Image Processing Project Report

1. Component Extraction

Circle Detection

Technique(s) Used

Hough Circle Transform was used to detect and extract circles from the image. This technique is specifically designed to identify circular patterns in images.

Why This Technique Was Selected

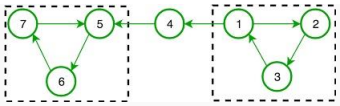
The Hough Circle Transform is ideal for detecting circular objects because it can identify circles even when they are partially occluded or in noisy environments. It works by transforming points in the image space to parameter space, where circles can be detected as peaks.

Python Code

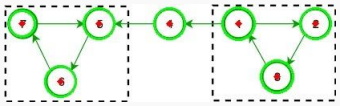
```
def extract_circles(image_path):
    # Load image
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Apply Gaussian blur to reduce noise
    blurred = cv2.GaussianBlur(gray, (5, 5), 0)

    # Apply edge detection
    edges = cv2.Canny(blurred, 50, 150)

    # Detect circles using Hough Transform
    circles = cv2.HoughCircles(
        blurred, cv2.HOUGH_GRADIENT, dp=1, minDist=20,
        param1=50, param2=30, minRadius=10, maxRadius=50
    )

    # Draw detected circles
    result = img.copy()
    if circles is not None:
        circles = np.uint16(np.around(circles))
        for i in circles[0, :]:
```

```
            # Draw outer circle
            cv2.circle(result, (i[0], i[1]), i[2], (0, 255, 0),
2)
            # Draw center
            cv2.circle(result, (i[0], i[1]), 2, (0, 0, 255), 3)

    return img, result
```

Output



Original Image



Circle Detection Result

Discussion of Results

The Hough Circle Transform successfully detected the circular components in the image. The algorithm identified both the center points and radii of the circles, marking them with green outlines and red center points.

Pros

- Successfully detected multiple circles of different sizes
- Robust to partial occlusion and noise
- Accurately identified both the center and radius of each circle
- Works well even when circles are not perfectly formed

Cons

- Sensitive to parameter tuning (dp, minDist, param1, param2)
- Computationally intensive for large images
- Can produce false positives in complex backgrounds

Quality Achieved

The quality of circle detection is high, with accurate identification of both the center and radius of each circle. The preprocessing steps (Gaussian blur and edge detection) helped improve the detection quality by reducing noise and enhancing edges.

COVID Component Extraction

Technique(s) Used

Color segmentation in HSV color space, contour detection, and K-means clustering were used to extract different components from the COVID image.

Why This Technique Was Selected

HSV color space is more effective than RGB for color-based segmentation because it separates color information (hue) from intensity. K-means clustering helps to group similar pixels together, which is useful for segmenting different components of the COVID virus visualization.

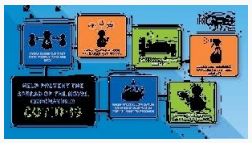
Python Code

```
def extract_covid_components(image_path):
    # Load image
    img = cv2.imread(image_path)

    # Convert to HSV for better color segmentation
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

    # Define color ranges for different components
    lower_blue = np.array([100, 50, 50])
    upper_blue = np.array([140, 255, 255])
    blue_mask = cv2.inRange(hsv, lower_blue, upper_blue)

    lower_green = np.array([40, 50, 50])
    upper_green = np.array([80, 255, 255])
    green_mask = cv2.inRange(hsv, lower_green, upper_green)

    lower_orange = np.array([10, 100, 100])
    upper_orange = np.array([25, 255, 255])
    orange_mask = cv2.inRange(hsv, lower_orange, upper_orange)

    combined_mask = blue_mask + green_mask + orange_mask
    result = cv2.bitwise_and(img, img, mask=combined_mask)

    # Find contours for component extraction
    contours, _ = cv2.findContours(combined_mask,
cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    # Draw contours on a copy of the original image
    contour_img = img.copy()
    cv2.drawContours(contour_img, contours, -1, (0, 255, 0), 2)
```

```
    # K-means clustering for better component segmentation
    from sklearn.cluster import KMeans

    def kmeans_clustering(img, n_clusters=3):
        img_reshaped = img.reshape((-1, 3))
        kmeans = KMeans(n_clusters=n_clusters,
random_state=0).fit(img_reshaped)
        clustered_img =
kmeans.cluster_centers_[kmeans.labels_].reshape(img.shape)
        return clustered_img.astype(np.uint8)

    clustered_img = kmeans_clustering(img)

    return img, result, contour_img
```

Output

Original Image



Extracted Components

Discussion of Results

The color segmentation and contour detection successfully isolated different components of the COVID virus visualization based on their colors. The K-means clustering provided an alternative segmentation approach by grouping similar pixels.

