*- Technical Paper -*

# A STUDY ON SEMI-AUTOMATIC CONCRETE CRACKS DETECTION USING INTERACTIVE GENETIC ALGORITHM

Cuong Nguyen KIM*[1], Kei KAWAMURA*[2], Amir TARIGHAT*[3]

ABSTRACT

Concrete surface crack detection is an important task of the inspection and diagnosis of concrete structures. This paper proposes a prototype software for semi-automatic crack detection from concrete surface images using an interactive genetic algorithm (iGA) and touch screen. The iGA is applied to optimize image processing parameters for each image with no knowledge of image processing techniques. Furthermore, the experimental results indicate that the proposed method has possibility of extracting crack with reasonable accuracy and working time.

Keywords: genetic algorithm, image processing, crack detection.

## 1. INTRODUCTION

The number of deteriorated concrete structures has increased dramatically in many countries. Therefore, the management of existing concrete structures has become a major social concern worldwide. concrete cracks are important indicators reflecting the safety of infrastructure. To keep concrete structures at good condition under all circumstances, the inspection and maintenance are important.

Crack detection on concrete surface is the first task of the inspection and maintenance of concrete structures. The conventional methods for crack detection are generally implemented by the naked eye. Therefore inspection results are usually subjective, and it takes a long time.

Recently, there have been many researchers have detected the cracks on concrete surface automatically such as Nagao et al. (2012) [1], Nishikawa et al. (2012) [2].

The full automation of crack detection from digital images is difficult due to many factors such as the difference in the photography environment and concrete surface condition. Therefore, image processing parameters to detect the cracks on concrete surface have to be adjusted for each image. Moreover, it is hard for inspectors to adjust the parameters without knowledge of image processing techniques.

The main target of this study was to propose the prototype software for semi-automatic crack detection from concrete surface images using an interactive genetic algorithm (iGA) and touch screen. The iGA was applied to optimize image processing parameters for each image. The output of the software was a crack map that was compared with a benchmark image to verify reasonable accuracy and working time from a practical engineering point of view.

## 2. PROPOSED METHOD

### 2.1 Overview

Fig.1 shows how the cracks are extracted intuitively on the touch screen where the crack is extracted from a concrete surface image by tracing user's finger. In addition, Fig.2 shows operation procedure of the proposed software. Fig.3 shows sequential 7 steps of image processing technique in which three parameters are adjusted to find out optimal values. Those are size of median filter (fsize), threshold of binarization (binary), and threshold of linear degree (linear). In this paper, the prototype software was development based on image processing technique and the iGA.

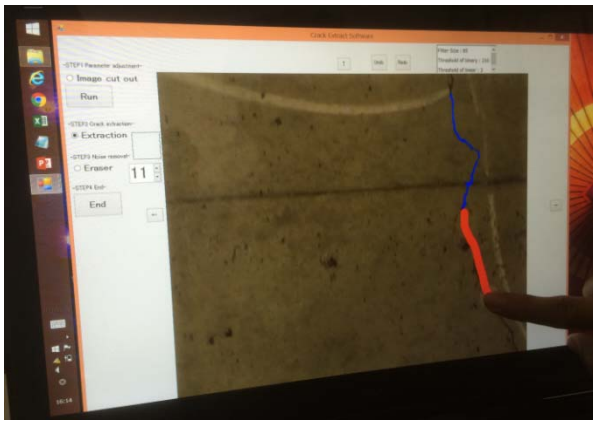



Fig.1 Touch screen for detecting crack by finger tracing

*1 Ph.D. Candidate, Dept. of information science & engineering, University of Yamaguchi, JCI Member
*2 Associate Prof. Dept. of information science & engineering, University of Yamaguchi, JCI Member
*3 Associate Professor, Civil Engineering Dept., Shahid Rajaee Teacher Training University, Teheran, Iran

