Photo Defect Detection for Image Inpainting

Rong-Chi Chang[1], Yun-Long Sie[2], Su-Mei Chou[2] and Timothy K. Shih[2]
[1]Department of Information and Design, Asia University, Taichung, Taiwan, R.O.C.
[2]Multimedia Information Networking Lab.
Department of Computer Science and Information Engineering, Tamkang University, Taiwan, R.O.C.
E-mail: roger@asia.edu.tw[1], tshih@cs.tku.edu.tw[2]

Abstract

Image inpainting (or image completion) techniques use textural or structural information to repair or fill damaged potions of a picture. However, most techniques request a human to identify the portion to be inpainted. We developed a new mechanism which can automatically detect defect portions in a photo, including damages by color ink spray and scratch drawing. The mechanism is based on several filters and structural information of damages. Old photos from the author's family are used for testing. Preliminary results show that most damages can be automatically detected without human involvement. The mechanism is integrated with our inpainting algorithms to complete a fully automatic photo defects repairing system.

Key words: *image inpainting, defect detection, structural features, image processing, image restoration*

1. Introduction

The discussion of image and video inpainting were found in the literature. In most image inpainting projects, users have to select the defect area. On the other hand, since it is impossible to select defects on all video frames, video inpainting uses motion estimation, shape features, and Kokaram's model [1] to detect defects such as line scratch and spike on video. Unfortunately, motion estimation and Kokaram's models can not be used in defect detection of a single photo image. Thus, we propose a naive but effective algorithm to detect photo defects. The detection of defects is similar but different from color image segmentation [5] to find fore ground objects. The detection is also different from finding defects in random color textures [4], using color cluster scheme. Basically, ink sprayed on image will destroy the homogeneity of a photo. Thus, the detection mechanism relies on the property of ink (e.g., ink color, shape, and decadence of gradient). Our mechanism can further distinguish the two types of damages (ink spray and scratch) in different ink colors. The proposed algorithms contain several filters, which will be discussed in the next section. The experimental results include more than 40 damaged photos. Some old photos are collected from the author's family history, with real cases of damages. A few photos are new photos with real ink spray and scratches scanned for testing.

2. The defect detection algorithm

Given an input defect picture, our goal is to detect defects in the test image and to remark the defects area with specially color. Our algorithm consists of two main steps: (a) defect intensity detector; (b) adaptive defect filter (to recover miss-detected areas). The procedure is illustrated in Figure 1. We explain each building block of our strategy in different sections below.

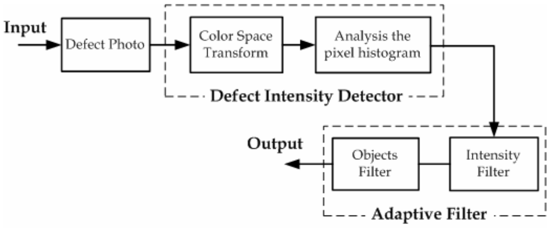


Figure 1: The Defect Photo Detection Procedure

2.1 *The Defect Intensity Detector*

We propose an algorithm which automatically adjusts threshold values to achieve the best results of detection. We use HSI color space since it is closer to human perception. It has been discussed that RGB color space is not quite suitable for comparing image colors due to the discontinuity of color values. Our experiments on different color spaces also reveal that

Proceedings of the Seventh IEEE International Symposium on Multimedia (ISM'05)
0-7695-2489-3/05 $20.00 © 2005 IEEE

HSI color space can be used in several successful examples.

In this paper, we aim at handling the ink spray defects, with an extension to detect scratch by color pens. There are plenty type of ink colors, and photo images usually contains lots of different colors. Hence, if we try to detect ink with color information, it would be almost impossible. We consider the intensity and shape of photo defects. Though, it is almost not possible to discriminate ink regions from objects in photo images. The intensity variation of ink regions is usually quite smooth and steady while comparing with objects in photos. Thus, we use the HSI color space and use intensity in the first filter. We decrease the value of *I* (i.e., intensity) from 255 to 0 at each step. In each step, we record the number of pixels detected. It has been mentioned that the intensity variation of ink is usually low. In decreasing the intensity of a photo, the ink part will remain while other parts will be filtered, up to an estimated *I* value. In the second filter, we compute a pixel histogram, with pixel numbers on y-axis and adjustment of *I* values on x-axis. That means we record the number of pixels been detected in each *I* variation. The histogram is represented as a curve chart in our tool (see Figure 2). We calculate the variance of pixel number been detected between two different steps of adjustment. If the variance is low, it means that pixels been detected in the last adjustment are not much affected by the present adjustment of *I*. And the pixels been detected are possible to be pixels of ink spray because of its steady intensity. Keeping on the calculation of variance of pixels been detected, a collection consists of consecutive adjustments of *I* can be constructed. The collection needs to be analyzed since sometimes the variance of pixels been detected is low only because it has low discrimination in a step of *I* adjustment. Then the collection been record has no use and we can't detect the ink. Thus, if the collection of adjustment contains too many passes of adjustment, we just leave it and go on to analysis the photo image. After the calculation and analysis, a set of adjustment of *I* can be found out to separate pixels of ink from objects in the photo. The following is our algorithm to find the possible minimal intensity of a defect area.

