Comprehensive Solutions for Removal of Dust and Scratches from Images

Ruth Bergman, Ron Maurer, Hila Nachlieli, Gitit Ruckenstein, Patrick Chase, Darryl Greig
HP Laboratories Israel

HPL-2007-20
February 6, 2007*

dust and scratch removal, defect detection

Dust, scratches or hair on originals (prints, slides or negatives) distinctly appear as light or dark artifacts on a scan. These unsightly artifacts have become a major consumer concern. There are several scenarios for removal of dust and scratch artifacts. One scenario is during acquisition, e.g., while scanning a photographic media. Another is artifact removal from a digital image in an image editor. For each scenario, a different solution is suitable, with different performance requirements and differing levels of user interaction. This report describes a comprehensive set of algorithms for the removal of dust and scratches from images. Our algorithms solve a wide range of use scenarios.

A dust and scratch removal solution has two steps: a detection step and a reconstruction step. Very good detection of dust and scratches is possible using side information, such as provided by dedicated hardware. Without hardware assistance, dust and scratch removal algorithms generally resort to blurring, thereby, losing image detail. We present algorithmic alternatives for dust and scratch detection. In addition we present reconstruction algorithms, that preserve image detail better than previously available alternatives. These algorithms consistently produce visually pleasing images in extensive testing.

Comprehensive Solutions for Removal of Dust and Scratches from Images

Ruth Bergman, Ron Maurer, Hila Nachlieli, Gitit Ruckenstein, Patrick Chase, Darryl Greig[*]
HP Laboratories

Abstract

Dust, scratches or hair on originals (prints, slides or negatives) distinctly appear as light or dark artifacts on a scan. These unsightly artifacts have become a major consumer concern. There are several scenarios for removal of dust and scratch artifacts. One scenario is during acquisition, e.g., while scanning a photographic media. Another is artifact removal from a digital image in an image editor. For each scenario, a different solution is suitable, with different performance requirements and differing levels of user interaction. This report describes a comprehensive set of algorithms for the removal of dust and scratches from images. Our algorithms solve a wide range of use scenarios.

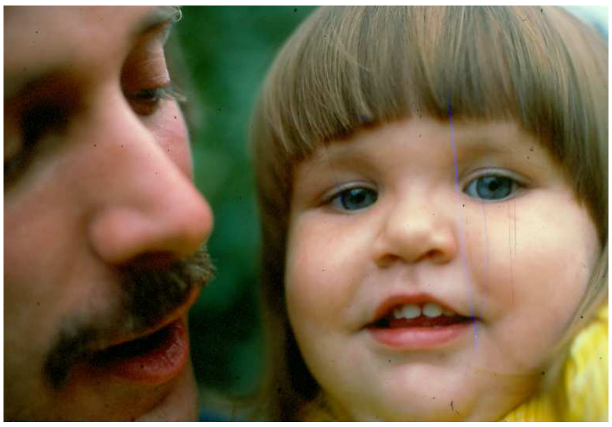
A dust and scratch removal solution has two steps: a detection step and a reconstruction step. Very good detection of dust and scratches is possible using side information, such as provided by dedicated hardware. Without hardware assistance, dust and scratch removal algorithms generally resort to blurring, thereby, losing image detail. We present algorithmic alternatives for dust and scratch detection. In addition we present reconstruction algorithms, that preserve image detail better than previously available alternatives. These algorithms consistently produce visually pleasing images in extensive testing.

1 Introduction

The ubiquitous acceptance of digital imaging is motivating many photography enthusiasts to transfer their photographic archive to digital form. Scans of negatives and positives (slides) are preferred since these transmissive media have a better range of tones than reflective prints. However, negatives and positives are small and must be scanned at high resolution to view on a monitor or reproduce in print. The high resolution scan of the image also makes small dust specks and scratches very apparent. These unsightly defects have become an important issue for consumers. Algorithms for removing dust and scratches vary in several aspects: speed and memory performance, utilization of side information, if any, and the image quality of the repaired image. Dust and scratch removal solutions available today both in scanners and image editors trade-off defect removal with loss of image details. Typically, the results are too blurry. This paper presents algorithms that improve the possible trade-off. For every dust and scratch removal scenario, we propose a solution that removes more defects and produces better image quality than available alternatives.

The image in Figure 1(a) contains severe dust and scratch defects. In the image in Figure 2(a) the defects are less severe, but still very disturbing. This image is more representative of a typical

[*]E-mail: {ruth.bergman, hila.nachlieli, gitit.ruckenstein}@hp.com


(a) An image with dust and scratch defects

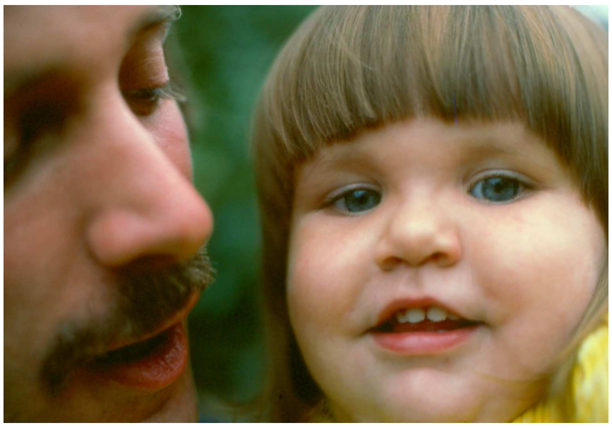
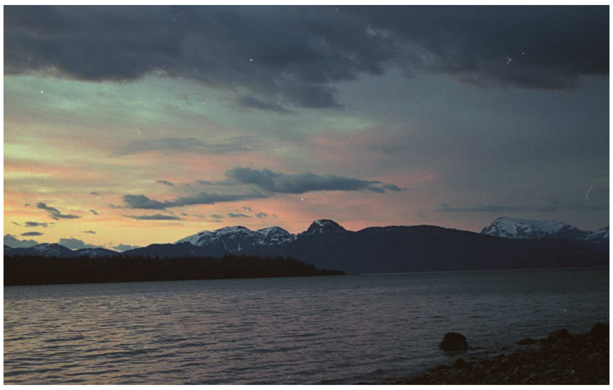

(b) Cleaned image

Figure 1: Example of dust and scratch removal.


(a) An image with dust defects

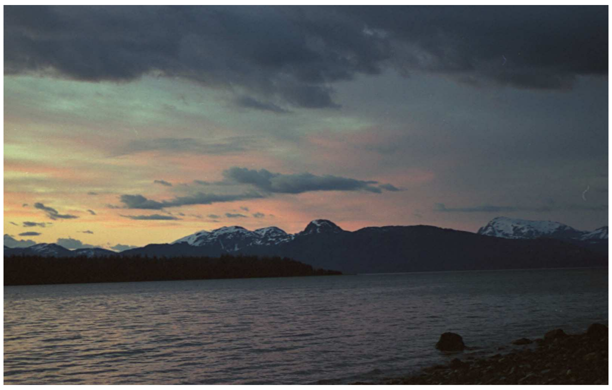

(b) Cleaned image

Figure 2: Example of dust removal in a typical consumer image.

consumer image. Dust is found in digital images of all types. Scratches exist primarily on images scanned from slides and negatives. The work described in this paper does not target tears that are often found on old photographs. Dust and scratches appear as light or dark artifacts in the image. Dust defects can appear as small specks in the image, and sometimes look narrow and long. Scratches are narrow and usually very long and straight. They tend to have very low contrast with the background. Dust typically has higher contrast making it easier to detect.

We are able to offer a wide range of algorithmic solutions for dust and scratch removal applications. We describe four tiers of solutions. Tier 1 addresses a real-time use case, for example in a capture device, with no specialized hardware. It is an efficient local software algorithm, and has low memory utilization. It produces more overall image blur than the higher tiers. Tier 2 addresses off-line or near real-time use cases, such as an image editor or post-capture processing. It, therefore, allows more processing time and higher memory utilization. It utilizes image analysis techniques on larger image regions to refine the detection of defects thereby avoiding overall image blur. Tier 3 is intended for the discriminating user who is willing to invest some manual intervention and longer running time to obtain very high image quality. It is also a software solution, like Tier 1 and 2, but utilizes state of the art image infilling algorithms for reconstruction. Tier 4 addresses both the real-time and the off-line scenarios and corresponds to the special case where hardware-assisted detection is available. Combining this side information with the efficient local reconstruction algorithm of Tier 1 attains very high image quality. At every tier our algorithms successfully reduce dust and scratch artifacts and produce superior image quality than competing algorithms [19, 2].

The removal of dust and scratches proceeds in two steps: a detection step and a reconstruction step. The detection step identifies defective pixels and the reconstruction step replaces the defective pixels.

The challenge for detection is that defects have to be found, while image features should not be detected erroneously. We divide detection algorithms according to several aspects

- Algorithms that use **side information**. Side information can come in the form of, e.g., user-selection in an image editor, or an infra-red (IR) image in a scanner. This information is the best way to avoid false detections.

