School

of

Electronics and Communication Engineering

Senior Design Project Report

on

Embedding Synthetic Patterns into Raw Images while Preserving Natural Image Statistics

By:

1. **Karthik** USN :01FE20BEC121
2. **Shaheen Khan** USN :01FE20BEC144
3. **Ananya Khed** USN :01FE20BEC149
4. **Radhika Horti** USN :01FE20BEC155

Semester: VII, 2023-2024

Under the Guidance of

Dr. Prabha C Nissimagoudar
Prof. Sahana Punagin

K.L.E SOCIETY'S
KLE Technological University,
HUBBALLI-580031
2023-2024

SCHOOL OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that project entitled **"Embedding Synthetic Patterns into Raw Images while Preserving Natural Image Statistics"** is a bonafide work carried out by the student team of **"Karthik: 01FE20BEC121; Shaheen Khan: 01FE20BEC144; Ananya Khed: 01FE20BEC149; Radhika Horti: 01FE20BEC155"**. The project report has been approved as it satisfies the requirements with respect to the Senior Design Project work prescribed by the university curriculum for BE (VII semester) in School of Electronics and Communication Engineering of KLE technological University for the academic year 2023-2024.

Dr. Prabha C N	**Dr.Suneeta.V.Budihal**	**Dr. Basavaraj S.A**
Guide	**Head of School**	**Registrar**

Prof. Sahana Punagin
Guide

External Viva:

Name of Examiners	**Signature with date**
1.	
2.	

ACKNOWLEDGMENT

The success of this initiative can be ascribed to the efforts and direction of a lot of people.Throughout the research, we were lucky to receive helpful advice from the Guides, Professors Prabha C. N. and Sahana Punagin. We would like to thank them from the bottom of our hearts. We are also appreciative to KLE Technological University, Hubballi, our prestigious university, for giving us this chance.

Dr. Suneeta V. Budihal, Head of the School of Electronics and Communication, has worked relentlessly to inspire and ensure the success of this attempt, and we would like to extend our sincere gratitude to her.

We also want to express our gratitude to the entire CIM-NVIDIA team for their help with this project. Last but not least, we would like to thank everyone who contributed significantly to this endeavour no matter how big or small.

-Karthik, Shaheen Khan, Ananya Khed, Radhika Horti.

ABSTRACT

The focal point of this report revolves around the innovative practice of embedding synthetic patterns into raw images, introducing a pioneering methodology to enrich visual data with artificially generated elements. The primary objective is to seamlessly incorporate computer-generated patterns into the raw image datasets, thereby achieving a harmonious fusion of naturalistic imagery and digitally augmented components. This exploration delves deep into the technical intricacies associated with the embedding process, dissecting its implications on image quality, authenticity, and its potential applications across diverse domains.

The study not only scrutinizes the theoretical foundations that underpin the integration of synthetic patterns but also offers practical insights into the challenges and opportunities encountered during implementation. By providing a thorough examination of the results obtained through experimentation, this report makes substantial contributions to the ever-evolving fields of computer vision and image processing. Through a nuanced analysis, the research endeavors to unravel the intricacies of synthetic pattern embedding, shedding light on its transformative potential in reshaping visual content manipulation and enhancement. This comprehensive investigation aims to broaden our understanding of the implications and applications of embedding synthetic patterns into raw images, thereby pushing the boundaries of technological innovation in the realm of visual data processing.

Contents

List of Figures

Chapter 1

Introduction

In computer vision, image processing, and related domains, embedding synthetic patterns onto pictures serves a variety of applications. Some examples may be: A dataset can be supplemented with synthetic patterns. Data augmentation is a typical strategy for increasing the quantity and variety of a training dataset, which can help machine learning models perform better. Fresh training examples that help the model generalize better by adding synthetic patterns to photos. Adding synthetic patterns to photos can be used to assess the resilience of image processing techniques or computer vision systems. These artificial patterns can be used to simulate real-world noise, distortions, or artifacts that the system may face in practice. Image processing or computer vision techniques must frequently be evaluated under controlled settings by researchers and engineers. They may evaluate how well these algorithms function in certain contexts by embedding synthetic patterns with known features. Synthetic patterns are useful for testing and troubleshooting. By providing predictable inputs, they can assist in identifying and resolving difficulties with image processing pipelines, object recognition, tracking, and other activities. Synthetic patterns can be used to watermark photos or movies in the context of digital media. Watermarks, which can be visible or invisible, are frequently used to protect intellectual property, authenticate material, and manage digital rights.

Synthetic patterns can be used to produce targets or markers that computer vision systems can easily identify. This is often employed in robotics, augmented reality, and computer vision applications that need the tracking of objects or markers. Synthetic patterns are employed in specific industries, such as computer graphics and virtual reality, to replicate natural textures, lighting conditions, or environmental impacts in order to create realistic and immersive virtual worlds. QR codes and barcodes are examples of synthetic patterns that are used for security and authentication. These patterns may include encoded information that machines can simply read and analyze.

In summary, embedding synthetic patterns onto images is a versatile technique with various applications in image processing, computer vision, data augmentation, testing, security, and more. These patterns are valuable tools for improving, evaluating, and extending the capabilities of image-based systems and technologies.

5. Histogram Comparison - Compare histograms of the two images. Histogram-based methods like the Bhattacharyya distance or the Earth Mover's Distance (EMD) can be used to measure image similarity.

The choice of method depends on the specific use case and what aspect of the images you want to compare (e.g., content, quality, structure, or objects). Some methods are more suitable for certain tasks, such as object detection, while others are more appropriate for measuring overall image similarity. The right approach may also involve a combination of these methods, depending on the complexity of the comparison task.

1.1 Motivation

In the fascinating realm of image processing, the integration of synthetic patterns into raw images stands as a captivating endeavor that merges creativity with technological innovation. This report delves into the intricacies of embedding synthetic patterns into raw images, exploring the potential to elevate visual aesthetics and unlock new dimensions of artistic expression. As we navigate through the technical nuances of this process, we uncover the transformative power it holds in augmenting the inherent beauty of raw images. Beyond the mere fusion of pixels, this exploration seeks to unravel the profound impact on visual storytelling, offering a glimpse into the future where the synthesis of reality and imagination converges seamlessly. This journey promises to unravel not only the technical advancements but also the artistic breakthroughs that lie at the intersection of raw imagery and synthetic ingenuity. Embarking on this report is an invitation to discover the boundless possibilities that arise when the artistry of human imagination intertwines with the precision of cutting-edge technology.