Algorithm* 1: *Defect Intensity Detector

```
Let $\lambda$ be a maximum length of a continuous curve ($\lambda \geq 5$)
Let $\boldsymbol{\alpha}$ be a threshold of difference of pixel number (initially, $\boldsymbol{\alpha} = 100$)
Let $n$ be a value of pixel number ($n = 100$)
```

```
Input: Image $C$
  Output: Intensity $\boldsymbol{\delta}$
  Algorithm:
  Step1: Convert image $C$ to HSI color space
  Step2: Compute the number of pixels, w. r. t.
         each $I$ variation; Let $\#P_i$ represents the
         number of pixels in the $i^{th}$ $I$ variation,
         where $1 \leq i \leq 255$
  Step3: Compute $\Delta P_i = (\#P_i - \#P_{i-1})$, if $\Delta P_i < \boldsymbol{\alpha}$
         store $i$ in an index list $L$
  Step4: Search $L$ and store the consecutive
         elements in set $S$, $S_i = \{\boldsymbol{\delta}, M_i\} \in S$; where
         $\boldsymbol{\delta}$ is the start intensity of consecutive
         elements, and $M_i$ is the length of the $i^{th}$
         consecutive elements
  Step5: Find $S_i$ with the maximal length $M_i$ in the
         set $S$, where the consecutive elements
         have $M_i$ less than or equal to $\boldsymbol{\lambda}$
  Step6: If no $M_i < \boldsymbol{\lambda}$ then $\boldsymbol{\alpha}=\boldsymbol{\alpha}+n$; goto Step3
         else Return $\boldsymbol{\delta}$
```

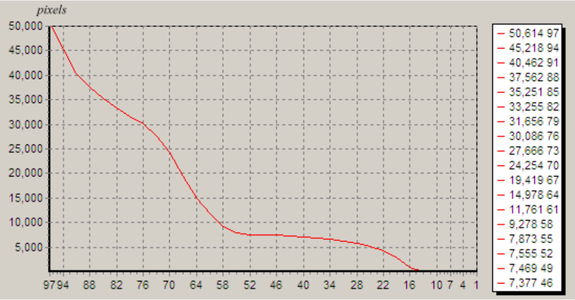


Figure 2: Histogram of Pixel Numbers by Intensity

2.2 *The Adaptive Filter*

The *defect intensity detector* operation is not ready to separate defect object and background object yet. We use an *adaptive filter,* which considers the details of object properties, and decide which part is defect. The intensity found in the defect intensity detector can be used on finding the potential defect objects of a photo. The *adaptive filter* will detect multiple objects. Each object O_j in an object set O has two parameters, A_j and B_j, to represent object size and object boundary, respectively. We define a threshold of object size, $\boldsymbol{\mu}$. If an object size is small than the $\boldsymbol{\mu}$ then we remove the object from set O. Two morphological operations, *opening* and *closing*, are then used. The *median filter* operation with 3×3 filter windows performs isolate object detection and elimination.

In the *adaptive filter* algorithm, the adjustment of I is achieved by decreasing the value of I from $\boldsymbol{\delta}$ to carry out the defect object to be detect by an intensity value of 5 at each step. In each step of adjustment, we record the size and locations of object O_j detected in a new set Q. The parameter n is a number of the objects in set Q. The new set Q is for intensity iteration while the original set O is used as a reference. In intensity iteration, an object can be split. The size of object (includes split objects) can be reduced. However, the center of the object is not likely to be altered. To tell whether split objects belong to an original object, we compare the center of split object against the boundary of the original object. It is important to keep track of which split object belong to an original object since we calculate the decreasing rate of area. If the decreasing rate is high, the object is not likely to be a defect. After several intensity iterations, the object left in Q are defects. We mark the original defect objects in O on the photo. The *Adaptive Filter* algorithm follows.

