An Automatic Approach to the Elevator Button Localization and Recognition

Yu-Ching Hsu and Yung-Sheng Chen

Abstract—**With the rapid progress of robotic technology, it conducts that the demand of automatic robot transportation grows immensely. Consider a scenario that a robot can navigate in a building with the ability of taking an elevator, how to identify a button from the elevator panel is a significant step in such a robotic vision application. Therefore, in this paper an automatic approach including several computer vision techniques is presented for the elevator button segmentation and recognition. In addition to our previous projection-and-checking method for preliminary button localization, the methodologies of convolution and template matching are developed for improving the localization performance, the CNN-based method is used to classify the localized buttons into the 'is-button' and 'non-button', and the symbol inside the button is recognized by the structural similarity comparison method for further identifying the localized button. There are 32 different elevator button panels with 328 buttons to be collected for experiments. The results show that 95.8% accuracy can be achieved and thus confirm the feasibility of the proposed approach.**

Index Terms—**computer vision, edge, elevator, image processing, localization, template matching, recognition.**

I. Introduction

WITH the rapid development in the field of computer science and its applications, the demand of automatically robotic transportation has been driven to an immense scale. To deal with such a massive trend, the computer vision based on camera plays a key role in several significant fields, such as automatic traffic surveillance system [1], reading analog multimeter [2], computer-using robot [3], autonomous vehicles [4], unmanned aerial vehicles (or drones) [5], deep learning [6], and so on. Considering the application of robotic transportation in the indoor environment, the robot vision can not only follow the moving path but also should have the ability of vertical moving, i.e., to take the elevator from one floor to the other floor since the buildings, undergrounds as well as skyscrapers with many floors presents more or less in the modern cities. In order to let a robotic system can take the elevator, the recognition of buttons controlling the elevator will be the first step. According to such a research way, in past years we have studied some related topics. For example, the developed CUBot [3] can manipulate the computer mouse to operate the computer like a human being, where the interaction between the operated computer and CUBot is based on the ability of computer vision to perceive the information on the computer screen [7]. In addition, the keyboard recognition is also further studied for possibly being integrated into the CUBot in the near future [8]. Therefore, developing an automatic approach for segmenting and recognizing the elevator buttons from the elevator inside door (EID) image is regarded as the goal of this study.

Manuscript received July xx, 2020; revised July xx, 2020.

Y.-C. Hsu and Y.-S. Chen are with the Department of Electrical Engineering, Yuan Ze University, Taoyuan, Taiwan, 320 ROC. e-mail: eeyschen@saturn.yzu.edu.tw.

Although currently there are few studies discussing the vertical navigation ability for a robot moving in the indoor environment, some methods proposed for the elevator button recognition can be found in literatures, such as the method based on clustering [9], template matching [10] and the neural network based approach [11]. These methods demonstrated the result with a small diversity on the elevator button panels or applied some constraints for specified button panel, e.g., only the US elevator button panel to be applicable [12]. Hence, in order to provide a solution with less limitations and more applicable for the general cases, and enhance the localization and recognition accuracy as well, a set of image processing algorithms will be adopted for designing our approach based on our previous studies [13], [14].

The related works in this study are introduced as follows. Considering the topic of object detection, traditionally the identification and localization are usually based on some image processing algorithms for feature extraction, e.g., thresholding [15], edge detection [16], contour analysis [17], thinning [18], [19], and so on. However, based on such a raw feature, a more complex approach needs to be performed onto them to obtain a more useful information for object identification. Fortunately, due to the much progress of integrated circuits and computer technologies as well as the neural network like Neocognitron developed by K. Fukushima [20], in recent two decades the convolutional neural network [21], [22] proposed by Y. Lecun et al. has pushed the field of object detection and recognition to a new era. It conducts that a human-mind structure makes the object detection no longer struggling in the solution case by case, but a general solution for recognizing the whole information included in an image. However, power as the neural network has its own weakness, such as the tremendous training data required, time-consuming training progress as well as the tedious ground truth labelling task, which make such a technique too heavy and bulky to apply in a small application like the study considered in this paper.

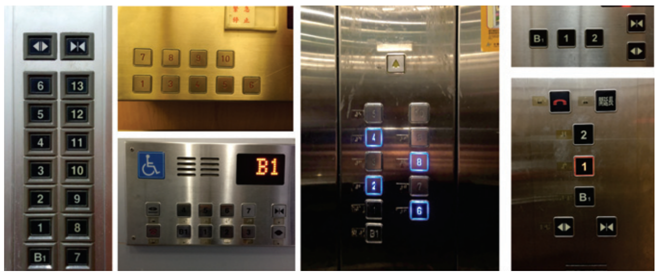


Fig. 1. The different elevator button panels, including buttons made of metal in square, that made of plastic in rectangle and that made of metal with accessible panel in square.

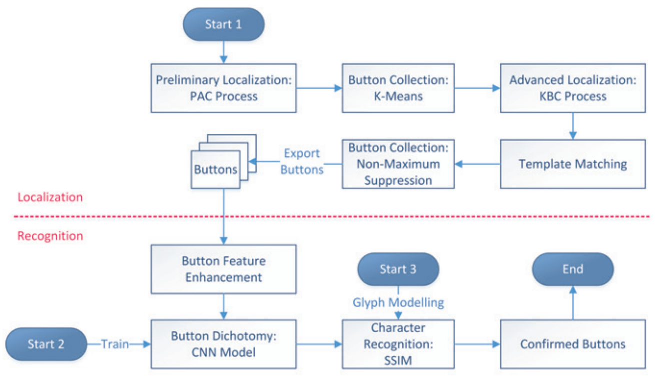

Start 1
Preliminary Localization: PAC Process
Button Collection: K-Means
Advanced Localization: KBC Process
Buttons
Export Buttons
Button Collection: Non-Maximum Suppression
Template Matching
Localization
Recognition
Button Feature Enhancement
Start 3
End
Glyph Modelling
Start 2
Train
Button Dichotomy: CNN Model
Character Recognition: SSIM
Confirmed Buttons

Fig. 2. The flowchart of the proposed approach, which can be divided into two main parts: button localization and character recognition with 3 start lines. Start 1 shows the flow of button localization and recognition, Start 2 prepares the model for button dichotomy, and Start 3 models the glyph table of button symbols.

Fig. 1 shows some examples of elevator button panels arranged in vertical and/or horizontal direction. Our goal is to recognize and locate the buttons just as a human does. Before the study, we had tried some nowadays techniques, such as SIFT [23], SURF [24] and YOLOv3[25] to observe the behaviours of localizing buttons. The results showed a bad accuracy due to only a small amount of data, i.e., 32 different panels with 328 buttons used in this study, to be trained. Moreover, it is expected that the developed system can be applied onto a small computer system (even more the embedded system). The proposed approach is therefore designed by means of a set of image processing techniques as the flowchart depicted in Fig. 2, which includes two parts: button localization and symbol recognition. For the part of button localization, except for the the preliminary localization using the projection-and-checking (PAC) process [13], [14], some other processes (K-Means, kernel-based convolution (KBC), template matching, and non-maximum suppression) are used for further refined the located buttons. For the part of symbol recognition, based on a simple CNN model and a glyph modelling, the symbol of button is recognized for further confirming the effectiveness of localized buttons.

