Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 313452, 10 pages
http://dx.doi.org/10.1155/2014/313452

Research Article

An Implementation of Document Image Reconstruction System on a Smart Device Using a 1D Histogram Calibration Algorithm

Lifeng Zhang,[1] Qian Fan,[1] Yujie Li,[1] Yousuke Uchimura,[2] and Seiichi Serikawa[1]

[1] *Kyushu Institute of Technology, Kitakyushu 804-8550, Japan*
[2] *Tokyo Electron Kyushu Limited, Kumamoto 861-1116, Japan*

Correspondence should be addressed to Lifeng Zhang; zhang@elcs.kyutech.ac.jp

Received 24 February 2014; Accepted 4 April 2014; Published 12 June 2014

Academic Editor: Her-Terng Yau

Copyright © 2014 Lifeng Zhang 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.

In recent years, the smart devices equipped with imaging functions are widely spreading for consumer application. It is very convenient for people to record information using these devices. For example, people can photo one page of a book in a library or they can capture an interesting piece of news on the bulletin board when walking on the street. But sometimes, one shot full area image cannot give a sufficient resolution for OCR soft or for human visual recognition. Therefore, people would prefer to take several partial character images of a readable size and then stitch them together in an efficient way. In this study, we propose a print document acquisition method using a device with a video camera. A one-dimensional histogram based self-calibration algorithm is developed for calibration. Because the calculation cost is low, it can be installed on a smartphone. The simulation result shows that the calibration and stitching are well performed.

1. Introduction

In recent years, there is a growing trend for document digitization in order to reduce the cost of storage and administration of printed documents. The traditional method is using a digital still camera or a scanner. However, the scanner does not work in some situations like a wet surface or a bended surface. Moreover, fragile historical manuscripts cannot be acquired by a scanner. In a noncontact scanning device, a digital camera is used. Liang et al. proposed an approach estimating 3D document shape from texture flow [1, 2]. Stamatopoulos et al. presented a goal-oriented rectification methodology to compensate for undesirable document image distortions in order to improve the OCR result [3]. But their purpose for the processed targets is full page image. In this study, a low profile device is used. When taking a picture of an entire page area, the characters in the image tend to blur due to the limited resolution. In order to take photos for characters with a sufficient resolution, we take photos for several subareas for the whole image, respectively, and then stitch these images to get the whole document.

When stitching natural images, a slight misalignment at the joint area of the reconstructed images cannot be detected clearly by human eyes. Somol and Haindl proposed a fast and adjustable suboptimal path search algorithm for finding minimum error boundaries between overlapping images [4]. It is a very attractive way for normalized nature images and patterned images. But for the character images, especially for the partially acquired images in which the camera state parameters are different, even a slight deviation at the joint part can be spotted immediately [5, 6]. This makes the produced documents hard to read and difficult to be recognized by an OCR software. To reduce such undesirable effect, each video frame is modified independently by estimating the camera state after the image acquisition so that the position matching can be performed accurately. Here we introduce a new approach for this purpose.

When people go outside, nobody would always take a camera or scanner with them. But nowadays, most people have a smart device in which a builtin web camera is equipped. How to take the advantage of such a device is the biggest challenge of this research.

2. System Outline

The proposed system includes image acquisition, self-calibration, joint point detection, and image synthesis. The details are as follows.

2.1. Image Acquisition. In this research, a web camera is used for image acquisition. That means a huge data volume needs be deposited if we save all the data streams coming from the camera. Furthermore processing all of these data needs a lot of calculation cost. Because we assume that the proposed algorithm should work on a smartphone and should be used everywhere, such a heavy load might become the biggest weakness of our system. Therefore, an autoimage acquisition approach is proposed in this study. Because of the redundancy of each video frame, the neighboring two frames contain almost the same information. That is to say, every two frames with a half overlapped area are enough to fulfill our purpose. We move the web camera by hand in a zigzag line scan manner. When the distance from the previous acquired image is larger than the half of the picture size, the next picture is acquired automatically. Figure 1 shows the flow chart of the image acquisition. In this work, optical flow is utilized for moving distance detection. In order to determine the flow instantly, a low computational cost method is employed. Because the printed document shows fewer changes in brightness value, we decided to use the gradient method called "Lucas-Kanade" which contributed in OpenCV library.

The details are as follows. If the image size is $N \times M$ pixels, then the amount of movement is $N/2$ or $M/2$, respectively. That means if the amount of movement along the X or Y-axis is x or y, respectively, the next image will be acquired when $|x| \leq N/2$ or $|y| \leq M/2$.

2.2. Calibration. For a handheld image acquisition system, the lighting condition, the imaging distance, and the imaging angle are changing frequently. Moreover the vibrations and view angle also influence the image quality. But such conditions cannot be improved perfectly by the users. Thus, we proposed a calibration method to deal with this problem.

In our previous work [7], a self-calibration algorithm was introduced. We just show the outline of this algorithm and the result of each step of simulation in this study. Please go for details in our published paper.

Distortion of the acquired image is assumed to occur due to the slope of the camera, which was separated into three different angles. Pitch angle, roll angle, and yaw angle are rotation angles of X-axis, Y-axis, and Z-axis, respectively (see Figure 2). In this work, affine transformation and projective transformation are used to correct these angles.

Because the camera state is unknown at first, we cannot get all of the parameters that should be set for a projective transformation at one time. The parameters are detected for each angle, respectively, and the modification is performed step by step. The procedure is as follows:

(a) image binarization.