1.2 Objectives

Objectives of the provided problem statement are as follows:

- Capture RAW Images.
- To generate synthetic pattern using image to image stable diffusion
- To embed the synthetic patterns into raw images
- Construct a local statistics comparison block.
- Develop a software tool for the same.

1.3 Literature survey

Embedding synthetic patterns into raw images explores various techniques and methodologies employed by researchers in the field of image processing and computer vision. Researchers have extensively investigated the integration of synthetic patterns into raw images for applications such as image watermarking, data hiding, and augmented reality [1].
Classic methods, such as spatial domain techniques and frequency domain transformations, have been employed for embedding synthetic patterns, with studies focusing on robustness, imperceptibility, and capacity. Depending on the task's unique purpose and needs, there are numerous ways for embedding synthetic patterns onto photographs. Here are some typical methods: Image Editing Software: - To manually apply synthetic patterns to images, use image editing software such as Adobe Photoshop, GIMP, or other graphic design tools. This approach is appropriate for activities that need artistic or

creative design, such as adding logos, typography, or watermarks. Image Processing Libraries: - Use image processing libraries and tools, such as OpenCV, Python's Pillow, or MATLAB, to programmatically add synthetic patterns to pictures. This enables automation and exact control over the embedding process. Data Augmentation: In Python, you may use libraries like Augmentor or imgaug to enhance data in machine learning and computer vision applications. These libraries include routines for applying different modifications, such as adding synthetic patterns, to a picture collection. Image Generation Models: Train or employ generative models such as Generative Adversarial Networks (GANs) or Variational Autoencoders (VAEs) to create synthetic patterns that may be superimposed on photographs. GANs, for example, may be taught to produce realistic textures or objects that can be superimposed on existing photos. At the core of stable diffusion methods is the concept of a diffusion process. The process starts with an initial noisy observation (e.g., an image with added noise). Over a series of steps or time intervals, the data is gradually refined, with noise being reduced and structure emerging. This process is meant to mimic the way data evolves or "diffuses" over time. Additionally, recent advancements in deep learning approaches, including generative models and neural network-based methods, have shown promising results in seamlessly integrating synthetic patterns into raw images [2].
The survey also delves into the challenges associated with this task, such as maintaining image quality, resisting attacks, and ensuring compatibility with diverse image formats. Stable diffusion methods, often referred to as "stochastic stable diffusion" or "diffusion models," are a class of generative models used in machine learning and deep learning. These models are designed for various tasks, including image generation, data denoising, and image super-resolution. They are based on the concept of simulating the diffusion process of data using a series of stochastic transformations.Notable examples of stable diffusion models include the "Image GPT" model and the "Denoising Diffusion Probabilistic Models (DDPM)".

By analyzing the existing literature, the report aims to provide a comprehensive understanding of the current state-of-the-art techniques and trends in embedding synthetic patterns into raw images, paving the way for further advancements in this dynamic and evolving field[3].
The job of comparing two pictures is prevalent in many domains, including computer vision, image processing, and pattern detection. The purpose of picture comparison might range from recognizing similarities or dissimilarities to determining the extent to which two photos differ. Here are several strategies and methods for comparing two images:

Pixel-by-Pixel Analysis Comparing photos pixel by pixel is one of the most basic approaches. You may determine the absolute difference by subtracting the matching pixel values in one image from the other. The overall difference score reveals how different the two photos are.

The Structural Similarity Index (SSI) measures how similar two structures are. The SSIM metric is a popular way to compare two photographs. It evaluates brightness, contrast, and structure and assigns a score ranging from -1 to 1.

1.4 Problem statement

Develop software tool to embed synthetic patterns into raw images while preserving natural image statistics.

1.5 Organization of the report

The chapters of this report on Embedding Synthetic Patterns into Raw Images while Preserving Natural Image Statistics are as follows:

- Chapter 2: System Design- The functional block diagram for the given problem statement is defined in this chapter. It also gives an overview of the different tone curves available and highlights our methodology for selecting the final solution that works best for the given statement.
- Chapter 3: Implementation Details- This chapter defines the algorithm used for implementing the chosen solution, as well as a flowchart highlighting the various stages present in the solution's execution and explaining each stage.
- Chapter 4: Optimization- This chapter discusses the techniques used for optimization of the selected solution.
- Chapter 5: Results and Discussions- This chapter discusses the results obtained after the selected solution is executed.
- Chapter 6: Conclusions and future scope- This chapter discusses the future scope of the solution for improving the performance of the algorithm.

Chapter 2

System design

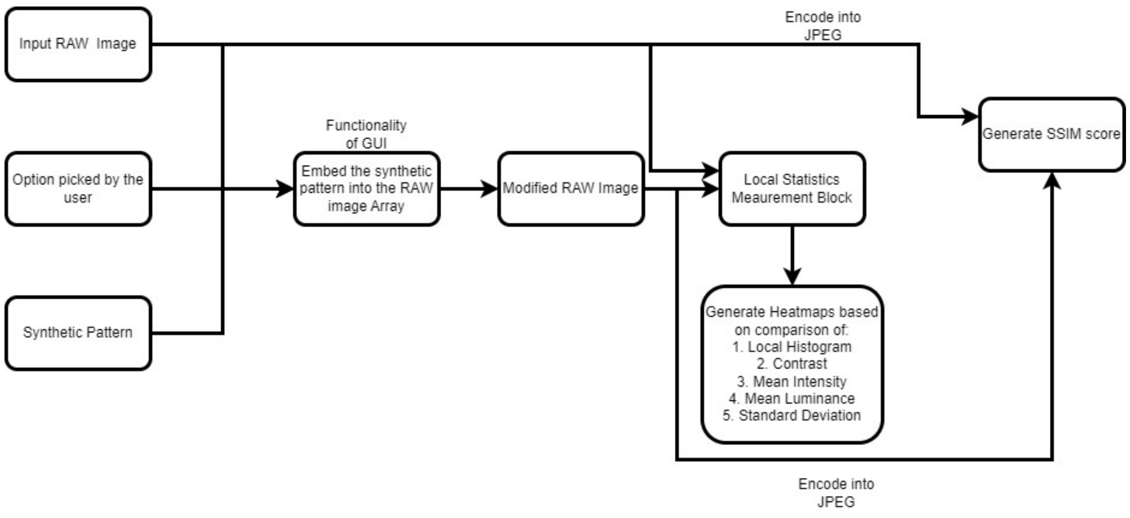
This chapter's goal is to provide a detailed description of the process of solving the given problem statement using a functional block diagram and architecture. Implementation for three different tone-curves is also discussed.

