Image Processing Coursework

1 Introduction

This report explores a series of image processing techniques aimed at finding a balance between improving classifier performance and visual quality. This will be done in three subsections: warping the images, enhancing the image content, and inpainting missing regions.

2 Warping

After some experimentation, it was determined that the best results were achieved by first rotating and then warping the image perspectives to fit the frame.

For rotating the images, a function was applied that detects the largest contour using adaptive thresholding [1]. The function then computes the minimum area bounding rectangle of the contour to determine the rotation angle. After adjusting the angle to ensure minimal rotation (under 45 degrees), an affine transformation [2] is used to rotate the image around its centre. This process increased the classifier accuracy from 55% to 71%.

The warping function performs a perspective warp to align a quadrilateral region within an image to a fixed frame size (256×256 pixels). The image is first preprocessed by converting it to grayscale and applying Gaussian blur (to improve edge detection), and then adaptive thresholding and Canny edge detection [3] are applied iteratively to identify contours. Once a valid quadrilateral is detected, the function sorts and slightly expands its corners before computing a perspective transformation matrix [2] to warp the image accordingly. This warping process increased accuracy to 91%. Figures 3 and 6 show both types of images (snow and rain) successfully warped using the above method.

Figure 1: Original Snow Image

Figure 2: Rotated Snow Image

Figure 3: Warped Snow Image

Figure 4: Original Rain Image

Figure 5: Rotated Rain Image

Figure 6: Warped Rain Image

3 Visual Improvements

The three main visual improvements to make were noise reduction, contrast and brightness adjustment, and colour channel imbalance correction.

3.1 Noise Reduction

Since the images contained both Gaussian and Salt&Pepper noise, multiple methods were tested to reduce the noise. Initially, a combination of Median filtering with Gaussian filtering was applied, followed by Bilateral filtering. However, both approaches only removed small amounts of noise and resulted in noticeably blurry images. Gaussian filtering alone significantly improved the classifier's accuracy (96%) but did not effectively eliminate much noise.

Subsequently, Non-Local Means (NLM) denoising was evaluated. Although NLM denoising produced a slightly lower classifier accuracy (90%) compared to Gaussian filtering, it significantly reduced the amount of noise, leading to more visually appealing images. To mitigate the increased blurring associated with NLM, the images were divided into smaller overlapping sections, NLM was applied to each subsection, and the images were then reconstructed.

Figures 7 - 10 compare Gaussian filtering to NLM, with the latter providing better results in both types of images.

Figure 7: Gaussian Blur (Snow)

Figure 8: Non-Local Means Denoising (Snow)

Figure 9: Gaussian Blur (Rain)

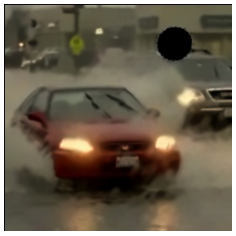

Figure 10: Non-Local Means Denoising (Rain)

3.2 Contrast & Brightness Adjustment

To address contrast and brightness adjustments, the following methods were evaluated: histogram equalisation, CLAHE, gamma correction, and linear stretching.

Gamma correction produced the best results both visually and in the classifier. An adaptable gamma correction was applied, which adjusts the brightness of an image using the gamma correction formula:

$$\text{output} = 255 \times \left(\frac{\text{input}}{255}\right)^{\frac{1}{\gamma}}$$

based on the image's average brightness; it chooses a higher gamma when the image is too dark and a lower gamma when it is too bright [4]. This method increased the classifier score to 91%.

We can see that Figure 12 shows a bright image successfully darkened and Figure 15 shows a dark image successfully brightened.

Figure 11: Denoised Image (Snow)

Figure 12: Gamma Correction (Snow)

Figure 13: Max-RGB (Snow)

Figure 14: Denoised Image (Rain)

Figure 15: Gamma Correction (Rain)

Figure 16: Max-RGB (Rain)

3.3 Colour Channel Imbalance Correction

For correction of colour channel imbalance, the following methods were tested: white patch (or max-RGB) and histogram matching across channels. Max-RGB yielded the best visual results and increased the classifier score to 93%. The function balances the colour channels of the image by scaling each channel so that the brightest point in each channel becomes equally bright, effectively correcting colour imbalances. [5]

Figures 13 and 16 show that Max-RGB removes the yellow tint in the images and reduces general colour imbalance.

4 Inpainting

To inpaint the circle in the image, OpenCV's standard TELEA inpainting function [6] was initially applied across the entire image. However, the resulting inpainting was not very realistic, and unnecessary areas outside the circle were also modified. Consequently, the Hough Circle Transform function [3] was employed to detect the largest black circle in the top-right corner of the image, which was then inpainted using the TELEA method. Since the circle's frame remained visible after inpainting, the inpainting radius was slightly increased to ensure a smoother blend between the inpainted area and the surrounding image. In a small percentage of images, where circles were not detected (possibly due to deformation from the warping process), a method was implemented that detects black shapes instead, using OpenCV's connectedComponentsWithStats function [7]. This alternative method applies a binary inverse threshold to highlight dark areas, creates a mask for the largest identified component, and subsequently inpaints it. The inpainting process did not alter the classifier results, maintaining an accuracy of 93%.

Figure 17: Inpainted Image (Snow)

Figure 18: Inpainted Image (Rain)

References

[1] OpenCV. *OpenCV Image Processing - Miscellaneous Functions.*

[2] OpenCV. *OpenCV Image Processing - Geometric Image Transformations.*

[3] OpenCV. *OpenCV Image Processing - Feature Detection and Extraction.*

[4] Adrian Rosebrock. *OpenCV Gamma Correction.* 2015.

[5] J. Manansala. *Image Processing with Python: Color Correction Using White Balancing.* 2021.

[6] OpenCV. *OpenCV Inpainting Tutorial.*

[7] J. Manansala. *Image Processing with Python: Connected Components and Region Labeling.* 2021.