- Algorithms that make a **hard** classification of pixels versus algorithms that make a **soft** decision. Hard classification creates a binary *defect map* where each pixel is labelled as *defective* or *not defective*. Soft classification attributes a *credibility* value to each pixel. A defect map provides a convenient representation of defects, but can lead to artifacts due to missed defects and false detections. When using soft classification, artifacts are usually less visible in the resulting repair.

- Algorithms that operate on a **local** neighborhood versus **non-local** algorithms that use large image regions. Local algorithms have the advantage in performance. However, they cannot distinguish defects from image features that have similar characteristics, without non-local context (unless they use side information).

This division defines $2 \times 2 \times 2 = 8$ categories of detection algorithms. Section 3 describes four algorithms, each of which fits into a different category. In particular, the algorithm in Subsection 3.4 uses an IR image and computes a credibility map, while the other algorithms detect from the input image. Our approach, like previous work[19], is to generate a detail-less image in which the defects are "erased". Our local detection algorithms compare several properties between the original and the detail-less image which results in detection of defects but not of image edges. Local measures, however, cannot avoid detection of image details that have similar characteristics to dust and scratches. We, therefore, examine regions in the defect map and the corresponding image regions to refine the classification. In this stage we may modify the classification of pixels that do or do not seem to create defects with plausible characteristics together with other pixels in their regions.

The reconstruction step should replace the defective pixels resulting in an image with no visible defects. We divide reconstruction algorithms to similar categories

- Algorithms that use a **hard** detection map (i.e., *defect map*) versus algorithms that use a **soft** detection map (i.e., a *credibility map*). The values in a defect map are binary, whereas a credibility map contains values in the range $[0, 1]$. Reconstruction algorithms that use the rich information contained in a credibility map are more complex, but have the potential to generate better image quality.

- Algorithms that operate on a **local** neighborhood versus algorithms that use **non-local** image regions. While local algorithms are fast, they are only able to perform a smoothing type of operation. Non-local algorithms are slow, but result in better image quality. In particular, good restoration of texture is only possible with non-local information.

To reconstruct based on a defect map the algorithms have to estimate the original pixel value for each defective pixel. The algorithms ignore the values of pixels labelled as defective. We suggest both a *local* approach and a *non-local* approach. Our local algorithm, referred to as the directional reconstruction, uses the local directionality to decide how defective pixels should be set. The non-local algorithm uses ideas from texture synthesis, in particular [10]. The latter algorithm is better able to mimic the existing image texture, while the directional algorithm reconstructs lines better. The directional algorithm is significantly faster and is also better at re-building image features that are erroneously detected as defects. Subsections 4.2 and 4.3 describe the local and non-local algorithms, respectively.

For reconstruction based on a soft classification, we developed a *local* filter that extends the bilateral filter [21]. While the bilateral filter makes a soft classification of its neighbors based on the gray-level difference of each neighbor with the central pixel and location relative to the central pixel, our new algorithm modifies the weighting scheme to include the soft credibility map computed in the detection step. This extension, the credibility-weighted bilateral filter is able to remove non-Gaussian noise, such as dust and scratch artifacts, as well as Gaussian noise simultaneously. This algorithm is presented in Section 4.1.

The rest of the paper is organized as follows. In Section 2, we describe related work on the removal of defects, particularly dust and scratches. The following two sections describe the detection algorithms (Section 3) and reconstruction algorithms (Section 4). Section 5 describes solutions for a variety of applications. Figures 1(b) and 2(b) preview some results of dust and scratch removal. In Section 6 we discuss directions for future work, and Section 7 summarizes our conclusions.

2 Related Work

Let us first consider the options that a user has for removing dust and scratch defects from a digital image. Adobe Photoshop [1], a standard image editing program, offers several options. One option is a dust and scratch filter that appears under the "Filter/Noise" menu. This operation blurs the whole image or a selected region. It does not do any form of detection. If applied to the whole image it results in a blurry image. The user can apply the filter regionally by selecting regions with defects, but this requires considerable manual effort. If a defect occurs in a textured area the filter will blur the texture. Another option in Photoshop would be to use the *stamp tool* to remove defects. This tool can repair defects in textured areas. Using this tool, however, is very time consuming and requires some practice.

This paper describes algorithms to replace the manual operation described above. The problem of dust and scratch removal is usually addressed as two distinct tasks. The first is the detection of the dust and scratch regions, in other words, defect detection. The second is the reconstruction of those defective regions.

2.1 Defect detection

We draw a strong distinction between solutions that use specialized hardware to detect dust and scratch in scanners and those that approach the problem algorithmically.

2.1.1 Hardware detection

Most scanners, particularly high-end scanners, have special purpose hardware to assist with the detection of dust and scratches on the original. Defects in transmissive material (negatives and positives) are reliably detected with an infra-red scan. Unlike visible light, infra-red light is not blocked by the colors in the image, but it is blocked by the opaque dust. The infra-red light is scattered differently in scratched areas, as compared with non-defective areas. Applied Science Fiction first introduced the use of infra-red hardware assisted detection for film in 2000 [2]. Figure 5(b) shows the results of an infra-red scan for a slide with dust and scratch defects. While the infra-red channel has a shadow of the image, the defects show clearly.

Infra-red hardware is not useful for scans of prints (reflective scans). One hardware assisted approach for reflective scans is offered by Digital ICE [2]. This solution involves scanning the print twice with lighting from two different directions. The shadow cast by a spec of dust should differ between the two images enabling detection. In practice, this approach does not lead to reliable detection. The algorithms used for the interpretation of the scans are proprietary in most cases. This approach is, however, reminiscent of the multiple-lighting approach used to model objects with polynomial texture maps [20]. In future work we intend to apply this modelling approach to the problem of detection from two (or more) scans of the print.

2.1.2 Software detection

Without the assistance of special purpose hardware detection must proceed with the limited information contained in a single digital image. Detection relies on characteristics of dust and scratch defects in the digital image [19, 3]. For example, the defects are assumed to be narrow and to have a significant difference in contrast compared with their background [19]. This approach uses local operations for detection, which cannot differentiate between dust and image features with similar characteristics. Hence, this approach suffers from false detections. Subsections 3.2 and 3.3 describe our detection algorithms which extends this approach, and has fewer false detections.

An alternate approach to detection is defect specific. For example, [9] detects straight or nearly straight anomalies in an image. Scratches appear as anomalies of this type in scans of negatives and slides.

All the defect detection approaches we have seen to date take a hard decision, i.e., they classify pixels as good or defective. We know of no prior work that uses a soft decision approach.

2.2 Reconstruction of defective pixels

The correction of defective pixels can be viewed as an image reconstruction problem. In this problem, the pixel being repaired as well as much of its neighboring pixels may be defective, i.e., their values are only loosely related to the respective original values. The values at these pixels should thus be ignored by the reconstruction algorithm.

At this stage of the solution, we assume that the defects are already detected. Simple reconstruction algorithms, such as the one in [19], merely smooth the defect regions, for example, using a median filter or another form of averaging. The median reconstruction is computationally very efficient, but is not very good from an image quality perspective. Figure 3(d) shows an example of median correction. The defective pixels were replaced, but the large scratch is still visible because the correction is too smooth, and does not match the texture in that area.

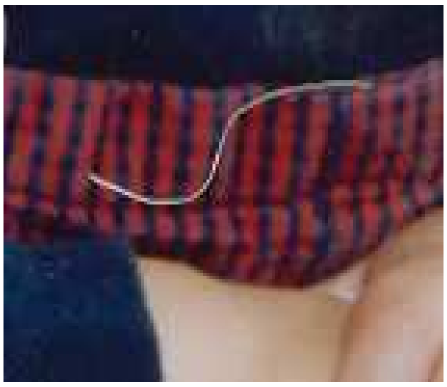
One of our local reconstruction algorithms extends the bilateral filter [21], a noise reduction filter. When the noise has a Gaussian distribution the bilateral filter distinguishes between noise and features; it removes noise and retains the features. When the noise has Gaussian distribution with a known variance the bilateral filter separates noise, which has relatively low local contrast, from features, which have high local contrast. When the distortion in the image does not follow a Gaussian distribution, the bilateral filter may interpret large local difference as image features rather than defects which should be smoothed. Dust and scratch defects would, unfortunately, be enhanced by the bilateral filter. In Subsection 4.1, we extend this filter to correctly clean images with non-Gaussian distortions using soft credibility values.