2.1 Functional block diagram

ARGUS is an API for acquiring images and associated metadata from cameras. The fundamental ARGUS operation is capturing that is acquiring an image from a sensor and processing it into a final output image.

For input, first the RAW image frames are extracted from the live capture.Other inputs include the option selected by the user and the type of of the synthetic pattern to be embedded. The options available to the user are as follows:

- Simple user input (defines the position where the synthetic pattern needs to be embedded).
- Based on matching the contrast of the synthetic pattern to the best suited position in the RAW image(defines only the size of the pattern that needs to be embedded).
- Based on matching the contrast of the synthetic pattern to the desired location so as to not disturb the input RAW image (defines the position where the synthetic pattern needs to be embedded).
- Based on matching the mean intensity of the synthetic pattern to the desired location so as to not disturb the input RAW image (defines the position where the synthetic pattern needs to be embedded).



Input RAW Image
Option picked by the user
Synthetic Pattern
Functionality of GUI
Embed the synthetic pattern into the RAW image Array
Modified RAW Image
Local Statistics Meaurement Block
Generate Heatmaps based on comparison of: 1. Local Histogram 2. Contrast 3. Mean Intensity 4. Mean Luminance 5. Standard Deviation
Encode into JPEG
Generate SSIM score
Encode into JPEG

Figure 2.1: Functional block diagram to embed synthetic patterns into raw images while preserving natural image statistics.

The synthetic pattern array that is selected is then embedded onto the input image array and the modified RAW image array is generated. The input RAW image and the modified RAW image is then fed into the local statistics measurement block that compares:

- Local histogram
- Contrast
- Mean luminance
- Mean intensity
- Standard Deviation

Heatmaps for the same are generated for better visual comparison Both the RAW images are enocded into JPEG format by making use of ARGUS APIs. The SSIM score for both the images is generated for performing the global statistics comparison.

2.2 Generating the Synthetic Patterns

In order to add unique visual features to a picture for uses such as data concealing, watermarking, or image enhancement, one must generate synthetic patterns for embedding onto images.In computer vision and image processing, the Image-to-Image (img2img) diffusion process is a method used for image creation, image-to-image translation, and image editing.We make use of this method to generate synthetic patterns:

2.2.1 Dot Pattern

TensorFlow is used in Python to build an img2img diffusion setup. In order to optimize the generation of a dot pattern over a gray backdrop, it defines a generator and discriminator network. The generator first generates random noise, which it then shapes to fit the form of the dot pattern and combines it with the backdrop and dot pattern to create a picture. After that, the discriminator evaluates these created pictures using both fictitious and authentic examples. The networks' parameters are adjusted during training to reduce the disparity between produced and real pictures over a period of epochs. Ultimately, a picture with a dot pattern is produced by the trained generator and stored as an png image

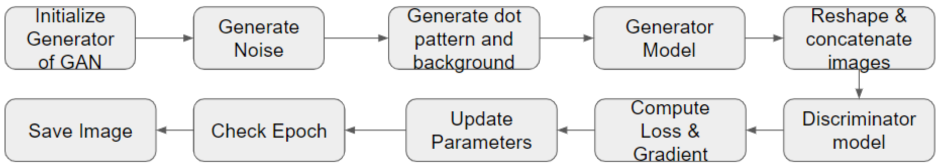


Figure 2.2: Functional block diagram for generating the Dot Pattern.

2.2.2 Uniform Color Patch

This setup uses an img2img diffusion technique to create a homogeneous color patch. It starts with a picture of random noise and, over a predetermined number of steps, diffuses it towards a desired color. The picture converges to the desired color by the repetitive adjustment of pixel values by the diffusion process. Once the picture has undergone 100 diffusion stages, it is transformed into a PIL image using the uint8 format and saved as an png image. To manage the diffusion process and the final look of the picture, the user may choose the target color (RGB values between 0 and 1), the number of diffusion stages, and the image size.

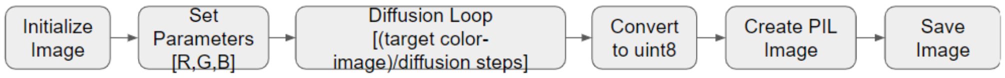


Figure 2.3: Functional block diagram for generating the Uniform Color Patch.

2.2.3 QR Code

This setup generates a QR code using the URL "https://www.nvidia.com/en-in/" using the ‘qrcode‘ library. It configures the version, error correction level, box size, and border of QR codes. The QR picture is created by the code with a white backdrop and a black foreground, and it is saved as "qr.jpg". It further uses the built-in image viewer to display the QR code. This script is handy for creating QR codes with text, contact information, or website URLs on them. It also lets you customize the characteristics of the QR code and the data that it encodes.

2.3 Framework for Embedding the Synthetic Patterns

2.3.1 Simple User Input

Capture RAW image and convert the raw file into array.Obtain synthetic pattern from diffusion model and convert the 8 bit image into an array. Obtain position and size of the pattern through the user.Check is the position defined by the user is within the image1.Compare the pixel values at the corresponding spots in picture 1 with those in image 2 multiplied by 256.Update the value of the pixel in image1 at that position to that the value of image2 multiplied by 256 if the pixel value in image1 is greater than the pixel value in image2 multiplied by 256.Then the pattern is embedded.

