Image Processing Report

1. Noise Processing

- **Salt-and-pepper noise processing**: median filtering (`medianBlur, kernel=3`) to remove noise points while preserving edges.
- **Gaussian noise processing:** Using Gaussian Blur (GaussianBlur, kernel=5×5) and Non-Local Means denoising (NLM) to smooth the image, reduce noise while preserving edge features.

2. Geometric Distortion Correction

Using gamma correction and CLAHE with dynamic Canny thresholds optimizes edge detection for perspective correction. Morphological analysis and contour filtering identify the target region, while coordinate validation ensures accurate vertex positioning. The final image quality is preserved through bicubic interpolation and sharpening.

3. Brightness/Contrast Optimization

- **Brightness adjustment**: Using **gamma correction** (`adjust_brightness`) to dynamically adjust brightness channels, avoiding over or underexposure, optimizing image details under various lighting conditions.
- **Contrast enhancement**: Using **CLAHE contrast enhancement** (`adjust_contrast`) for local equalization, enhancing detail and texture features, improving recognition accuracy in complex scenes.

4. Color Channel Balancing

- **White balance adjustment**: Using **white balance** to correct color deviation, enhance color naturality, and optimize classifier recognition for different weather scenes.
- **Enhance blue highlight areas**: Through **blue enhancement** (`highlight_blue_boost`) adjusting LAB space B channel to emphasize snowy scene features.

5. Missing Region Repair

Adopting a **progressive repair solution** instead of single-pass `cv2.inpaint` processing to improve repair **naturalness and accuracy**. Compared to traditional one-time filling, this solution's **outside-to-inside progressive repair** better utilizes surrounding pixel information, ensuring smooth transitions, reducing incorrect filling and artifacts, and improving image coherence.

`detect_black_defect` function — Detect black defect areas in images.

Processing Flow

1. **Color space conversion**: RGB to HSV (`cv2.cvtColor`) for black region detection.
2. **Color threshold processing**: Set HSV thresholds (`cv2.inRange`) to generate defect area masks.
3. **Morphological optimization**: Opening and closing operations (`cv2.morphologyEx`) to optimize boundary quality.
4. **Contour analysis and verification**:

- Use `cv2.findContours` to detect and sort boundaries. Calculate circularity to verify geometric properties. Output defect area parameters (center coordinates and radius).

`progressive_inpaint` function

Implements high-precision progressive repair of image defect areas, ensuring natural and accurate repair effects through a systematic approach:

Implementation Steps

1. **Initialization processing**:
   - Use `detect_black_defect` to precisely locate center points and radius. Construct circular mask markers.
2. **Iterative repair process**:
   - Use `cv2.Canny` and `cv2.dilate` to identify and expand boundaries, optimizing repair range.
   - Establish repair priority mechanism through `cv2.distanceTransform`, optimizing pixel reconstruction order.
   - Adopt stepped `cv2.inpaint` strategy to achieve precise area repair, ensuring natural repair transitions.

6. Additional Functions

Granularity processing is for **emphasizing snow and raindrop traces**. By adding **grain noise** (`add_grain`) to simulate snowflake and water droplet effects, enhance detail presentation, helping classifiers better identify weather features.

Optimization Effects

The table shows the **accuracy** of each function when used **independently** and in **series**, improving quality through multiple image processing steps.
(The following problem-solving functions are ordered according to their repair sequence in the code)

Function	Accuracy when applied alone	Accuracy when combined with previous functions
remove_salt_and_pepper	0.92	0.92（First function）
remove_gaussian_noise	0.94	0.91
perspective correction	0.68	0.97
repair	0.60（Did not work when applied alone）	0.96
sharpen_image	0.54	0.93

adjust_contrast	0.64	0.94
white_balance	0.63	0.91（Struggles with rainy day scenarios）
adjust_brightness	0.62	0.85（Struggles with rainy day scenarios）
highlight_blue_boost	0.60	0.93
add_grain	0.62	0.99

Denoising:

remove_salt_and_pepper: When applied alone, achieves **92%** accuracy. As the first step in the processing pipeline, it effectively removes salt and pepper noise, ensuring cleaner images for subsequent processing.

remove_gaussian_noise: Reaches **94%** accuracy alone, showing its importance in noise removal. Combining with salt noise processing reduces accuracy to **91%**. This shows that while it can remove noise, it slightly affects image quality, requiring adjustments in subsequent processing.

(remove_salt_and_pepper）(remove_gaussian_noise） （perspective correction） （repair missing piece)
（All images are generated after applying functions from the current and previous processes）

Perspective Correction:

perspective correction: Perspective correction alone achieves **68% accuracy**. When combined with other steps, it improves to **97%**, demonstrating its importance in correcting distortion and enhancing analysis precision.

Defect Restoration

Repair: Accuracy is 60% when used alone due to black edges interfering with circle detection. When combined with the previous steps, the accuracy increases to 96 per cent, allowing for effective repair of damaged areas.


(sharped)


(adjust_contrast)


(white_balance)


(adjust_brightness)

Contrast and Brightness

`sharpen_image`, `adjust_contrast`, and `adjust_brightness` have limited effectiveness when used alone. `sharpen_image` only enhances details without denoising, `adjust_contrast` strengthens features but increases noise, and `adjust_brightness` may reduce contrast. When combined with other functions, `sharpen_image` and `adjust_contrast` enhance details to **94%**, while `adjust_brightness` performance decreases, mainly because brightness changes reduce contrast, affecting subsequent enhancement algorithms and interfering with the feature recognition of rainy and cloudy scenes.

Color Processing

white_balance used alone can improve color balance, but combining it with other functions affects color distribution. By adjusting **local color** to enhance blue highlights, accuracy increased to **93%**, effectively improving snow scene feature recognition.

Additional Effects

add_grain: Improves to 99% when combined with other functions. Grain processing enhances image texture, improving rain scene recognition rates by simulating water droplets and rainfall characteristics.