(b) modification of yaw angle,

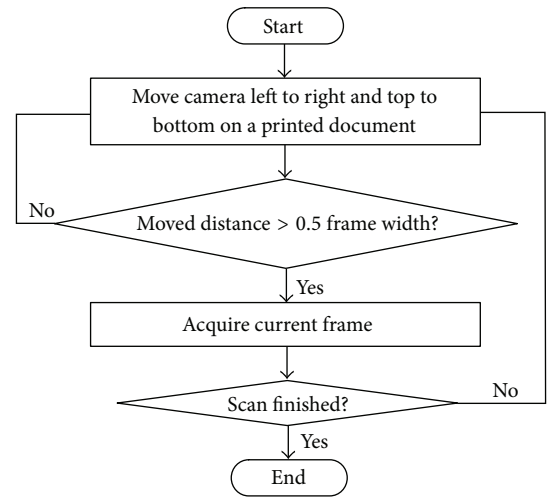


Figure 1: Flow chart for image acquisitions.

(c) modification of roll angle,

(d) modification of pitch angle.

The details of these steps are as follows.

2.2.1. Binarization. Normally, the unique threshold cannot be determined for the binarization process because the text color and the brightness are changing in the ambient lighting environment. In this study, we provide an adaptive threshold [8] for each local area in the image for binarization processing. Consider

$$B(i,j) = \begin{cases} 1 & \text{if } m_{W\times W}(i,j) < l(i,j) * \text{bias} \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where $m_{W\times W}(i,j)$ is the local mean over a $w = 10$ pixel-sized window and bias $= 52$. Figure 3 shows one processing example.

2.2.2. One-Dimensional Histogram. In this study we use a one-dimensional histogram to detect the correction parameters. Here is the overview.

For a binarized character image, with consideration of the number of black pixels on the horizontal line, we get the one-dimensional histogram as shown in Figure 4.

In Figure 4(a), the histogram appears regularly like a bar chart if the characters are laid in a horizontal direction. Suppose the number of the bar is K and the most frequently appearing pixel number in each bar area is $n_{a-\max}$ ($0 < a < K$; a : integer). In order to get the bar width, we find out two y positions R_a and L_a and when the pixel number is less than $n_{a-\max}/4$ on both sides of each bar, the width $W_a = R_a - L_a$ of each bar (character line width) can be calculated, respectively, and the blank width between the two neighboring character lines can be calculated as $I_b = L_{a+1} - R_a$ ($0 \leq a \leq K-1$; b : integer). These parameters are used to modify the pitch angle described in following process.

Generally, the obtained image has a distortion in some degree. The one-dimensional histogram rarely appears the

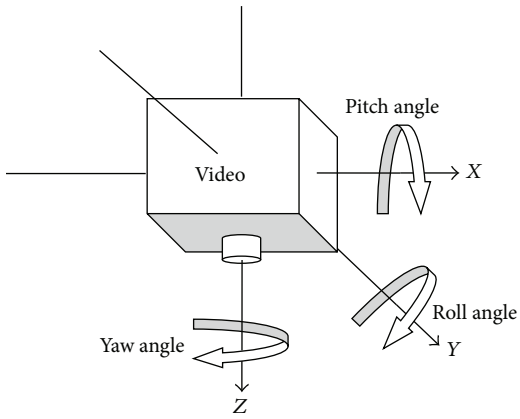
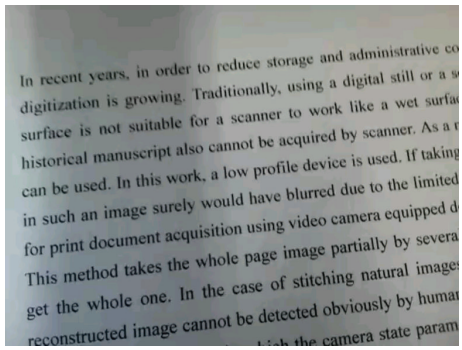
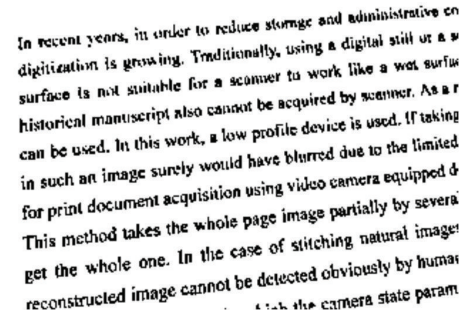


Figure 2: Definition of pitch, roll, and yaw angle for camera state estimation.



(a) Original image



(b) Binarized image

Figure 3: Local area image binarization.

same as Figure 4(a). In most cases the bar blurs as in Figure 4(b). This confused us in determining the width of the character line and the distance between the two neighbor lines. In order to detect the parameters in this case, we developed a local histogram method as illustrated in Figure 5. In this study, we set three strap areas on the image in advance and then make one-dimensional histogram of them. In Figure 5(b) the resulting histograms reflect the condition of the character line's distortion even if the overall character line is not horizontal. The modification parameters can be derived from such information. The details will be described in the following process.

2.2.3. Yaw Angle Modification. In this research, all of the calibration parameters are derived from one-dimensional histogram. Such histograms were made from not only the overall image but also the local areas of an image. Each gravity point shown in Figure 6(a) is calculated from its neighboring local area histograms. By calculating the line directions and the disappearing point from these gravity points, the yaw angle is modified as in Figure 6(c).

The calculation of the gravity point of each gathered histogram is shown in Figure 5(b). The gravity point g_{C_k} is calculated by

$$g_{C_k} = \frac{\sum_{y=l_{ck}}^{r_{ck}} B(y) \cdot y}{\sum_{y=l_{ck}}^{r_{ck}} B(y)}, \tag{2}$$

where k is the gathered histogram number, $B(y)$ is the pixel number when the coordinate of each strap is $y \cdot l_{ck}$, and r_{ck} is the boundary coordinate of each gathered histogram. The resulting image is shown in Figure 6. The average θ can be calculated from θ_n.