2.3.2 Matching the Contrast of the Synthetic Pattern to the best suited location in the input image

Capture RAW image and convert the raw file into array.Obtain synthetic pattern from diffusion model and convert the 8 bit image into an array. Obtain position and size of the pattern through the user.Calculate the standard deviation & determine the contrast of the pattern & Retrieves the dimensions (height and width) .Extract a block of the same size as the pattern from the image data array and then compare the contrast of the block with the pattern. If the contrast of the current block is smaller than the previous one, update the coordinates to embed the pattern to the current coordinates.

2.3.3 Matching the Contrast of the Selected block in the input image to the Synthetic Pattern

Capture RAW image and convert the raw file into array.Obtain synthetic pattern from diffusion model and convert the 8 bit image into an array. Obtain position and size of the pattern through the user.The function calculates the contrast of the selected pattern and the contrast of the specified block from the image.It adjusts the contrast of the pattern iteratively until it aligns within a threshold (in this case, within 5 units).Once the contrast is adjusted, it embeds the modified pattern into the specified block in the image.

2.3.4 Matching the Mean Intensity of the Selected block in the input image to the Synthetic Pattern

Capture RAW image and convert the raw file into array.Obtain synthetic pattern from diffusion model and convert the 8 bit image into an array. Obtain position and size of the pattern through the user.The function calculates the mean intensity of the selected pattern and the mean intensity of the specified block from the image.It computes the intensity difference between the block and pattern, then normalizes the pattern array.The function then adjusts the intensity of the pattern by applying the intensity difference and normalization. Then the pattern is embedded.

We have chosen to implement four different variations to embed synthetic pattern into the input RAW image as mentioned above to be able to compare the local and global statistics. The local statistics that are being compared are local histogram, contrast, mean luminance, mean intensity and standard deviation. The statistics are displayed in the form of a heat map for better visual representation.Then the RAW images are encoded into JPEG format to be able to compare the global statistic in the terms of SSIM (Structural Similarity Index).The methods are then compared on the basis of these statistics to find the most optimized method to embed synthetic pattern onto the RAW image

Chapter 3

Implementation details

This chapter discusses the solution chosen and defines the algorithm used in the project. The project flowchart is also included in this chapter.

3.1 Specifications and final system architecture

3.1.1 NVIDIA Jetson AGX Orin kit specifications:

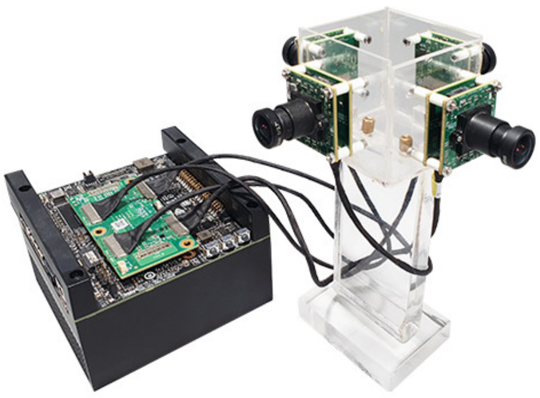
The NVIDIA Jetson AGX Orin kit is created with the highly effective and power efficient AGX Orin module that is capable of generating up to 275 Terra Operations Per Second (TOPS) and is eight times more powerful than the NVIDIA Jetson AGX Xavier in the same small shape. Power can be varied from 15W to 50W. The kit consists of the module, a heat sink, and a reference carrier board, alongside an 802.11ac/abgn wireless Network interface controller, USB-C power controller, and cord, in addition to the quick start.

The NVIDIA Jetson AGX Orin architecture has its foundation on the NVIDIA Ampere architecture GPU and the Arm Cortex - A78AE CPU. It also has next-generation deep learning and vision accelerators built in. High-speed IO, memory bandwidth of 204GB/s, and 32GB of DRAM is provided by the kit. It can also feed numerous AI application pipelines at a given time.The key advantage of Jetson Orin is that developers can now deploy massive and intricate models to address real world problems in the field of AI.
To summarize its features:

- It can perform up to 275 TOPS and performs up to eight times better than than its counterpart Xavier in a comparable small shape factor, with power varying from 15W to 50W. It signifies a major improvement in robotics and edge AI.
- 12-core Arm Cortex-A78AE v8.2 64-bit CPU, 2048 NVIDIA CUDA cores, 64 tensor cores, and Ampere architecture included
- Multiple AI programs may run simultaneously thanks to the on-board 64GB eMMC, 204 GB/s of memory bandwidth, and the kit also provides 32 GB of RAM.
- High-speed I/O includes 22 PCIe Gen4 lanes, Gigabit Ethernet, four XFI interfaces for 10 Gigabit Ethernet, a Display Port, sixteen MIPI CSI-2 lanes, USB3.2 interfaces, and a 40-pin header.
- PVA v2, a next-generation vision accelerator engine, in addition to a multi-standard video encoder and decoder is also included in the kit.

- Jetpack 5.0 supports the entirety of JetPack and use-case specific software platforms, including specific applications for robotics and smart cities , and installs Ubuntu 20.04.

(a) Jetson Orin Kit



(b) Camera Sensors

3.1.2 e-CAM22_CUOAGX IMX 462 camera specifications:

This is an 8 MP ultra low light MIPI CSI 2 camera for Jetson AGX Orin kit. This camera is based on the SONY STARVIS IMX 462 CMOS image sensor which helps it to produce superior quality images in both visible and low light conditions. This is made possible by its large sensor size of 1/1.2" and pixel size of the frame captured by the sensor is around of 2.8 micrometer.This big sensor size, in addition to the camera sensor's outstanding resolution, allows it to capture additional features at the scene of interest.

