of “4”, a special process was performed to estimate the tilt.

Figure 7. Circumscribed rectangle of the digit “4”.
To separate “4” from the other digits, we focus on the line crossing part “+” of the “4”. The status of the perpendicular crossing of two lines is not available in the other digits, and it has a unique feature for “4”. Here, we first estimated the circumscribed rectangle and determined its short length (\(w\)). Then we detect the straight lines within the rectangle. If at least two lines are detected, then we estimated the angles between each two-line pair (\(\varphi_i\)). Here, \(i = 1,2\). In addition, the lengths of the detected lines \(l_j\) were determined. The line pair that fulfills the conditions in Equations (4) and (5) were extracted as the crossing line pair of “4”. Through this idea, we were able to separate “4” from the other digits with a high success rate.

\[
80^\circ < \varphi_i < 100^\circ
\]

\[
l_j \geq w \times 0.8
\]

In the above process, line detection was performed following the Hough transform. The Hough transform is used to detect shapes with mathematical forms, such as lines, circles, and ellipses [26–29]. In this study, we used the Hough transform process for line detection. Generally, a straight line in an \((x, y)\) space can be represented by Equation (6). Here, \(a\) and \(b\) constants. In the Hough transform, if the \((x, y)\) space line is converted to the \((r, \theta)\) space by Equation (7), then the line almost appears as a single dot. As shown in Figure 8, if we calculate all the lines that pass through the single dot in the \((x, y)\) space, they appear as curved lines in the \((r, \theta)\) space, as shown in Figure 9. In the case where the dots are on a straight line in \((x, y)\) space, there is a crossing point for curved lines in the \((r, \theta)\) space (Figure 9). By following these conditions, line detection can be performed [30–32].

\[
y = a_0 x + b_0
\]

\[
\rho = x_i \cos \theta + y_i \sin \theta
\]

![Figure 8. Line in the \((x, y)\) space.](image)
2.3. Tilt Estimation of “4”

The digit “4” structure includes a diagonal line. We used this diagonal line part to estimate the tilt of “4”. As shown in Figure 10, the detected digit “4” is divided into four parts following the perpendicular cross lines. Then, we calculated the angle required for the diagonal line to move to the top-left part (see Figure 10 right). This angle is the tilt of the “4”, considering that “4” is inclined.

In this study, we detected diagonal lines by calculating the white pixels in each of the four parts gained following the cross part of a digit “4” image (Figure 11). The part containing the most white pixels includes the diagonal line. The diagonal line was detected using this feature.
3. Tilt Correction

In the above sections, we mainly discuss the estimation of the tilt of digits that exist in individual locations. After estimating the tilt, correction must be performed. In this study, tilt correction was performed following an affine transformation. Affine transform is a method that can translate images to different sizes and rotate them in different directions [33–35].

In this study, we applied an affine transformation to correct the digits by rotating their image parts. Following the above equation, if we rotate an image setting using (0, 0) as the image original, the above equation becomes Equation (8), where $\theta$ is the rotation angle.

$$
\begin{bmatrix}
X \\
Y \\
1
\end{bmatrix} =
\begin{bmatrix}
\cos \theta & -\sin \theta & 0 \\
\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
1
\end{bmatrix}
$$

Figure 12 shows an example of the tilt correction by the affine transformation. Here, the tilt estimation is performed using the following method.

4. Experimental Evaluations

4.1. Experimental Environment

We prepared document images with 300 handwritten digits ranging from 0 to 9. Most of the digits were written in an inclined manner, not vertically. The images were taken while keeping the camera almost 10–15 cm away from the documents. Here, we kept the camera approximately parallel to the documents when the images were captured. The resolution of the image was 480 × 640 pixels. We conducted experiments to confirm the effectiveness of the proposed digit tilt estimation and correction. Considering the application issues, we conducted experiments on a microcontroller (Raspberry Pi 4 hardware). The execution time per image under the above condition is 176 ms. In addition to that, we picked up some relevant image data from famous digit image datasets [36–38] and conducted experiments.

4.2. Experimental Results

The input and output images of numbers 0–9 are shown in Figures 13–22. Table 1 shows the success rate of the angle corrections for each number. Table 2 shows the success rate of the angle corrections for each number when the tilt angle is less than 30°. Table 3 shows the success rate of distinguishing between “4” and other digits.
Figure 13. Tilt correction examples of “0” following the proposal (left: original, right: tilt correction results).

Figure 14. Tilt correction examples of “1” following the proposal (left: original, right: tilt correction results).

Figure 15. Tilt correction examples of “2” following the proposal (left: original, right: tilt correction results).

Figure 16. Tilt correction examples of “3” following the proposal (left: original, right: tilt correction results).
Figure 17. Tilt correction examples of “4” following the proposal (left: original, right: tilt correction results).

Figure 18. Tilt correction examples of “5” following the proposal (left: original, right: tilt correction results).

Figure 19. Tilt correction examples of “6” following the proposal (left: original, right: tilt correction results).

Figure 20. Tilt correction examples of “7” following the proposal (left: original, right: tilt correction results).
Figure 21. Tilt correction examples of “8” following the proposal (left: original, right: tilt correction results).

Figure 22. Tilt correction examples of “9” following the proposal (left: original, right: tilt correction results).

Table 1. Overall success rate.