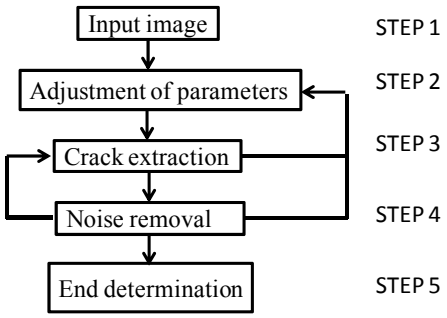



Fig.2 Operation procedure of the software

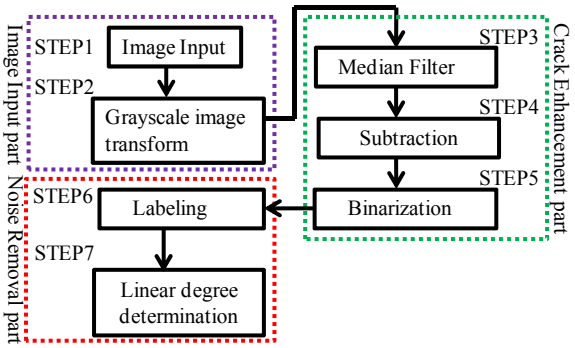



Fig.3 Image processing technique

## 2.2 Image processing technique

The algorithm for accurate crack detection has composed of three major process parts shown in Fig.3. The first part was image input part which included image input and grayscale image transformation. The second part was crack enhancement part which included the following preprocessing steps: median filter, subtraction, binarization. The final part was noise removal part, comprising two steps: labeling and linear degree determination. This image processing technique was presented in our previous research [4].

## 2.3 Application of iGA to the image processing parameters optimization

iGA, proposed in the 1980s, is one of the optimization algorithms for solving an implicit or uncertainty problem. The iGA combines the traditional genetic algorithm (GA) with human's intelligent evaluation. The difference between iGA and GA is the individual's fitness evaluation of a population in every generation. In GA, the individual's fitness is calculated by the fitness function. However, in iGA, the individual's fitness is evaluated by the user. The outline of the basic iGA is shown in Fig.4 [3]. It presents sequential steps of the iGA procedure. In this study, the iGA is used for searching optimal values of the image processing parameters because it has many advantages such as the evaluation for each individual's fitness easier than the individual's fitness function of GA, small population size, less number of generations [5]. Full details of the iGA procedure are presented as following four steps:

### (1) Generating initial population randomly

Firstly, the three parameters are combined to create an individual in a population. Secondly, each individual is represented by a chromosome encoded to a binary string as Fig.5. Namely, the fsize is expressed by 6 bits, the binary is expressed by 8 bits and the linear is expressed by 6 bits. Finally, an initial population was generated randomly with a predetermined size. In addition, Table 1 shows some parameter values. The ranges of the shown values in Table 1 are based on the preliminary experiments [4].

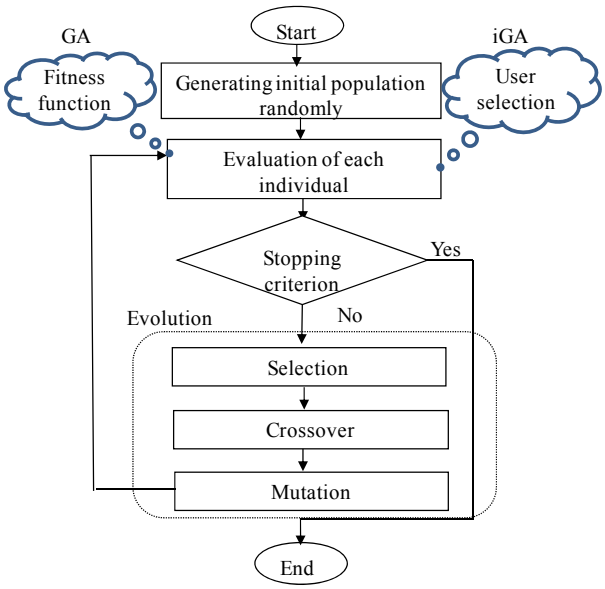



Fig.4 iGA procedure

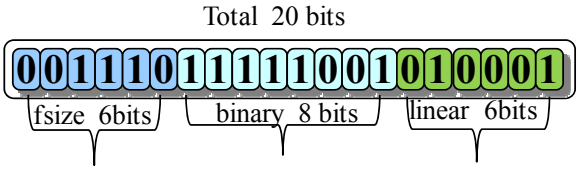



Fig.5 Coding presentation of a solution candidate

Table 1 Properties of parameter [3]

| Variable | Range | Steps | Bits |
|---|---|---|---|
| fsize | 1≤fsize≤127 | 2 | 6 |
| binary | 0≤binary≤255 | 1 | 8 |
| linear | 1.0≤linear≤32.5 | 0.5 | 6 |

### (2) Evaluation of each individual

The fitness of each individual (solution candidate) in a population is evaluated based on user's preference. Therefore, the evaluation result is often a relative fitness. This point was different from traditional GA.