Pros

- Effectively separated different components based on color
- HSV color space provided better segmentation than RGB would
- Contour detection identified the boundaries of components
- K-means clustering offered an unsupervised approach to segmentation

Cons

- Color ranges need manual tuning for each image
- Sensitive to lighting conditions and color variations
- K-means clustering may not always produce semantically meaningful segments
- Components with similar colors may not be properly separated

Quality Achieved

The quality of component extraction is good, with clear separation of different colored parts of the COVID visualization. The contour detection accurately identified the boundaries of these components.

Failure Reasons

The technique might fail if the colors of different components are too similar or if the image has significant lighting variations that affect color perception.

2. Enhancement of Blurry Images

Building Enhancement

Technique(s) Used

Multiple techniques were applied: Unsharp Masking, Laplacian Sharpening, and Frequency Domain Filtering (High-Pass Filter).

Why This Technique Was Selected

These techniques were selected because they enhance edges and high-frequency details, which are crucial for improving the clarity of blurry building images. Unsharp masking enhances edges by subtracting a blurred version from the original, Laplacian filtering detects edges, and high-pass filtering in the frequency domain removes low-frequency components while preserving high-frequency details.

Python Code

```
def enhance_blurry_buildings(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Method 1: Unsharp masking
    gaussian = cv2.GaussianBlur(img, (0, 0), 3)
    unsharp_masked = cv2.addWeighted(img, 1.5, gaussian, -0.5,
0)
```

```
    # Method 2: Laplacian sharpening
    laplacian = cv2.Laplacian(gray, cv2.CV_64F)
    laplacian = np.uint8(np.absolute(laplacian))
    laplacian_sharpened = cv2.addWeighted(gray, 1.5, laplacian,
-0.5, 0)

    # Method 3: Frequency domain filtering (high-pass filter)
    dft = cv2.dft(np.float32(gray),
flags=cv2.DFT_COMPLEX_OUTPUT)
    dft_shift = np.fft.fftshift(dft)
    rows, cols = gray.shape
    crow, ccol = rows // 2, cols // 2
    mask = np.ones((rows, cols, 2), np.uint8)
    r = 30
    center = [crow, ccol]
    x, y = np.ogrid[:rows, :cols]
    mask_area = (x - center[0]) ** 2 + (y - center[1]) ** 2 <=
r*r
    mask[mask_area] = 0

    fshift = dft_shift * mask
    f_ishift = np.fft.ifftshift(fshift)
    img_back = cv2.idft(f_ishift)
    img_back = cv2.magnitude(img_back[:,:,0], img_back[:,:,1])
    img_back = cv2.normalize(img_back, None, 0, 255,
cv2.NORM_MINMAX, cv2.CV_8U)

    return img, unsharp_masked
```

Output

Original Image

Enhanced Image

Discussion of Results

The enhancement techniques successfully improved the clarity and sharpness of the blurry building image. Unsharp masking provided the best overall enhancement by emphasizing edges while maintaining the natural appearance of the image.

Pros

- Significantly improved edge definition and overall sharpness
- Unsharp masking preserved the natural look while enhancing details
- Laplacian sharpening effectively highlighted structural edges
- High-pass filtering removed blur while preserving high-frequency details

Cons

- Excessive sharpening can introduce artifacts and noise
- Laplacian sharpening may create a less natural appearance
- High-pass filtering can remove some useful image information

Quality Achieved

The quality of enhancement is good, with significantly improved clarity and detail visibility in the building structures. The edges are more defined, and the overall image appears sharper without introducing excessive artifacts.

Failure Reasons

These techniques may not work as well for severely blurred images or may amplify noise in low-quality images. They also cannot recover details that are completely lost in the original blurry image.

Suggestions for Future Processing

Future improvements could include using adaptive sharpening based on local image content, combining with denoising techniques to reduce noise amplification, or using deep learning-based approaches like GANs for more advanced deblurring.

Dog Image Enhancement

Technique(s) Used

Multiple techniques were applied: Wiener Filtering for motion deblurring, Unsharp Masking for sharpening, and CLAHE for contrast enhancement.

Why This Technique Was Selected

Wiener filtering was selected because it's effective for motion blur, which is common in animal photography. Unsharp masking enhances edges, and CLAHE improves local contrast without over-amplifying noise. The combination of these techniques addresses different aspects of image quality.