- Sensor: IMX462 sensor from SONY®Focus
- Type: Fixed focus
- Sensor Resolution: 8MP
- Chroma: Color
- Shutter Type: Electronic Rolling
- ShutterOptical Format: 1/1.2"
- Output Format: RAW Bayer 10/12-bit
- Pixel size: 2.8 um x2.8 um
- Sensor Active Area: 3864H x 2180V
- Array Size:3864(H) $\times$ 2200(V)
- Holder: CS-Mount
- DFOV:35.75 (with the lens provided by e-con)

3.2 Hardware Setup

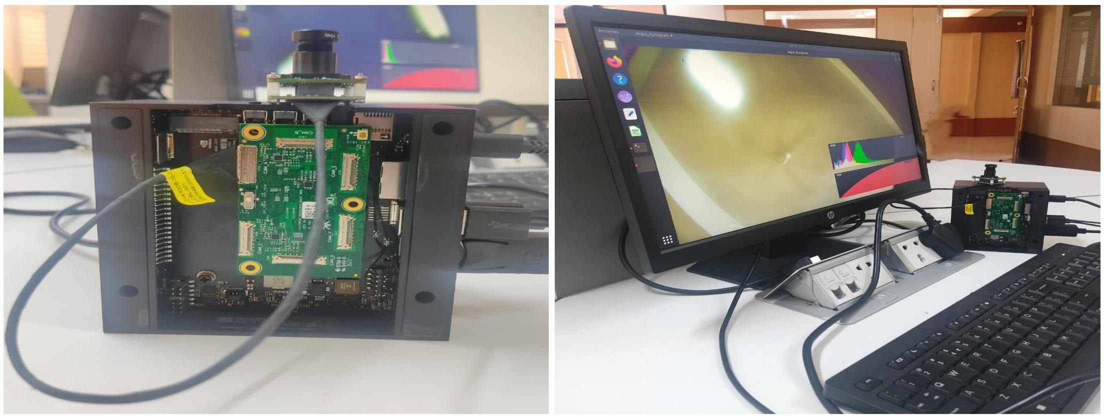

(a) Jetson Orin Kit and Camera sensor (b) Connections

Figure 3.2: Hardware Setup

3.3 Algorithm

The algorithm of the solution for dynamic display of histogram and tone-curve has the following steps:

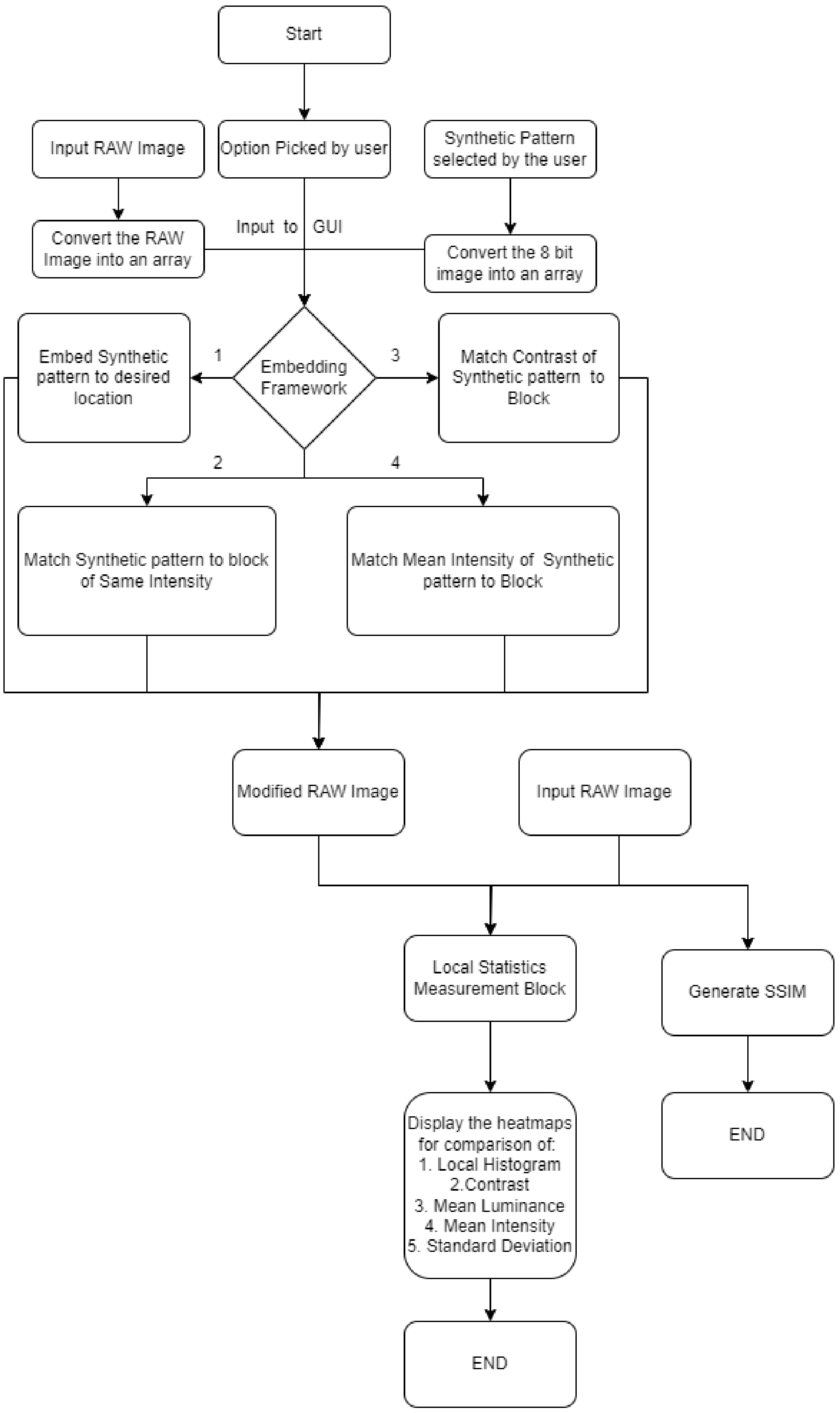
- Step1:From the live capture obtain the RAW Image frames.
- Step2:The user can pick an option from the available methods to embed
- Step3:Prompt the user to select the Synthetic Pattern to be embedded amongst the available options (QR Code, Dot Pattern, Uniform Color Patch and Noise schedule).
- Step4:Input the above three values into the GUI for processing.
- Step5:Select the framework for embedding based on the option selected by the user.
- Step6:After obtaining the modified raw image array, feed both the input RAW image and the modified RAW image into the Local Statistics measurement block.
- Step7: It determines dissimilar blocks based on contrast, mean brightness, mean intensity, and standard deviation differences using the Chi-square dissimilarity measure for histogram comparison.
- Step8: The algorithm iterates across blocks with predetermined sizes, comparing their properties, and storing the locations of blocks that differ. Then, by creating heatmaps that show the locations of the dissimilar blocks in each comparison, it visualizes these differences.
- Step9:Encode both the RAW images into JPEG format.
- Step10:Generate the SSIM score after comparing the two images for comparing the global statistics.