### (3) Stopping criterion

Evaluation of each individual is terminated by the user when the user chooses the best parameters.

### (4) Evolution

When stopping criterion is not satisfied, evolution procedure is implemented for operating selection, crossover, and mutation until the user chooses the best parameters.

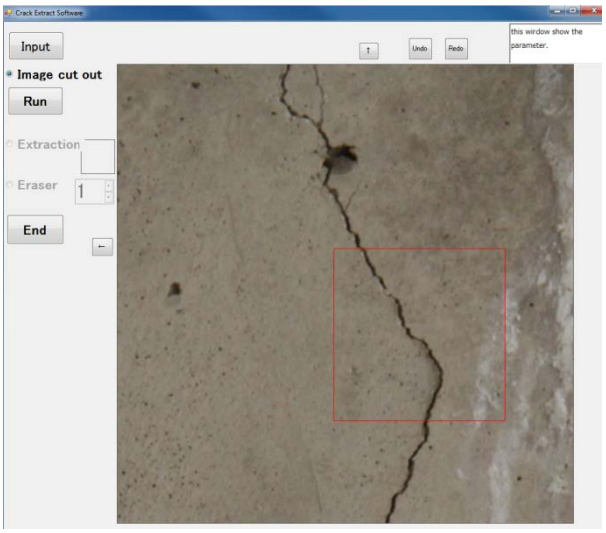



Fig.6 Input image window

## 3. PROTOTYPE SOFTWARE DEVELOPMENT

In this study, the image processing algorithm shown in Fig.3 is developed in the C program language. In addition, an interactive interface of the software for extracting crack and searching optimal parameter values are developed in C # program language (see Fig.6, Fig.7, Fig.9, Fig.10).

The following describes the prototype software that implements the flow shown in Fig.2. In addition, each STEP below corresponds to the STEP number of Fig.2.

### 3.1 STEP1 (Input image)

Firstly, the user inputs an original image by clicking "input" button. Secondly, the user selects a cut-out image to adjust the image processing parameters. Here, the size of the cut-out image is 300 x300 pixels. Finally, the user chooses an interested region of the cracks. Accordingly, a red frame appeared automatically as Fig.6. The purpose of the crack region selection is to search the optimal image processing parameter values for the cut- out image.

### 3.2 STEP2 (Adjustment of parameters)

The interactive interface is presented as Fig.7 so that the user adjusts the parameters. The software generates 15 sample images (individuals) as an initial population. The user can keep elites (better individuals) to next new generation. If the user cannot keep any individuals, the user has to operate upon "Initialization" to generate a new population.

In this window, the user evaluates the fitness of each individual with five buttons from 1 to 5 corresponding to the fitness from the worst to the best. These buttons are located below sample images. Subsequently, the user chooses "Evolution" button to perform gene operators (crossover, mutation).

The purpose of evolution is to create a better individual in the new generation until finding out the best individual. As an illustration, Fig.7 also shows that the user finds out sample image (2) with the best parameter values.