In this study, the average rotation angle $\theta = (1/2K) \sum_{n=0}^{2K} \theta_n$ is obtained as the yaw angle. The rotation is made by using an affine transform shown as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix},$$
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{3}$$

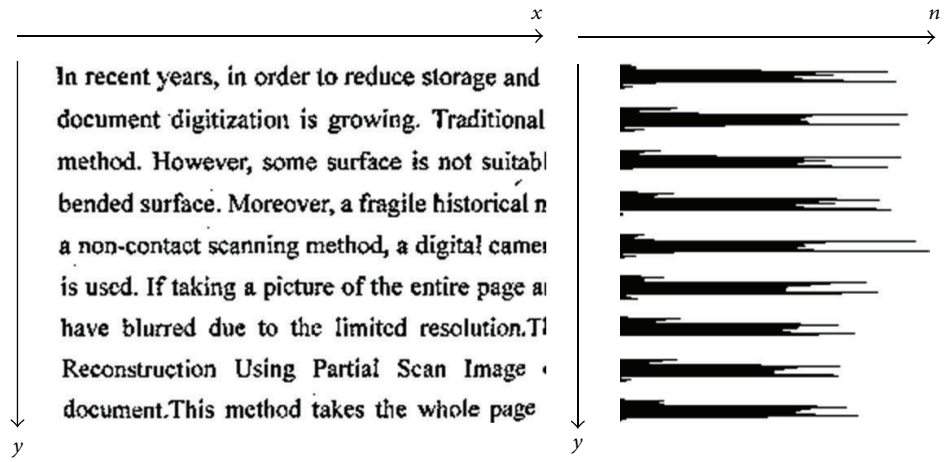
where θ is the rotation parameter and T_x and T_y are the shift parameters of the X-axis and Y-axis direction, respectively. By adopting the nearest neighboring interpolation and a rapid calculation method to give the absent pixels, we got the yaw angle calibration image shown as Figure 6(c).

2.2.4. Roll Angle Modification. After yaw angle modification, we perform local histogram processing again to the previous result again, but this time we set two straps on the image automatically. And finally we find four gravity points on two separated character lines. For an image without roll angle distortion, these four points must be the vertex of a rectangle. Using this characteristic, the roll angle can be modified by using projective transformation.

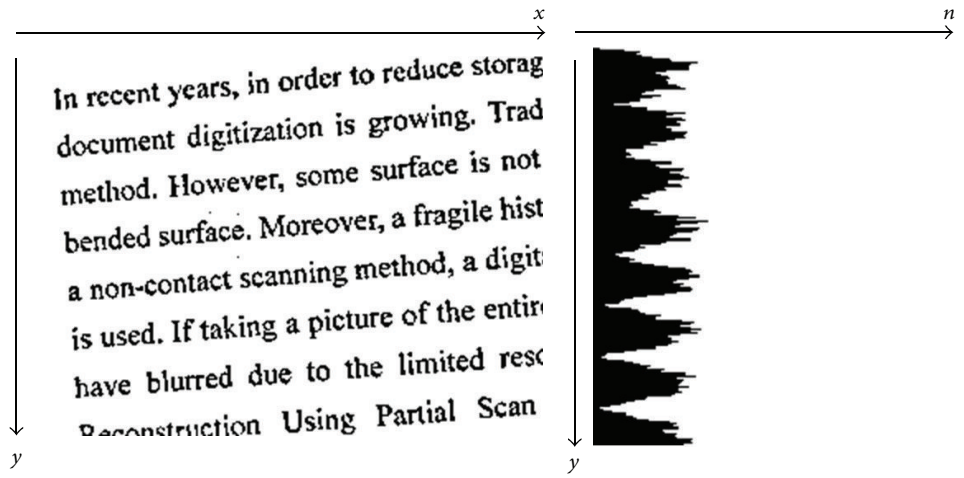
Generally, the projective transformation can be expressed as follows:

$$X = \frac{a_1 x + b_1 y + c_1}{A_0 x + b_0 y + c_0},$$
$$Y = \frac{a_2 x + b_2 y + c_2}{A_0 x + b_0 y + c_0}, \tag{4}$$

where X, Y are transformed coordinates and x, y are pre-transformed coordinates. And a_0, b_0, c_0, a_1, b_1, c_1, a_2, b_2, and c_2 are transformation parameters. If these transformation parameters are known, the projection can be made easily. But,