The rest of this paper is organized as follows. In Sec. II, based on the edge information of the EID image, a kernel function called projection-and-checking (PAC) and some refining processes are developed for preliminarily localizing buttons. Since some buttons may be mis-localized or not localized well in the process or preliminary localization, based on the methodologies of convolution and template matching an advanced localization is presented in Sec. III. In order to further enhance the accuracy of wanted buttons, Sec. IV presents a CNN-based method for distinguishing wanted buttons and unwanted buttons; and adopts the structural similarity comparison method to further recognize the symbols inside the buttons. The results and discussions are give in Sec. V. The conclusion of this paper is finally drawn in Sec. VI.

II. Preliminary Localization

Consider an elevator button panel, it is observed that the buttons are usually arranged in vertical and/or horizontal directions. With this characteristic, the following projection-and-checking (PAC) algorithm has been developed for locating the buttons [13], [14]. In order to enhance its feasibility and stability as well as clearly present the proposed approach, the related procedures and illustrations of original PAC will be redescribed in this section.

A. Preparation of Images

By observing an EID-image as shown in Fig. 3(a), the color of the door panel and that of buttons are very similar and not easily distinguished. In addition, there exists a serious reflection over the EID-image. According to our previous study, the edge information provides a useful cue [13], [14] and thus it is needed to be transformed from the original EID-image for the further PAC process, where the Canny edge detection method [16] is adopted.

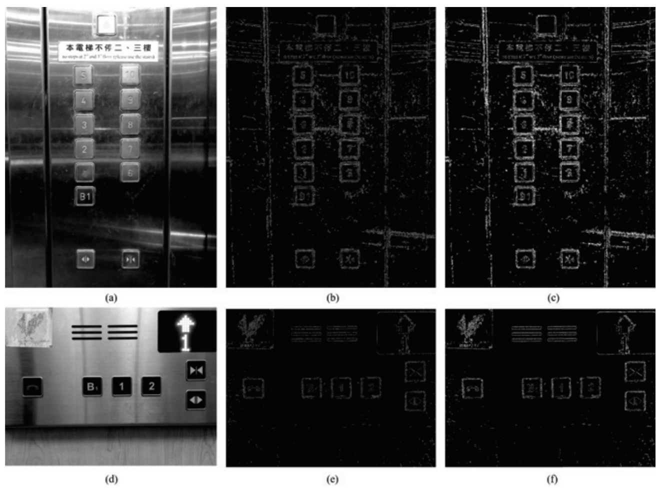


Fig. 3. The example of vertical panel shown in (a) grayscale, (b) edge information (E-image) and (c) the result after morphology (M-image). (d)-(f) show that of horizontal panel with the same procedure.

Fig. 3(a) and 3(d) show the EID-images in grayscale, where Fig. 3(b) and 3(e) are the obtained edges (called as E-image here) after Canny edge detection. From the E-image, it is observed that the edge information are usually intermittent and discontinuous even the wanted structure of a button becomes apparent. This phenomenon may result from different light condition, material made of panel, the uneven reflection, and so on. In order to reduce the noisy pixels in the edge image and make the appearance information more enough, a morphological operation of three times erosions and dilations with 3×3 structure element is applied for the E-image, and the filtered image is thus obtained (called as M-image in this study) as the results of current illustrations shown in Fig. 3(c) and 3(f). Both E-image and M-image are binary images, we let 0 and 1 denote respectively the black and white pixel for the further processing. Note here that the current cases as given in Fig. 3(a) and 3(d) are more complex than those in our previous study [13], [14], where only the vertical or horizontal EID images but not mixed ones were dealt with. Therefore, the step of collecting main blocks in Sec. II-C and that of merging and deleting in Sec. II-D will be modified to improve our preliminary localization of buttons.

B. Preprocessing: PAC Process

Based on the obtained E-image and M-image, the button information could be extracted roughly by the projection-and-checking (PAC) process. This process first collects the main blocks, i.e., button columns, with the vertical projection along x-axis, and then apples the merging-and-deleting process to merge or filter out the trivial blocks. In addition, the collected blocks are confirmed by means of the blocks' area information. Finally, by repeating the similar process with the horizontal projection along y-axis, the whole button coordinates can be obtained as the flowchart depicted in Fig. 4.

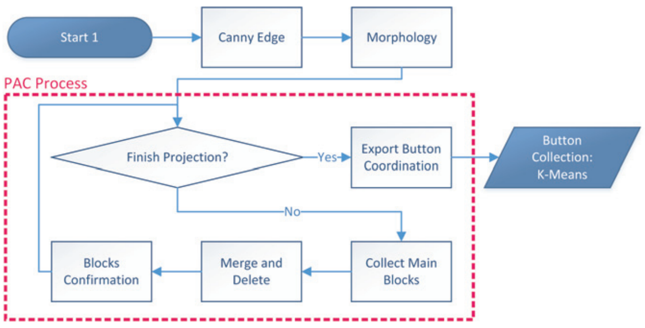


Fig. 4. The flowchart including image preparation and PAC process is designed for preliminarily locating the apparent buttons shown in the EID image.

The basic idea of developing our PAC process is to project the accumulation of 1-pixel in the M-image onto x-axis and y-axis iteratively followed with the checking process to identify a useful range ($R_x = [x_1 : x_2]$, or $R_y = [y_1 : y_2]$), which can be used to form a region-of-interest (ROI) image cropped from the M-image and denoted as $ROI(R_x, R_y) = ROI([x_1 : x_2], [y_1 : y_2])$. For example, in this way, the original M-image can be represented as $ROI([0 : W-1], [0 : H-1])$, where W and H represent the width and height of the M-image. Alternatively, the PAC may be formulated as follows.

PAC: projection-and-checking:

Input an ROI-image.
Output the selected coordination in ROI-image.

1) Given an image with $ROI([x_1 : x_2], [y_1 : y_2])$.
2) Perform projection process:
   a) If vertical projection along x-axis (P_x) is considered, we have the projection array $P_x(i), i = x_1, \ldots, x_2$.
   b) If horizontal projection along y-axis (P_y) is considered, we have the projection array $P_y(i), i = y_1, \ldots, y_2$.
3) Perform checking process:
   a) If checking process (C_x) is performed on the P_x, we obtain N_x wanted ranges, $R_x(i), i = 1, \ldots, N_x$.
   b) If checking process (C_y) is performed on the P_y, we obtain N_y wanted ranges, $R_y(i), i = 1, \ldots, N_y$.

The PAC function can be further denoted as PAC_x and PAC_y, depending on the process performed along the x-axis or y-axis, respectively, In addition, the PAC process completes only if both PAC_x and PAC_y are finished.

