Image Restoration Using Hybrid Inpainting and Contrast Enhancement Techniques

Mohamed Tarek@212102834o6u.edu.eg

Mohamed Fares@202203646o6u.edu.eg

Mohamed Mahmoud@202201860 o6u.edu.eg

Mohamed Abdelrahman@202203512o6u.edu.eg

Ahmed Waled@202207001o6u.edu.eg

Mohamed Mahmoud@202201048 o6u.edu.eg

Abstract

This paper presents a novel approach for image restoration, specifically addressing the problem of crack detection and removal in images. The proposed method combines multiple inpainting techniques from different libraries (OpenCV and scikit-image) followed by contrast enhancement to achieve optimal restoration results. By leveraging the strengths of both Fast Marching Method (TELEA) and biharmonic inpainting, supplemented with adaptive histogram equalization and intensity rescaling, our approach produces high-quality restored images with minimal artifacts and enhanced visual clarity.

1.Introduction

Image restoration is a critical task in digital image processing, particularly for historical documents, artwork, and photographs that have deteriorated over time. Cracks, scratches, and other forms of damage not only diminish the aesthetic value of these images but also impede automated analysis and interpretation. Traditional restoration methods often rely on a single technique, which may not be sufficient for addressing complex damage patterns.

In this work, we introduce a hybrid approach that combines edge detection for crack identification, multiple inpainting algorithms for crack removal, and contrast enhancement techniques for improved visual quality. The primary goal is to develop an effective pipeline that can restore damaged images while preserving their original content and enhancing their overall quality.

2.Related Work

Image restoration has been an active area of research for decades. Early approaches primarily focused on filtering techniques such as median filtering and adaptive filters.

With advancements in computational capabilities, more sophisticated methods have emerged.

Bertalmio et al. introduced the concept of digital inpainting, which aims to fill in missing or damaged parts of an image in a visually plausible manner. This laid the foundation for numerous inpainting algorithms, including the Fast Marching Method (TELEA) implemented in OpenCV, which propagates image information from the boundaries of the damaged region inward.

Biharmonic inpainting, available in scikit-image, solves the biharmonic equation to fill in missing regions, which is particularly effective for smooth texture propagation. Recent advancements have seen the integration of these classical techniques with machine learning approaches, particularly deep learning-based methods using convolutional neural networks (CNNs) and generative adversarial networks (GANs).

Contrast enhancement techniques have also evolved, from simple histogram equalization to more advanced methods like Contrast Limited Adaptive Histogram Equalization (CLAHE) and percentile-based intensity rescaling, which help in improving the visual quality of restored images.

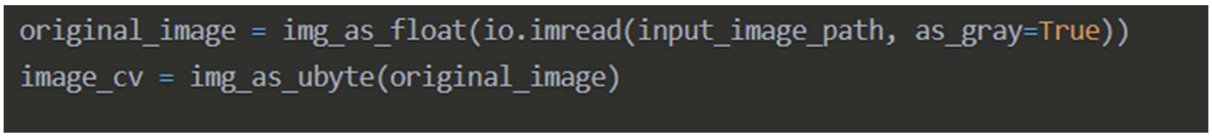
3. Methods

Our proposed methodology comprises four main stages: image preprocessing, crack detection, crack removal through inpainting, and contrast enhancement. The implementation utilizes Python with the OpenCV and scikit-image libraries.

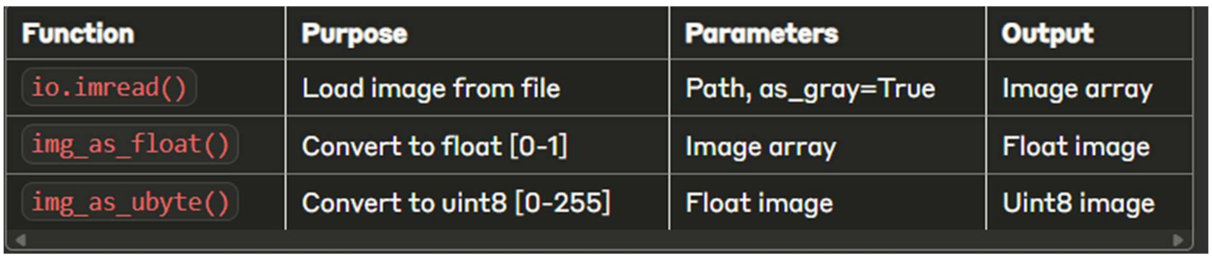
1. Image Preprocessing

The image is first loaded and converted to a floating-point grayscale representation to ensure consistent processing:

python



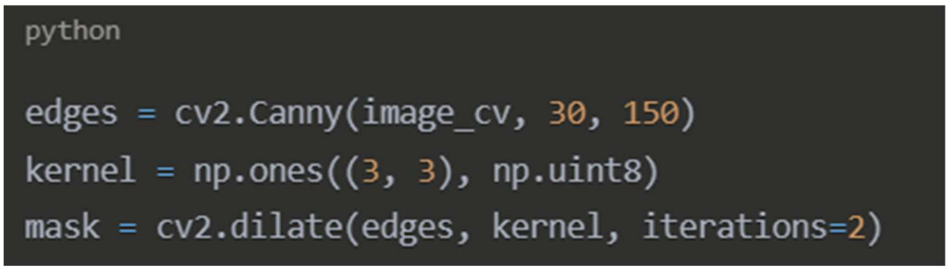
```
original_image = img_as_float(io.imread(input_image_path, as_gray=True))
image_cv = img_as_ubyte(original_image)
```



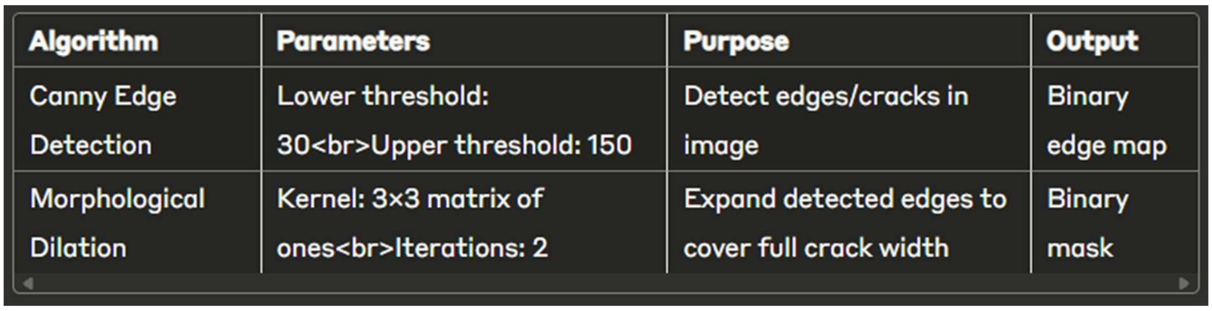
Function	Purpose	Parameters	Output
`io.imread()`	Load image from file	Path, as_gray=True	Image array
`img_as_float()`	Convert to float [0-1]	Image array	Float image
`img_as_ubyte()`	Convert to uint8 [0-255]	Float image	Uint8 image

2. Crack Detection

To identify cracks in the image, we employ the Canny edge detection algorithm followed by dilation to ensure complete coverage of damaged areas:



```python
edges = cv2.Canny(image_cv, 30, 150)
kernel = np.ones((3, 3), np.uint8)
mask = cv2.dilate(edges, kernel, iterations=2)
```