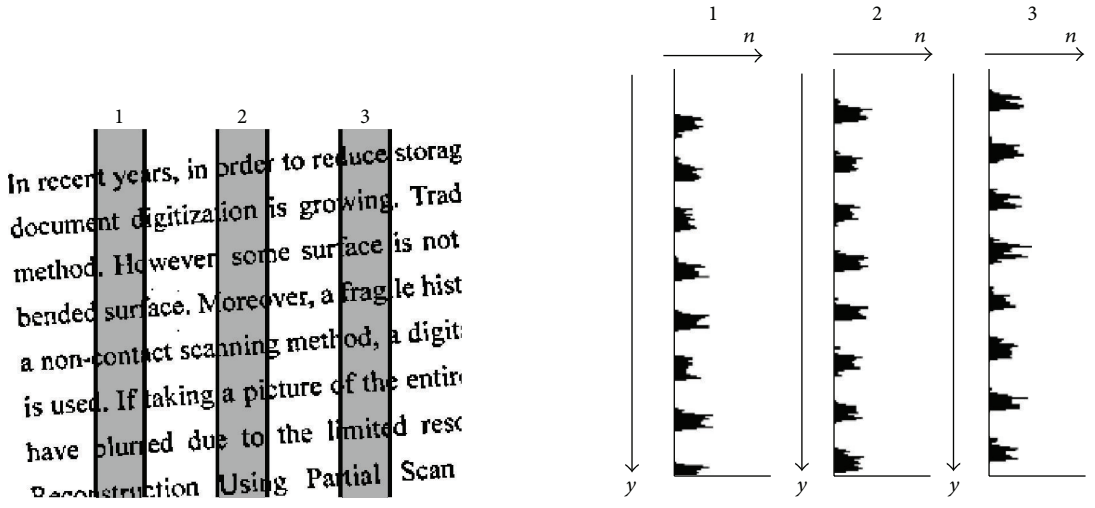


(a) Histogram without distortion



(b) Histogram with distortion

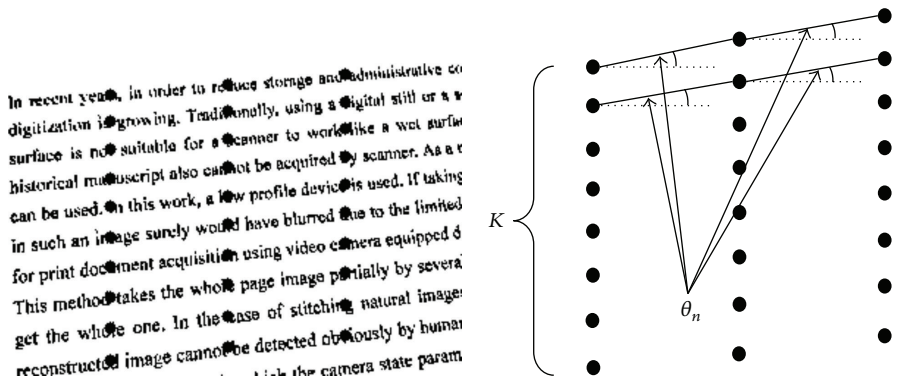
Figure 4: Comparison of one-dimensional histogram on horizontal direction.



(a) Select strap area on image

(b) 1D histogram of each strap

Figure 5: Local area 1D histogram detection.



(a) Gravity point detection result (b) Rotation angle θ_n detection

(c) Yaw angle modified result

Figure 6: Yaw angle modification procedure.

at this stage, we need to figure out these unknown parameters first based on the coordinates of G_1, G_2, G_3, and G_4 which were obtained previously and the reasonable relationship of these points.

As shown in (4), there are nine transformation parameters which are unknown. The known information is four detected gravity point coordinates and their expected positions. In fact, we can reduce one parameter by dividing c_0 into both the numerator and the denominator of (4). The alternate form is shown as in

$$X = \frac{a_1 x + b_1 y + c_1}{A_0 x + b_0 y + 1},$$
$$Y = \frac{a_2 x + b_2 y + c_2}{A_0 x + b_0 y + 1}. \tag{5}$$

The number of the unknown transform parameters is reduced to eight. Therefore, we can make eight equations as follows:

$$A \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \\ a_3 \\ b_3 \end{bmatrix} = \begin{bmatrix} X_1 \\ Y_1 \\ X_2 \\ Y_2 \\ X_3 \\ Y_3 \\ X_4 \\ Y_4 \end{bmatrix}, \tag{6}$$

where

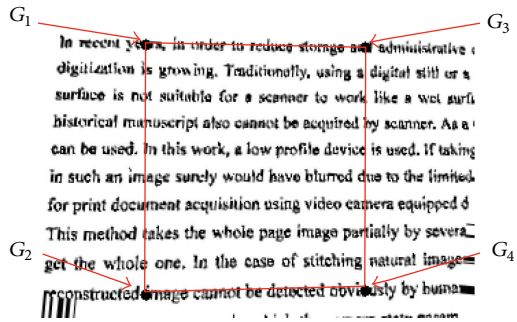
$$A = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -X_1 x_1 & -X_1 y_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -Y_1 x_1 & -Y_1 y_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -X_2 x_2 & -X_2 y_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -Y_2 x_2 & -Y_2 y_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -X_3 x_3 & -X_3 y_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -Y_3 x_3 & -Y_3 y_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -X_4 x_4 & -X_4 y_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -Y_4 x_4 & -Y_4 y_4 \end{bmatrix}, \tag{7}$$

and then calculate the inverse matrix A^{-1}; the eight unknown transform parameters can be obtained as

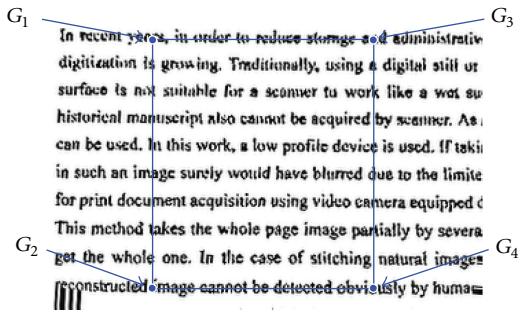
$$\begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ a_2 \\ b_2 \\ c_2 \\ a_3 \\ b_3 \end{bmatrix} = A^{-1} \begin{bmatrix} X_1 \\ Y_1 \\ X_2 \\ Y_2 \\ X_3 \\ Y_3 \\ X_4 \\ Y_4 \end{bmatrix}, \tag{8}$$

and finally the transform can be performed by using (5). The result is shown in Figure 7(b). Tables 1 and 2 show the gravity points coordinate before and after the roll angle calibration.

2.2.5. Pitch Angle Modification. Pitch angle modification parameter is estimated from an overall one-dimensional histogram that was created based on the above steps. Different



(a) Gravity point detection result



(b) Gravity point after modification

Figure 7: Roll angle modification result.

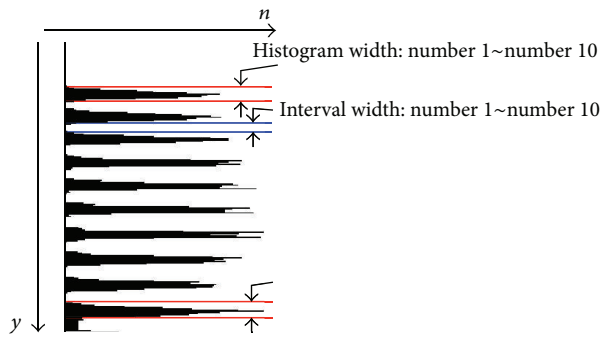


Figure 8: Calculate the line width and interval width.