3.4 Flowchart for Embedding Synthetic Patterns into Raw Images while Preserving Natural Image Statistics

The flowchart of the solution for Embedding Synthetic Patterns into Raw Images while Preserving Natural Image Statistics can be represented as follows in Figure 3.4.

As shown in the flowchart, the input to the GUI are taken in the form of the input RAW image, the option of embedding format from the user and the the synthetic pattern to be embedded selected by the user. After which the appropriate form of embedding is carried out based on the option selected by the user.The corresponding modified image array is obtained as result. The modified array contains the synthetic pattern embedded onto the input RAW image.

The next step is comparison of the modified image and the input image which is carried out by comparison of local as well as global statistics. In terms of local statistics, we compare local histogram, contrast, mean luminance, mean intensity and standard deviation. This comparison is then presented in terms of a heatmap wherein the dissimilar blocks are represented as cold regions. For global statistics comparison we make use of SSIM.



Start
Input RAW Image
Option Picked by user
Synthetic Pattern selected by the user
Convert the RAW Image into an array
Input to GUI
Convert the 8 bit image into an array
Embedding Framework
1
3
Embed Synthetic pattern to desired location
Match Contrast of Synthetic pattern to Block
2
4
Match Synthetic pattern to block of Same Intensity
Match Mean Intensity of Synthetic pattern to Block
Modified RAW Image
Input RAW Image
Local Statistics Measurement Block
Generate SSIM
Display the heatmaps for comparison of: 1. Local Histogram 2.Contrast 3. Mean Luminance 4. Mean Intensity 5. Standard Deviation
END
END

Figure 3.3: Flowchart for Embedding Synthetic Patterns into Raw Images while Preserving Natural Image Statistics

Chapter 4

Optimization

This chapter discusses the techniques used for optimization of the selected solution.

4.1 Method of Optimization

The optimized solution for the given problem statement used is as follows :

- Embedding the synthetic pattern onto the best matched block in terms of contrast

  Go through the image iteratively in terms of blocks and compare the contrast of each block with the synthetic pattern. Embed the pattern onto the best suited block.

- Matching the contrast of the selected block by adjusting the contrast of the synthetic pattern

  Adjust the contrast of the pattern iteratively until it aligns within a threshold (in this case, within 5 units).Once the contrast is adjusted,embed the modified pattern into the specified block in the image.

- Matching the mean intensity of the selected block by adjusting the mean intensity of the synthetic pattern

  Compute the intensity difference between the block and pattern, then normalize the pattern array. The function then adjusts the intensity of the pattern by applying the intensity difference and normalization.

4.2 Selection and justification of optimization method

A comparative study of all the above methods was carried out. The most above appropriate method to embed synthetic patterns onto the input image such that all the natural statistics are preserved is by embedding the pattern onto the block that has the same or similar contrast to it, as in this case the SSIM was found to 1.0 that is the image is virtually unchanged even though the pattern has been embedded onto it and the local statistics measurement block did not find any dissimilar blocks when both the images where compared. However, in case of practical scenarios it would be more appropriate to match the contrast/ intensity of the selected block to that of the synthetic pattern, as in these cases the SSIM score was near 0.99 which shows that the modified image is almost similar to the input image

Chapter 5

Results and Discussions

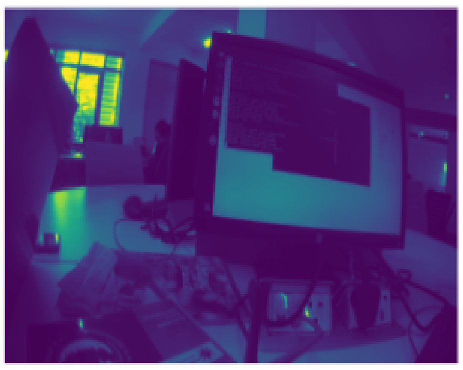
In this chapter, we discuss the results and compare the results to the desired output.

5.1 Result Analysis

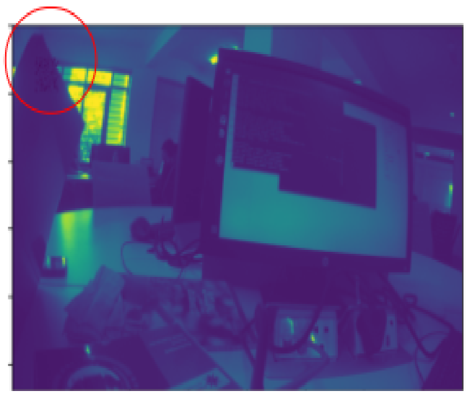
We were able to obtain the modified RAW image with the synthetic pattern embedded onto it after executing our Sample Code.Both the RAW images were also converted in JPEG images using ARGUS sample application.The local statistics that were considered for comparison were local histogram, contrast, mean luminance, mean intensity and standard deviation. To give the user a better idea of the dissimilar blocks present in the modified image, we displayed them in the form of a heatmap where the dissimilar blocks where represented as cold regions. We also verified the global statistics for the both images by making use of SSIM .The frames collected before and after the synthetic pattern was embedded onto them are shown in the following pictures, for indoor lighting conditions:

5.1.1 Different Embedding Frameworks:

CASE 1 (Simple User Input):We have presented the frames to which the synthetic pattern was embedded on the location of (100,100). The pattern that is embedded is highlighted.