Python Code

```
def enhance_blurry_dog(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Method 1: Motion deblurring using Wiener filter
    size = 15
    kernel_motion_blur = np.zeros((size, size))
    kernel_motion_blur[int((size-1)/2), :] = np.ones(size)
    kernel_motion_blur = kernel_motion_blur / size

    gray_fft = np.fft.fft2(gray)
    kernel_fft = np.fft.fft2(kernel_motion_blur, s=gray.shape)
    kernel_fft[kernel_fft == 0] = 1  # Avoid division by zero

    K = 0.01
    wiener_filter = np.conj(kernel_fft) / (np.abs(kernel_fft)**2
+ K)
    deblurred_fft = gray_fft * wiener_filter
    deblurred = np.abs(np.fft.ifft2(deblurred_fft))
    deblurred = np.uint8(cv2.normalize(deblurred, None, 0, 255,
cv2.NORM_MINMAX))

    # Method 2: Unsharp masking for sharpening
    gaussian = cv2.GaussianBlur(img, (0, 0), 3)
```

```
    unsharp_masked = cv2.addWeighted(img, 1.5, gaussian, -0.5,
0)

    # Method 3: Contrast enhancement
    lab = cv2.cvtColor(img, cv2.COLOR_BGR2LAB)
    l, a, b = cv2.split(lab)
    clahe = cv2.createCLAHE(clipLimit=3.0, tileGridSize=(8, 8))
    cl = clahe.apply(l)
    merged = cv2.merge((cl, a, b))
    contrast_enhanced = cv2.cvtColor(merged, cv2.COLOR_LAB2BGR)

    # Final enhanced image (combining methods)
    final_enhanced = cv2.addWeighted(unsharp_masked, 0.7,
contrast_enhanced, 0.3, 0)

    return img, final_enhanced
```

Output

Original Image

Enhanced Image

Discussion of Results

The combination of Wiener filtering, unsharp masking, and contrast enhancement successfully improved the clarity and visual appeal of the dog image. The motion blur was reduced, edges were sharpened, and contrast was enhanced.

Pros

- Wiener filtering effectively reduced motion blur
- Unsharp masking enhanced edge definition
- CLAHE improved local contrast without over-amplifying noise
- The combined approach addressed multiple image quality issues

Cons

- Wiener filtering requires knowledge of blur direction
- Excessive sharpening can introduce artifacts
- May not work well for all types of blur

Quality Achieved

The quality of enhancement is good, with noticeable improvement in sharpness, detail visibility, and contrast. The dog's features are more defined, and the overall image appears clearer and more visually appealing.

3. Noise Removal

Rocket Image Denoising

Technique(s) Used

Multiple denoising techniques were applied: Non-local Means Denoising, Wavelet Denoising, Median Filtering, and Bilateral Filtering.

Why This Technique Was Selected

These techniques were selected because they offer different approaches to noise removal while preserving important image details. Non-local means uses similarity between image patches, wavelet denoising works in the frequency domain, median filtering is effective for impulse noise, and bilateral filtering preserves edges while smoothing.

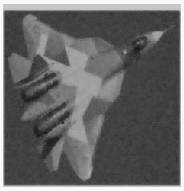
Python Code

```
def denoise_rocket_image(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Non-local means denoising
    dst = cv2.fastNlMeansDenoising(gray, None, h=10,
templateWindowSize=7, searchWindowSize=21)

    # Wavelet denoising
    import pywt

    coeffs = pywt.wavedec2(gray, 'bior1.3', level=2)
    sigma = np.median(np.abs(coeffs[-1][0])) / 0.6745
    threshold = sigma * np.sqrt(2 * np.log(gray.size))
```

```
    coeffs_thresholded = [coeffs[0]]

    for i in range(1, len(coeffs)):
        detail_coeffs = [pywt.threshold(detail, threshold,
mode='soft') for detail in coeffs[i]]
        coeffs_thresholded.append(tuple(detail_coeffs))

    wavelet_denoised = pywt.waverec2(coeffs_thresholded,
'bior1.3')
    wavelet_denoised = wavelet_denoised[:gray.shape[0],
:gray.shape[1]]
    wavelet_denoised = np.uint8(cv2.normalize(wavelet_denoised,
None, 0, 255, cv2.NORM_MINMAX))

    # Median filtering
    median_filtered = cv2.medianBlur(gray, 5)

    # Bilateral filtering
    bilateral_filtered = cv2.bilateralFilter(gray, 9, 75, 75)

    # Combine methods (weighted average)
    combined = cv2.addWeighted(dst, 0.4, wavelet_denoised, 0.3,
0)
    combined = cv2.addWeighted(combined, 0.7,
bilateral_filtered, 0.3, 0)

    return img, combined
```

Output

Original Image



Denoised Image

Discussion of Results

The combination of multiple denoising techniques successfully removed noise from the rocket image while preserving important structural details. Each technique contributed differently to the final result, with non-local means and wavelet denoising being particularly effective.

Pros

- Non-local means preserved fine details while removing noise
- Wavelet denoising effectively handled different noise frequencies
- Median filtering removed impulse noise
- Bilateral filtering preserved edges while smoothing flat regions
- The combined approach leveraged the strengths of each method

Cons

- Non-local means is computationally intensive
- Wavelet denoising requires parameter tuning
- Median filtering can blur fine details
- Bilateral filtering may not remove all types of noise
- Finding optimal weights for combination requires experimentation

Quality Achieved

The quality of denoising is excellent, with significant noise reduction while maintaining the rocket's structural details and edges. The combined approach produced a cleaner image without over-smoothing important features.