C. Collect Main Blocks

By the illustrative example given in Fig. 3(a)-(c), with the PAC result showing in Fig. 5(a), the N_x main blocks is firstly located for obtaining the button columns' range R_x. To filter out the non-button accumulation and noise, as well as preserve the main column information, a threshold TH is needed to be derived. Let the scheme start searching downward from the value of $y = \bar{P}_x$, where $\bar{P}_x$ denotes the mean of $P_x(i), i = 0, ..., W-1$; and stop at y' if $\Delta w > 0.2$, where $\Delta w = \frac{y' - \bar{P}_x}{\bar{P}_x}$. The following threshold TH is experimentally used in this study,

$$TH = \frac{3}{8}\left(\bar{P}_x + \Delta w\right), \text{ if } \Delta w > 0.2, \tag{1}$$

and the thresholded result $\tilde{P}_x$ can thus be obtained by

$$\tilde{P}_x(i) = \begin{cases} 0, & \text{if } P_x(i) < TH, \\ P_x(i), & \text{otherwise.} \end{cases} \tag{2}$$

Following the current illustration, the obtained TH and related parameters are drawn in Fig. 5(b) and 5(d), and the thresholded result $\tilde{P}_x$ is shown in Fig. 5(c).

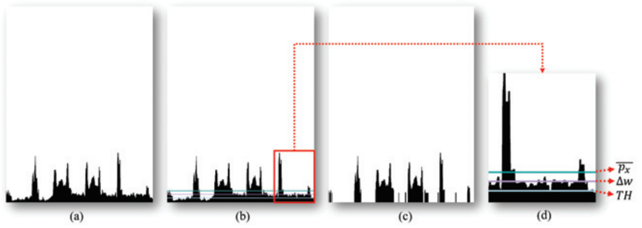


Fig. 5. (a) The projection result P_x in x-axis, (b) the derived TH and related parameters, (c) the thresholded result $\tilde{P}_x$, and (d) the magnified details of $\bar{P}_x$, Δw and the obtained TH for the range highlighted in (b).

Based on the thresholded information as the $\tilde{P}_x$ illustrated in Fig. 5(c), the initially wanted range, such as $[x_1 : x_2]$ (or $[y_1 : y_2]$) can be readily collected first by the following checking mechanism.

$$ROI_k([x_1 : x_2], [y_1 : y_2]) = \begin{cases} x_1, & \text{if } \tilde{P}_x(i-1) = 0 \text{ and } \tilde{P}_x(i) > 0, \\ x_2, & \text{if } \tilde{P}_x(i-1) > 0 \text{ and } \tilde{P}_x(i) = 0, \\ y_1, & \text{if } \tilde{P}_y(i-1) = 0 \text{ and } \tilde{P}_y(i) > 0, \\ y_2, & \text{if } \tilde{P}_y(i-1) > 0 \text{ and } \tilde{P}_y(i) = 0, \end{cases} \tag{3}$$

Following the current illustration of x-axis projection, the initial ranges collected from P_x can be obtained as detected in Fig. 6(a) with the colored blocks denoted by $B_k, k = 0, 1, 2, ..., N_x$, where $N_x = 11$ here. In addition, let the range R_x of B_k be denoted as $[x_1^k : x_2^k]$ for the following presentation. For the situation of y-axis projection, the results can also be obtained by the similar way.

D. Merging and Deleting Operations

In order to locate the effective blocks, some interferences should be removed. Along the current illustration, let W_k and $\overline{W}$ be the width of the block B_k and the mean width for the whole blocks, and be expressed respectively as

$$W_k = |x_2^k - x_1^k|, \tag{4}$$

and

$$\overline{W} = \frac{1}{N_x}\sum_{k=0}^{N_x} W_k. \tag{5}$$

Assume that there exist N blocks whose width $> \overline{W}$, we may collect them and compute their mean width again as below

$$\widehat{W} = \frac{1}{N} \sum_{\forall W_k > \overline{W}} W_k. \quad (6)$$

Then we merge the narrower block to the wider adjacent block if both of their widths are $< 0.8\widehat{W}$ and the distance between them is < 10 as the result shown in Fig. 6(b). Moreover, for each of the N collected blocks, its sum of projections S_k is computed as

$$S_k = \sum_{i=x_1^k}^{x_2^k} P_x(i), \quad (7)$$

and thus we have their mean value $\widehat{S}$ as expressed by

$$\widehat{S} = \frac{1}{N} \sum_{\forall W_k > \overline{W}} S_k. \quad (8)$$

The block satisfying $S_k < \widehat{S}$ will be deleted from the set of collected blocks as the final result shown in Fig. 6(c). Such a filtering process is thus called "merging and deleting" operation.

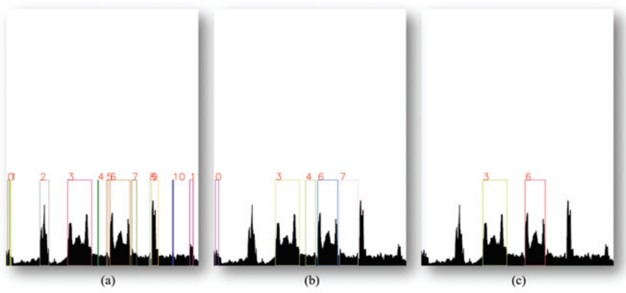


Fig. 6. (a) The collected blocks after checking mechanism, (b) the result of merging and deleting operation, and (c) the final obtained blocks B_3 and B_6.

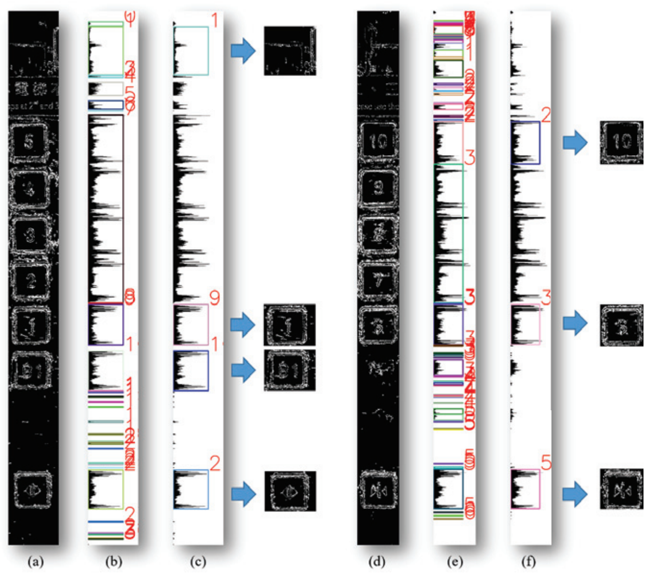


Fig. 7. (a)(d) The cropped image based on the final collected blocks B_3 and B_6 with $y_1 = 0, y_2 = H - 1$. After performing the relative (b)(e) main block collecting process in y-axis and the merging and deleting process, there are (c) 4 and (f) 3 possible buttons to be collected.

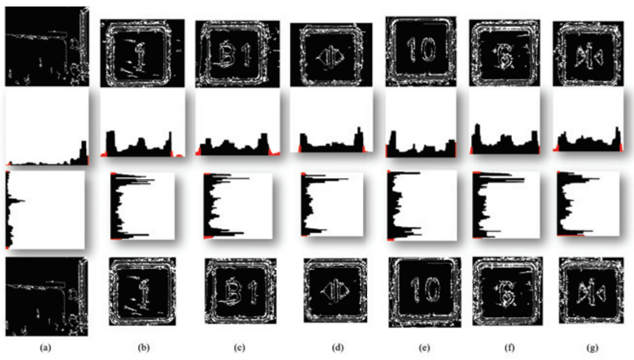


Fig. 8. (a)-(g) illustrate each located buttons with the refining results. The 1st row displays the buttons located previously, where the 2nd row and the 3rd row show the refining regions during the process of PAC_x and PAC_y, respectively. The 4th row displays the finally refining results.