(a) Input Image



(b) Output RAW Image (100,100)

Figure 5.1: Simple User Input

(a) Input Image

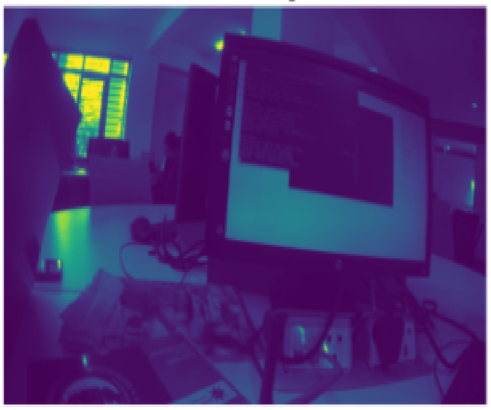

(b) Output RAW Image (1212,707)

Figure 5.2: Embedding the synthetic pattern onto the best matched block in terms of contrast

CASE 2 (Best suited Block): We have presented the frames to which the synthetic pattern was embedded on the location of (1212,707) where best suited block is located

CASE 3 (Matching the Mean Contrast):We have presented the frames to which the synthetic pattern was embedded on the location of (100,100).

(a) Input Image

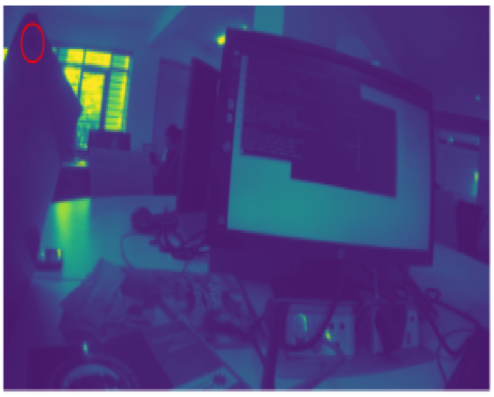
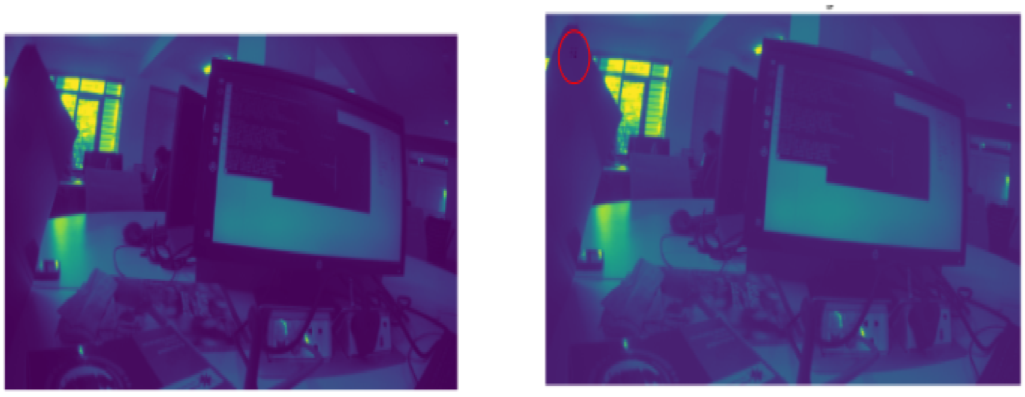

(b) Output RAW Image (100,100)

Figure 5.3: Matching the contrast of the selected block by adjusting the mean intensity of the synthetic pattern

CASE 4 (Matching the Mean Intensity): We have presented the frames to which the synthetic pattern was embedded on the location of (100,100).



(a) Input Image (b) Output RAW Image (100,100)

Figure 5.4: Matching the mean intensity of the selected block by adjusting the contrast of the synthetic pattern

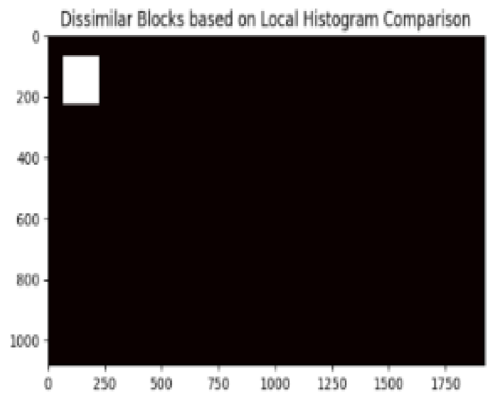
Table 5.1: Comparison of SSIM scores of the Embedding Framworks

Embedding Framework	SSIM Score
Simple User Input	0.9973
Contrast of Pattern	1.0
Contrast of block	0.9988
Mean Intensity of block	0.9990

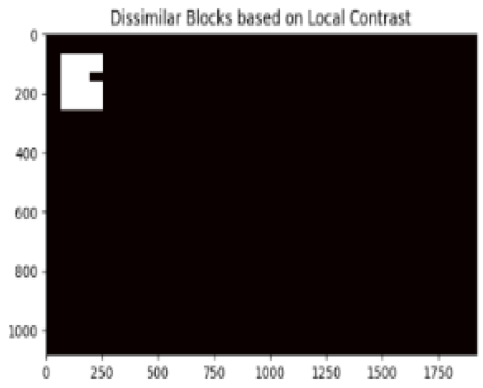
The table above compares the SSIM scores of the various embedding frameworks that the GUI makes use of. High similarity is shown across a range of characteristics in the examined picture comparison utilizing SSIM scores and the embedding architecture. Strong similarity is demonstrated by the remarkably high SSIM score of 0.9973 for simple user input. It is clear that the option two that is embedding based on the contrast of the pattern is the most efficient option with the SSIM score of 1.0, so as to be able to fully preserve the natural image statistics.Furthermore, there is a notable similarity between the contrast and mean intensity comparisons, with scores of 0.9988 and 0.9990, respectively, indicating almost comparable characteristics.Overall, the findings show a strong and constant similarity between the photos on a variety of measures, confirming their integrity and close likeness within the assessed limits.