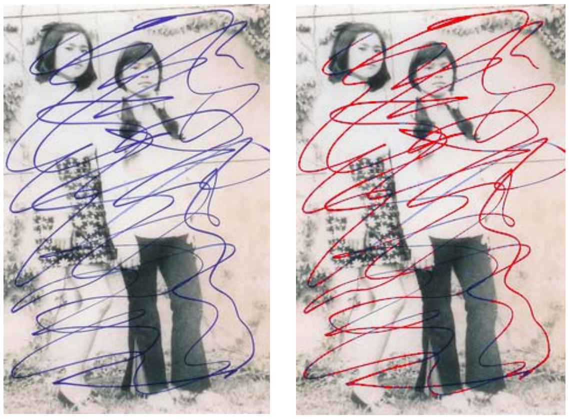
Algorithm* 2: *Adaptive Filter

```
Let $O$ be a set of objects; $O_j = \{A_j, B_j\} \in O$, where
   $A_j$ and $B_j$ are the area size and boundary of
   object $O_j$, respectively
Let $\Omega$ be a percentage threshold of object size ($\Omega$
   = 50%)
Let $\boldsymbol{\mu}$ be a threshold of number of pixels ($\boldsymbol{\mu}$= 5 or
   10)
Let $\varepsilon$ be a decreasing intensity amount ($\boldsymbol{\varepsilon}$ = 10)
Algorithm:
Step1: Store initial objects obtained from color
   segmentation in set $O$
Step2: Remove small objects form set $O$ if the
   area of object size $A_j < \boldsymbol{\mu}$
Step3: Use filter Opening(Closing(C)) to remove
   isolated objects in $O$
Step4: Copy set $O$ to a new set $Q$; Let $\boldsymbol{n}$ be the
   number of objects in $Q$; Let $s$ = $\boldsymbol{\delta}$ be an initial
   intensity value (from the defect intensity
   detector)
 Step5: Intensity Iteration: Repeat till $\boldsymbol{n}$ has no
        change
        {
          For all object $O_i$ in $Q$ do
 If $A_i$ decreases > $\Omega$ from last Intensity Iteration
   and $O_i$ is not split then
 remove $O_i$ from $Q$ and update $\boldsymbol{n}$
          Set $s = s - \boldsymbol{\varepsilon}$
 }
 Step6:  For each element $O_j$ in set $O$ not
        removed in $Q$
 Mark $O_j$ in image $C$
```

3. Experiments results and discussion

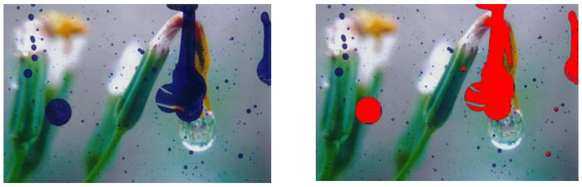
In this section, we present the experimental results from a variety of tests. We divide the photos into three categories, with damages by spray of color ink (blue, red or black), scratch by ink pen and scratch line. Examples of defect pictures and detections are presented in Figure 3 to 8. A simple example of scratching defect is shows in Figure 3. Scratch lines are hard to detect, especially when the line is very thin. In Figure 3, we take an alternative strategy, which use gray levels of image. The gray value detected is 128.

Figure 4 shows a defect photo by blue ink. In this experiment, we set the threshold of object size μ to 10 pixels. Therefore, there are some small objects not detected. Figure 5 has defects by black ink. The threshold of object size μ is also 10 in this case. Note that, the strings in black of Figure 5(a) is not changed mostly.