E. Horizontal Analysis

After finishing PAC_x, in current illustration there are two button columns are extracted as shown in Fig. 7(a) and 7(d), respectively. Next with the similar procedure, the PAC_y is applied onto these two button columns for finding the possible buttons as the profiles depicted in Fig. 7(b) and Fig. 7(e). After using the merging and deleting process, seven possible buttons are extracted as shown in Fig. 7(c) and Fig. 7(f), and denoted as $ROI_k, k = 0, ..., 6$.

F. Refining the Located Buttons

In order to make the range of the located button more close to the button's frame, based on the found ROI for each button the processes of PAC_x and PAC_y are applied again onto these ROIs for refining them. The 1st row of Fig. 8 displays the buttons located previously. The 2nd row and the 3rd row in Fig. 8 show the refining regions during the process of PAC_x and PAC_y, respectively. Here the red mark area will be shrunk, and then the refining results can be finally obtained as the 4th row of Fig. 8 displays.

From the result of current illustration, it is obvious that some buttons may be not localized or the non-button may be localized as a button by using the previous PAC method [13], [14]. Therefore, in order to enhance the accuracy and stability of the button identification, in addition to the preliminary localization presented in this section, an advanced localization will be presented in next section for further collecting and confirming the buttons' locations. Thereafter, with the confirmed buttons, a neural network based method will also be introduced to identify the symbols inside buttons.

III. Advanced Localization

Based on the current illustration of 14-button panel, there are only six real buttons and one mistake button to be located by means of the preliminary localization method presented previously. Fortunately, these results can provide a useful cue to develop a button kernel for finding other real buttons. The useful cue is that since the most of located buttons contain the information of button frame and its inside symbols, the K-means analysis [26] can be used to classify these buttons. In this study, K = 3 is empirically used and all the buttons clustered in a group containing the most of located buttons can be further derived as a kernel button, which will be served as the kernel used in the advanced localization presented in this section.

A. Kernel Preparation from Edge Information

Assume there are N buttons, say $ROI_i, i = 1, 2, \ldots, N$, selected for the kernel button computation, i.e., to yield the kernel image $K(W', H')$ with the kernel width W' and kernel height H'. Each ROI_i is then resized and denoted as $K_i, \forall i$ according to the kernel image size. Finally the kernel image may be piled up pixel-by-pixel and expressed by

$$K(W', H') = \sum_{i=1}^{N} K_i(W', H') \tag{9}$$

Fig. 9 shows the found kernel for the current example by piling up the selected buttons shown in (b), (d) and (g) at the 4^{th} row of Fig. 8. Note here that since the piled up buttons are with edge information, the kernel prepared in this stage is also with edge information, which will be used for the next convolution process to derive the so-called star feature as indicated in the flowchart of Fig. 10.

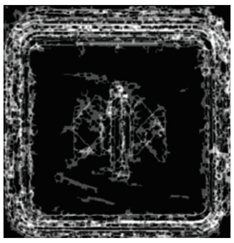
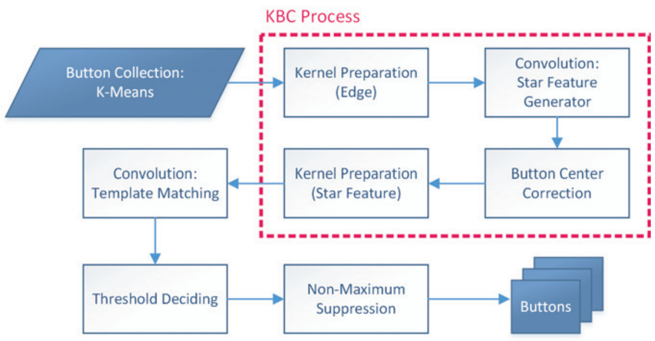


Fig. 9. The found kernel with edge information.


KBC Process
Button Collection: K-Means
Kernel Preparation (Edge)
Convolution: Star Feature Generator
Button Center Correction
Kernel Preparation (Star Feature)
Convolution: Template Matching
Threshold Deciding
Non-Maximum Suppression
Buttons
Fig. 10. The detailed flowchart including KBC process, which is designed for further localizing the missing buttons from the previous PAC process.

B. Kernel with Star Feature

Based on the found kernel image K as given in Fig. 9, it can be used to perform the convolution onto the M-image for highlighting the button area, which is helpful for the later button retrieving process. Fig. 11 illustrates the convolved result, which is called here as the star feature intensity (or SFI) image and explained as follows. Because the shape of our kernel is of rectangle and the direction of convolution slides from left to right and top to bottom, within the button area it will look like a '+' star symbol in the SFI image. Moreover, within the non-button area the SFI will become more obliterated as shown in Fig. 11. Therefore, based on such a SFI information, a useful kernel will be derived from the SFI image and be regarded as a template for further retrieving buttons.

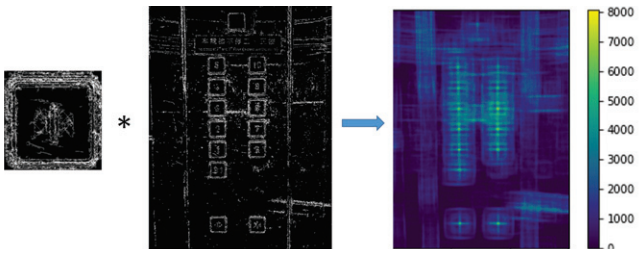


Fig. 11. Convolution process in edge feature domain, and the generated SFI image.

Along the current illustration, since we have three selected buttons as shown in (b), (d) and (g) at the 4^{th} row of Fig. 8 used for the kernel derivation, the corresponding region (or K_i) in the SFI image can be confined by the kernel frame as the red rectangle plotted in Fig. 12(a). According to the star feature property, the center of a K_i should have the maximum intensity, however it is not usually true in real situation. Therefore the original center ($Center_{org}$) of K_i should be adjusted to the 'true' center ($Center_{adj}$) having the maximum intensity and thus the new region of K_i is obtained and denoted as $\tilde{K}_i$. Fig. 13 demonstrates such a center adjustment. Based on this center adjustment process, the original three kernel frames plotted in Fig. 12(a) can be corrected as shown in Fig. 12(b), and their adjustment details are given in Fig. 12(c)-(e), respectively.

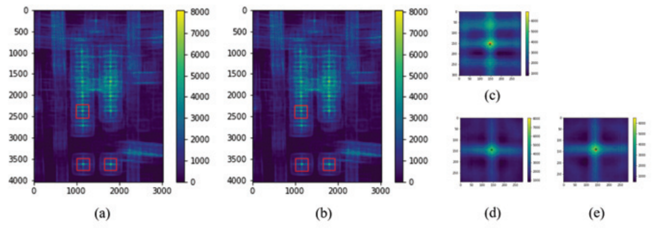


Fig. 12. (a) The illustration of the K_i before center correction on the SFI image; (b) the result after correcting each obtained ROI, which shows in (c)-(e) respectively.

