Principles of Biological Vision

Assignment1: Edge detection

Objectives:-

This assignment's goal is to test our ability to analyze images and detect edges by putting different algorithms into practice and performing in-depth research. The tasks of this assignment includes:

- ☐ **Objective 1:** Image Preprocessing by converting each image to grayscale using techniques that ensure optimal contrast enhancement and dynamic range preservation.
- ☐ **Objective 2:** Complex Orientation Filters based on Gabor wavelets.
- ☐ **Objective 3:** Winner-Takes-All and Normalization.
- ☐ **Objective 4:** Comparative Analysis between original, gaussian noisy image and motion-blurred image by implementation of Structural Similarity Index (SSIM) and Edge F1-Score calculator.
- ☐ **Objective 5:** Visualization by highlighting the edges and features detected and creating gradient magnitudes and orientation map.

Challenges:-

The challenges of this assignment includes:

- ☐ **Challenge 1:** Selecting appropriate parameters like kernel sizes, cliplimit, gamma in image preprocessing part.
- ☐ **Challenge 2:** Implementation of gabor filters. Selecting the right frequency range for the gabor filter was a major challenge. After convolution, lower frequencies generated an overall white and an overall black output. In order to eliminate extreme values, gabor filters should also be normalized before being applied to the image.

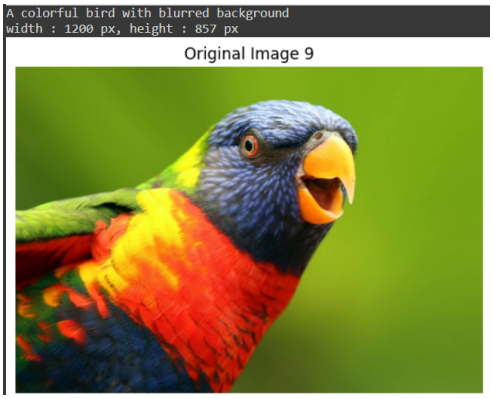
Python and its dependencies :

Libraries	Functions
CV2	Operations on images.
Numpy	Matrix Math etc.
Matplotlib	3D Graphs plots etc.

Objective 1: Image Preprocessing

Approach/Implementation Explanation:

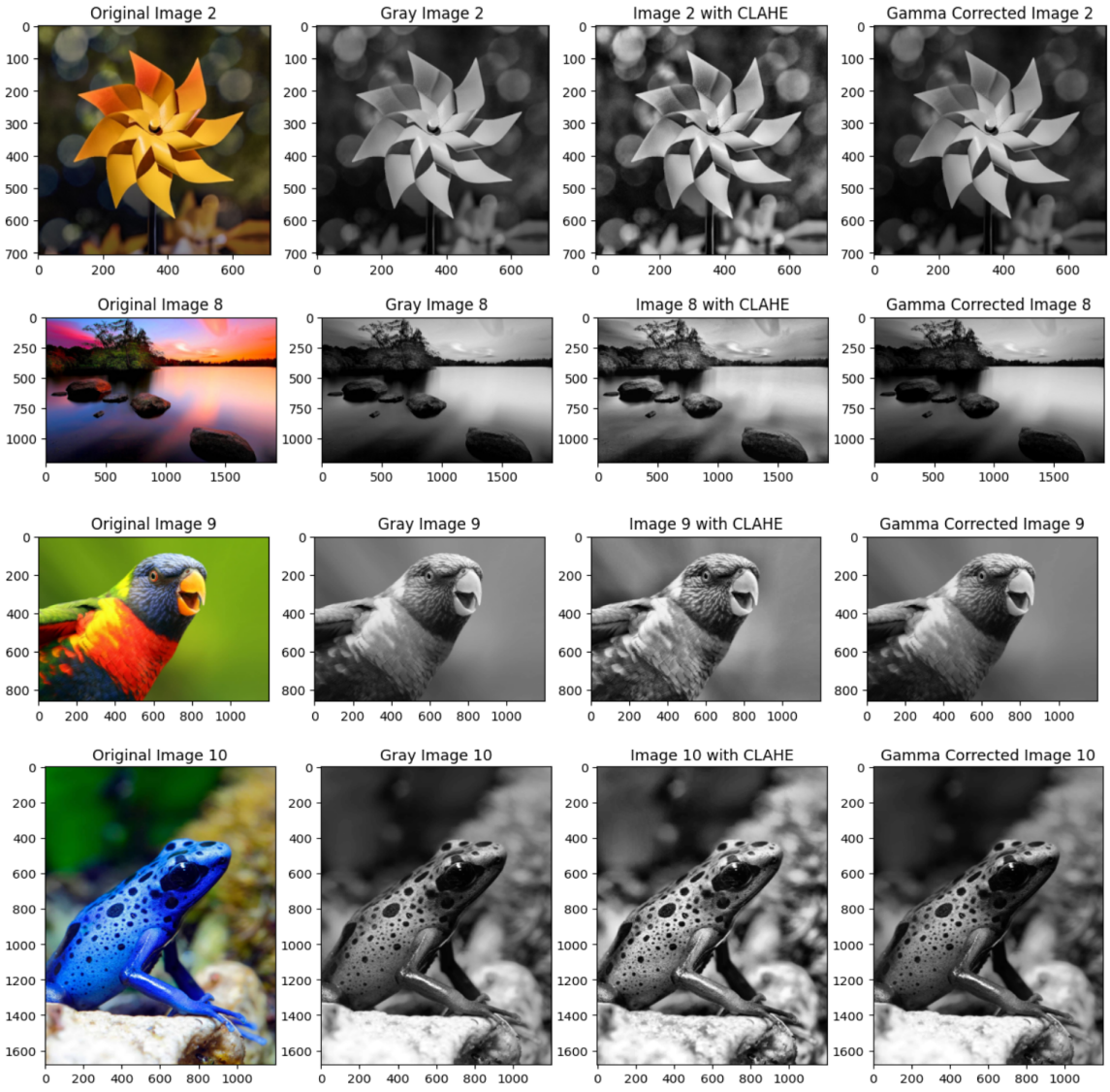
- A google drive folder is created containing the images used for this assignment . The link for the same is : Click here .
- The images with a wide range of complexities, perspectives, and lighting conditions are loaded using *Image* class from *PIL library*.
- Following are few images with brief description:



- All images are converted to using techniques that ensure optimal contrast enhancement and dynamic range preservation. To do the same following steps are taken:
    - First, the cv2.cvtColor(color_img, cv2.COLOR_BGR2GRAY) function is used to convert color images to grayscale images.
    - Next, grayscale images are subjected to Contrast Limited Adaptive Histogram Equalisation (CLAHE) in order to boost dynamic range while maintaining local details. As opposed to AHE, CLAHE limits contrast

enhancement and avoids over-amplification of noise by enhancing contrast within a block cell.

- Gamma correction is applied to the resulting CLAHE output in order to modify pixel intensities and enhance contrast and brightness perception. Gamma typically has a value between 0 and 3. Gamma = 1 indicates no correction, >1 indicates a positive value, while <1 further reduces the perceived brightness.I have observed that gamma = 1.1 is appropriate for my work.
- Initially, the grayscale picture is divided by 255.0 to normalize it. After that, it is boosted to the gamma power before returning to normal.
- Following are the gray scale images :

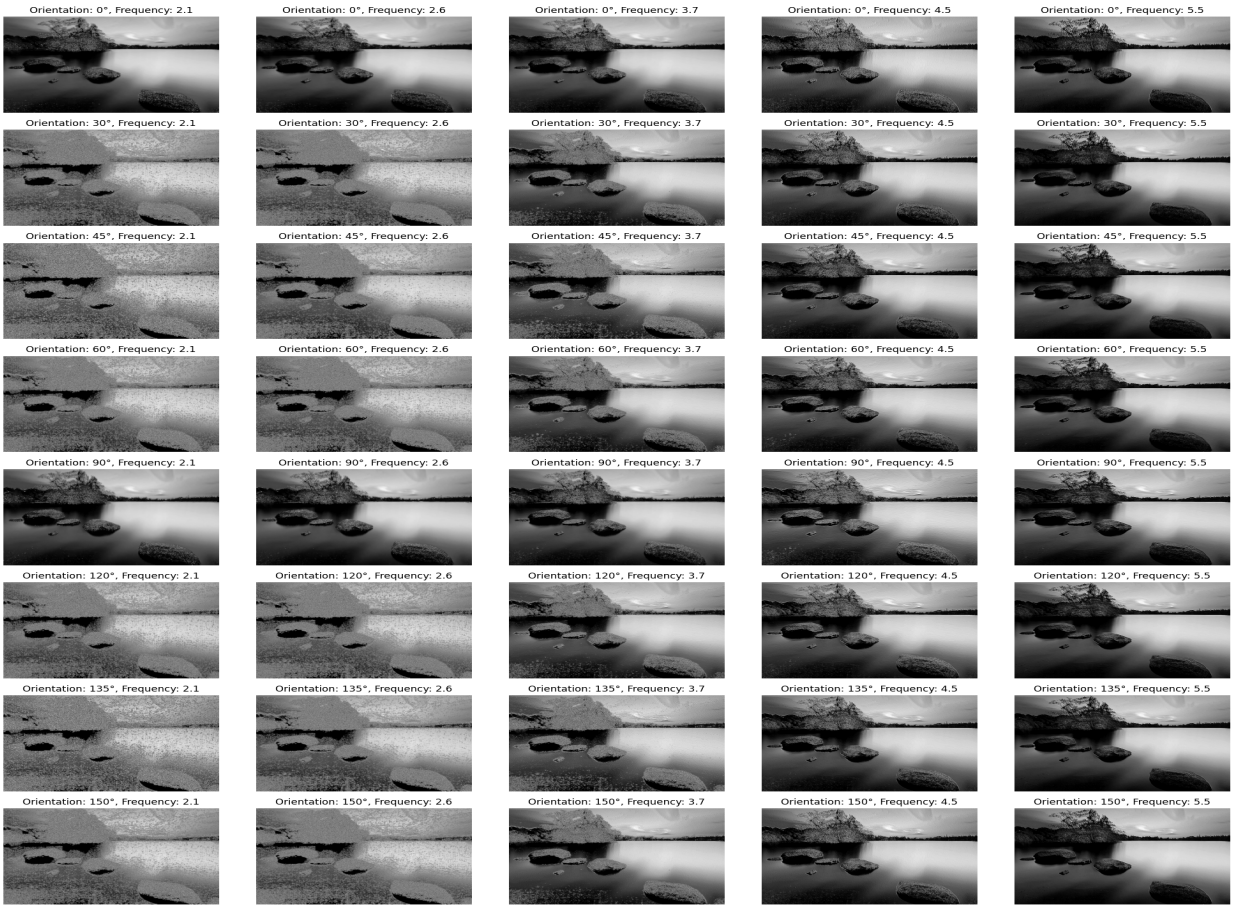


Objective 2:Complex Orientation Filters

Approach/Implementation:

- One high-resolution grayscale image from Part 1 for further processing is selected.The following image is selected for testing Gabor filters in this part and WTA algorithm in next part.

- Gabor filter is formed for theta values of {0,30,45,60,90,120,135,150} with frequencies {2.1,2.6,3.7,4.5,5.5}. For creating Gabor's filter, kernel size of 9*9 is selected. Gabor filter also performs Gaussian smoothing internally, for which sigma is selected as 1.1.
- The test image is now subjected to Gabor's filters of varying orientations and frequency using the cv2.filter2d() function, which carries out the convolution of the test image with filters.