Table 1: Gravity points coordinate.

Gravity point	x	y
G_1	80	42
G_2	80	224
G_3	240	43
G_4	240	220

from the previous two steps, the modification parameter cannot be derived from the gravity point. Because no difference can be obtained from the gravity point between the expected final result and the processing result at this moment, we have to estimate the modification parameters through the slight variation of the width of the horizontal direction's histogram bar and the width of blank interval shown as Figure 8. Figure 9 is a binarized image for making histogram shown

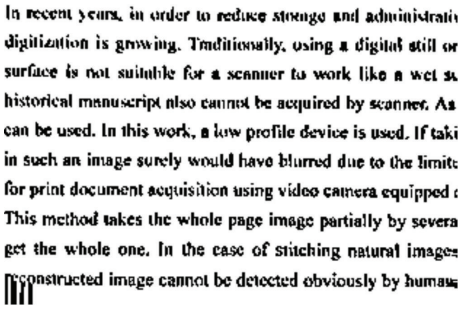


Figure 9: Pitch angle modification result.

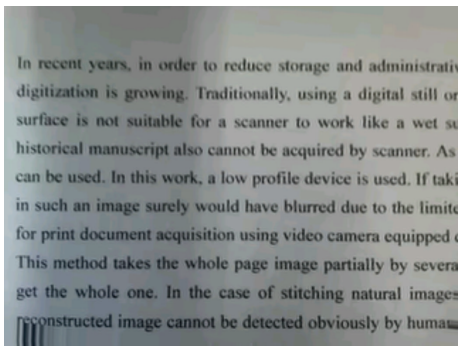


Figure 10: Final calibration modification result.

Table 2: Calibrated gravity points coordinate.

Gravity point	x	y
G_1	80	42
G_2	80	224
G_3	240	42
G_4	240	224

Table 3: Width of each histogram bar and interval.

Index	Bar width [pixels]	Interval width [pixels]
1	8	10
2	9	10
3	9	11
4	8	11
5	9	12
6	8	12
7	9	12
8	9	12
9	9	13
10	10	—

as Figure 8 to acquire the modification parameters. Table 3 shows a measurement result of horizontal 1D histogram bar width and an interval width of each character line. Pitch angle modification parameter detection is an irritation procedure; thus Figure 10 seems to blur because of the cumulative error. But once the parameter is decided, the resulting image can

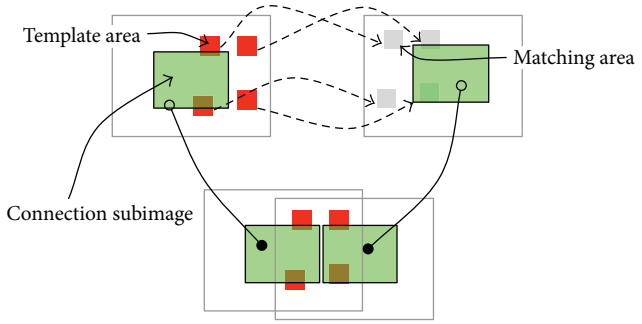


Figure 11: Illustration of image connection.

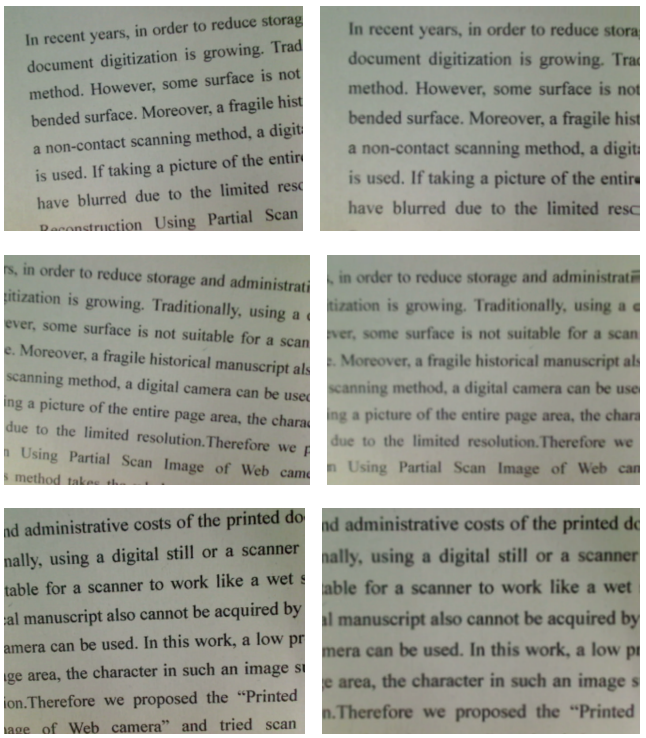


Figure 12: Calibration results. Left: original images and right: calibrated images.

be calculated just one time and no cumulative error occurs in the final calibration image (see Figure 10).