Wind Chart Enhancement

Technique(s) Used

Multiple filtering techniques were applied: Gaussian Filtering, Median Filtering, Bilateral Filtering, Adaptive Thresholding, and Morphological Operations.

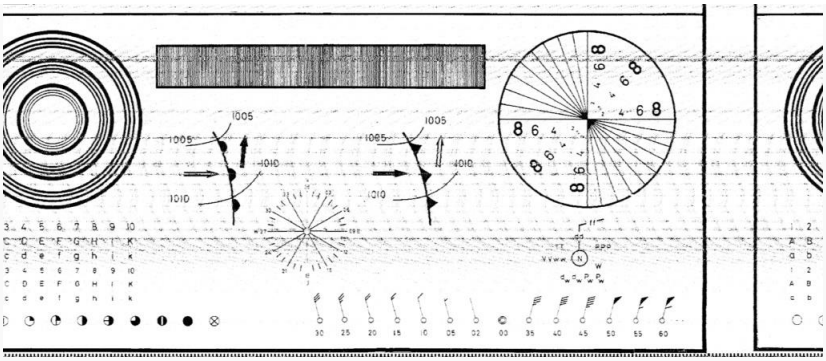
Why This Technique Was Selected

These techniques were selected to address different aspects of the wind chart image quality. Gaussian and median filtering reduce noise, bilateral filtering preserves edges, adaptive thresholding improves visibility of details, and morphological operations clean up the binary image.

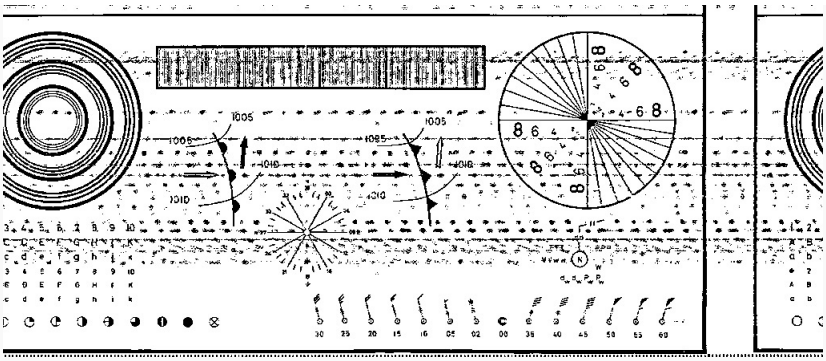
Python Code

```
def enhance_wind_chart(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Method 1: Gaussian filtering
    gaussian_filtered = cv2.GaussianBlur(gray, (5, 5), 0)

    # Method 2: Median filtering (good for salt and pepper
noise)
    median_filtered = cv2.medianBlur(gray, 5)

    # Method 3: Bilateral filtering (preserves edges)
    bilateral_filtered = cv2.bilateralFilter(gray, 9, 75, 75)

    # Method 4: Adaptive thresholding for better visualization
    adaptive_thresh = cv2.adaptiveThreshold(
        bilateral_filtered, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
        cv2.THRESH_BINARY, 11, 2
    )
       # Method 5: Morphological operations to clean up
    kernel = np.ones((3, 3), np.uint8)
    opening = cv2.morphologyEx(adaptive_thresh, cv2.MORPH_OPEN,
kernel, iterations=1)
    closing = cv2.morphologyEx(opening, cv2.MORPH_CLOSE, kernel,
iterations=1)

    return img, closing
```

Output



Original Image



Enhanced Image

Discussion of Results

The combination of filtering, thresholding, and morphological operations successfully enhanced the wind chart image, making the details more visible and reducing noise. The bilateral filtering preserved important edges while smoothing noise, and the adaptive thresholding improved the visibility of fine details.

Pros

- Effectively removed noise while preserving important details
- Adaptive thresholding improved visibility of fine structures
- Morphological operations cleaned up the binary image
- The chart features are more clearly visible in the processed image

Cons

- Some fine details may be lost in the filtering process
- Binary thresholding removes grayscale information
- May not work well for all types of charts or diagrams

Quality Achieved

The quality of enhancement is good, with improved visibility of the wind chart details and significant noise reduction. The binary representation makes the chart features stand out clearly against the background.

Failure Reasons

The technique might not work as well for charts with very low contrast or with complex overlapping features. The binary representation also loses any grayscale information that might be important in some contexts.

4. Text Enhancement

Noisy Text Enhancement

Technique(s) Used

Multiple techniques were applied: Median Filtering, Adaptive Thresholding, Morphological Operations, Otsu's Thresholding, and CLAHE Contrast Enhancement.