<table>
<thead>
<tr>
<th>Number</th>
<th>Success Rate of Inclination Correction (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>96</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>92</td>
</tr>
<tr>
<td>3</td>
<td>91</td>
</tr>
<tr>
<td>4</td>
<td>86</td>
</tr>
<tr>
<td>5</td>
<td>93</td>
</tr>
<tr>
<td>6</td>
<td>91</td>
</tr>
<tr>
<td>7</td>
<td>92</td>
</tr>
<tr>
<td>8</td>
<td>95</td>
</tr>
<tr>
<td>9</td>
<td>90</td>
</tr>
</tbody>
</table>

Table 2. Success rate when the inclined angle is within 30°.

<table>
<thead>
<tr>
<th>Number</th>
<th>Success Rate of Inclination Correction (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>97</td>
</tr>
<tr>
<td>1</td>
<td>99</td>
</tr>
<tr>
<td>2</td>
<td>94</td>
</tr>
<tr>
<td>3</td>
<td>93</td>
</tr>
<tr>
<td>4</td>
<td>90</td>
</tr>
<tr>
<td>5</td>
<td>93</td>
</tr>
<tr>
<td>6</td>
<td>92</td>
</tr>
<tr>
<td>7</td>
<td>93</td>
</tr>
<tr>
<td>8</td>
<td>97</td>
</tr>
<tr>
<td>9</td>
<td>94</td>
</tr>
</tbody>
</table>
Table 3. Classification success rate of $\lceil 4 \rceil$ and other numbers.

<table>
<thead>
<tr>
<th>Number</th>
<th>Success Rate of Classification (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>98</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>95</td>
</tr>
<tr>
<td>3</td>
<td>94</td>
</tr>
<tr>
<td>4</td>
<td>97</td>
</tr>
<tr>
<td>5</td>
<td>93</td>
</tr>
<tr>
<td>6</td>
<td>93</td>
</tr>
<tr>
<td>7</td>
<td>93</td>
</tr>
<tr>
<td>8</td>
<td>90</td>
</tr>
<tr>
<td>9</td>
<td>91</td>
</tr>
</tbody>
</table>

The angles of the digits other than 4 were successfully corrected with a probability of 90% or more. The cause of the failure to correct the angle was that the circumscribed rectangle considering rotations was not generated to surround the entire number. The reason is that during the binarization process, some pixels belonging to the digit line were detected as background pixels. An example of the failure to generate a circumscribed rectangle is shown in Figure 23. As shown in the figure, if one number is split into several parts and recognized, the processing will fail. The number is split and recognized because the lines of the numbers are thin, and the lines may disappear at the binarization stage. However, if the line is clearly written this error will not occur. Solving this issue by proposing the new binarization approaches or applying the latest methods in the literature is an important future work [39]. In Tables 1 and 2, the slope correction rate is higher regarding many digits when the slope is within approximately $30^\circ$. Therefore, this method is considered to be effective when the angle of the number is within $30^\circ$.

Figure 23. Example of an error in the circumscribed rectangle generation.

Table 4 shows the success rate of the tilt correction. Based on the findings, the success rate is very high when the numbers are neatly binarized.

Table 4. Success rate when the digits and background are clearly separated in the binarized process.

<table>
<thead>
<tr>
<th>Number</th>
<th>Success Rate of Inclination Correction (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>99</td>
</tr>
<tr>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>2</td>
<td>97</td>
</tr>
<tr>
<td>3</td>
<td>96</td>
</tr>
<tr>
<td>4</td>
<td>92</td>
</tr>
<tr>
<td>5</td>
<td>95</td>
</tr>
<tr>
<td>6</td>
<td>96</td>
</tr>
<tr>
<td>7</td>
<td>96</td>
</tr>
<tr>
<td>8</td>
<td>98</td>
</tr>
<tr>
<td>9</td>
<td>95</td>
</tr>
</tbody>
</table>
Future prospects include handling the thin lines of numbers, increasing the accuracy rate of circumscribed rectangle acquisition and “4” recognition processing, and handling arbitrary input images. To increase the accuracy of circumscribed rectangle acquisition, other processes can be added, such as increasing the contrast. In this study, we used an input image with less noise and less distortion. Therefore, to support an arbitrary input image, it is necessary to remove noise and correct image distortion.

5. Conclusions

In this study, we propose a method that estimates and corrects the slope of independent numbers using a circumscribed rectangle and other specific digit features. We corrected the slope of the digits when the circumscribed rectangle on the slope could be obtained correctly. In the case of the digit “4”, the tilt angle cannot be easily estimated with the circumscribed rectangle, but it could be corrected using the positional relationship between the cross part of “4” and the diagonal line portion. We conducted experiments to confirm the proposed method by preparing appropriate document images with inclined digits. The results revealed that the proposed method achieved a success rate of approximately 90% or more. The circumscribed rectangle was not estimated correctly, mainly due to the disappearance of the digit line parts at the binarization stage. Improving this issue will be one of the main future stages of this work. In addition, the proposed method has a short processing time, and it can be implemented in real-time with a small computer, such as a Raspberry Pi. Therefore, the proposed method is at an applicable level.

Author Contributions: Conceptualization, H.W.H.P. and H.K.; methodology, H.W.H.P. and M.Y.; software, H.W.H.P. and M.Y.; investigation, C.P.; resources, H.W.H.P.; data curation, H.W.H.P.; writing—original draft preparation, H.W.H.P.; writing—review and editing, H.K. and C.P.; supervision, H.K.; project administration, C.P.; funding acquisition, C.P. All authors have read and agreed to the published version of the manuscript.

Funding: Shibaura Institute of Technology.

Conflicts of Interest: The authors declare no conflict of interest.

References