- Now the visualization is done for the filtered images for each orientation emphasizing the extracted features.



- In the acquired image, we can see the change in feature direction. The way we rotate the orientation of Gabor's filter allows us to search for edges or features in those directions. For instance, the Gobor's filter will be more sensitive to the vertical characteristics if we alter theta between 0-90.
- Also it is evident from the aforementioned pictures that coarse features or low-frequency patterns are captured by gabor filters with low frequency values. Broad edges and smooth textures are examples of large-scale features that can be effectively extracted with these filters. We are able to modify the Gabor filter's response scale by changing the frequency parameter.
- Gabor filters that respond selectively to edges and features at different orientations in the image can be made by adjusting the orientation parameter. This is especially helpful for jobs where object orientations can vary, such as texture analysis and object recognition.

Objective 3:Winner-Takes-All and Normalization

Approach/Implementation:

- In this part a Winner-Takes-All algorithm is implemented, considering both magnitude and orientation information of the complex filtered images.
- WTA selects the most prominent response among all the Gabor filters with different orientations and frequencies for all the pixels considering both magnitude and orientation.
- The WTA output images are normalized using Adaptive Contrast Normalization (ACN) with cliplimit=2 and frame size = 10*10 to enhance feature visibility while preserving contextual relationships.
- Now the visualization is done for the WTA and normalized images, capturing intricate features and texture patterns.

Objective 4:Comparative Analysis

Approach/Implementation:

- At first the whole pipeline is implemented for edge detection which includes preprocessing , gabor filter creation , WTA implementation and normalization using ACN.