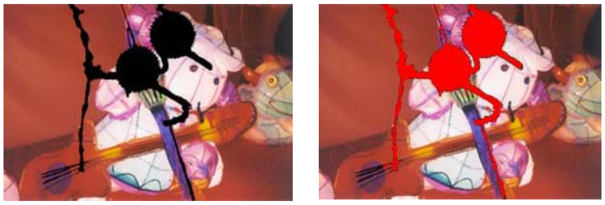
(a) The original picture **(b) The detection result**

Figure 3: Defect Detection with Ink Pen Scratch (some thin line scratches are not detected)



(a) The defect photo **(b) The detection result**

Figure 4: Detection of Defect with Blue Ink Spray

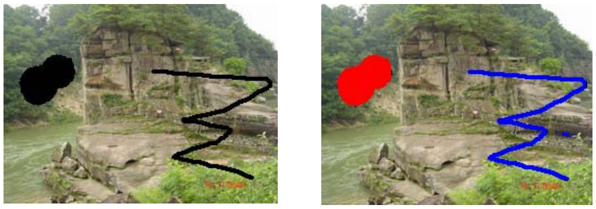


(a) The defect photo **(b) The detection result**

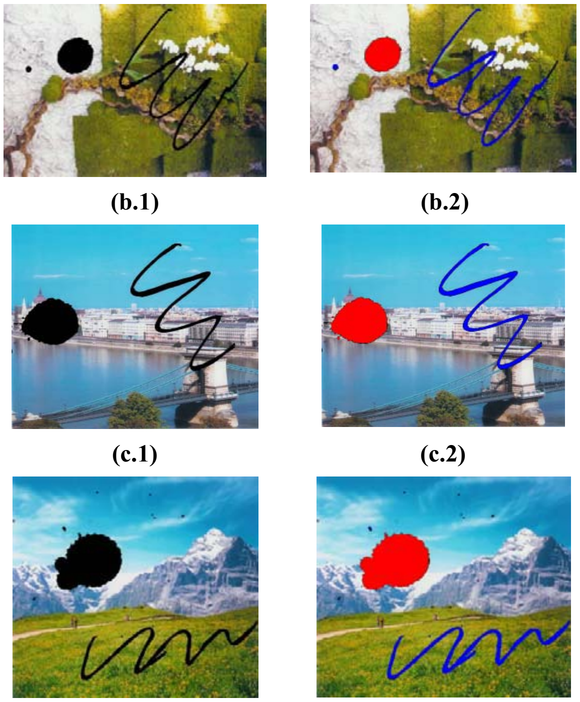
Figure 5: Detection of Defect with Black Ink Spray

We also consider the type of defects. A naive but effective strategy is to compute the ratio of area to perimeter on defect objects. We use a line detection algorithm to find the approximation of pixel numbers of perimeters. Roughly speaking, ink sprays will have a ratio higher than line scratches due to shape. In Figure 6, we illustrate four sets of photos with type of defects separated. The first row shows the test images and the second row presents the result of detection. We found that the ratio of ink spray is usually higher than 1 and the ratio of line scratch is close to 1. However, small ink spray with a diameter close to line thickness is detected as a line. A threshold set to 2.0 is used to separate the two types of defects, using red to show ink spray and blue to show line scratch. Table 1 shows the ratios of defect objects in Figure 6. Each photo contains from 2 to 3 defect objects. The numbers of pixels in object area (i.e., O_A) and object perimeter (i.e., O_P) are also provided.

We also detect multiple ink colors in the same photo. The strategy detects two different intensity values. The values detected for blue/black are 50/37 (Fig. 7a), 50/40 (Fig. 7b), and 60/22 (Fig. 7c). As shown in Figure 7, the black ink detected is shown in red and the blue ink detected is shown in yellow. The output of the scratch lines detection was shown in Figure 8. The values detected for scratch lines/ink are 213/16 (Fig. 8b). Table 2 summarizes parameters used in our algorithm for Figures 3 to 8.