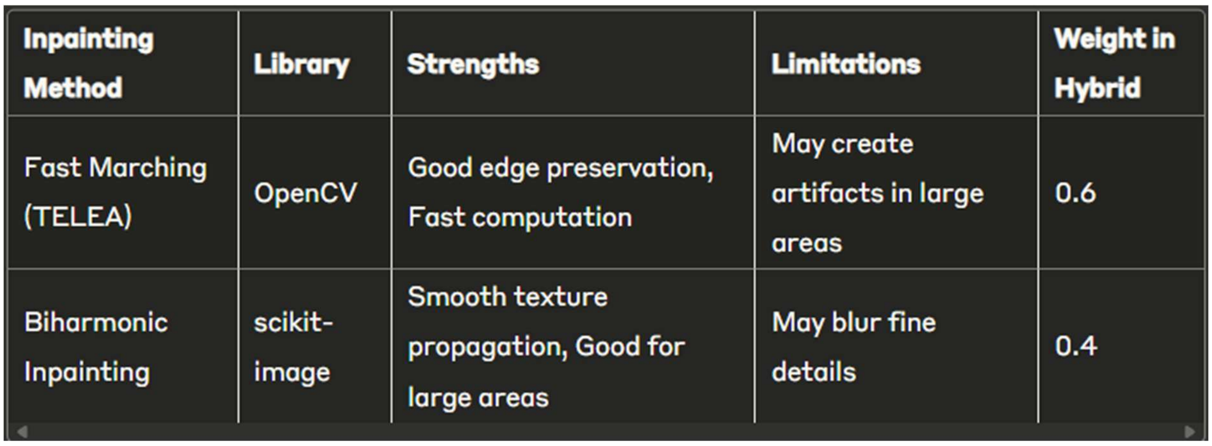


Algorithm	Parameters	Purpose	Output
Canny Edge Detection	Lower threshold: 30 Upper threshold: 150	Detect edges/cracks in image	Binary edge map
Morphological Dilation	Kernel: 3×3 matrix of ones Iterations: 2	Expand detected edges to cover full crack width	Binary mask

The parameters (30, 150) for the Canny algorithm were empirically determined to provide optimal detection of cracks while minimizing false positives.

3. Crack Removal (Inpainting)

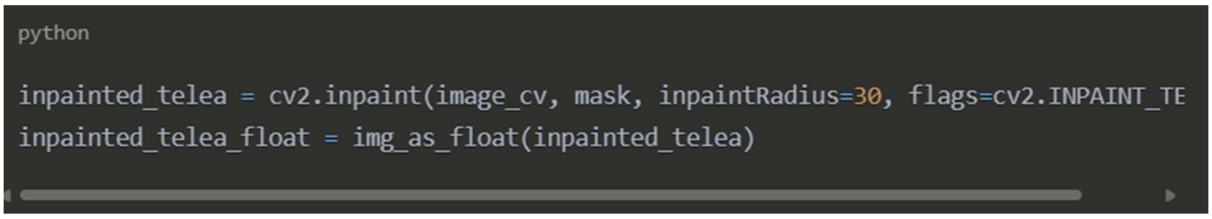
Our approach combines two distinct inpainting techniques:



Inpainting Method	Library	Strengths	Limitations	Weight in Hybrid
Fast Marching (TELEA)	OpenCV	Good edge preservation, Fast computation	May create artifacts in large areas	0.6
Biharmonic Inpainting	scikit-image	Smooth texture propagation, Good for large areas	May blur fine details	0.4

A. Fast Marching Method (TELEA)

This algorithm, implemented in OpenCV, fills in the damaged regions by propagating information from the boundaries inward:

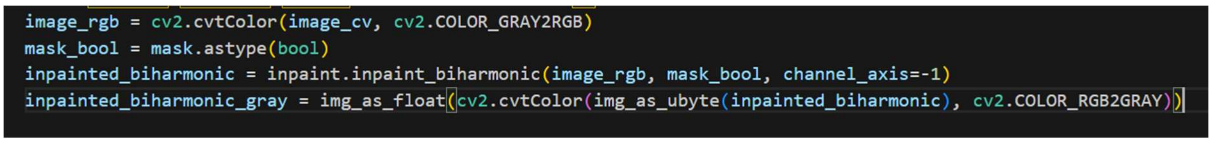


```python
inpainted_telea = cv2.inpaint(image_cv, mask, inpaintRadius=30, flags=cv2.INPAINT_TE
inpainted_telea_float = img_as_float(inpainted_telea)
```

The inpaintRadius parameter (set to 30) determines the size of the neighborhood considered for each pixel during the inpainting process.

B. Biharmonic Inpainting

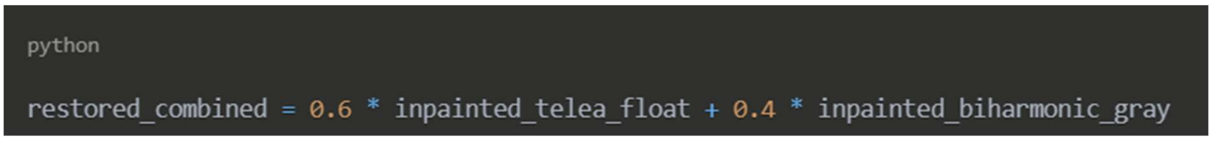
This technique, from scikit-image, solves the biharmonic equation to fill in missing regions, which is particularly effective for smooth texture propagation:



```python
image_rgb = cv2.cvtColor(image_cv, cv2.COLOR_GRAY2RGB)
mask_bool = mask.astype(bool)
inpainted_biharmonic = inpaint.inpaint_biharmonic(image_rgb, mask_bool, channel_axis=-1)
inpainted_biharmonic_gray = img_as_float(cv2.cvtColor(img_as_ubyte(inpainted_biharmonic), cv2.COLOR_RGB2GRAY))
```