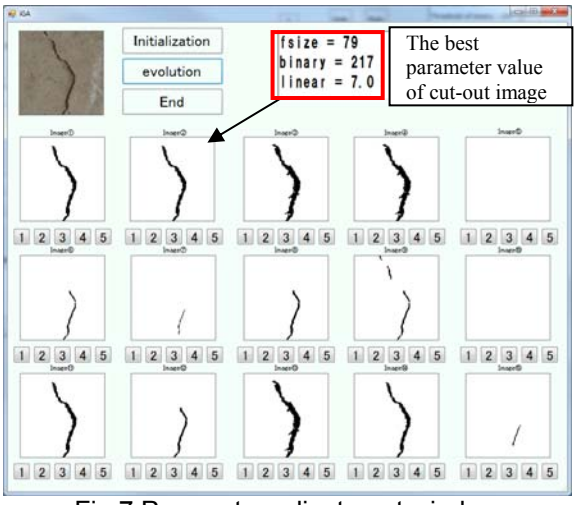



Fig.7 Parameter adjustment window

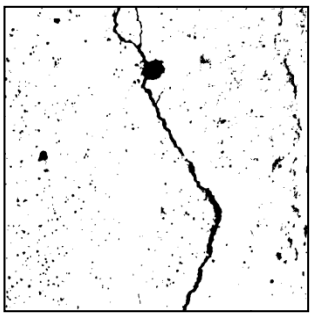


Fig.8 Background image

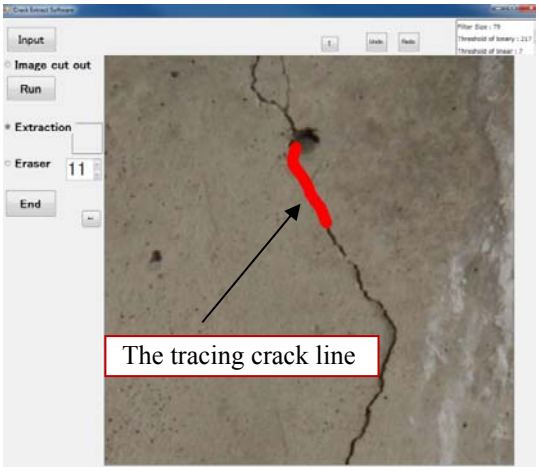



Fig.9 Tracing crack on background image

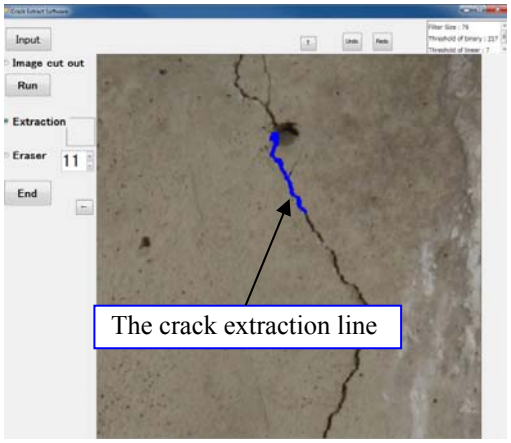



Fig.10 Crack extraction image

The user clicks “End” button to determine at the end of the parameter adjustment procedure. As a result, a background image is created according to the optimal image processing parameter values as Fig.8. However, the prototype software does not show the user’s background image which is saved automatically in a specific folder.

### 3.3 STEP3 (crack extraction)

To extract cracks from the original image smoothly, the user traces roughly the shape of the cracks by a finger on the touch panel as Fig.9. This operation is also capable of processing with a computer mouse. The place in which the user traces on the touch panel is displayed as a red line. That is an interested region of crack extraction. Additionally, the width of the extraction area can be adjusted by a size selection which is set in advance from 1pixel to 51pixel (the size increments in 5pixels). After limiting the crack region by a user's fingertip, the cracks are automatically displayed in the selected region with a blue line as Fig.10. The user has to trace region of the crack to extract it from the background image. Hence, the crack of the final result image will be extracted with reduced noises. As a matter of course, depending on the chosen parameter value, the cracks can be appeared with noise pixels or loss pixels. If some crack parts are not displayed, then the user returns STEP 2 to choose other parameter values. If many noises are also displayed as a portion of the crack, then the user moves to next step 4 for noise removal.

### 3.4 STEP4 (Noise removal)

When some noises appear with the cracks are extracted from STEP3, the user selects the “Eraser” button in Fig.10 to carry out a process of noise removal. If the noises are not presented, the user moves to STEP5 without selecting the “Eraser” button.

### 3.5 STEP5 (End determination)

When the user completes the crack extraction process, the user presses “End” button at the bottom left of the window shown in Fig.10. On the other hand, when the user continues working crack extraction, the user returns to STEP3. If the user wants to choose the parameters again, the user returns to STEP2.