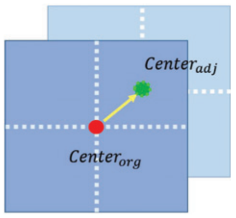


Fig. 13. Demonstration of moving a selected K_i from $Center_{org}$ to $Center_{adj}$ and obtaining the new $\tilde{K}_i$.

In order to obtain the kernel $\tilde{K}$ based on the SFI information for next template matching, the equation of finding kernel K based edge information in Eq. (9) is reused as follows.

$$\tilde{K}(W', H') = \sum_{i=1}^{N} \tilde{K}_i(W', H') \tag{10}$$

Fig. 14(a)-(c) show the process of consequently finding such a kernel by piling up the kernel frames displayed in 12(c)-(e), and the final result $\tilde{K}$ is obtained as shown in Fig. 14(d), which will be used in next section for template matching and retrieving more wanted buttons.

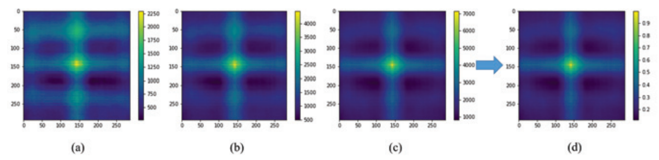


Fig. 14. (a)-(c) show the pileup process consequently, and (d) shows the obtained kernel $\tilde{K}$ based on the SFI information for template matching.

C. Template Matching

Based on the found kernel $\tilde{K}$ and the SFI image, to locate all the wanted buttons, the template matching (TM) method [27] is adopted here. By fixing the width W' and height H' of the kernel $\tilde{K}(W', H')$, let T and I denote the kernel $\tilde{K}$ and the SFI image. The matching score (*MS*) for the location (x, y) may be defined as

$$MS(x, y) = \sum_{i=0}^{W'-1} \sum_{j=0}^{H'-1} T(i, j) \cdot I(x + i, y + j). \tag{11}$$

By sliding window to perform such a template matching as illustrated in Fig. 15(a), the template matching intensity (or TMI) image can be obtained as shown in Fig. 15(b) according to the Eq. (11). Since all the kernel $\tilde{K}_i, \forall i$ in the TMI domain, are selected for highlighting all possible wanted button regions, based on these kernels a reasonable threshold called TH_{tmi} may be expressed by

$$TH_{tmi} = \frac{1}{2} \min_{\forall i} \left[\max_{\forall (x,y) \in \tilde{K}_i} MS(x, y) \right]. \tag{12}$$

By this thresholding process, all the cases satisfying $MS > TH_{tmi}$ can be found as the red frames plotted onto the E-image shown in Fig. 15(c).

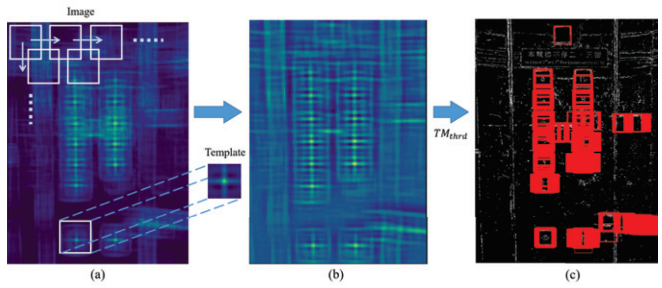


Fig. 15. (a) Illustration of template matching by means of sliding window, (b) the obtained TMI image, and (c) the found ROIs satisfying $MS > TH_{tmi}$.

D. Non-Maximum Suppression

From the result of Fig. 15(c), it seems that too many ROIs are intersected. In this study, the non-maximum suppression [28] method, which is often used in the field of object detection (e.g., human face detection), is adopted to filter out the overlapping ROIs of the same object. Its procedure is described as follows.

1) Find the score for each ROI.
2) Take the ROI having the maximum score as reference.
3) Compute the Intersection over Union (IoU) between the reference and the other one.
4) Suppress the ROI whose IoU is greater than a given threshold TH_{IoU}.
5) Repeat step 3) to step 4) until no ROI needed to be suppressed.
6) Sort the left ROIs which are not yet been processed, and repeat step 2) to step 5).

In this study, $TH_{IoU} = 10\%$ is heuristically selected since the real button would not be overlapped together. Fig. 16 demonstrates the non-maximum suppression (NMS) using the operation of IoU. Fig. 17 shows the result of NMS after the TM process, which is called as 'TM+NMS'. This TM+NMS image demonstrates that all the wanted buttons have been located at this stage of advanced localization even some mistakes still exist.

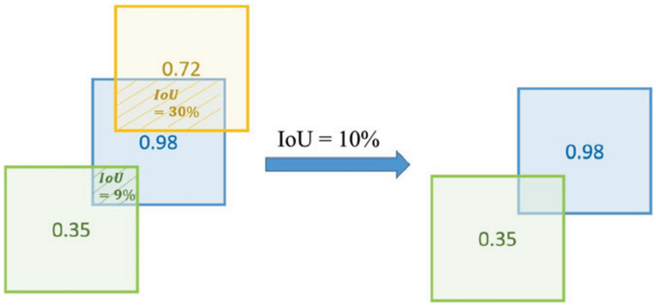


Fig. 16. Demonstration of NMS using the operation of IoU.

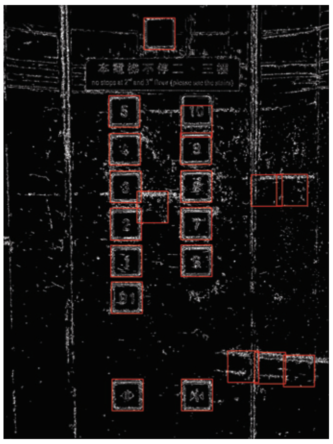


Fig. 17. The NMS result after the TM process.

IV. Recognition

After finishing the button localization, in order to make the located button more meaningful, the symbol inside the button should be recognized. In this study based on the button information within a small data set of buttons, the outer frame of a button should be removed as clear as possible in advance for facilitating the recognition of its inside symbol information, which is presented and illustrated as follows. We collect the gray button as illustrated in Fig. 18(a) and extract its edge information shown in Fig. 18(b) as the reference. Then the dilation operation is performed twice for enhancing the symbol as shown in Fig. 18(c). And finally by excluding the outer frame with its $\frac{width}{6}$ and $\frac{height}{6}$, the final symbol information of this button can then be confined and obtained as shown in Fig. 18(d).

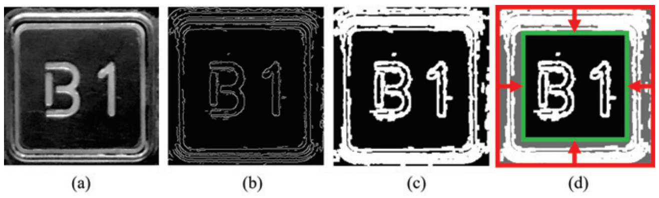


Fig. 18. (a) Button in gray scale, (b) its edge information, (c) the dilation result, and (d) the used button information by excluding the outer frame.