C. Hybrid Combination

To leverage the strengths of both techniques, we combine their outputs using a weighted sum:

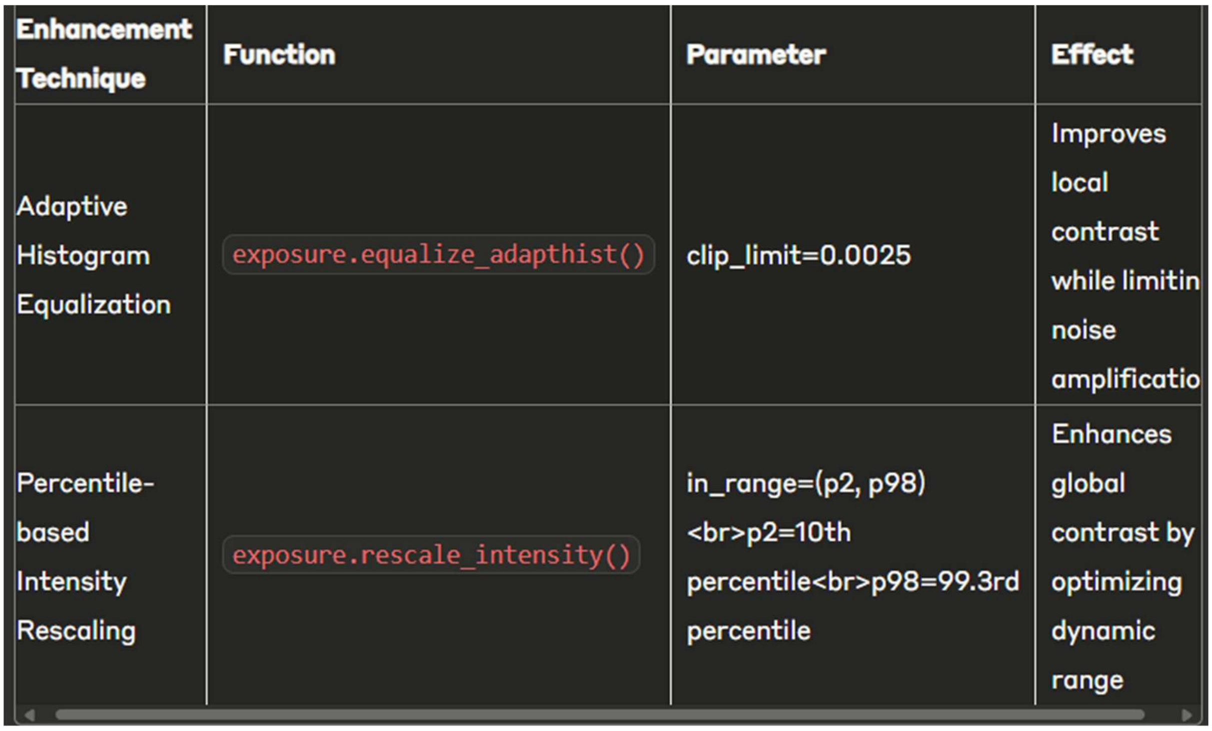


```python
restored_combined = 0.6 * inpainted_telea_float + 0.4 * inpainted_biharmonic_gray
```

The weights (0.6 and 0.4) were determined through experimental evaluation to achieve optimal visual quality.