Consequently, the outputs of the software have two images automatically saved in the specified folder. Those are a crack extraction image as Fig.10 and a crack map (final result image) as Fig.12.

## 4. EXPERIMENT

### 4.1 Sample test

The experiment is implemented with 4 original images which shown as Fig.11. Additionally, a characteristic feature of the original images is shown as Table 2. To assess the effectiveness of the software objectively, 3 users who know about image processing and know this software usage are invited to make the

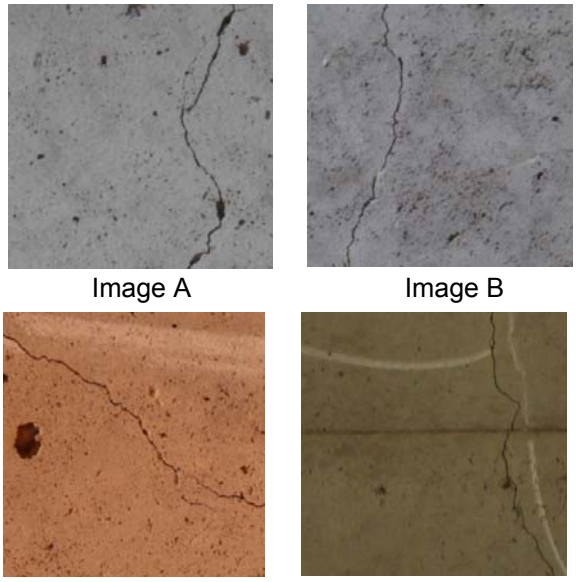



Fig.11 Original image

Table 2 Characteristic feature of the original images

| Image | Image size (pixel) | Resolution (mm/pixel) | Maximum crack width (mm) | Visual characteristic |
|---|---|---|---|---|
| A | 800x800 | 0.06 | 0.4 | Dirty spot |
| B | 800x800 | 0.2 | 0.35 | Rough surface |
| C | 800x800 | 0.06 | 0.4 | Honeycombs |
| D | 800x800 | 0.06 | 0.3 | Mold mark |

Fig.12 Final result image

Fig.13 Benchmark image

experiment.

Each user detected and extracted the cracks for 4 selected original images, their upside down images, and their left side right images. Thus, totally 36 test samples were available to conduct experiments.The purpose of the direction change of the original images is to test the accuracy of the crack extraction.

The validity of the software is expressed by two criteria comprising working time and accuracy of the crack extraction. To verify these criteria, a benchmark image is made from the original image by a manual tool as Fig.13. Working time, the loss and noise rate of final result image were compared with the benchmark image.

Table 3 Average working time of 3 users

| Image | $T_1$(second) | User | $T_2$ (second) | $T$% |
|---|---|---|---|---|
| A | 1120 | #a | 142 | 87 |
| | | #b | 112 | 90 |
| | | #c | 106 | 91 |
| B | 1161 | #a | 106 | 91 |
| | | #b | 100 | 91 |
| | | #c | 94 | 92 |
| C | 1492 | #a | 123 | 92 |
| | | #b | 119 | 92 |
| | | #c | 95 | 94 |
| D | 1368 | #a | 95 | 93 |
| | | #b | 71 | 95 |
| | | #c | 60 | 96 |

### 4.2 Working time measure

To measure working time for the benchmark image, and final result image, the Stopwatch is used during the experiment progress. Effectiveness of the software usage in regards to working time is expressed as following equation:

$$T = \frac{T_1 - T_2}{T_1} \times 100\% \quad (1)$$

where,

$T_1$ : working time to make the correct image.

$T_2$ : working time to make the final result image.

$T$ : the percent of reduced time.

Table 3 shows average working time for 3 users to make the correct image and the crack extraction of 36 sample tests. Comparing the values of $T_1$ and $T_2$ of all images, it is expressed that the highest value of $T$ is 96%, and the lowest value of $T$ is 87%. Additionally, the average value of $T$ for all images is 92%. It is concluded that the time is significantly shortened when applying this software to extract the cracks.