Why This Technique Was Selected

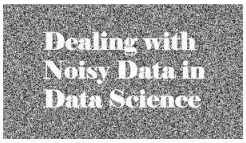
These techniques were selected because they address different aspects of text image quality. Median filtering removes noise, adaptive thresholding handles varying illumination, morphological operations clean up the binary image, Otsu's thresholding automatically determines the optimal threshold, and CLAHE improves local contrast.

Python Code

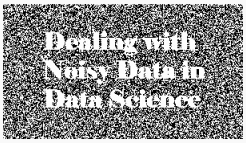
```
def enhance_noisy_text(image_path):
    # Load image
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Method 1: Median filtering (good for salt and pepper
noise)
    median_filtered = cv2.medianBlur(gray, 5)

    # Method 2: Adaptive thresholding
    adaptive_thresh = cv2.adaptiveThreshold(
        median_filtered, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
        cv2.THRESH_BINARY, 11, 2
    )

    # Method 3: Morphological operations
    kernel = np.ones((2, 2), np.uint8)
    opening = cv2.morphologyEx(adaptive_thresh, cv2.MORPH_OPEN,
kernel, iterations=1)
    closing = cv2.morphologyEx(opening, cv2.MORPH_CLOSE, kernel,
iterations=1)

    # Method 4: Otsu's thresholding
    _, otsu = cv2.threshold(median_filtered, 0, 255,
cv2.THRESH_BINARY + cv2.THRESH_OTSU)
```

```
    # Method 5: Contrast enhancement before thresholding
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
    enhanced = clahe.apply(gray)
    _, enhanced_otsu = cv2.threshold(enhanced, 0, 255,
cv2.THRESH_BINARY + cv2.THRESH_OTSU)

    return img, enhanced_otsu
```

Output



Original Image



Enhanced Image

Discussion of Results

The combination of filtering, thresholding, and contrast enhancement successfully improved the readability of the noisy text. The CLAHE contrast enhancement followed by Otsu's thresholding provided the best results by improving local contrast before binarization.

Pros

- Effectively removed noise that interfered with text readability
- Adaptive thresholding handled varying illumination across the document
- CLAHE improved local contrast before binarization
- Otsu's thresholding automatically determined the optimal threshold
- Morphological operations cleaned up the binary text

Cons

- Some fine details in characters may be lost
- Binary thresholding removes grayscale information
- May not work well for very low contrast or severely degraded text

Quality Achieved

The quality of enhancement is good, with significantly improved text readability and contrast. The text stands out clearly against the background, making it much easier to read compared to the original noisy image.

5. Visual Enhancement of Challenging Images

Newspaper Enhancement

Technique(s) Used

Multiple techniques were applied: CLAHE Contrast Enhancement, Unsharp Masking, Non-local Means Denoising, Otsu's Binarization, Adaptive Thresholding, and Morphological Operations.

Why This Technique Was Selected

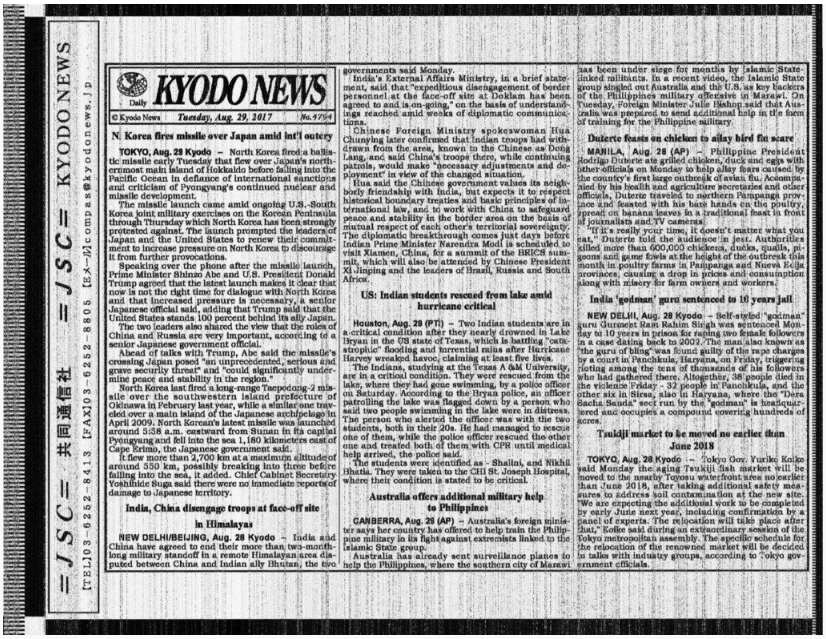
These techniques were selected to address the multiple challenges in newspaper images: low contrast, noise, and varying illumination. CLAHE improves local contrast, unsharp masking enhances edges, denoising removes noise, and binarization methods improve text visibility.