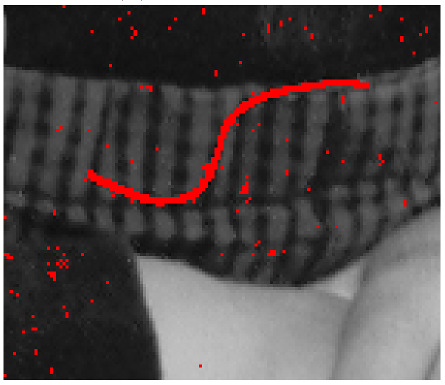
Two approaches to image restoration of missing regions of an image in the literature are image inpainting and texture synthesis. Algorithms for both approaches are computationally slow compared with typical local averaging algorithms.

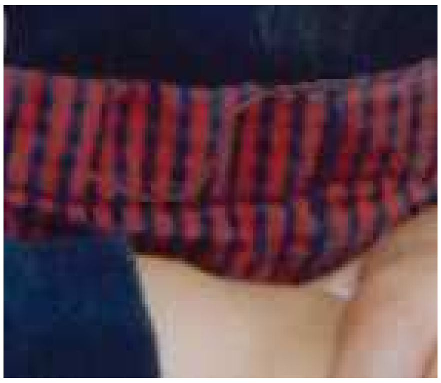
Texture synthesis algorithms are concerned with producing a large texture image from a small sample of a texture. The resulting texture image should appear to arise from the same texture to a human observer. The texture should not appear to be duplicated or artificial. Many texture synthesis algorithms have been developed since the 1970's. One approach is to simulate the physical generation of the texture [15]. This approach is not realistic for most textures. Another approach uses statistical models of texture such as Markov random fields [8, 10, 24] and marginal statistics stored in a co-occurrence matrix [17, 7]. More recently, multi-scale feature extraction methods that rely upon wavelet decomposition have been used to define texture models [6, 18, 25]. A non-parametric approach, introduced in [10], uses simplistic operations to produce visually superior results in many cases, particularly for highly structured texture. We elaborate on this method in sub-section 4.

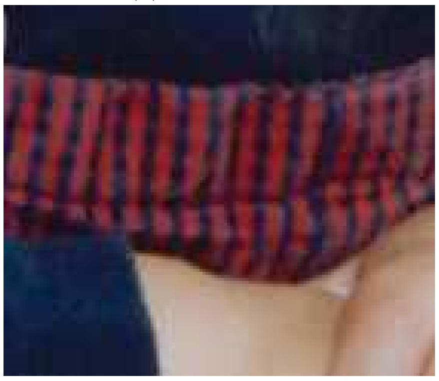
Image inpainting [16] attempts to fill in a defective region in the image in a natural way. The pixels in the defective image are treated as missing. The algorithm iteratively solves some partial differential equations that smoothly propagate the information surrounding the missing region to preserve the gradients apparent in the boundary. The original inpainting algorithm [16] repairs the image structure quite well, but tends to smooth away texture. In more recent work[5], the authors introduce a solution for both structure and texture inpainting. This solution decomposes the image into a structural component and a texture component and works on each component separately. Likewise, the texture synthesis approach of [10] has been used to fill in holes in images [4]. Finally, the technique of Projection Onto Convex Sets (POCS) was used in [14] to remove scratches or wires using both global frequency and local spatial information in the image.

(a) Original scan


(b) Enlarged defect


(c) defect map


(d) Median


(e) Directional

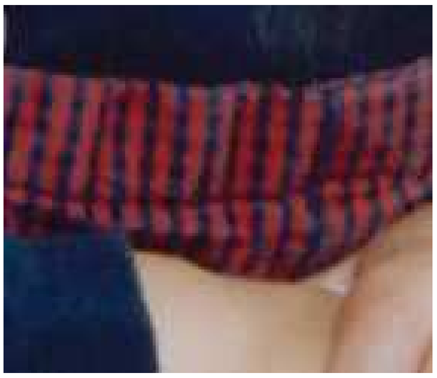

(f) contextual

Figure 3: Comparison of the reconstruction results. The original image is shown in (a) with an enlarged view of the dust in (b). (c) shows the defect map. (d), (e) and (f) show the median, directional, and contextual reconstructions respectively. After the median (d) the defect, though blurred, can still be seen. The directional (e) reconstruction successfully restore the texture of the overalls, further disguising the dust. The contextual (f) fully reconstructs the texture leaving no trace of the defect.

	Side Information	Soft/Hard	Local/Non-Local	Algorithm
1	No	Soft	Local	Section 3.1
2	No	Soft	Non-local	
3	No	Hard	Local	Section 3.2
4	No	Hard	Non-Local	Section 3.3
5	Yes	Soft	Local	Section 3.4
6	Yes	Soft	Non-Local	
7	Yes	Hard	Local	
8	Yes	Hard	Non-Local	

Table 1: Detection approaches classified by type and complexity.

3 Detection

This section describes our detection algorithms in detail. Table 1 summarize the categories of detection algorithms, as discussed in Section 1. The entries in the Algorithm column refer to categories for which we have developed algorithms. The number in each box corresponds to the number of the subsection that describes the algorithm for this category.

3.1 A Credibility Measure Based on the Image

The candidate pixels are those that are much lighter (or darker) than their neighbors and are therefore suspected to be parts of defects. To find these pixels, we follow the direction of [19] and create a so-called *detail-less* image. The goal is to erase dust and scratch defects, such as lines and spots, while keeping other image features such as edges and texture. One simple and efficient option for generating a detail-less image is a median filter. Another good approach for creating a detail-less image include morphological closing and opening. We chose the median filter which removes spots and lines effectively. It also preserves straight edges, but distorts edges with sharp curves. This distortion can translate to false detection, which other aspects of our algorithms treat.

The size of the median filter is a parameter indicating the size of the defect. It has been our experience that the size of the physical dust and scratch defects can be taken as fixed (i.e.,the distribution of the sizes of these objects is narrow). The size of the defect in pixels, then, depends on the resolution of the digital image. Higher resolution implies bigger defect sizes.

The algorithm uses a gray representation of the image. A detail-less image is created using a median filter. The algorithm then uses the gray image and the detail-less image to compute two measures at each pixel, where each measure indicates how likely this pixel is to be defective. The two measures are gray-level difference and contrast dis-similarity. The computation of each measure is very fast and simple.

Gray level difference

This measure is a pixel-by-pixel difference between the gray image and the detail-less image. Defects have high values in the gray-level difference measure. Image edges have high values as well, as do small edges in textured areas.

Contrast dis-similarity

The purpose of the contrast dis-similarity measure is to exclude edges and texture edges that have high values in the gray-level difference measure. The intuition behind this measure is that image edges, unlike defects, are still evident in the detail-less image. While the pixel values along the edge are usually changed by the filtering process, the contrast in the region around each pixel remains high even in the detail-less image. When defects are erased in the detail-less image, however, the contrast is high in the original image, but low in the detail-less image.

The local contrast difference between the gray image and the detail-less image at pixel i is computed as follows:

1. Let σ_i^g be the standard deviation of the pixel values in the $D \times D$ neighborhood of pixel i in the gray image.

2. Let σ_i^{dl} be the standard deviation of the pixel values in the $D \times D$ neighborhood of pixel i in the *detail-less* image.

3. The contrast difference at location i is defined as

$$\frac{(\sigma_i^g - \sigma_i^{dl})^2}{(\sigma_i^g)^2 + (\sigma_i^{dl})^2 + C}, \tag{1}$$

   where C is a positive constant used to avoid division by 0.

Note that other measures of contrast can be used in place of the standard deviations. A related contrast similarity measure has been proposed by Wang et al. [23] as one element of a general measure designed to compare between images.

Combining the measures

For each pixel of the image we use the two measures described above to decide whether it is defective or not. The hard approach to pixel labelling would threshold each of the gray-level difference and contrast difference measure separation and combine with an **AND** operation. The disadvantage of the hard approach is that defects that are not evident in both measures will be missed.

The approach we selected to combine the measure treats each measure as a "probability" of defect. We want to label pixels that have both high gray-level difference and high contrast dissimilarity as defects. We therefore multiply the value at each pixel to obtain a single measure, and map the resulting measure into the range $[0, 1]$ to obtain the credibility map. (Other operations in place of the multiplication that correspond to a fuzzy-AND operation could be utilized [26].) Figure 4(b) shows an example of a defect map computed by this detection algorithm.

3.2 Pixel Labelling based on Local Information

The pixel labelling algorithm computes a hard classification by thresholding the credibility map computed in Section 3.1. Figure 4(d) shows an example of a defect map. This hard threshold is typically selected to balance missing defects and false detections. Non-local context is necessary to improve upon this trade-off.

3.3 A Refined Defect Map based on Non-Local Information

This algorithm examines regions in the defect map and the corresponding image regions to refine the local pixel labelling. Three heuristic are currently used to detect the contour of defects, to avoid erroneous detections of textural features, and to avoid detection of eye-glint.