- After the first step, comparison and analysis of the results across different images, focusing on the impact of complexity, texture, and lighting on edge detection is done. For doing the same SSIM and edge f1 score are used.
    - Structural Similarity Index (SSIM) : It is employed to measure the degree of structural similarity while taking brightness, contrast, and structure into account—between the grayscale and ACN normalized images.The range of SSIM values is -1 to 1. When two images have the same luminance, contrast, and structure, they are represented by SSIM=1, and when two images completely differ in these respects, they are represented by SSIM=-1. Greater similarity between the photos is indicated by a higher SSIM score. The results of SSIM is :

```
SSIM score for image 1: 0.22383628990889223
SSIM score for image 2: 0.1969667465944559
SSIM score for image 3: 0.1352731537089994
SSIM score for image 4: 0.3101763100217681
SSIM score for image 5: 0.2579973989262218
SSIM score for image 6: 0.13482777840255863
SSIM score for image 7: 0.12393860725175722
SSIM score for image 8: 0.24737000014531826
SSIM score for image 9: 0.3446641274985803
SSIM score for image 10: 0.225634882845303
```

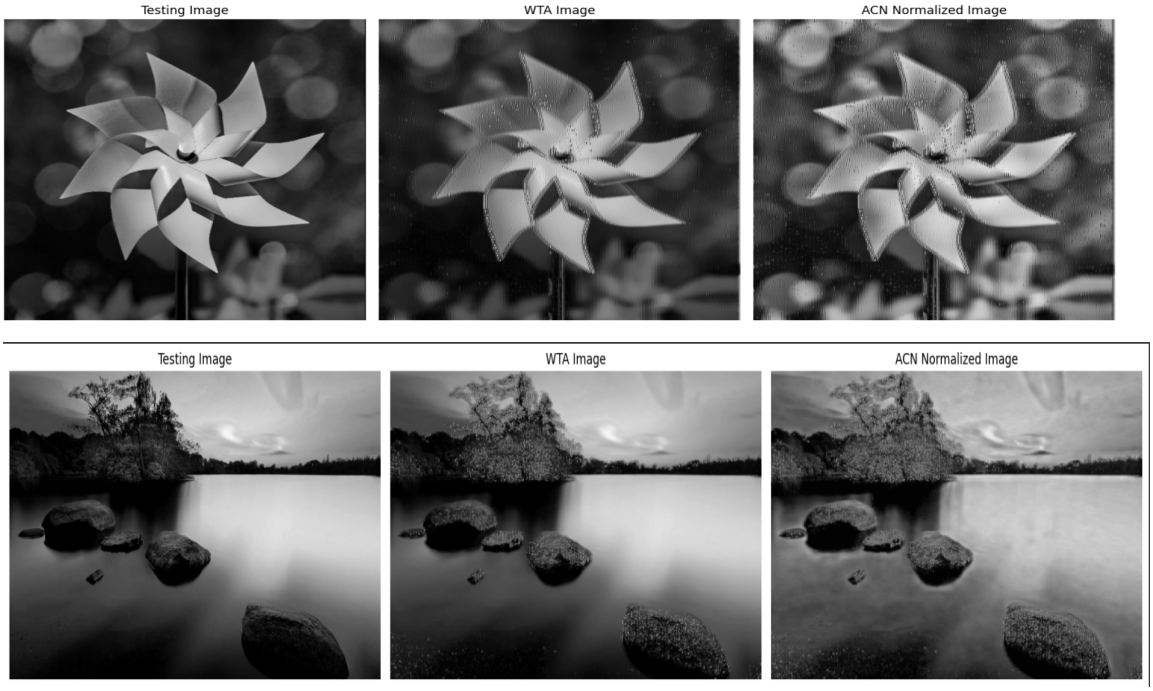
    - Edge f1 score : The F1 score is a metric commonly used in binary classification problems to evaluate the model's performance. The F1 score is the harmonic mean of precision and recall, providing a balance between the two metrics. To find f1 score first we have to convert detected edges vector to 0-1 vector by applying a threshold of on values. Any value < threshold is 0, otherwise 1. Then TP,TN,FP,FN are calculated to calculate edge f1 score .The result for edge f1 score is :

```
Edge f1 score for image 1: 0.8077341603416851
Edge f1 score for image 2: 0.588753716988196
Edge f1 score for image 3: 0.4293732884477857
Edge f1 score for image 4: 0.9312451137455944
Edge f1 score for image 5: 0.7137830268412999
Edge f1 score for image 6: 0.4426259402780944
Edge f1 score for image 7: 0.7109559503029058
Edge f1 score for image 8: 0.8469974476201162
Edge f1 score for image 9: 0.638402099970423
Edge f1 score for image 10: 0.7376361394004618
```