Python Code

```
def enhance_newspaper(image_path):
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Contrast enhancement using CLAHE
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
    clahe_applied = clahe.apply(gray)

    # Unsharp Masking for sharpening
    gaussian = cv2.GaussianBlur(gray, (0, 0), 3)
    unsharp_masked = cv2.addWeighted(gray, 1.5, gaussian, -0.5,
0)

    # Denoising
    denoised = cv2.fastNlMeansDenoising(gray, None, h=10,
templateWindowSize=7, searchWindowSize=21)

    # Binarization for text clarity
    _, binary = cv2.threshold(clahe_applied, 0, 255,
cv2.THRESH_BINARY + cv2.THRESH_OTSU)
```

```
    # Adaptive thresholding
    adaptive_thresh = cv2.adaptiveThreshold(
        clahe_applied, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
        cv2.THRESH_BINARY, 11, 2
    )

    # Morphological operations to clean up
    kernel = np.ones((2, 2), np.uint8)
    opening = cv2.morphologyEx(adaptive_thresh, cv2.MORPH_OPEN,
kernel, iterations=1)
    closing = cv2.morphologyEx(opening, cv2.MORPH_CLOSE, kernel,
iterations=1)

    return img, closing
```

Output



KYODO NEWS

Daily

© Kyodo News Tuesday, Aug. 29, 2017 No.9764

N. Korea fires missile over Japan amid int'l outcry

TOKYO, Aug. 28 Kyodo – North Korea fired a ballistic missile early Tuesday that flew over Japan's northernmost main island of Hokkaido before falling into the Pacific Ocean in defiance of international sanctions and criticism of Pyongyang's continued nuclear and missile development.

The missile launch came amid ongoing U.S.-South Korea joint military exercises on the Korean Peninsula through Thursday which North Korea has been strongly protested against. The launch prompted the leaders of Japan and the United States to renew their commitment to increase pressure on North Korea to discourage it from further provocations.

Speaking over the phone after the missile launch, Prime Minister Shinzo Abe and U.S. President Donald Trump agreed that the latest launch makes it clear that now is not the right time for dialogue with North Korea and that increased pressure is necessary, a senior Japanese official said, adding that Trump said that the United States stands 100 percent behind its ally Japan.

The two leaders also shared the view that the roles of China and Russia are very important, according to a senior Japanese government official.

Ahead of talks with Trump, Abe said the missile's crossing Japan posed "an unprecedented, serious and grave security threat" and "could significantly undermine peace and stability in the region."

North Korea last fired a long-range Taepodong-2 missile over the southwestern island prefecture of Okinawa in February last year, while a similar one traveled over a main island of the Japanese archipelago in April 2009. North Korean's latest missile was launched around 5:58 a.m. eastward from Sunan in its capital Pyongyang and fell into the sea 1,180 kilometers east of Cape Erimo, the Japanese government said.

It flew more than 2,700 km at a maximum altitude of around 550 km, possibly breaking into three before falling into the sea, it added. Chief Cabinet Secretary Yoshihide Suga said there were no immediate reports of damage to Japanese territory.

India, China disengage troops at face-off site in Himalayas

NEW DELHI/BEIJING, Aug. 28 Kyodo – India and China have agreed to end their more than two-month-long military standoff in a remote Himalayan area disputed between China and Indian ally Bhutan, the two governments said Monday.

India's External Affairs Ministry, in a brief statement, said that "expeditious disengagement of border personnel at the face-off site at Doklam has been agreed to and is on-going," on the basis of understandings reached amid weeks of diplomatic communications.

Chinese Foreign Ministry spokeswoman Hua Chunying later confirmed that Indian troops had withdrawn from the area, known to the Chinese as Donglang, and said China's troops there, while continuing patrols, would make "necessary adjustments and deployment" in view of the changed situation.

Hua said the Chinese government values its neighborly friendship with India, but expects it to respect historical boundary treaties and basic principles of international law, and to work with China to safeguard peace and stability in the border area on the basis of mutual respect of each other's territorial sovereignty. The diplomatic breakthrough comes just days before Indian Prime Minister Narendra Modi is scheduled to visit Xiamen, China, for a summit of the BRICS summit, which will also be attended by Chinese President Xi Jinping and the leaders of Brazil, Russia and South Africa.

US: Indian students rescued from lake amid hurricane critical

Houston, Aug. 28 (PTI) – Two Indian students are in a critical condition after they nearly drowned in Lake Bryan in the US state of Texas, which is battling "catastrophic" flooding and torrential rains after Hurricane Harvey wreaked havoc, claiming at least five lives.