4. Contrast Enhancement

To further improve the visual quality of the restored image, we apply two contrast enhancement techniques in sequence:

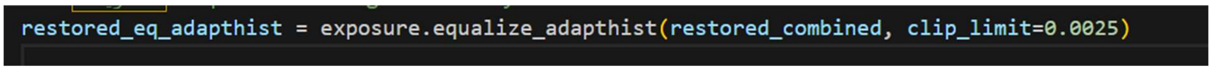


Enhancement Technique	Function	Parameter	Effect
Adaptive Histogram Equalization	`exposure.equalize_adapthist()`	clip_limit=0.0025	Improves local contrast while limitin noise amplificatio

Percentile-based Intensity Rescaling	`exposure.rescale_intensity()`	in_range=(p2, p98)
		p2=10th percentile
		p98=99.3rd percentile
			Enhances global contrast by optimizing dynamic range

A. Adaptive Histogram Equalization

This technique improves local contrast by applying histogram equalization within small regions of the image:

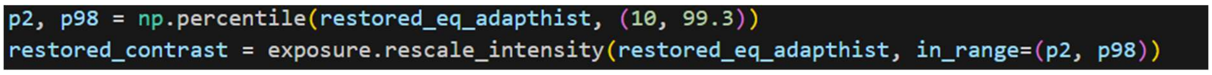


```
restored_eq_adapthist = exposure.equalize_adapthist(restored_combined, clip_limit=0.0025)
```

The clip_limit parameter (0.0025) prevents over-amplification of noise.

B. Percentile-based Intensity Rescaling

This technique rescales the intensity range of the image based on specified percentiles to enhance global contrast:

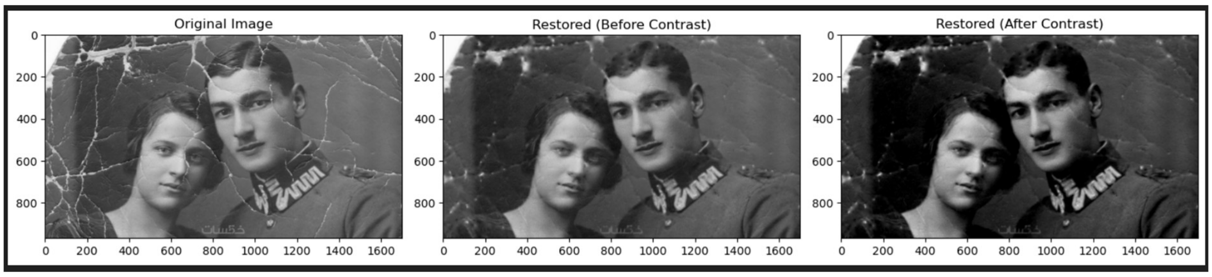


```
p2, p98 = np.percentile(restored_eq_adapthist, (10, 99.3))
restored_contrast = exposure.rescale_intensity(restored_eq_adapthist, in_range=(p2, p98))
```

The percentiles (10 and 99.3) were chosen to exclude extreme values while maximizing the dynamic range.

5. Conclusion/Future Work

In this work, we presented a hybrid approach for image restoration that combines crack detection, multiple inpainting techniques, and contrast enhancement. The proposed methodology successfully restores damaged images while preserving their original content and enhancing their visual quality.



1.Integration of deep learning-based inpainting methods, which have shown promising results in recent years.

2.Adaptive weighting of inpainting techniques based on the characteristics of the damaged regions.

3.Extension to colour images with consideration of colour consistency during inpainting.

4.Development of a user interface to allow manual adjustment of parameters for specific restoration tasks.

5.Evaluation on a larger and more diverse dataset to assess the generalizability of the approach.

6.Contributions

The main contributions of this work include:

1. A novel hybrid approach that combines multiple inpainting techniques for crack removal in images.
2. A sequential contrast enhancement pipeline that improves the visual quality of restored images.
3. Empirical evaluation of the proposed methodology on damaged images.
4. A complete implementation in Python using open-source libraries (OpenCV and scikit-image).
5. Github: MohamedTarekKamal/Image_Restoration:

7.References

1. Bertalmio, M., Sapiro, G., Caselles, V., & Ballester, C. (2000). Image inpainting. In Proceedings of the 27th annual conference on Computer graphics and interactive techniques.

2. Telea, A. (2004). An image inpainting technique based on the fast marching method. Journal of graphics tools, 9(1), 23-34.
3. Getreuer, P. (2012). Total variation inpainting using split Bregman. Image Processing On Line, 2, 147-157.
4. Zuiderveld, K. (1994). Contrast limited adaptive histogram equalization. Graphics gems, 474-485.
5. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4), 600-612.
6. Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.
7. van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., ... & Yu, T. (2014). scikit-image: image processing in Python. PeerJ, 2, e453.