Detecting the defect contour using classification

The detection steps thus far find high-contrast defects. If the defect is sharp, then the contrast with the background is sufficient to find all the defective pixels. The process of scanning, unfortunately, tend to blur the edges of the dust or scratches, particularly at high scanning resolutions. The outcome of this optical blur is that the contour of the dust does not have enough contrast to be detected. When the contour of a defect is not detected the result of reconstruction is not visually pleasing. First, the contour which is not reconstructed remains visible, so that the defect appears reduced, but does not disappear. Second, the pixels of the defect contour are used for the estimation of the new pixel value in reconstruction, which misleads the algorithm.

We cast the problem of finding the defect contour as a classification problem. Given the pixel labelling from the first stage, pixels that surround defective pixels are examined and classified as defective/not defective. We use Quadratic Discriminant Analysis (QDA)[13] on the gray level at the pixels (a 1-dimensional classification).

This method utilizes intensity information, for the classification, as well as geometric information, i.e., the proximity to a defect. The combined information attains the right balance between finding the low-contrast defect contour and preserving features. From an implementation viewpoint, the method has low computation time.

Excluding high activity areas

Some images contain high activity texture that is very similar to dust or scratches. In such images, the density of detections is too high. Even a very dusty picture will not contain the abundance of defects detected in textured areas. To improve the robustness of the algorithm to such textures we added a filter which ignores the detections in an area of the image if there are too many detections. This simple heuristic greatly improves the results of the algorithm in images with high activity. The result of this heuristic can be observed in the difference of the defect maps in Figure 4(f) as compared with Figure 4(d).

Avoiding detection of eye glint

We have found that glint in a person's eye (the tiny bright flash of light) has similar characteristics as light dust on an image. It is therefore typically labelled as a defect by the pixel labelling algorithm. Removing the glint results in an image that appears lifeless. This artifact is one to which people are particularly sensitive. The solution to this problem is the observation that we have to treat the eye area with care. We implemented an eye detection algorithm based on the Viola-Jones method [22], which enabled us to detect when a particular labelled defect was actually due to eye glint.

3.4 Infra-Red Assisted Detection

Infrared dust and scratch detection is based upon the observation that image-forming dyes in common film originals are transparent in the infrared, whereas defects are typically opaque. IR source selection is constrained by both cost and the spectral sensitivity limitations of sensors optimized for visible-band scanning, and the resulting emission spectra are closer to the visible band than would be ideal. The spectral absorptivities of common film dyes therefore overlap the spectral emissivities of the available IR sources, resulting in significant crosstalk between the visible and IR images.

We developed a decorrelation procedure that removes the film-dye contribution from IR images. We have observed that the absorptivity spectra of common photographic films closely follow the Beer-Lambert law as exposure and thereby image-forming dye concentration is varied. The algorithm therefore assumes that in the density domain the contribution of the image forming dyes on the film to IR response is linear. That is the response, in the density domain to the IR light (D^{IR}) is modelled as

$$D_i^{IR} - a \cdot D_i^I + b$$

where D^I is the response of the visible light in the density domain. The procedure estimates the parameters, a and b of this linear model using the visible image and the IR image and essentially removes this contribution from the IR image.

We have further observed that the cyan dye layer is the principal source of IR absorptivity in common films. To reduce computation, only the red image channel is used for parameter estimation.

The outcome of the decorrelation procedure is demonstrated in Figures 5(c). Note that decorrelation removes all the image details from the IR image, while leaving dust and scratches.

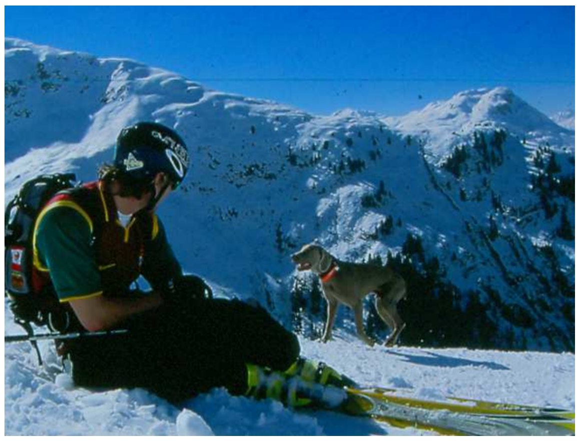
From IR Data to Defect Likelihood

Once image data is removed from the IR image, we are left with an image containing noise, from, e.g., the sensor, and some dark regions indicating defects. The decorrelated IR data is assumed to follow a Gaussian distribution. Under this assumption the probability of pixels below the *background* level goes down quickly, however, it is not zero even for low values of IR. Thus a thresholding scheme is not sufficient to detect defects even after decorrelation.

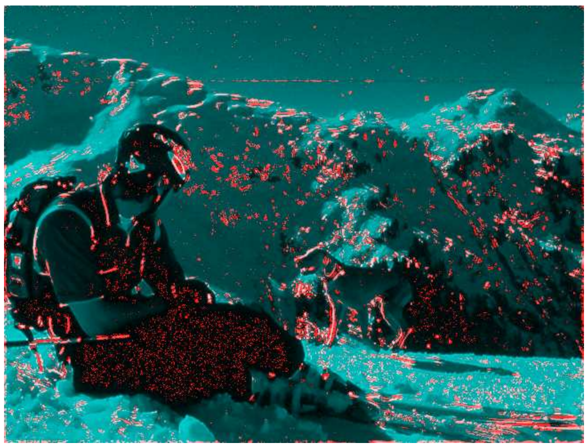
The credibility measure is computed probabilistically using the statistics of the IR data.

1. Set the background level during decorrelation.
2. Estimate IR noise variance by the IR sample variance using a straightforward computation
3. For every pixel compute the distance of that IR value from the background, normalized by the noise standard deviations.
4. Compute the probability of this value for a standard normal variable.

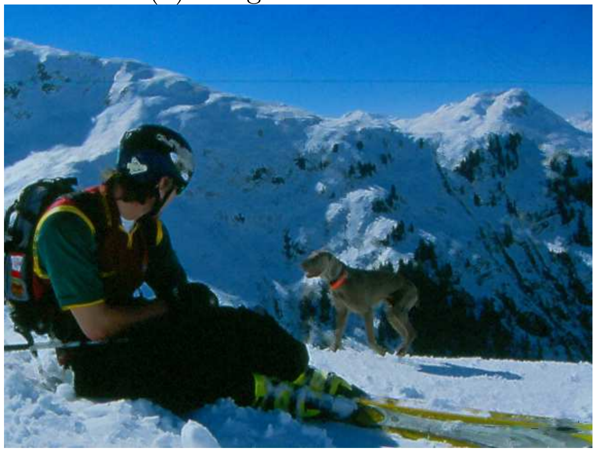
Rather than using the standard normal probability, a parameter may be used to obtain lower or higher defect removal. This parameter corresponds to a normalized distance range for defects, e.g. $[-3, -1]$ would indicate that pixels whose value is at least 3 noise deviations below the background are defective, and pixels whose value is greater than 1 noise deviation below the background contain good data. The credibility value for each pixel is computed by mapping the distance in the given range to the range $[0, 1]$. An example of the resulting credibility measure for a high level of defect removal is shown is Figure 5(d).