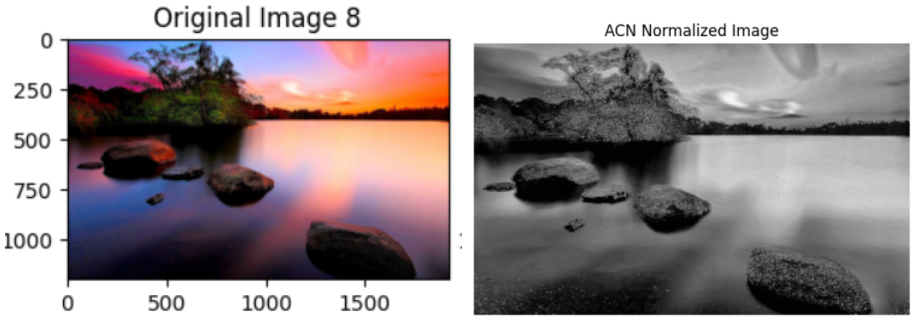
- Impact of illumination, texture, and complexity on edge detection
    - Complexity: Edge detection might be difficult in complex settings with lots of objects, occlusions, or messy backdrops. An increased frequency of false positives may result from the existence of several edges and overlapping structures. It may be difficult for edge detection algorithms to distinguish between secondary edges and the major limits of objects. Edge detection usually works effectively in smaller settings with distinct object boundaries and less clutter. In the below example if we use the train example edges are bot

detected properly due to its complex nature. On the other hand the edges of the bird image are clearly detected.

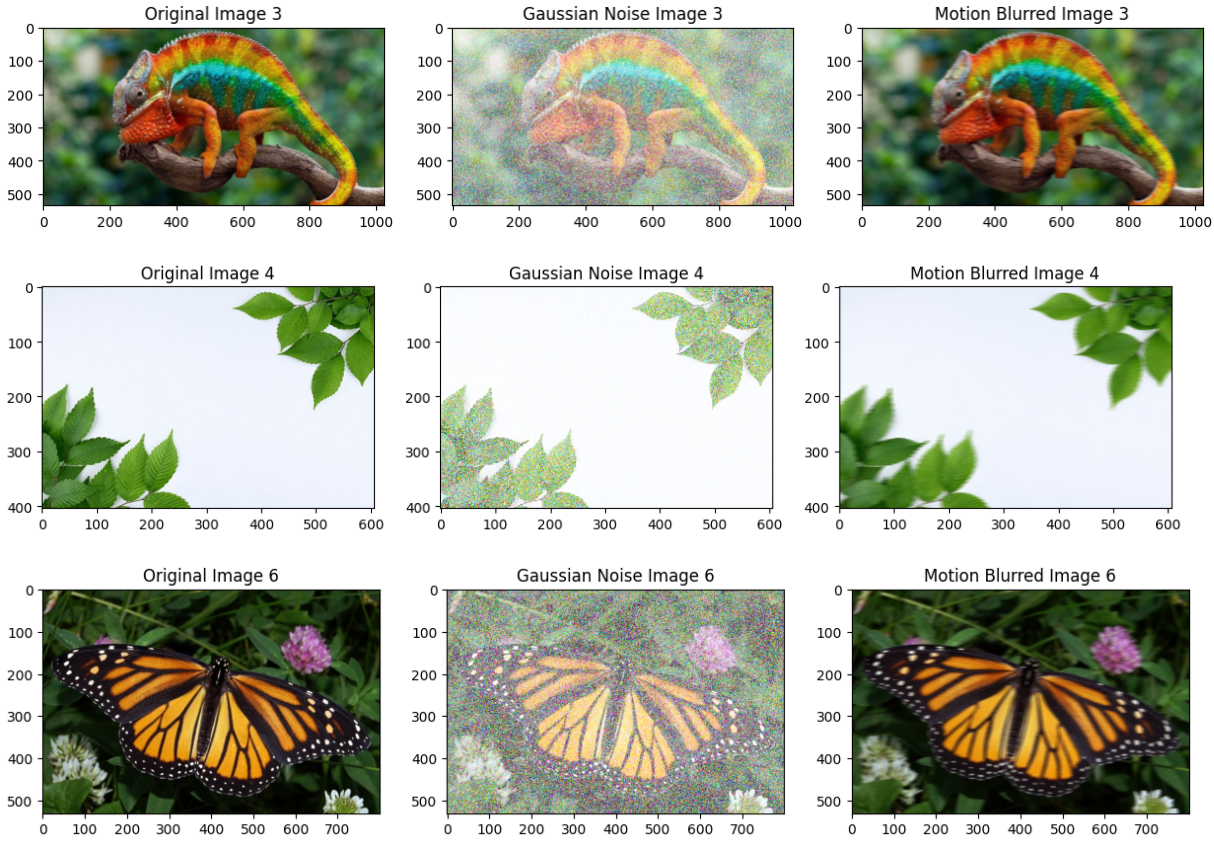
- Texture: A lot of small-scale edges can be produced by the presence of fine textures. These textures may cause edge detection algorithms to become more sensitive and generate more edge responses, which could result in a higher number of false positives. Accurate edge identification by edge detection algorithms may be difficult in areas with flat surfaces or little texture. Since there are fewer edge cues when there is no texture, true edges may be overlooked, which could lead to a lower recall. Below are two examples of less diverse and more diverse texture images and their detected edges.