In addition, the $T_1$ values of image C are the highest of all images. In contrast, the $T_1$ value of image A is the smallest of all images. However, the average $T_2$ value of 3 users for extracting crack of image A is the highest of all images. It is due to that the shape of image A is more complex than the other images.

In the other hand, the $T_1$ value of image D is higher than $T_1$ value of images A and B, but the $T_2$ value of image D is less than $T_2$ value of images A and B. Finally, the Table 3 shows that the $T$ value of image D is the highest of all images which the average value is 95%.

### 4.3 Crack extraction accuracy measure

To assess the accuracy of the software, evaluation value is determined by following equations:

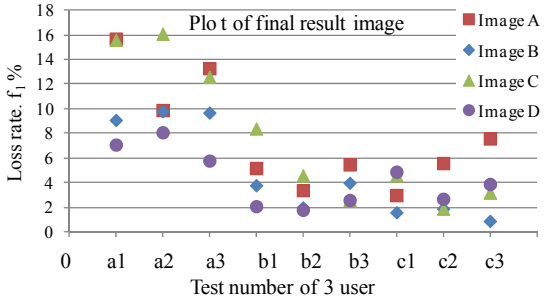



Fig.14 Loss rate- Test number

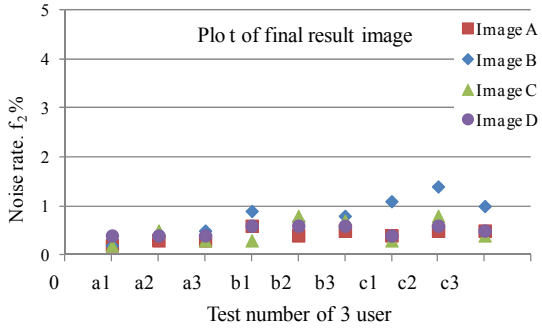



Fig.15 Noise rate –Test number

$$f_1 = \frac{m}{M}; f_2 = \frac{n}{N} \quad (2)$$

$$f = 1 - \sqrt{\frac{1}{2}(f_1)^2 + \frac{1}{2}(f_2)^2} \quad f \in [0,1] \quad (3)$$

where,

$m$, $n$: the number of loss pixels and noise pixels, respectively.

$M$, $N$: the number of background pixels and crack pixels, respectively.

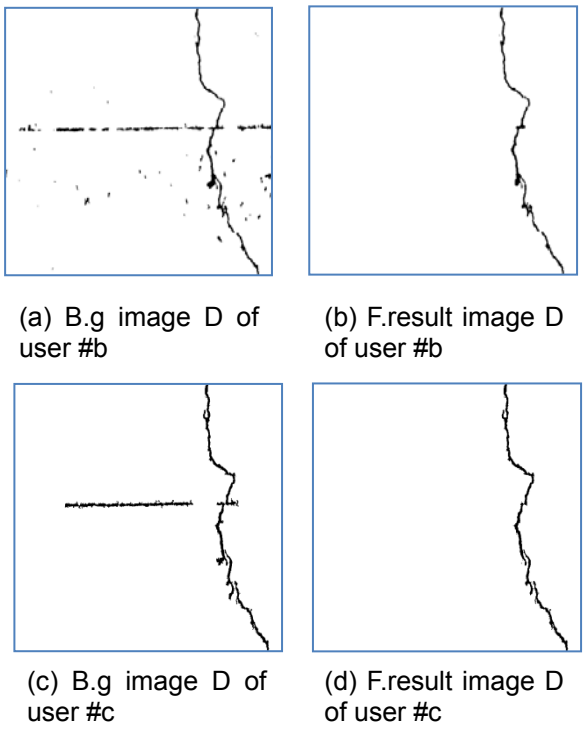
$f_1$, $f_2$ : loss rate and noise rate, respectively.

$f$: evaluation value. The higher value is calculated, the higher accuracy of the software is obtained.

Figs.14 and 15 show Loss rate and Noise rate of the four original images, their upside down images, and left side right images. The horizontal axis of the plot shows test number of the three users. Namely, user #a has test number (a1, a2, a3), user #b has test number (b1, b2, b3) and user #c has test number (c1, c2, c3).