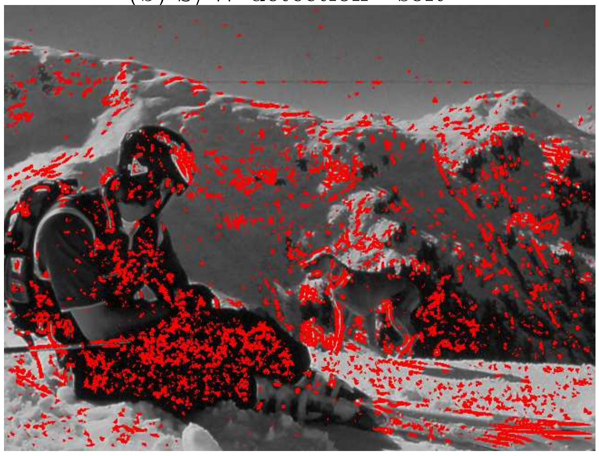
(a) Image with defects



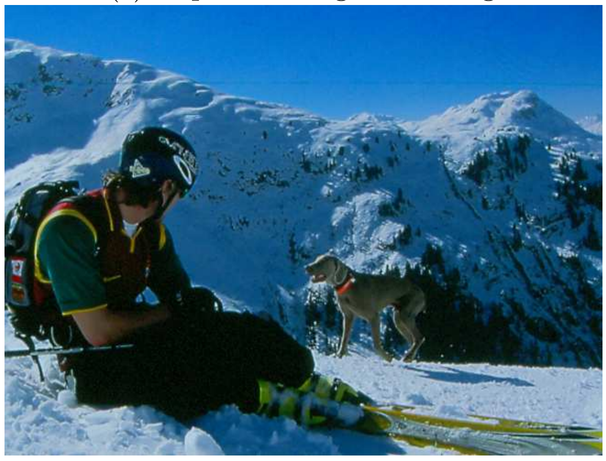
(b) S/W detection - soft



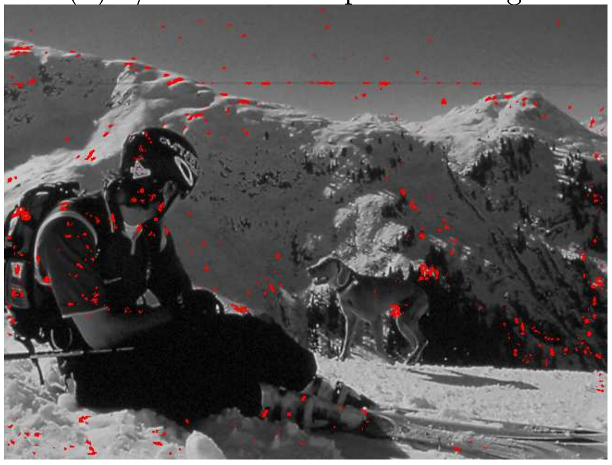
(c) Repaired image - filtering



(d) S/W detection - pixel labeling

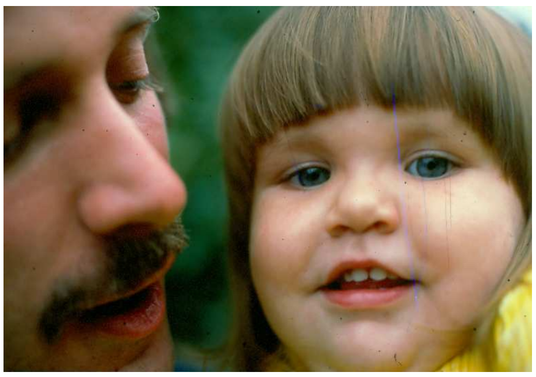


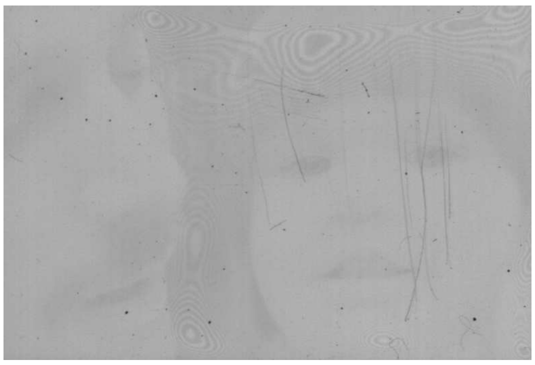
(e) Repaired image - infilling

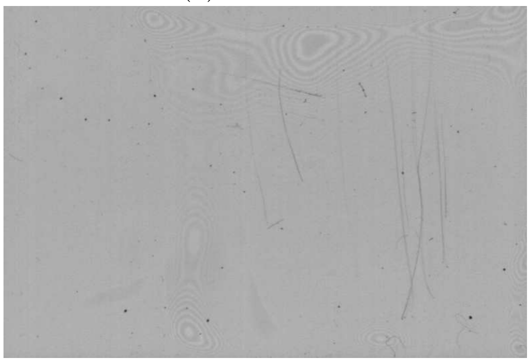


(f) S/W detection - regional classification

Figure 4: Examples of software only dust and scratch removal algorithms. The original image (a) has a large scratch in the sky right above the mountain on the right which extends across the entire image. It also has several dust defects in the sky, e.g., to the left of the mountain on the right. The credibility map (b) is the result of the soft detection algorithm. Repaired image (c) is the result of the credibility-weighted filter using the credibility map (b). The defect map (d) shows the map detected by the pixel labelling algorithm, which finds all the details in the snow texture. The defect map in (f) shows the result of regional classification, which removes most of the false detections. In the repaired image (e), the dust defects in the sky are removed and the scratch is reduced significantly. These details may be difficult to see in print. We suggest viewing them in the digital format.


(a) Slide scan


(b) IR scan of slide


(c) Decorrelated IR data

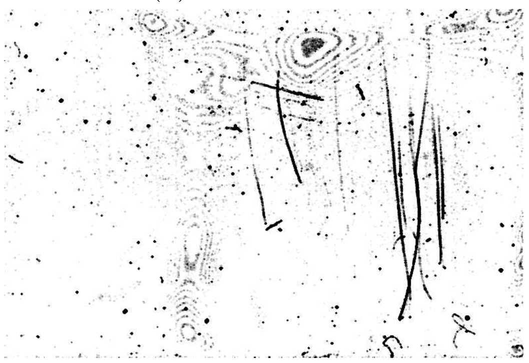

(d) Credibility measure

Figure 5: Example of detection using infra-red. The resulting repair is shown in Figure 1(b)

	Soft/Hard	Local/Non-Local	Algorithm
1	Soft	Local	Section 4.1
2	Soft	Non-local	
3	Hard	Local	Section 4.2
4	Hard	Non-Local	Section 4.3

Table 2: Reconstruction approaches classified by type and complexity.

4 Reconstruction

Like our detection algorithms, we developed reconstruction algorithms in several of the categories described in Section 1. Table 2 summarizes the categories. The first algorithm in this section uses a soft credibility. The two following algorithms use a defect map.

4.1 A credibility weighted bilateral filter

The bilateral filter is given by the following formula, which operates on a local neighborhood of pixels. The neighborhood, for example, may consist of $k \times k$ pixels symmetric about the pixel to be filtered, where k is an odd integer greater than 1.

$$u_0 = \frac{\sum_n L_n \cdot g\left(\beta|f_n - f_0|\right) \cdot f_n}{\sum_n L_n \cdot g\left(\beta|f_n - f_0|\right)} \tag{2}$$

where the index 0 refers to the central pixel to be filtered, and the index n refers to its n-th neighbor, $g(\cdot) \in [0,1]$ is the photometric function, i.e., a function that relates gray-level differences to perceptible edge strengths, β is a scaling constant related to the edge strength, f is the measured single channel signal, L_n is the spatial weight at each neighbor. The sum over n includes the central pixel $n = 0$.

The bilateral filter computes a weighted average of pixels in the neighborhood. The contribution from each of the pixels depends on:

1. **The spatial distance from center pixel.** Typically a Gaussian function is used to reduce the weight of pixels further out from the central pixel.
2. **Contrast difference from central pixel.** The pixel contribution is weighted according to a *photometric function* such that if the contrast difference is low the weight is high, and if the contrast difference is high, the weight is low.

We may write the bilateral filter equivalently as

$$u_0 = f_0 + \frac{\sum_{n\neq 0} L_n \cdot g\left(\beta|\delta_n|\right) \cdot \delta_n}{L_0 + \sum_{n\neq 0} L_n \cdot g\left(\beta|\delta_n|\right)} \tag{3}$$

where $\delta_n = f_n - f_0$. This formulation lends itself to higher order bilateral filters, e.g., replacing the first order differences δ_n by higher order differences, such as $\delta_n^{(2)} = (f_{-n} + f_n)/2 - f_0$.

Like the classic bilateral filter, the credibility-weighted bilateral filter computes a weighted average of pixels in the neighborhood. The contribution from each of the pixels depends on the two factors used by the bilateral filter and one additional factor:

3. **Pixel credibility.** When setting the weight of any pixel in the neighborhood, we balance the effect of the credibility of both the pixel under consideration and the central pixel. When the credibility of a neighborhood pixel is low its relative weight is also low. In addition, if the credibility in the central pixel is low, the sensitivity of the photometric function is reduced so that the relative weight of other neighborhood pixels may increase.

The rest of this section describes this filter in detail. More details on the bilateral filter may be found in [21].

Zeroth order filter

We extend the bilateral filter to take advantage of pixel credibility in the following way:

$$u_0 = \frac{\sum_n L_n C_n \cdot g\left(\beta C_0 |f_n - f_0|\right) \cdot f_n}{\sum_n L_n C_n \cdot g\left(\beta C_0 |f_n - f_0|\right)} \tag{4}$$

where C_0 is the credibility of the central pixel and C_n is the credibility of the nth pixel in the neighborhood. Again, we re-write the credibility-weighted bilateral filter as follows.

$$u_0 = f_0 + \frac{\sum_{n \neq 0} L_n C_n \cdot g\left(\beta C_0 |\delta_n|\right) \cdot \delta_n}{L_0 C_0 + \sum_{n \neq 0} L_n C_n \cdot g\left(\beta C_0 |\delta_n|\right)} \tag{5}$$

where $\delta_n = f_n - f_0$.

The behavior of the credibility weighted bilateral filter ranges between the following extreme cases:

1. If the central pixel has zero credibility, it is replaced by a weighted average of its neighbors regardless of their similarity to the central pixel, but where their relative weights are their credibility-values (in other words, an average of the non-defective neighbors).