A. *Button Dichotomy Based on CNN*

As shown in Fig. 17, after a series of processes for button localization, all the buttons have been collected without any missing but some non-button information still exists. Therefore, before performing the recognition of the symbol inside a button, a button dichotomy based on CNN is designated for confirming whether the located button is a real one or not, which is presented as follows.

1) Model Introduction: With total 476 button samples, we divided the data into three parts: training set, validation set, and testing set with the proportion of $7:2:1$. Because only a small data amount and the low diversity in elevator patterns we could collect, some well-known object identification methods such as YOLO and Mask-RCNN are not suitable for this application. Thus to perform the so-called button dichotomy, our simple CNN-like model can be regarded as a is-button/non-button classifier and designed as the structure illustrated in Fig. 19, where only three convolution layers are used.

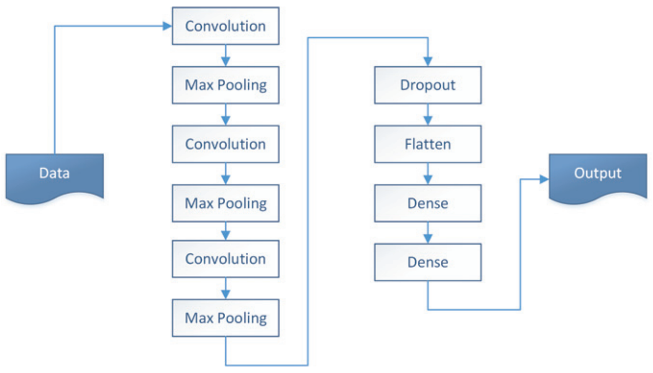


Fig. 19. The structure of our CNN model.

2) Model Evaluation: Fig. 20 shows the performance of our model with (a) model accuracy and (b) model loss, where the model converges between the 10^{th} and 16^{th} epoch, and becomes overfitting after 17^{th} epoch as indicated in Fig. 20(b). In addition, the accuracy of the model in Fig. 20(a) shows that our model learned well in the early epoch. Because of the small amount of training data and the easy task assigned, the behaviour of our performance evaluation is reasonable. Therefore, the 15^{th} epoch is adopted in our final model, where 95.16% accuracy is obtained after testing.

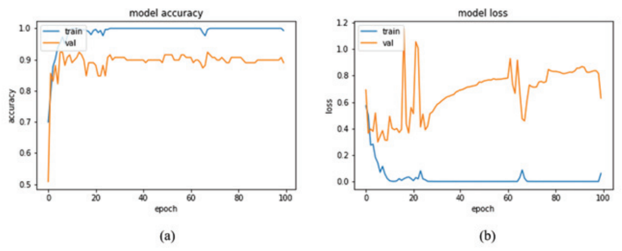


Fig. 20. The performance of our model explored with (a) model accuracy and (b) model loss.

Along with the current illustrative example, based on the TM+NMS image shown in Fig. 17, the obtained result by our button dichotomy method is shown in Fig. 21, where the predicted ‘non-button’ and the predicted ‘is-button’ is marked in red and green rectangle, respectively. Note here that the score is ranged from 0 to 100. The lower the score, the higher the possibility of the ROI belonging to ’is-button’; and vice versa.

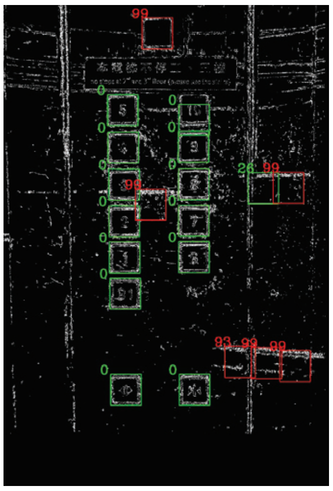


Fig. 21. The result after our button dichotomy process. The predicted ‘non-button’ and the predicted ‘is-button’ is marked in red and green rectangle, respectively, where the score is ranged from 0 to 100.

B. *Symbol Recognition*

After the button dichotomy process, the symbol inside the button will be further recognized, where a glyph table containing the button’s symbol should be modelled in advance. To identify the symbol of a unknown button, the structural similarity (SSIM) comparison method [29] is adopted in this study. Fig. 22 shows our glyph table with 17 classes including “1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, B1, Call, Close, and Open,” where the same meaning symbols are put into the same row. Due to the limitation of the collected button panels, the amount of each symbol is vary.

Given two images as illustrated in Fig. 23, the SSIM can be obtained by the following expression, where the image luminance l, contrast c and structure s are used. The sliding

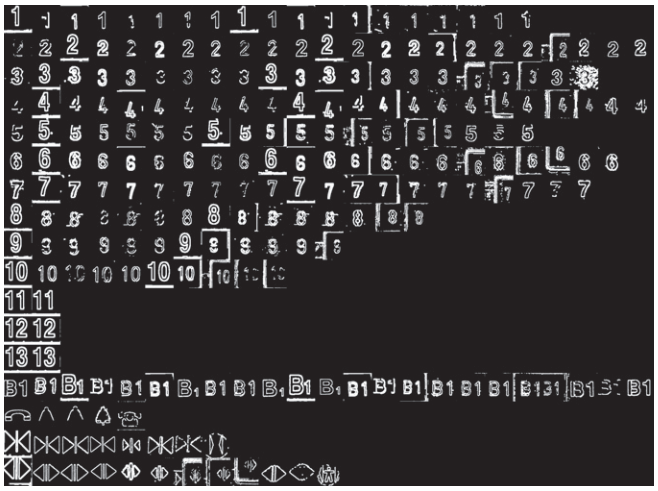


Fig. 22. The glyph table for elevator button symbol recognition.

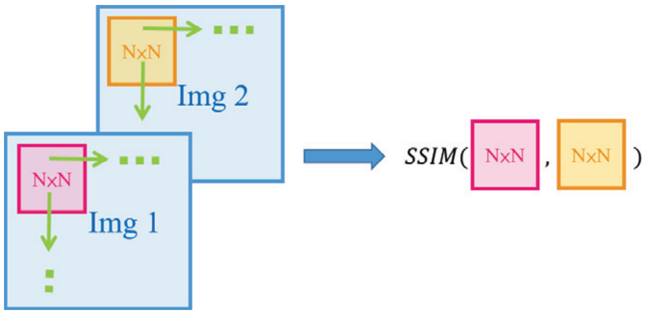


Fig. 23. Illustration of SSIM.

window size is defined by $N \times N$.

$$SSIM(x,y) = [l(x,y)]^{\alpha}[c(x,y)]^{\beta}[s(x,y)]^{\gamma} \tag{13}$$

$$\begin{cases} l(x,y) = & \frac{2\mu_x\mu_y+c_1}{\mu_x^2+\mu_y^2+c_1} \\ c(x,y) = & \frac{2\sigma_x\sigma_y+c_2}{\sigma_x^2+\sigma_y^2+c_2} \\ s(x,y) = & \frac{\sigma_{xy}+c_3}{\sigma_x\sigma_y+c_3} \end{cases} \tag{14}$$

In our study, we let $\alpha = \beta = \gamma = 1$ and $c_3 = c_2/2$, then the SSIM can be derived as below

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{15}$$

where the meanings of the relative parameters are listed as follows.

$$\begin{cases} \mu_x, \mu_y & \textit{average of x, average of y} \\ \sigma_x^2, \sigma_y^2 & \textit{variance of x, variance of y} \\ \sigma_{xy} & \textit{covariance of x and y} \\ c_1, c_2 & \textit{constant used to maintain stability} \end{cases} \tag{16}$$

After every local SSIM is calculated on each position of the entire image, the average of these local (SSIM)s is regarded as the final SSIM of the two images. The SSIM is ranged from 0 to 1 and the value of SSIM becomes 1 if the two images are identical. For the current example, the result of SSIM is shown in Fig. 24 with the text format of 'class: confidence' in green. Since the SSIM recognizes all the buttons classified as 'is-button' from the previous CNN button dichotomy, the accuracy of SSIM recognition discussed in this paper would not include the error-classified button like "Call: 85" in Fig. 24, which is the CNN's error, not the SSIM's error.