- Lighting: Because there is less contrast between objects and backgrounds in low light, edge recognition algorithms may have trouble identifying edges. Edge detection may be impacted by lighting changes, such as highlights, shadows, and fluctuations in luminance. Edge detection methods may generate artifacts, such as false edges or breaks in detected edges, in regions with significant fluctuations in lighting. Below is the image for the same. In the normalized image there are some extra edges detected .

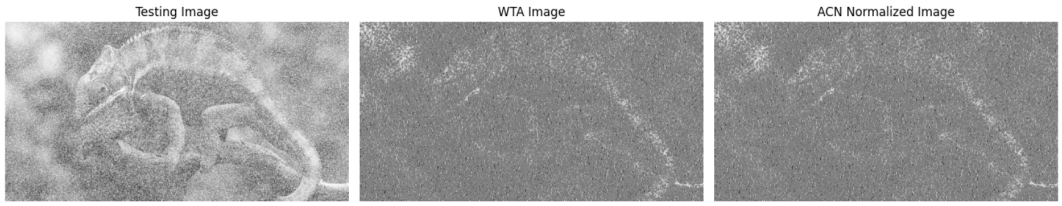


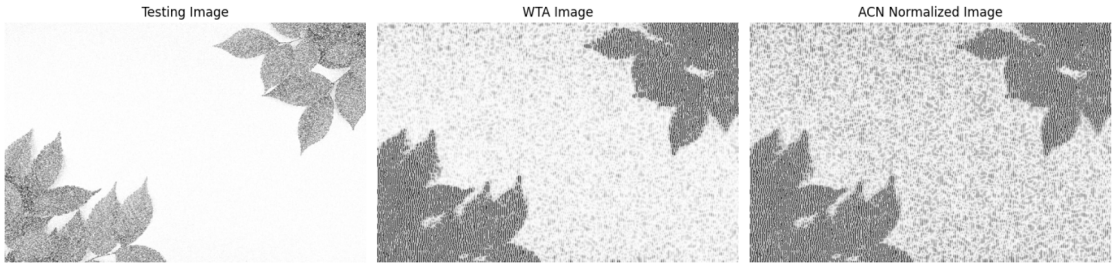
- After the above step the same experiment is done with gaussian noise added image and motion blurred images. Gaussian noise is added to the original image using a gaussian distribution with mean 0 and standard deviation 20. Motion blurring on images is done by using a motion-blur filter of kernel size 12*12.
- Following are the images for the same:

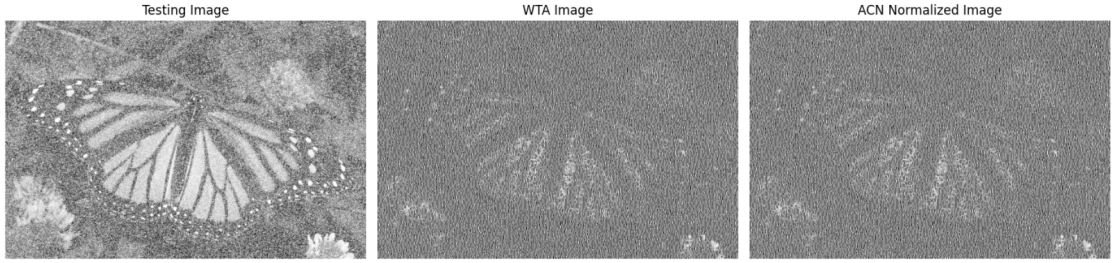


- Edge detection is performed on both gaussian noise images and motion blurred images. Below are some examples of the same :