Here we use the coordinates of the gravity points shown in Figure 7(b) as the initial reference points, where the coordinates are $(p1_x, p1_y)$, $(p2_x, p2_y)$, $(p3_x, p3_y)$, and $(p4_x, p4_y)$ and the estimation coordinates of the modified points are $(p1'_x, p1'_y)$, $(p2'_x, p2'_y)$, $(p3'_x, p3'_y)$, and $(p4'_x, p4'_y)$, respectively. The width of histogram bar W_i, the width of blank interval S_j, Num_W, and Num_S are numbers of histogram bar and blank interval, respectively. Because we use an iterative algorithm for parameter estimation, first we define an evaluation function as

$$E = \frac{\mathrm{comp}_S - \mathrm{center}_S}{2} + \frac{\mathrm{comp}_W - \mathrm{center}_W}{2}, \tag{9}$$

where

$$\begin{aligned}
\mathrm{center}_W &= \frac{\sum_{i=1}^{\mathrm{Num}_W} W_i \times i}{\sum_{i=1}^{\mathrm{Num}_W} W_i},\\
\mathrm{center}_S &= \frac{\sum_{j=1}^{\mathrm{Num}_S} S_j \times j}{\sum_{j=1}^{\mathrm{Num}_S} S_j},\\
\mathrm{comp}_W &= \frac{\sum_{i=1}^{\mathrm{Num}_W} i}{\mathrm{Num}_W},\\
\mathrm{comp}_S &= \frac{\sum_{j=1}^{\mathrm{Num}_S} j}{\mathrm{Num}_S}.
\end{aligned} \tag{10}$$

If $\mathrm{comp}_W < \mathrm{center}_W \cap \mathrm{comp}_S < \mathrm{center}_S$,

$$\begin{aligned}
p1'_x &= p1_x + \frac{F_W + F_S}{2}, \quad & p1'_y &= p1_y,\\
p2'_x &= p2_x, & p2'_y &= p2_y,\\
p3'_x &= p3_x - \frac{F_W + F_S}{2}, & p3'_y &= p3_y,\\
p4'_x &= p4_x, & p4'_y &= p4_y.
\end{aligned} \tag{11}$$

Otherwise,

$$\begin{aligned}
p1'_x &= p1_x, \quad & p1'_y &= p1_y,\\
p2'_x &= p2_x - \frac{F_W + F_S}{2}, & p2'_y &= p2_y,\\
p3'_x &= p3_x, & p3'_y &= p3_y\\
p4'_x &= p4_x + \frac{F_W + F_S}{2}, & p4'_y &= p4_y,
\end{aligned} \tag{12}$$

where $F_W = (\mathrm{comp}_W - \mathrm{center}_W) \times (\mathrm{comp}_W)^2$ and $F_S = (\mathrm{comp}_S - \mathrm{center}_S) \times (\mathrm{comp}_S)^2$.

Perform the projective transformation iteratively using the estimated coordinates, and then evaluate the accuracy of the correction based on the evaluation function E. Terminate the iteration when $E \le T$ (T is threshold). Otherwise the algorithm remakes a one-dimensional histogram and repeats this procedure again. Finally, we get the completely modified result.

In fact, the size normalization is also achieved at this step. Because the width of the histogram bar and the interval is calculated, we can use such figure as a universal parameter for the normalization procedure.

2.3. Image Synthesis. After calibrating all the acquired partial images of a character document, discovering the same area of the neighborhood images becomes the important task for image synthesis. In this study, we set four template subimages on a "base image to connect" and search the same local

Table 4: Experiment conditions.

Camera imaging size	5.11 M pixels
Video size	640 × 480
Video codec	AVC/H.264
Frame rate	29.63 fps

Table 5: Tolerance angle.

	Range [deg.]
Yaw angle	−35° ~35°
Row angle	−17° ~17°
Pitch angle	−15° ~15°

Table 6: Processing time.

Procedure	Processing time [sec]
Video frame selection	146
Calibration	97
Image synthesis	22
Total	**265**

position on "the image to be connected." Then, we calculate the "connecting coordinate" based on the resulting positions. Figure 11 demonstrates the image connection. Because the surrounding areas are noisy and the neighbored images have an almost half overlapped area, only the center part of the subimages can be used for image synthesis.

3. Simulation

3.1. Comparison Experiment. Because this research is designed for smart device using, after developing the algorithm on a PC system, implementation for Android device is also needed.

The PC system construction is ACER notebook which has a core i5 1.6 GHz processor, 4 GB memory, and a web camera (Logicool Webcam Pro 9000). The development tool is Microsoft Visual Studio 2010 professional edition, and OpenCV1.1 library was included.

The Android tablet is SONY SGPT112JP/S which has a NVIDIA Tegra 2 mobile processor, 1 GB main memory, and a mobile CMOS sensor HD camera; the display is 9.4 inch WXGA. The development tool is Eclipse with Android SDK, also OpenCV 2.3.4 for Android was included. Table 4 shows other specifications.

In order to evaluate the performance of these two systems, the same images are used which are captured by a web camera (Logicool Webcam Pro 9000; frame size: 640 × 480). The left side of Figure 12 is the original images and the right side of Figure 12 is PC system calibrated images, respectively. The processed image is initialized to the same character scale and trimmed off the surrounding area which includes pseudo information. Therefore, the size is different from the original one. Figure 13 shows the connection result of the PC system. The processing time is within 5 seconds.

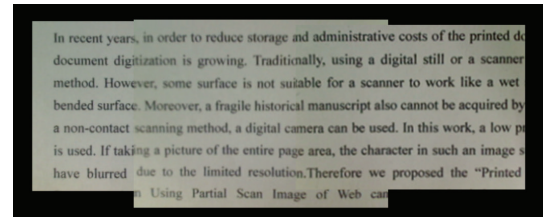


Figure 13: Connection result on PC with Windows 7.