2. If the central pixel has full credibility, it is replaced by a modified bilateral weighted average of itself and its neighbors, where the bilateral weight of each neighbor depends both on the similarity to the central pixel (as in bilateral), and on its credibility (in other words, a bilateral average of the non-defective neighbors).

If the central pixel has medium credibility, we want the similarity measure between neighbors and center to have less importance than in case 2, but more importance than the zero-importance in case 1.

First order filter

We know from previous experience with the bilateral filter that for higher resolution images the zeroth order bilateral filter is not sufficient. In such images edges can span a number of pixels and appear as gradual changes, so that the filter misses the edge and smooths across it. Higher order bilateral filters have been developed that perform much better for high resolution images. Whereas the zeroth order bilateral filter uses each pixel in the neighborhood independently, the 1st order bilateral filter uses opposing pairs of pixels from the neighborhood together.

We have extended the credibility-weighted bilateral filter to the first order formulation. The differences between the zeroth order formulation and the first order formulation are:

- The index n refers to a pair of neighborhood pixels with indices n, n', e.g., we may choose opposing pixel pairs.

- The neighboring pixel value f_n is replaced by the average of the pair n, n'.
- The credibility C_n is replaced by a function that combines the two pixel credibility values C_n and $C_{n'}$. One may use any function corresponding to a *fuzzy-AND* operation [26].

$$\begin{aligned} C^1_{n,n'} &= min(C_n, C_{n'}) && (6)\\ C^2_{n,n'} &= C_n \cdot C_{n'} && (7)\\ C^3_{n,n'} &= \lfloor C_n + C_{n'} - 1 \rfloor_0 && (8) \end{aligned}$$

where the notation $\lfloor \cdot \rfloor_0$ indicates clipping of negative values to 0. These functions are ordered by how severely they penalize the pair based on individual pixel credibility. We prefer the combination function 7.

Thus, the first-order credibility-weighted bilateral filter is

$$u_0 = f_0 + \frac{\sum_{n,n'} L_{n,n'} C_{n,n'} \cdot g\left(\beta C_0 |\delta_{n,n'}|\right) \cdot \delta_{n,n'}}{L_0 C_0 + \sum_{n,n'} L_n C_{n,n'} \cdot g\left(\beta C_0 |\delta_{n,n'}|\right)} \tag{9}$$

where $\delta_{n,n'} = (f_n + f'_n)/2 - f_0$, and $C_{n,n'}$ is one of the equations above. For symmetric pair $L_{n,n'} = L_n = L'_n$.

Combined zeroth & first order filter

While the first order bilateral filter is better suited to high resolution images, it does not remove distortions as well. Near a defect we often do not find opposing pairs of high credibility pixels. The best trade-off for reducing both defects and noise, while preserving edges, is obtained by weighting the response of both filters. The weighting function is based on the similarity between credibility values of the pair of pixels.

$$\begin{aligned} u_0 &= f_0 + \frac{\sum_{n,n'}[\alpha_{n,n'} \cdot w^{(1)}_{n,n'} \cdot \delta_{n,n'} + (1 - \alpha_{n,n'}) \cdot \left(w^{(0)}_n \delta_n + w^{(0)}_{n'} \delta'_n\right) \cdot \frac{1}{2}]}{L_0 C_0 + \sum_{n,n'}[\alpha_{n,n'} \cdot w^{(1)}_{n,n'} + (1 - \alpha_{n,n'}) \cdot \left(w^{(0)}_n + w^{(0)}_{n'}\right) \cdot \frac{1}{2}]} && (10)\\ w^{(1)}_{n,n'} &= L_{n,n'} C_{n,n'} \cdot g\left(\beta C_0 |\delta_{n,n'}|\right) && (11)\\ w^{(0)}_n &= L_n C_n \cdot g\left(\beta C_0 |\delta_n|\right) && (12) \end{aligned}$$

where the weight given to the first order filter is

$$\alpha_{n,n'} = \frac{min(C_n, C'_n)}{max(C_n, C'_n)}.$$

This weighting factor ensures that the first order filter is used when the pair of pixels have similar credibilities. However, when there is a significant difference between the two credibilities, e.g., only one of the pixels is credible, the zeroth order filter will have greater weight. The aim of this formulation is to avoid blurring of features.

Figures 1(b) and 2 show an example of dust and scratch removal using the credibility weighted bilateral filter with credibilities computed using IR assisted detection.

4.2 A Directional Reconstruction Algorithm

The directional reconstruction algorithm is a local reconstruction algorithm that uses a defect map. Like other local algorithms it computes a weighted average of the neighborhood of the defective pixels. It is different from other averaging operations in two ways. First, it does not use the value of defective pixels in the neighborhood of the pixel to repair. Second, it attempts to determine whether there is a feature in the block with a definite direction. The algorithm determines the dominant direction, if any, in a local block surrounding the defective pixel, and corrects the missing pixel so that the existing direction is emphasized. Image features that are mistakenly detected as defects can be partially restored in this way. If no definite direction is observed, however, the algorithm falls back to the median option or simply computes the average of the block.

The steps of the algorithm are

1. Compute the gradient at each pixel in the block $D \times D$.

2. Estimate the reliability of gradient information in each pixel: Gradients involving defected pixels are not reliable at all, gradients with small magnitude might indicate noise instead of local directionality. The importance of such measure is stressed by Medioni et al. [12].

3. Compute the weighted average of the gradients in the block, taking the reliability estimations as weight-measures.

4. If the local gradient information is unreliable or the average gradient is small, replace the defective pixel with the median or average value of the non-defective pixels.

5. Find the direction of the local edge in the block. This direction is perpendicular to the average gradient. Now find a line in the same direction that crosses the pixel of interest. This line is parallel to the local edge.

6. Replace the defective central pixel with the average value of the non-defective pixels along this line, giving higher weight to pixels closer to the central pixel than to distant ones.

The directional reconstruction attains better image quality than median reconstruction; compare, for example, Figure 3(d) versus 3(e). Yet it does not reconstruct textures very well.

4.3 A Non-Local Context Dependent Algorithm

The non-local reconstruction algorithm also uses a defect map, and is intended to overcome the shortcomings of local reconstruction, in particular in textured areas. It is motivated by a texture synthesis algorithm [10]. Rather than averaging, it copies like textures from elsewhere in the image.

Let D and R be positive integers such that $D < R$. Typical values for D and R, for example, are $D = 5$ and $R = 100$. At each defective pixel i,

1. Let C_i be the context of the defective pixel i. The context includes all color channels of the pixels in a $D \times D$ neighborhood around pixel i.

2. Look at all the $D \times D$ neighborhoods, N_j, in an $R \times R$ region around pixel i. Neighborhoods with defective pixels are excluded from the search.

Tier	Time Performance	Memory Utilization	Image Quality	Scanner/Image Editor
Tier 1	Real-Time	Image strip	Acceptable	Scanner
Tier 2	Efficient (seconds)	Two full images	Good	Both
Tier 3	Slow (minutes)	Two full images + data structures	Excellent	Image editor
Tier 4	Real-Time	Two Image strips	Excellent	Scanner

Table 3: Dust and scratch solution tiers.

3. Find the neighborhood N_j which is most similar to C_i. Specifically, for every neighborhood N_j, compute the sum of squared differences between the non-defective pixels of C_i and the corresponding pixels of N_j. Let N_{j*} be the neighborhood with the lowest sum of squared differences.

4. If the sum of squared differences is below a pre-defined threshold, replace the defective pixels in C_i with the corresponding pixel values from N_{j*}.

5. Otherwise replace pixel i with the median value of its $D \times D$ neighborhood.

Note that in step 4 several pixels are typically replaced. Replacing several pixels together has two advantages. The first advantage is speed, because the number of searches for matching neighborhoods is reduced. The second is, perhaps surprisingly, in image quality. Work in texture synthesis [11] has shown that texture is duplicated better when groups of pixels are copied rather than single pixels. In step 5, when no "good enough" context is found, the algorithm defaults to the median solution.

Figure 3(f) demonstrates the results of the non-local reconstruction. The advantage of this reconstruction in restoring textures is very evident.

5 Applications

Dust and scratch removal applications arise in various scenarios. For example, a solution in a scanner pipeline must be fully-automatic and efficient. A scanner may also utilize side information, such as an IR scan. In an image editor, efficiency constraints may be relaxed, but, for the naive user, the application should require minimal intervention. On the other hand, there are some users that want excellent results and are willing to make the necessary effort. We describe four tiers of applications and a solution that is suited to each level. Table 3 summarizes the characteristics and requirements of each application tier, and Table 4 indicates the categories of detection and reconstruction algorithms we propose for each tier.