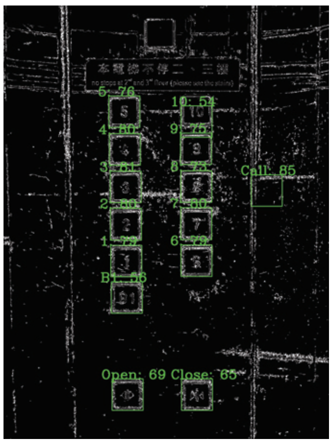
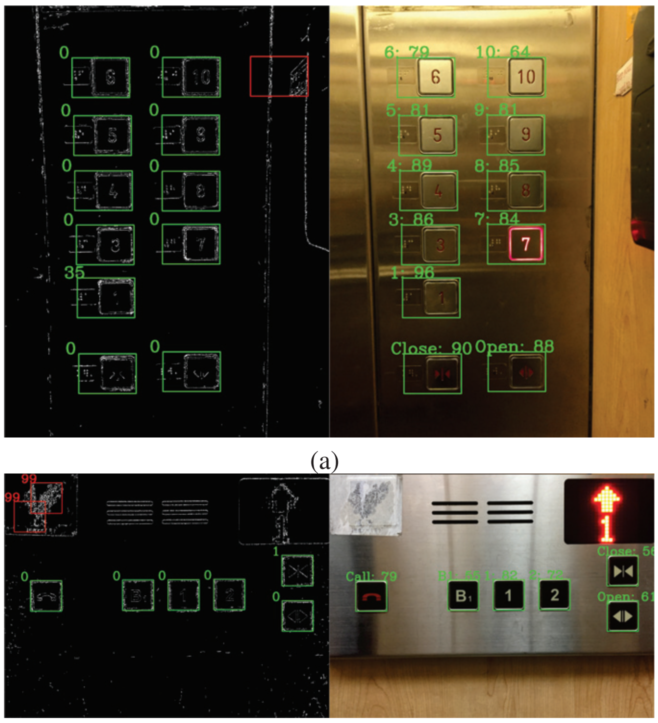


Fig. 24. The SSIM result with the text format in 'Class: confidence.' For example: button 5 is predicted as the class '5' with 76% confidence.

V. Results and Discussions

The proposed methods are implemented by Microsoft® Visual Studio® 2013 C/C++ and Python 3.6 under Windows 10. The developed codes are run on a desktop computer with Intel® Core™ i5-4590 3.30 GHz CPU and 8G RAM as well as NVIDIA Geforce GTX 1050Ti. There are 32 different elevator button panels with 328 buttons to be collected for our experiments. Fig. 25 demonstrates the other two experimental results, where the results of button localization and symbol recognition are shown in left-side and right-side, respectively.



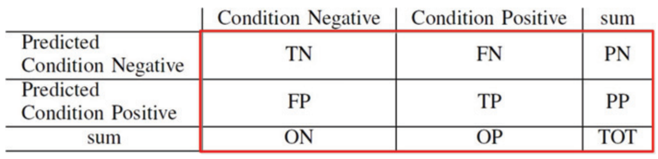
(a)

(b)

Fig. 25. Two other demonstrations show the feasibility of our approach, where the left-side and right-side present the results of localization and symbol recognition, respectively.

Table I lists the related parameters for the experimental analyses, where TN denotes True Negative, FN denotes False Negative, FP denotes False Positive, TP denotes True Positive, PN denotes TN+FN, PP denotes FP+TP, ON denotes TN+FP, OP denotes FN+TP, and TOT denotes TP+FP+FN+TN, respectively.

TABLE I
THE ADOPTED 9 PARAMETERS USED FOR OUR EXPERIMENTAL ANALYSES.



	Condition Negative	Condition Positive	sum
Predicted Condition Negative	TN	FN	PN
Predicted Condition Positive	FP	TP	PP
sum	ON	OP	TOT

Table II reports the classification performance of PAC, TM+NMS, CNN and SSIM based on the 9 parameters highlighted within the red box in Table I. The first two PAC and TM+NMS are the methods belonging to button localization, whereas the latter two are for the button's symbol recognition. Based on the 9 information reported in Table II, the useful performance, e.g., recall, specificity, accuracy, miss rate, precision, can be obtained for each methodology as reported in Table III and discussed as follows.

TABLE II
THE CLASSIFICATION TABLE OF PAC, TM+NMS, CNN AND SSIM CORRESPONDING TO THE RED HIGHLIGHTED AREA IN TABLE I.

PAC			TM+NMS			CNN			SSIM		
0	133	133	0	5	5	111	8	119	0	0	0
9	191	200	9	323	332	4	314	318	0	314	314
9	324	333	9	328	337	115	322	437	0	314	314

TABLE III
THE PERFORMANCE EVALUATION FOR LOCALIZATION (PAC, TM+NMS) AND RECOGNITION (CNN, SSIM). HERE '-' DENOTES NOT AVAILABLE.

	PAC	TM+NMS	CNN	SSIM
TPR(Recall)	0.589	0.985	0.975	1
TNR(Specificity)	0	0	0.965	-
ACC	0.574	0.958	0.973	1
FNR(Miss rate)	0.410	0.015	0.025	0
PPV(Precision)	0.955	0.973	0.987	1
NPV	0	0	0.933	-

In the performance evaluation, the original PAC localization and the newly presented TM+NMS method with the relative true positive (TP), false negative (FN) and false positive (FP) are compared and discussed. For the index of true negative (TN), it is zero in both methods as reported in Tabel II and thus not discussed here. For button localization, the accuracy (ACC) only obtains 0.574 with the PAC method, whereas 0.958 accuracy could be achieved by further including the 'TM + NMS' processes as reported in Table III.

In Section IV-A, a simple CNN model is implemented to perform the button dichotomy for distinguishing whether the collected ROI belongs to a real button or not. Apart from the model evaluation obtaining the 0.952 accuracy presented in Subsection IV-A2, Table III further reports that our CNN shows 0.973 accuracy for button dichotomy and the SSIM symbol recognition perform 100% accuracy. Note here that the accuracy of SSIM is computed under the condition that the button dichotomy in CNN is correct. That is, only the TP class in CNN is used for computing the accuracy of SSIM.