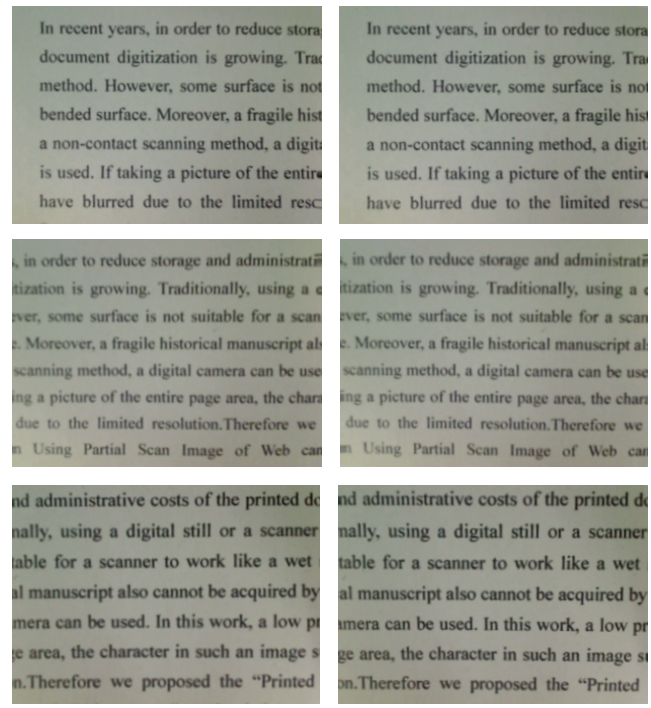


Figure 14: Comparison of calibration results processed by Windows 7 and Android 4.1 tablet. Left: processed by Windows 7 and right: processed by Android 4.1 tablet.

Figure 14 shows the comparison of the image calibration results between the PC system and the Android tablet. The size of the left images is different from the right side. This means that the result of these two systems is somehow different. We made investigation about this problem and realized that the difference is mainly caused by the calculation accuracy and the version difference of OpenCV library. Figure 15 shows the connection result performed on Android tablet. The processing time is about 30 seconds.

Table 5 shows the tolerant angle of each direction defined in Figure 2. Ordinarily, people can take a video with a portable device within this range easily. Although it is possible to extend the tolerant angles using an additional algorithm, it will cost more calculation time. Such tolerant angles are considered enough in this study.

3.2. Overall Experiment on Android Pad. In order to test the performance of implementation on a smart device, an overall experiment has been performed.

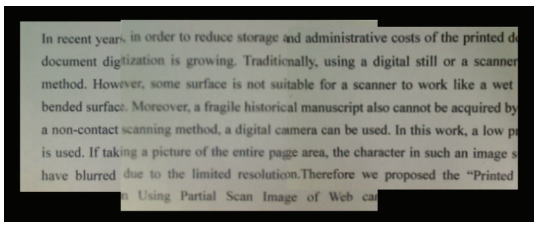


Figure 15: Connection result by Android 4.1 tablet.

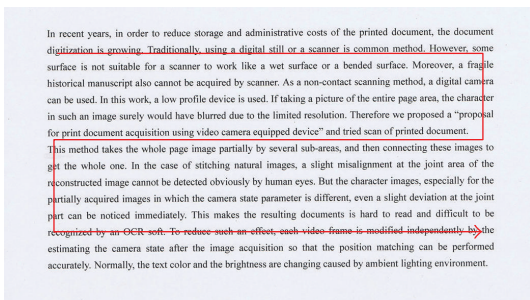
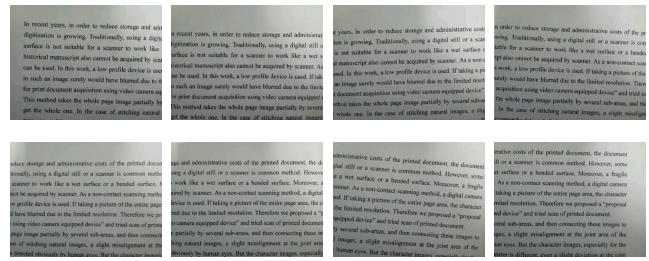
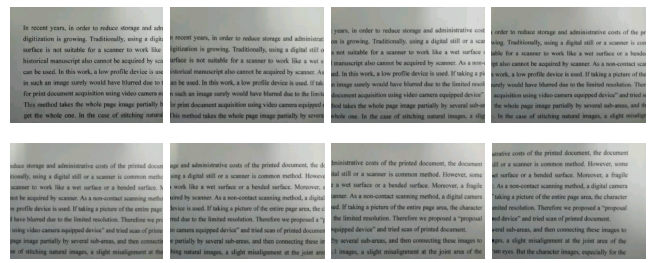


Figure 16: Original printed document.



(a) Autocaptured frames



(b) Autocalibrated images

Figure 17: Calibration result examples.

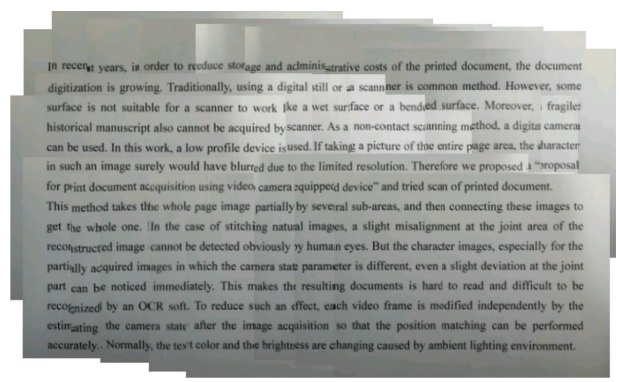


Figure 18: Completely reconstructed document image.

The image processing on a portable device is a time-consuming process. Therefore, we divide the whole process into two parts. One is for video acquisition and the other is for image synthesis. The base function of smart device was used for image acquisition and the video file is stored on the flush memory of the device. Image synthesis is performed using all of the procedure introduced in this work.

Approximately a 10-second video was taken with a target as shown in Figure 16. 23 photo frames were selected automatically. Figure 17 shows a part of the autocaptured frames and their autocalibration results. Figure 18 shows the completely reconstructed document image. Table 6 shows the time consumption.

4. Conclusion

In this work, we implement the self-calibration algorithm for partial character image synthesis on a PC and on an Android tablet separately. The simulation results show that the distortions were well repaired, and the image was reconstructed well on a PC system, but the Android system still shows a low performance because of the lack of calculation ability. For future work, we would improve the image synthesis algorithm to get a more satisfying result and speed up the calculation on a tablet device so that it can better meet the needs for practical application.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is partially supported by NEC Foreign Doctorate Research Grant and Global Security Design, Inc.