5.1 Tier 1: Dust and Scratch Removal Combined with Denoising with Best Performance

Tier 1 use case assumes the most demanding performance requirements, e.g., a real-time system such as a scanner, without specialized hardware. For best performance, the imaging pipeline of a capture device processes the image in strips. Thus, algorithms in the pipeline can only access local image information. This is the worst case for detection, because with local information false

Tier	Detection			Reconstruction	
	Soft/Hard	Local/Non-Local	Side Information	Soft/Hard	Local/Non-Local
Tier 1	Soft	Local	No	Soft	Local
Tier 2	Hard	Non-local	No	Hard	Local
Tier 3	Hard	Non-Local	User selection	Hard	Non-Local
Tier 4	Soft	Local	IR image	Soft	Local

Table 4: Characteristics of detection and reconstruction algorithms for application tiers.

detections are inevitable. To avoid noticeable artifacts due to false alarms we advocate a soft classification. Our solution for tier 1, therefore uses the detection algorithm in Subsection 3.1 and the credibility weighted bilateral filter from Subsection 4.1. This combination is particularly well suited to this application, because a capture device requires denoising in addition to dust and scratch removal. The credibility-weighted filter does both in one operation and a single pass over the image. This solution has a trade-off between defect removal and image detail preservation that is worse than the higher tiers. An example of the results of our solution may be found in Figure 4(c). The trade-off was tuned for effective defect removal, at the cost of detail loss.

5.2 Tier 2: An Automatic, Non-Blurring Software Solution

Tier 2 case assumes an automatic software-only solution. The performance requirements at this tier allow more processing time and more memory. This enables the algorithm to access the entire image as well as the defect map. We are, therefore, able to take advantage of non-local image analysis using the detection algorithm in Subsection 3.3. The reconstruction algorithm we propose for this application is the local infilling algorithm from Subsection 4.2. The refined defect map enables the solution to avoid overall image blur, but inevitably includes some false detections. The reconstruction algorithm is able to repair some image details to compensate for false detections. While this solution uses more memory and computation time than tier 1, it is fast enough to be used in the imaging pipeline of a scanner as well as in an image editor.

Figure 6 shows a comparison of this solutions with other available software solutions. The original image (a) has two strong scratches across the sky and bridge. The repair in Adobe Photoshop[1] (c) has blurred away the scratch along with the entire image. This solution is better suited to dust defects, where the user can select the small area containing dust thereby avoiding overall image blur. By comparison our Tier 2 software only solution (d) does not blur the image overall. It obscures the scratches, but does not remove them entirely. The result of Digital ICE (e) removes the scratches well in the sky, but leaves section of the scratch on the bridge. There is also some blur of image features that leave a noticeable scar where the scratch was near the cables. The result of our hardware assisted solution (f) removes the scratch across the entire image. It blurs the cables a bit, but not enough to be visually disturbing.

Figure 4 demonstrates the advantage of this solution over the local solution from Tier 1. The refined defect map, Figure 4(f), retains few false detections as compared with the credibility map in Figure 4(b). The repaired image, Figure 4(e), therefore, remains sharp. The dust defects were removed. The scratch in the sky is reduced, but not removed completely.

5.3 Tier 3: The Discriminating User

Tier 3 is another software-only solution intended for the discriminating user who is willing to invest some time and effort to obtain very high image quality. In an image editor, user interaction may provide side information to the detection algorithm. For example, an initial defect map may be computed using the algorithm in Subsection 3.3. The user can select pixels labelled as defective in the defect map and re-label them as clean. Given an accurate defect map, this tier has two advantages. The number of pixels that need to be repaired is typically small, so that we can utilize a slower reconstruction algorithm. In addition, we can assume that image features are not labelled as defective. In this setting, the non-local reconstruction algorithm from Subsection 4.3 gives the best repair, in particular in textured areas. This solution can provide excellent image quality, assuming useful side-information from the user.

5.4 Tier 4: An IR Assisted Solution

The solution for Tier 4 assumes that special IR hardware is available for detection. Since specialized hardware would only be available in a scanner, we assumed very strict requirements in speed and memory utilization. This tier uses the detection algorithm described in Subsection 3.4. The credibility weighted bilateral filter described in Section 4.1 is a perfect fit for this application. The imaging pipeline of the scanner includes denoising, and we can combine these two steps into one using this filter. With detection based on IR data this algorithm performs at its best. The credibility values are computed from an independent source, so the credibility map has high accuracy. The filter is able to remove defects completely because these have values near 0 in the credibility map. It does not cause the blur that we saw in the Tier 1 application since the credibility value of non-defective pixels is high. Figures 1(b) and 6(f) demonstrate the application of the hardware assisted solution to images with dust and scratch defects.

Results of this solution are shown in Figure 1(b), 2(b), and 6(f). Figure 6 compares the result of this solution with several the dust and scratch removal solutions. This image is marred by two scratches. One scratch across the entire image about a quarter of the way from the top of the image. There is a smaller scratch on the right below the first scratch. Both scratches have low contrast, and the first scratch is a challenge to repair because it crosses the texture of the bridge cables. The Photoshop algorithm removes the scratch, but leaves the image very blurry, see Figure 6(c). Our solution is sharper. Compare our hardware-assisted results with Digital ICE. The result of our best automatic software-only solution from Tier 2, Figure 6(d), is not able to remove these challenging scratches and produces artifacts on the bridge. The competitive hardware solution, Digital ICE, does not blur the image. It is able to remove the scratches in the sky, but leaves parts of the scratches on the bridge and blurring artifacts in the cable texture where the scratch was originally. The removal results of our hardware solution in Figure 6(f) are superior, demonstrating better defect removal and less disturbing artifacts.

6 Future Work

Our recent work for software-only algorithms is aimed at further reducing false detections as well as improving low-contrast defect detection.

- Scratch defects often have low-contrast which results in partial detection. Likewise, large dust defects are not always fully detected. We would like to connect between detections to complete the defect using tensor voting [12].

- Make use of image analysis, e.g. noise and texture, to adapt the algorithm to the image. Global and regional image information can be useful to adaptively tune detection parameters and for removal of dense detections from textural areas.

In the future we plan to continue improving detection in the following directions:

- Analyze geometric and photometric characteristics of defect clusters to decide whether they are dust, scratches or image features. This requires finding the connected components of the defective pixels and analyzing the components.
- Introduce special handling for additional features of importance, as we did for eyes.

We are researching a new detection approach for prints using multiple scans. The approach is related to modelling objects with polynomial texture maps [20].

A limitation of the credibility-weighted bilateral filter is that if a defect lies on an edge in the image or in a textured region, and the defect pixels have very low credibility, then the filter will compute a new value that blurs the edge. We have experimented with a heuristic that increases the credibility of pixels in high-activity areas of the image. A disciplined approach that avoids such edge artifacts remains future work.

The non-local reconstruction algorithm gives us the best repair results, but is too slow for most applications. Recent work aims to improve the running time of the this algorithm significantly by using search heuristics. This speedup also enables us to attempt more complex image reconstruction applications which we are investigating.

Tables 1 and 2 include several entries for which we have not developed algorithms. Some of these are directions for future work. We do not have a detection or reconstruction algorithm that makes a soft classification and uses non-local image regions. Such algorithms may reduce artifacts even further than the current algorithms, but are expected to be slower.

7 Conclusions

We presented a comprehensive set of algorithm for the removal of dust and scratches from digital images. We introduced several categories of algorithms that differ in several aspects: making soft vs. hard classification of pixels, using local vs. non-local information, and taking advantage of external assistance. The task of removal was divided to two steps: detection and reconstruction. For each step several algorithms were developed that fall into different categories.

We have introduced several innovations in our algorithms. The software-only local detection step takes advantage of local contrast difference which enables the algorithm to separate defects from large image edges better than the commonly used gray level difference.

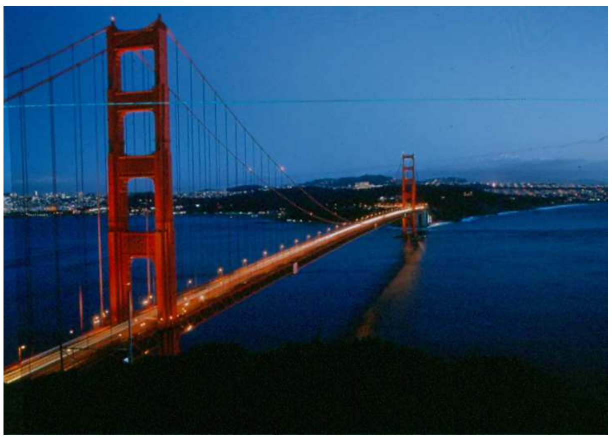
We have made the observation that without side information regional considerations are necessary to separate dust and scratch defects from image details with similar characteristics. The regional classification step effectively removes false detections using simple but insightful heuristics. For example, the heuristic for detecting texture is critical to avoid blurry results. Avoiding glint removal is important because most pictures contain faces.