  Gaussian Noise images:







  Motion Blurred Images:

- A comparison is done between original edge detection and gaussian noise edge detection and original edge detection and motion blurred edge detection.
- Original vs Gaussian noise:
    - SSIM score :

```
SSIM score for image 1: 0.08655252382368901
SSIM score for image 2: 0.010983297077169992
SSIM score for image 3: 0.025353708667450504
SSIM score for image 4: 0.053537898854808
SSIM score for image 5: 0.05669750581429413
SSIM score for image 6: 0.02930050893797888
SSIM score for image 7: 0.030949988840858724
SSIM score for image 8: 0.014496829197991397
SSIM score for image 9: 0.010109387196246917
SSIM score for image 10: 0.029713778349899706
```

    - Edge f1 score :

```
Edge F1-Score for Image 1 is: 0.6985971367263667
Edge F1-Score for Image 2 is: 0.41467569089254785
Edge F1-Score for Image 3 is: 0.46513064971751406
Edge F1-Score for Image 4 is: 0.8867854152205972
Edge F1-Score for Image 5 is: 0.5687101370282184
Edge F1-Score for Image 6 is: 0.3907767133208896
Edge F1-Score for Image 7 is: 0.5552403610409893
Edge F1-Score for Image 8 is: 0.530454476904035
Edge F1-Score for Image 9 is: 0.4559820579713578
Edge F1-Score for Image 10 is: 0.5398240080308011
```

- Original vs Motion Blurred:
    - SSIM score :

```
SSIM score for 1th image: 0.1708437322293673
SSIM score for 2th image: 0.1947058048469566
SSIM score for 3th image: 0.14086088044531112
SSIM score for 4th image: 0.2827124034289408
SSIM score for 5th image: 0.25123393495145974
SSIM score for 6th image: 0.22773171174073475
SSIM score for 7th image: 0.15679968782345233
SSIM score for 8th image: 0.23539738892665088
SSIM score for 9th image: 0.3040494719029552
SSIM score for 10th image: 0.21276454605874157
```

    - Edge f1 score :

```
Edge F1-Score for Image: 1 is 0.8081415519833468
Edge F1-Score for Image: 2 is 0.7947110141766631
Edge F1-Score for Image: 3 is 0.67562151223401
Edge F1-Score for Image: 4 is 0.934922429055845
Edge F1-Score for Image: 5 is 0.7781359740453384
Edge F1-Score for Image: 6 is 0.6455561422034495
Edge F1-Score for Image: 7 is 0.7547223790119924
Edge F1-Score for Image: 8 is 0.8574197607777312
Edge F1-Score for Image: 9 is 0.7223981466149493
Edge F1-Score for Image: 10 is 0.8261803576677048
```

- The SSIM score for edge identification of normal, motion blurred, and Gaussian noise-induced images is highest for normal images, followed by blurred images, and lowest for Gaussian noisy images in nearly all instances.Motion blur is not

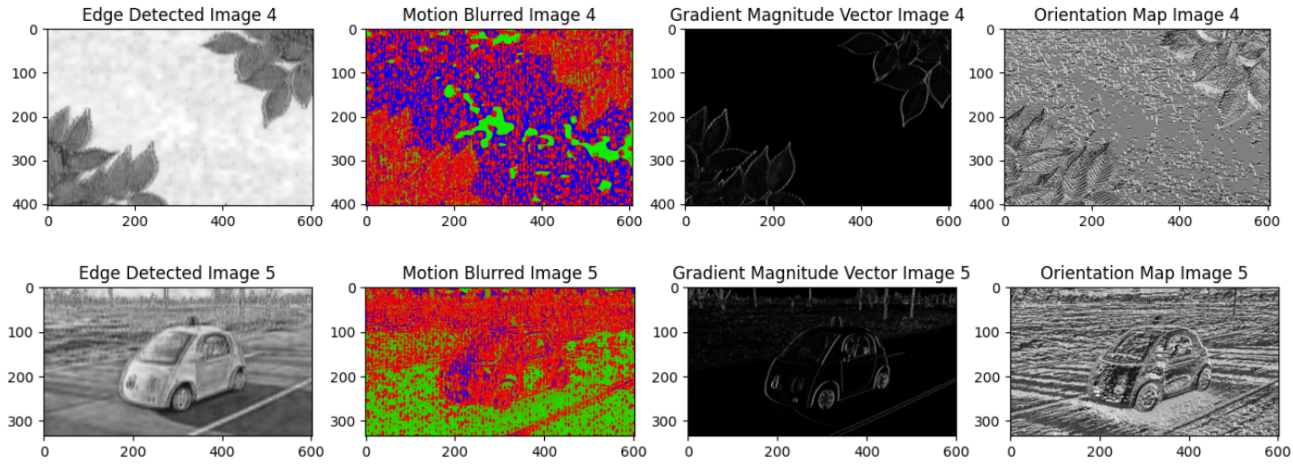
nearly as detrimental to edge recognition as noise is.Nevertheless, the normal image performs better than motion-blurred images when examining F1-scores, with a gaussian noisy image coming in second. This is because a high number of false negatives are induced by the motion blur, which lowers the f1-score in turn. In line with the SSIM score, the F1-score for edge identification of normal and motion-blurred images is nearly comparable.High-frequency noises can be lessened by the smoothing effect produced by averaging the blur pixel values along the edge direction. This will enhance the performance of edge detection.