VI. CONCLUSIONS

This paper presents a method to localize (with 95.8% accuracy) and recognize (with 100% accuracy) the button with a series of processes including PAC, KBC, TM, CNN and SSIM methods. This study could be helpful for enabling a robot possessing the ability of recognizing the buttons in the elevator, which is a useful function to enhance the indoor navigation ability for a mobile robot. Moreover, the experiments with 32 panels in different pattern, different light condition and different material, confirm the feasibility of the proposed approach.

The main contribution of this paper compared to our early PAC method is twofold. Firstly, the designed advanced localization improves the recall of the buttons from 58.9% to 98.5% to retrieve the most of missing buttons from the preliminary localization stage. Secondly, the symbol recognition based on the processed glyph table is involved in our system and a 100% accuracy of symbol recognition is obtained. However, this is only for the recognition accuracy. If the CNN button dichotomy predicted the button as the non-button, our system could not be aware since CNN is our last goalkeeper for collecting the appeared buttons. That is, if our template matching plus NMS process did not collect the appeared buttons, our system could not correct them, neither.

Although the proposed method has tried best to confirm and retrieve the buttons, so far it still has no self-correction mechanism to enhance the system accuracy. We are not satisfied enough with the accuracy due to the button missing situation happened, even it is very few. Accordingly, by including a feedback mechanism to our system for retraining our CNN model is worthy of further studying and could be considered as our future work.

REFERENCES

[1] J.-W. Hsieh, S.-H. Yu, Y.-S. Chen, and W.-F. Hu, "Automatic traffic surveillance system for vehicle tracking and classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, vo. 2, pp. 175-187, 2006.

[2] Y.-S. Chen and J.-Y. Wang, "Computer vision-based approach for reading analog multimeter," *Applied Sciences*, vol. 8, no. 8, 1268, 2018.

[3] Y.-S. Chen and K.-L. Lin, "CUBot: Computer vision on the eye-hand coordination with a computer-using robot and its implementation," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 32, no. 4, 1855005, 2018.

[4] Z. Cui, L. Heng, Y.-C. Yeo, A. Geiger, M. Pollefeys, and T. Sattler, "Real-Time Dense Mapping for Self-Driving Vehicles using Fisheye Cameras," in *Proc. International Conference on Robotics and Automation (ICRA)*, pp. 6087-6093, 2019.

[5] C. Patruno, M. Nitti, A. Petitti, E. Stella, and T. D'Orazio, "A vision-based approach for unmanned aerial vehicle landing," *Journal of Intelligent & Robotic Systems*, vol. 95, pp. 645-664, 2019.

[6] S. Hayat, S. Kun, Z. Tengtao, Y. Yu, T. Tu, and Y. Du, "A deep learning framework using convolutional neural network for multi-class object recognition," in *Proc. International Conference on Image, Vision and Computing (ICIVC)*, pp. 194-198, 2018.

[7] Y.-S. Chen and K.-L. Lin, "Screen image segmentation and correction for a computer display," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 28, no. 4, 1454002, 2014.

[8] M.-T. Chao and Y.-S. Chen, "Keyboard recognition from scale-invariant feature transform," in *Proc. IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, pp. 207-208, 2017.

[9] H. Kim, D. Kim and K. Park, "Robust Elevator Button Recognition in the Presence of Partial Occlusion and Clutter by Specular Reflections," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 3, pp. 1597-1611, March 2012.

[10] H. Yang, Q. Wu, P. Li, K. Lin and L. Hou, "Automatical encoding of button products based on visual recognition," in *Proc. IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 399-403, 2017.

[11] Z. Dong, D. Zhu and M. Q.-H. Meng, "An autonomous elevator button recognition system based on convolutional neural networks," in *Proc. IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 2533-2539, 2017.

[12] E. Klingbeil, B. Carpenter, O. Russakovsky and A. Y. Ng, "Autonomous operation of novel elevators for robot navigation," in *Proc. IEEE International Conference on Robotics and Automation*, pp. 751-758, 2010.

[13] Y.-C. Hsu and Y.-S Chen, "Button Segmentation from an Elevator Inside Door Image," in *Proc. The International MultiConference of Engineers and Computer Scientists IMECS 2018*, Hong Kong, vol. I, pp. 343-347, 2018.

[14] Y.-S Chen and Y.-C. Hsu, "Approach to the segmentation of buttons from an elevator inside door image," *Transactions on Engineering Technologies - IMECS 2018* (Eds. by Sio-Iong Ao, Haeng Kon Kim, Oscar Castillo, Alan Hoi-shou Chan, Hideki Katagiri), Springer Nature Singapore Pte Ltd., pp. 105-118, 2020.

[15] J.-D. Yang, Y.-S. Chen and W.-H. Hsu, "Adaptive thresholding algorithm and its hardware implementation," *Pattern Recognition Letters*, vol. 15, no. 2, pp. 141-150, 1994.

[16] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698, 1986.

[17] Y.-S. Chen, "Registration of seal images using contour analysis," in *Proc. of 13th Scandinavian Conference on Image Analysis* (Series Editors: G. Goos, J. Hartmanis, and J. van Leeuwen, Volume Editors: J. Bigun and T. Gustavsson), Springer-Verlag Berlin Heidelberg, vol. LNCS 2749, pp. 255-261, 2003.

[18] Y.-S. Chen and W.-H. Hsu, "A modified fast parallel algorithm for thinning digital patterns," *Pattern Recognition Letters*, vol. 7, no. 2, pp. 99-106, 1988.

[19] Y.-S. Chen and M.-T. Chao, "Pattern reconstructability in fully parallel thinning," *Journal of Imaging*, vol. 3, no. 3, 29, 2017.

[20] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybernetics*, vol. 36, pp. 193-202, 1980.

[21] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998.

[22] Y. Lecun, Y. Bengio and G. Hinton, "Deep learning," *Nature*, vol. 521, 2015.

[23] D. G. Lowe, "Object recognition from local scale-invariant features," *IEEE International Conference on Computer vision*, vol. 2, pp. 1150-1157, 1999.

[24] Bay H., Tuytelaars T., Van Gool L., "SURF: Speeded Up Robust Features," *Lecture Notes in Computer Science, Springer*, vol 3951, 2006.

[25] Redmon, Joseph and Ali Farhadi, "YOLOv3: An Incremental Improvement, Published in *ArXiv 2018*.

[26] Hartigan, J. A., and M. A. Wong., "Algorithm AS 136: A K-Means Clustering Algorithm, *Journal of the Royal Statistical Society*, Series C (Applied Statistics), vol. 28, no. 1, pp. 100108, 1979.

[27] Brunelli, Roberto, "Template Matching Techniques in Computer Vision: Theory and Practice," in *Wiley Publishing*, 2009.

[28] A. Neubeck and L. Van Gool, "Efficient Non-Maximum Suppression," *"18th International Conference on Pattern Recognition (ICPR'06)*, Hong Kong, pp. 850-855, 2006.

[29] K. Gu, M. Liu, G. Zhai, X. Yang and W. Zhang, "Quality Assessment Considering Viewing Distance and Image Resolution," in *IEEE Transactions on Broadcasting*, vol. 61, no. 3, pp. 520-531, Sept. 2015.