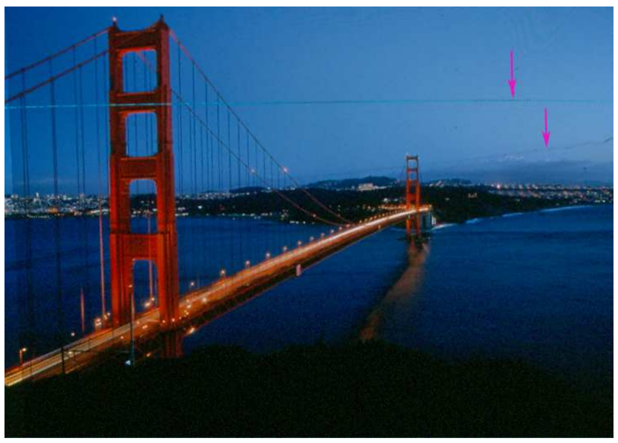
For reconstruction, we also employ regional considerations, in order to repair texture. The contextual reconstruction achieves very credible repair of defective pixels. In particular the repair is superior to local algorithms for textures.

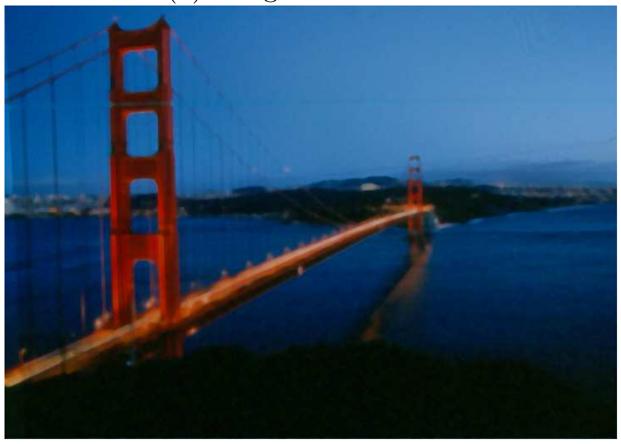
Our local approach for repair from a defect map, the directional reconstruction, is good at repairing defective pixels, and it is often able to compensate for false detections by reconstructing the features. Since it is a local algorithm, it is better suited to applications where efficiency is important.

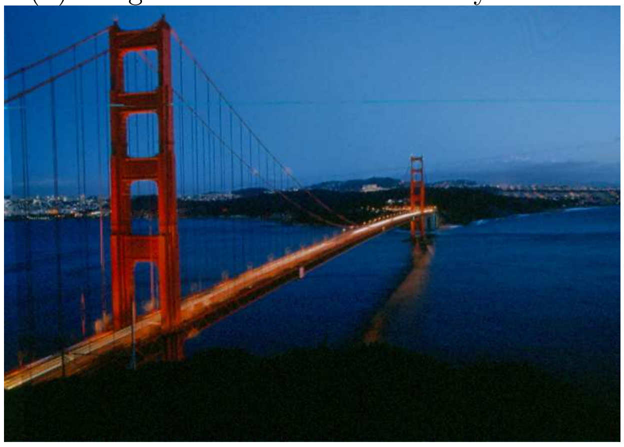
We introduced the credibility weighted bilateral filter for local repair using a credibility map. In this efficient filter, we have incorporated the ability to use pixels with partial credibility. It is, therefore, perfectly suited to use in a scanner with side information in the form of an IR scan.

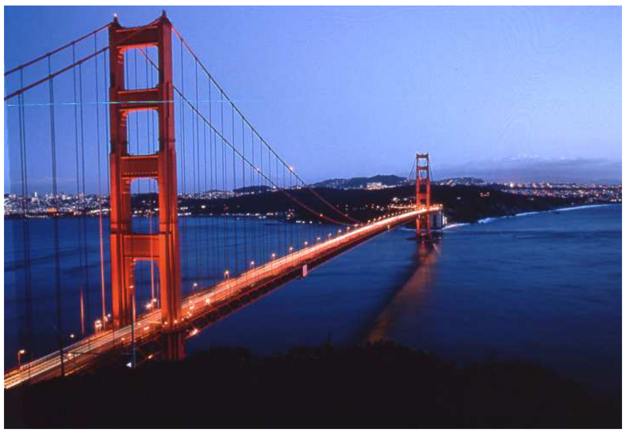
We described four tiers of dust and scratch removal solutions that are suitable for different applications. We show that either side information or looser performance requirements lead to higher image quality. The results of the solution at each tier compare well with competitive solutions.


(a) Image with defects


(b) Image with defects indicated by arrows


(c) Repaired image - Photoshop


(d) Repaired image - Tier 2


(e) Repaired image - Digital ICE

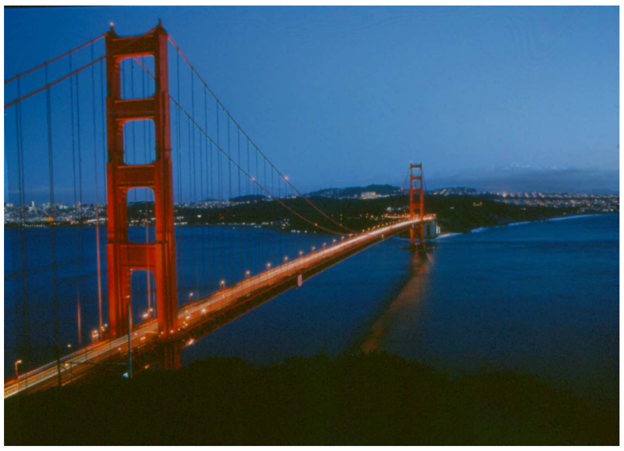

(f) Repaired image - Tier 4

Figure 6: Comparison of dust and scratch removal algorithms.

References

[1] Adobe photoshop. http://www.computer-darkroom.com/tutorials/tutorial_5_1.htm.

[2] Digital ice, eastman kodak company. http://www.asf.com/products/ice/FilmICEOverview.shtml.

[3] Polaroid dust & scratch utility. http://www.polaroid.com/service/software/poladsr/poladsr.html.

[4] K. Toyama A. Criminisi, P. Prez. Object removal by exemplar-based inpainting. In *Proceedings of IEEE CVPR 2003*, 2003.

[5] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher. Simultaneous structure and texture image inpainting.

[6] J. S. De Bonet. Multiresolution sampling procedure for analysis and synthesis of texture images. In *Proceedings of ACM SIGGRAPH*, 1997.

[7] A. C. Copeland, G. Ravichandran, and M. M. Trivedi. Texture synthesis using gray-level co-occurrence models: algorithms, experimental analysis, and psychophysical support. *Optical Engineering*, 40:2655–2673, 2001.

[8] G. Cross and A Jain. Markov randoim field texture models. *IEEE Transactions PAMI*, 5:25–39, 1983.

[9] J. F. Dupont and C. E. Papin. Method for the removal of scratches from digital images, 2002. US Patent No. US20030068096.

[10] A.A Efros and T.K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of IEEE Internation Conference on Computer Vision*, 1999.

[11] A.A Efros and Freeman W.T. Image quilting for texture synthesis and transfer. In *Proceedings of SIGGRAPH 2001*, pages 341–346, 2001.

[12] M.S. Lee G. Medioni and C.K. Tang. *Computational Framework for Segmentation and Grouping.* Elsevier, 2000.

[13] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning - Data Mining, Inference, and Prediction.* Springer-Velag, 2001.

[14] A. N. Hirani and T. Totsuka. Combining frequency and spatial domain information for fast interactive image noise removal. In *In Proceedings of ACM SIGGRAPH*, 1996.

[15] J. Legakis H.W. Jensen J. Dorsey, Al Edelman and H.K. Pdersen. Modeling and rendering of weathered stone. In *Proceedings of SIGGRAPH 99*, page 225.

[16] V. Caselles M. Bertalmio, G. Sapiro and C. Ballester. Image inpainting. In *Proceedings of SIGGRAPH 2000*, 2000.

[17] T. Malzbender and S. Spach. A context sensitive texture nib. In *Proc. of Computer Graphics International*, 1993.

[18] J. Portilla and E.P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40:49–71, December 2000.

[19] D. R. Robins and J.C. Ye. Method and apparatus for detection and removal of scanned image scratches and dust, 2001. US Patent No. US20030039402.

[20] H. Wolters T. Maltbender, D. Gelb. Polynomial texture maps. In *Proceedings of SIGGRAPH 2001*, page 519.

[21] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *ICCV*, pages 839–846, 1998.

[22] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 52(2):137–154, 2004.

[23] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing*, 13:600–612, April 2004.

[24] L-Y Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of SIGGRAPH*, 2000.

[25] D. Lischinski M. Werman Z. Bar-Joseph, R. El-Yaniv. Texture mixing and texture movie synthesis using statistical learning. *IEEE Transactions on Visualization and Computer Graphics*, 7.

[26] L. A. Zadeh. *Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi A. Zadeh.* Advances in Fuzzy Systems - Applications and Theory. World Scientific Publishing Co., 1996.