- Now we are going to discuss the pros and cons of Gabor-based edge detection in challenging scenarios.
- <u>Pros of Gabor-Based Edge Detection:</u>
    - Because gabor filters are good at collecting textures, they can be used to identify edges in textured areas in difficult situations.
    - Gabor filters are flexible in collecting edges of varied sizes and orientations because they may be set to different scales and orientations, enabling multiscale and multi orientation analysis.
    - Because of their strong frequency localization capabilities, gabor filters are useful for precisely identifying edges at particular spatial frequencies.
    - Compared to certain other edge detection techniques, gabor-based edge detection is typically more noise-resistant, enabling improved performance in noisy settings.
    - By supplying pertinent edge information, gabor-based edge detection can help with later feature extraction tasks like texture analysis or object recognition.
- <u>Cons of Gabor-Based Edge Detection:</u>
    - Gabor-based edge detection can be computationally demanding, particularly when processing high-resolution images or employing a lot of Gabor filters. For real-time applications or systems with constrained processing resources, this complexity may be a disadvantage.
    - The choice of parameters, including the size, direction, and bandwidth of the Gabor filters, affects how well Gabor-based edge detection performs. Choosing the right parameters can be difficult and necessitate domain knowledge.
    - Certain situations may result in blurring or spreading of detected edges when using Gabor-based edge detection, especially if the parameters are not selected properly. This may cause the edges to lose their sharpness and delicate details.
    - Because of their preset structures and forms, gabor filters may not be as flexible in situations with intricate edge patterns or other non-conforming structures.

Objective 5:Visualization

Approach/Implementation:

- Following are maps used for visualization to highlight detected edges.
    - Edges overlay
    - Gradient Magnitude Map
    - Orientation Map
- Gradient magnitude and orientation maps were generated to visualize the strength and direction of edges in the processed images.Gradient magnitude and orientation maps encode the strength (magnitude = (sqrt(Gx 2 + Gy 2))) and orientations (theta = tan -1 (Gy /Gx)) of gradients at each pixel respectively. It represents the variation of pixel values and direction of edges and features.
- Below are the visualizations of some images .



- Comparison and discussion of the effectiveness of these visualization methods.
    - The particular task at hand and the required information determine how effective gradient magnitude and orientation maps are. In applications involving edge recognition, localization, or highlighting significant intensity change regions, gradient magnitude maps seem to be both efficient in terms of computation and effectiveness.
    - When you want to rapidly locate and highlight the edges in a photo, Edges Overlay is an extremely helpful tool. It sheds light on edge detection and local characteristics and offers a useful qualitative assessment of the edge detection results.

- However, gradient orientation maps are more appropriate if the task calls for orientation-based analysis, like texture classification or object recognition. They offer vital direction information that can be used to differentiate between various textured areas or objects with particular orientations.
- Ultimately, the selection between gradient magnitude and orientation maps is contingent upon the particular image processing task at hand and the nature of the information that is needed.

Conclusion summarizing the findings and lessons learned

The experiments explore edge-detection techniques in images with varying complexities, perspectives, and luminance. The approach includes image preprocessing, Gaussian smoothing, gamma correction, Gabor's filter, Winner Takes All algorithm, normalization, and visualization techniques. Techniques like Contrast Limited Adaptive Histogram Equalization (CLAHE) and gamma correction enhance contrast while preserving local features. Gabor filters extract features and textures at pixel level, while WTA algorithm selects optimal edges at each pixel. Normalization techniques like ACN improve contrast around edges. The experiments also evaluates the impact of noise and motion blurring on edge detection.