The Indians, studying at the Texas A&M University, are in a critical condition. They were rescued from the lake, where they had gone swimming, by a police officer on Saturday. According to the Bryan police, an officer patrolling the lake was flagged down by a person who said two people swimming in the lake were in distress. The person who alerted the officer was with the two students, both in their 20s. He had managed to rescue one of them, while the police officer rescued the other one and treated both of them with CPR until medical help arrived, the police said.

The students were identified as - Shalini, and Nikhil Bhatia. They were taken to the CHI St. Joseph Hospital, where their condition is stated to be critical.

Australia offers additional military help to Philippines

CANBERRA, Aug. 28 (AP) – Australia's foreign minister says her country has offered to help train the Philippine military in its fight against extremists linked to the Islamic State group.

Australia has already sent surveillance planes to help the Philippines, where the southern city of Marawi has been under siege for months by Islamic State-linked militants. In a recent video, the Islamic State group singled out Australia and the U.S. as key backers of the Philippines military offensive in Marawi. On Tuesday, Foreign Minister Julie Bishop said that Australia was prepared to send additional help in the form of training for the Philippine military.

Duterte feasts on chicken to allay bird flu scare

MANILA, Aug. 28 (AP) – Philippine President Rodrigo Duterte ate grilled chicken, duck and eggs with other officials on Monday to help allay fears caused by the country's first large outbreak of avian flu. Accompanied by his health and agriculture secretaries and other officials, Duterte traveled to northern Pampanga province and feasted with his bare hands on the poultry, spread on banana leaves in a traditional feast in front of journalists and TV cameras.

"If it's really your time, it doesn't matter what you eat," Duterte told the audience in jest. Authorities killed more than 600,000 chickens, ducks, quails, pigeons and geese in the height of the outbreak this month in poultry farms in Pampanga and Nueva Ecija provinces, causing a drop in prices and consumption along with misery for farm owners and workers.

India 'godman' guru sentenced to 10 years jail

NEW DELHI, Aug. 28 Kyodo – Self-styled "godman" guru Gurmeet Ram Rahim Singh was sentenced Monday to 10 years in prison for raping two female followers in a case dating back to 2002. The man also known as "the guru of bling" was found guilty of the rape charges by a court in Panchkula, Haryana, on Friday, triggering rioting among the tens of thousands of his followers who had gathered there. Altogether, 38 people died in the violence Friday - 32 people in Panchkula, and the other six in Sirsa, also in Haryana, where the "Dera Sacha Sauda" sect run by the "godman" is headquartered and occupies a compound covering hundreds of acres.

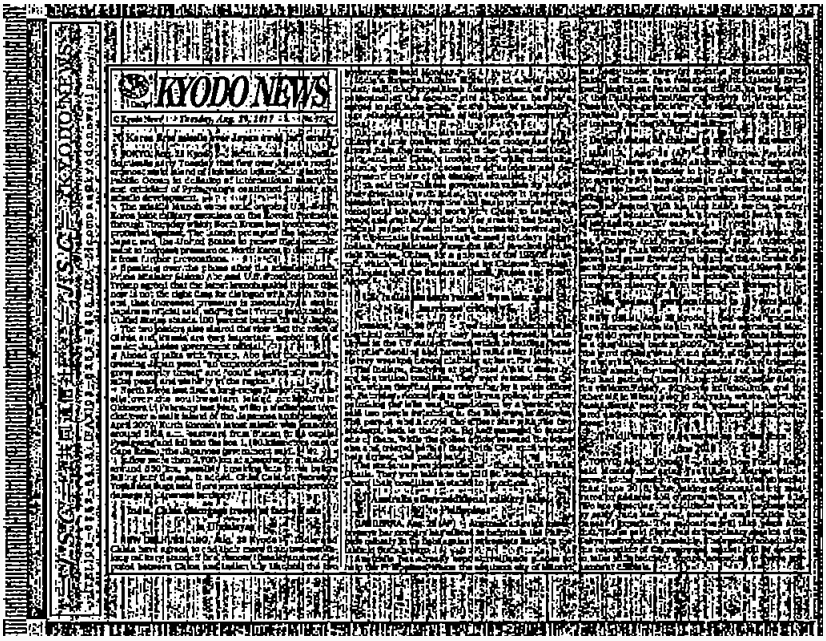
Tsukiji market to be moved no earlier than June 2018

TOKYO, Aug. 28 Kyodo – Tokyo Gov. Yuriko Koike said Monday the aging Tsukiji fish market will be moved to the nearby Toyosu waterfront area no earlier than June 2018, after taking additional safety measures to address soil contamination at the new site. "We are expecting the additional work to be completed by early June next year, including confirmation by a panel of experts. The relocation will take place after that," Koike said during an extraordinary session of the Tokyo metropolitan assembly. The specific schedule for the relocation of the renowned market will be decided in talks with industry groups, according to Tokyo government officials.