(a.1) **(a.2)**

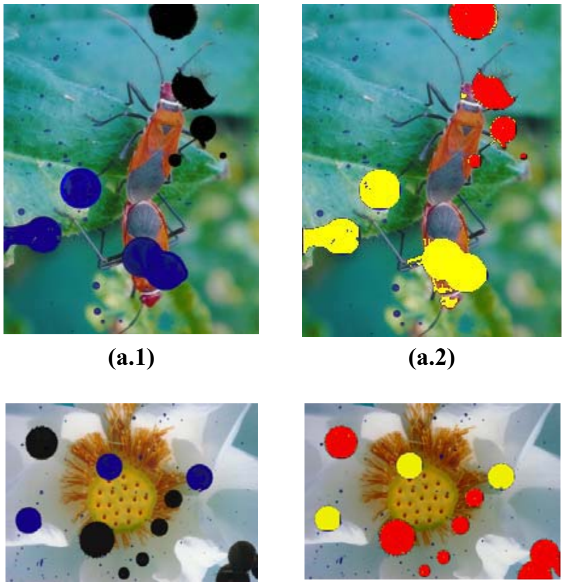


(b.1) **(b.2)**

(c.1) **(c.2)**

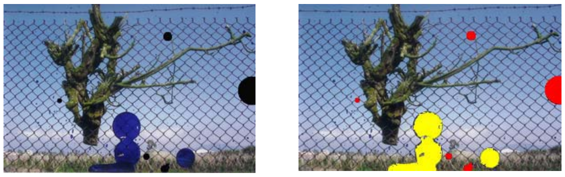
(d.1) **(d.2)**

Figure 3: Result of Separation of Defect Types. Each pair shows the original image in the left side and the result of the defect detection in the right side.



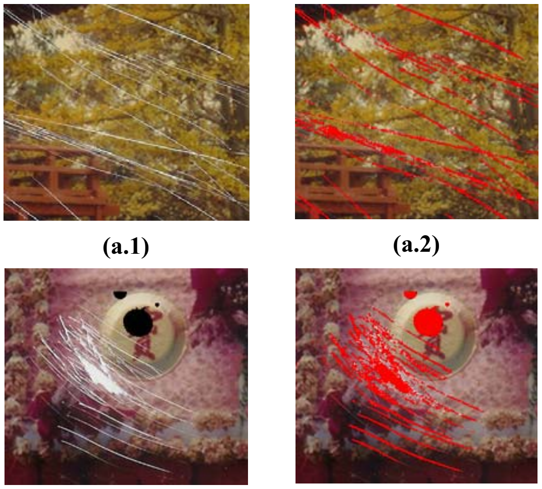
(a.1) **(a.2)**

(b.1) **(b.2)**



(c.1) (c.2)

Figure 4: Result of Separation of Defect Ink Colors Each pair shows the original image in the left side and the result of the defect detection in the right side.



(a.1) (a.2)

(b.1) (b.2)

Figure 5: Result of Defect Detection with Scratch Lines. Each pair shows the original image in the left side and the result of the defect detection in the right side.

4. Conclusions

Digital image inpainting algorithms usually rely on user to select defect areas to be inpainted. We propose a naive but effective strategy to detect ink spray and line scratch with different ink colors. We found that, on different type of photographic papers, ink spray results in different types of intensity detection. Also, thickness of ink results differently. In addition, if scratch lines are very thin, it is hard to detect. In such case, we may subdivide the photo into multiple segments for separated detection. Our future research will detect other types of defects, such as ripped and frazzled photos.

Table 1: Ratio of Pixel Numbers

Fig.6	(a)		(b)			(c)		(d)		
O_{item}	#1	#2	#1	#2	#3	#1	#2	#1	#2	#3
O_A	2252	2013	1915	2029	48	1636	2902	19	6452	2958
O_P	411	1953	2744	354	48	2085	479	28	733	2749
Ratio	5.47	1.03	0.69	5.73	1	0.78	6.05	0.67	8.8	1.07

Table 2: Parameters for Defect Detection

Figures	δ	α	μ	Gray value
3		100	5	128
4	46	300	10	
5	37	100	10	
6(a)	43	100	10	
6(b)	19	100	10	
6(c)	28	100	10	
6(d)	26	100	10	
7(a)	50/37	200	10	
7(b)	50/40	100	10	
7(c)	60/22	200	10	
8(a)	225	200	50	
8(b)	213/16	200	50	

References

[1] Vittoria Bruni and Domenico Vitulano, “A generalized model for scratch detection,” IEEE Transactions on Image Processing, 13(1), January 2004, pp. 44-50.

[2] L. Joyeux, O. Buisson, B. Besserer, S. Boukir, “Detection and removal of line scratches in motion picture films,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition, June 23-25, 1999, pp. 548-553.

[3] A. Machi, F. Collura, “Accurate spatio-temporal restoration of compact single frame defects in aged motion picture,” in Proceedings on 12th International Conference on Image Analysis and Processing, 17-19 Sept. 2003, pp. 454-459.

[4] K. Y. Song, J. Kittler and M. Petrou, “Defect detection in random colour textures,” Image and Vision Computing, 14(9), October 1996, pp. 667-683.

[5] Swee-Seong Wong and Wee Kheng Leow, “Color segmentation and figure-ground segregation of natural images,” in Proceedings on International Conference on Image Processing, ICIP’02, 2002, pp. 120-123.