As a result, with Figs.14 and 15, user #b and user #c have relatively equivalent loss rate and noise rate. The loss rate of user #a is higher than the loss rate of the others, and the noise rate of user #a is lower than the noise rate of the others. User #a has loss rate tolerance. However, user #b and user #c have noise tolerance. In Fig.15, the noise rate of final result image of each original image is almost invariance for the three tests of each user. It is evident that the direction change of the original image does not make influence to the output results.

In addition, in the aspect of crack extraction, images A and C have the loss rate more than images B and D. In contrast, the noise rate of image B has the highest of the four images. The loss and noise rate values of image D have the lowest of the four images. It is concluded that the order of the crack extraction of



(a) B.g image D of user #b

(b) F.result image D of user #b

(c) B.g image D of user #c

(d) F.result image D of user #c

Fig.16 crack extraction results

images are from the easiest as D, following B, A and C images.

Fig.16 also shows an example of crack extraction results of users #b and #c for image D. In the background image, the noise pixels are appeared. However, in the final result, the noise pixels are reduced significantly because the user used "Extraction" and "Eraser" buttons which are shown Fig.10. In the other hand, the background images of the two users are different because they refer to the different optimal parameter values. Therefore, with the same way of the crack extraction, the both of two users have similar final result images.

Furthermore, Table 4 shows that all users achieve the high evaluation values. The average evaluation value of user #c is 0.98. This value is the highest evaluation value of three users. Meanwhile, the average evaluation value of user #b is 0.97 higher than the average evaluation value of user #a. The average evaluation value of user #a is 0.92.

As a result, with the two criteria, it is concluded that user #c achieves the best skill of using the software, and user #b gains the skill of software usage better than user #a.

## 5. CONCLUSIONS

With the two evaluation criteria, the experiment results indicated that the proposed method has achieved reasonable accuracy and working time. Furthermore, in this experiment, the average working time for the extracting crack of each image reduced about 87%. If the crack extraction is performed by the proposed software for a thousand of images, it will save a great deal of time.

Table 4 Average Evaluation value of three users

| User | Value Evaluation value (f) | | |
|---|---|---|---|
| | Max | Min | Ave |
| User a | 0.96 | 0.89 | 0.92 |
| User b | 0.99 | 0.94 | 0.97 |
| User c | 0.99 | 0.97 | 0.98 |

This paper proposed a robust method for crack semi-automatic detection from concrete surface images. Moreover, the prototype software is developed based on iGA with the image processing technique. Especially, using a touch monitor, the easiness and efficiency of the proposed crack detection software is remarkable.

However, the loss pixels still remain with a high rate, this software needs to will be improved further in the future. Furthermore, the size of the extracted cracks (width, length, orientation) also would be measured automatically from this proposed software.

## ACKNOWLEDGEMENT

This work was supported by JSPS KAKENHI Grant Number 15K06180 and the scholarship of Ministry of Construction Vietnam.

## REFERENCES

[1] H.Ba, N.Yata and T. Nagao, "Automatic Finding of Optimal Image Processing for Extracting Concrete Image Cracks Using Features ACTIT" IEEJ Transactions On Electrical And Electronic Engineering, IEEJ Trans 7, 2012, pp.308–315.

[2] T. Nishikawa, J.Yoshida, T.Sugiyama, Y.Fujino "Concrete crack detection by multiple sequential image filtering", computer-Aided civil and infrastructure engineering 27, 2012, pp 29-47.K.

[3] Kawamura, A. Miyamoto, H.Nakamura, R. Sato: "Proposal of a crack pattern extraction method from digital images using an interactive genetic algorithm" Proc. Japan Soc. Civ. Eng. Vol.742, 2003, pp. 115-131.

[4] K. Kawamura, K.YOSHINO, A. Tarighat and H. Nakamura "A valid parameter range identification method of a digital image processing algorithm for concrete surface cracks detection using genetic Algorithm and decision tree", JSCE, Vol.69,(2013) No2 p.I_13-I_23.

[5] H. Farooq, M.T. Siddique, "A Comparative Study On User Interfaces Of Interactive Genetic Algorithm", procedia computer Science, Vol.32, 2014, pp.45-52.