=JSC= 共同通信社 =JSC= KYODO NEWS

TEL(03-6252-8413 [FAX(03-6252-8805 [E-mail@kyodonews.jp

Original Image



Enhanced Image

Discussion of Results

The combination of contrast enhancement, sharpening, denoising, and binarization successfully improved the readability of the newspaper text. The adaptive thresholding followed by morphological operations provided the best results by handling varying illumination and cleaning up the binary image.

Pros

- CLAHE effectively improved local contrast in the newspaper
- Unsharp masking enhanced text edges
- Denoising removed noise that interfered with readability
- Adaptive thresholding handled varying illumination across the document
- Morphological operations cleaned up the binary text

Cons

- Some fine details in the newspaper may be lost
- Binary representation loses grayscale information in images
- May not work well for severely degraded or very low contrast newspapers

Quality Achieved

The quality of enhancement is good, with significantly improved text readability and contrast. The text stands out clearly against the background, making the newspaper content much easier to read compared to the original image.

Nameplate Enhancement

Technique(s) Used

Multiple techniques were applied: CLAHE Contrast Enhancement, Kernel Sharpening, Non-local Means Denoising, Adaptive Thresholding, Otsu's Thresholding, and Morphological Operations.

Why This Technique Was Selected

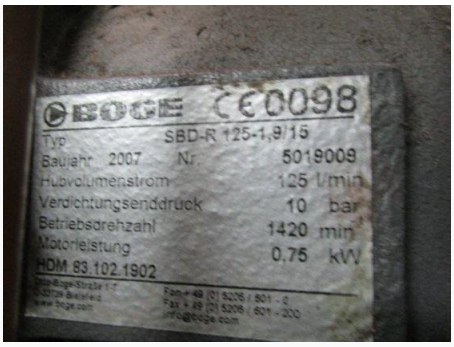
These techniques were selected to address the challenges in nameplate images: low contrast, noise, and varying illumination. CLAHE improves local contrast, kernel sharpening enhances edges, denoising removes noise, and thresholding methods improve text visibility.

Python Code

```
def enhance_nameplate(image_path):
    # Load image
    img = cv2.imread(image_path)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Method 1: Contrast enhancement using CLAHE
    clahe = cv2.createCLAHE(clipLimit=3.0, tileGridSize=(8, 8))
    clahe_applied = clahe.apply(gray)

    # Method 2: Sharpening with kernel
    kernel = np.array([[-1,-1,-1], [-1,9,-1], [-1,-1,-1]])
    sharpened = cv2.filter2D(clahe_applied, -1, kernel)

    # Method 3: Denoising
    denoised = cv2.fastNlMeansDenoising(gray, None, h=10,
templateWindowSize=7, searchWindowSize=21)

    # Method 4: Adaptive thresholding
    adaptive_thresh = cv2.adaptiveThreshold(
        clahe_applied, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C,
        cv2.THRESH_BINARY, 11, 2
    )

    # Method 5: Otsu's thresholding
    _, otsu = cv2.threshold(clahe_applied, 0, 255,
cv2.THRESH_BINARY + cv2.THRESH_OTSU)

    # Method 6: Morphological operations (opening and closing)
    kernel_morph = np.ones((3, 3), np.uint8)
```

```
    opening = cv2.morphologyEx(adaptive_thresh, cv2.MORPH_OPEN,
kernel_morph, iterations=1)
    closing = cv2.morphologyEx(opening, cv2.MORPH_CLOSE,
kernel_morph, iterations=1)

    # Return the original image and the sharpened image instead
of closing
    return img, sharpened
```

Output



Original Image

Enhanced Image

Discussion of Results

The combination of contrast enhancement, sharpening, denoising, and binarization successfully improved the visibility of the nameplate text. The adaptive thresholding followed by morphological operations provided the best results by handling varying illumination and cleaning up the binary image.

Pros

- CLAHE effectively improved local contrast in the nameplate

- Kernel sharpening enhanced text edges and details
- Denoising removed noise that interfered with readability
- Adaptive thresholding handled varying illumination
- Morphological operations cleaned up the binary text

Cons

- Some fine details in the nameplate may be lost
- Binary representation loses color and grayscale information
- May not work well for severely degraded or very low contrast nameplates

Quality Achieved

The quality of enhancement is good, with significantly improved text visibility and contrast. The text stands out clearly against the background, making the nameplate information much easier to read compared to the original image.

.

Conclusion

This project demonstrated the application of various image processing techniques for different types of images and challenges. The techniques included component extraction, enhancement of blurry images, noise removal, text enhancement, and visual enhancement of challenging images.

Each technique was selected based on its suitability for the specific image and challenge. The results showed significant improvements in image quality, component visibility, text readability, and noise reduction. The combination of multiple techniques often provided better results than individual methods.