References

[1] J. Liang, D. DeMenthon, and D. Doermann, "Geometric rectification of camera-captured document images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 591–605, 2008.

[2] J. Liang, D. Doermann, and H. Li, "Camera-based analysis of text and documents: a survey," *International Journal on Document Analysis and Recognition*, vol. 7, no. 2-3, pp. 84–104, 2005.

[3] N. Stamatopoulos, B. Gatos, I. Pratikakis, and S. J. Perantonis, "Goal-oriented rectification of camera-based document images," *IEEE Transactions on Image Processing*, vol. 20, no. 4, pp. 910–920, 2011.

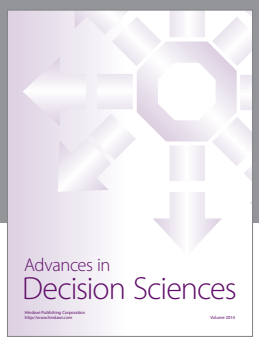
[4] P. Somol and M. Haindl, "Novel path search algorithm for image stitching and advanced texture tiling," in *Proceedings of the 13th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG '05)*, pp. 155–162, February 2005.

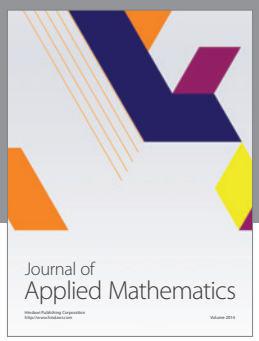
[5] Y. Suematu and H. Yamada, *Image Processing Engineering*, Corona Publishing, 2000.

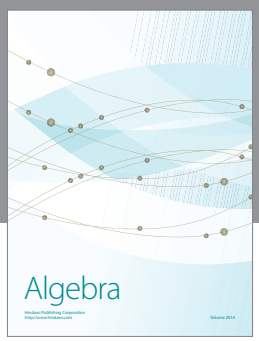
[6] Nara Institute of Science and Technology, *OpenCV Programming Book*, Mynavi Corporation, 2008.

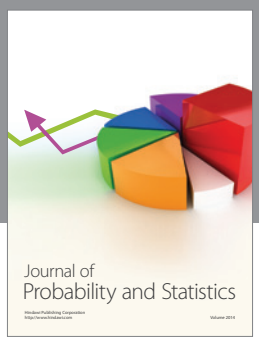
[7] Y. Uchimura, L. Zhang, T. Kouda, H. Lu, and S. Serikawa, "A calibration method for partially acquired print document image by web camera," *ICIC Express Letters*, vol. 7, no. 3, pp. 861–866, 2013.

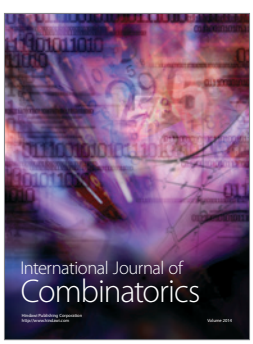
[8] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146–165, 2004.

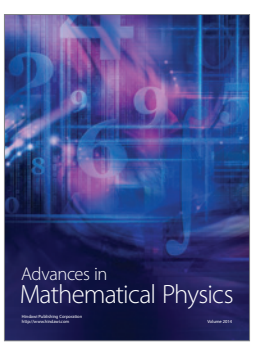


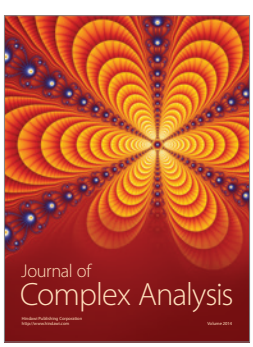


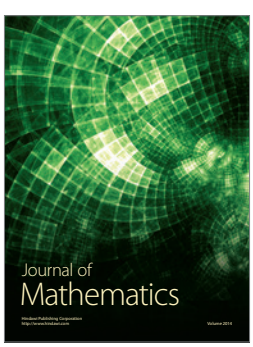


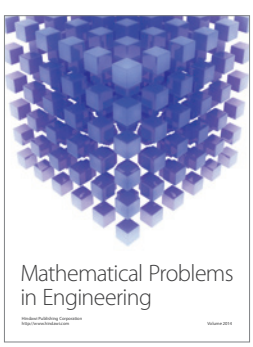


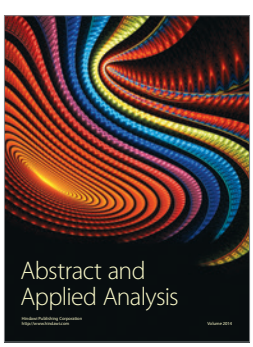


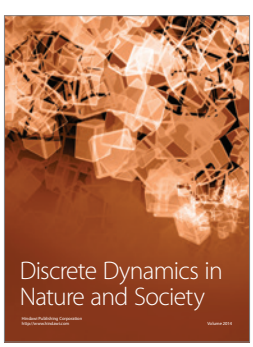


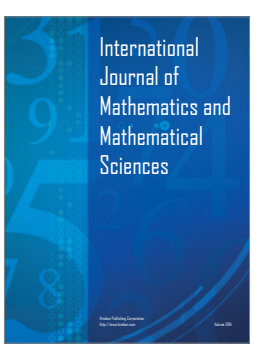


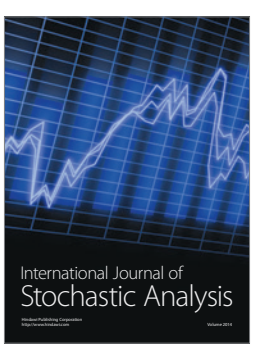












Hindawi

Submit your manuscripts at
http://www.hindawi.com