5.1.2 Local Statistics Comparison:

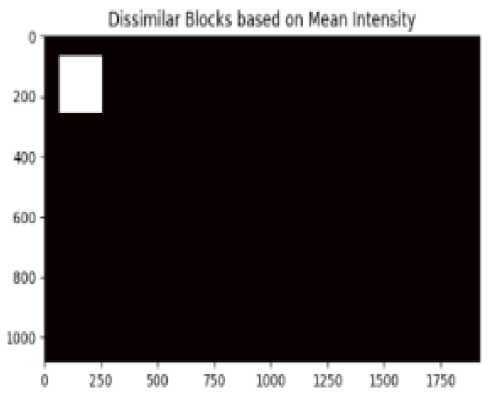
A heatmap generated using the 'hot' colormap and 'nearest' interpolation is typically used to visualize dissimilarity or intensity values between pixels or elements in an array. White often symbolizes the hottest (highest) values in a 'hot' colormap display. In the context of picture dissimilarity, this might refer to places where the patterns are highly similar or have very low dissimilarity to the surrounding regions. In the context of picture dissimilarity, orange zones may represent places where the embedded patterns or elements are fairly similar or dissimilar to their surroundings.



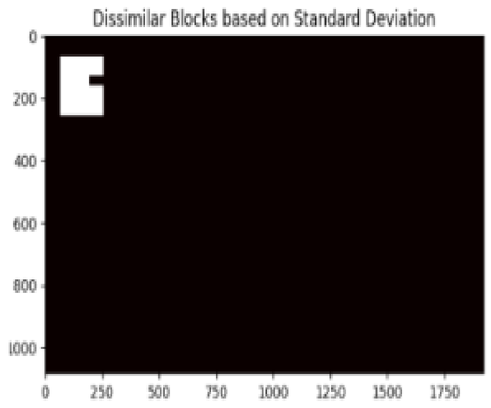
(a) Local Histogram



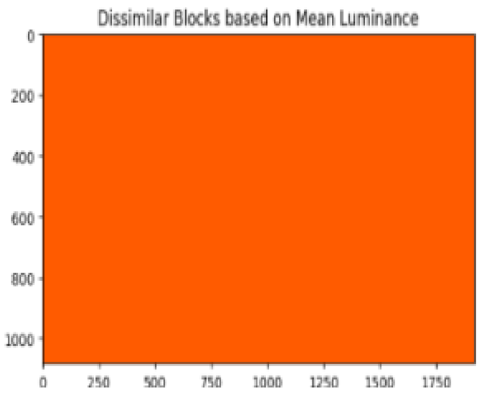
(b) Contrast



(c) Mean Intensity

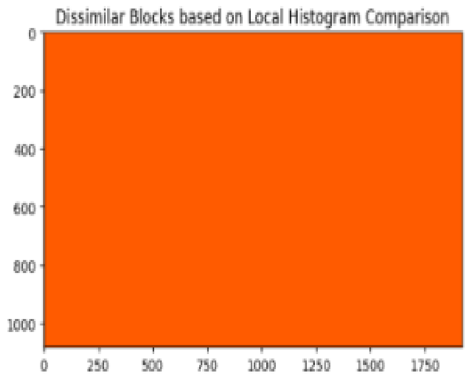


(d) Standard Deviation

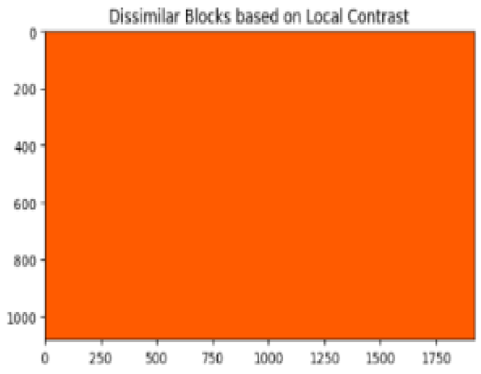


(e) Mean Luminance

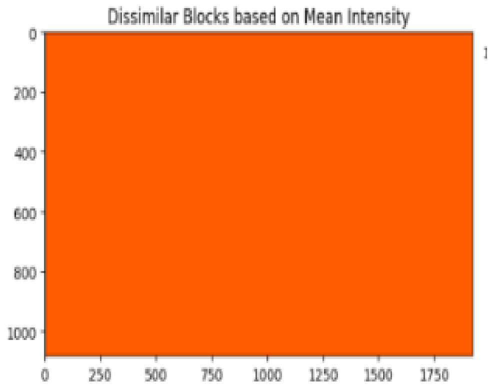
Figure 5.5: Heat-maps representing the dissimilar blocks in terms of local statistics for CASE 1 (Simple User Input)



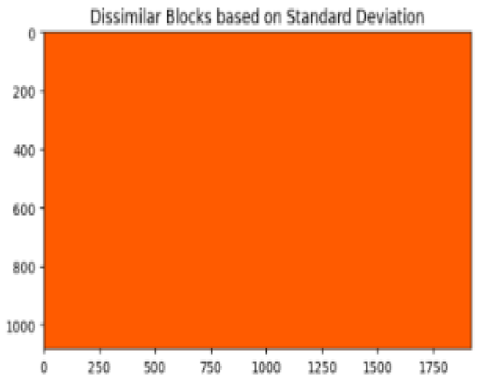
(a) Local Histogram



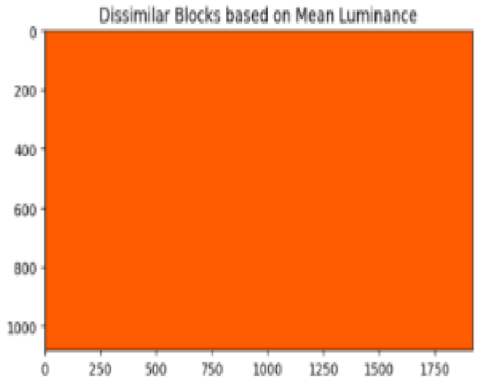
(b) Contrast



(c) Mean Intensity

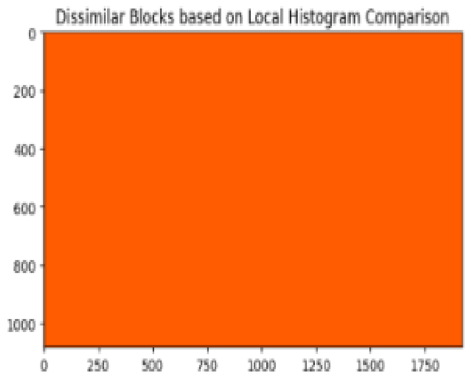


(d) Standard Deviation

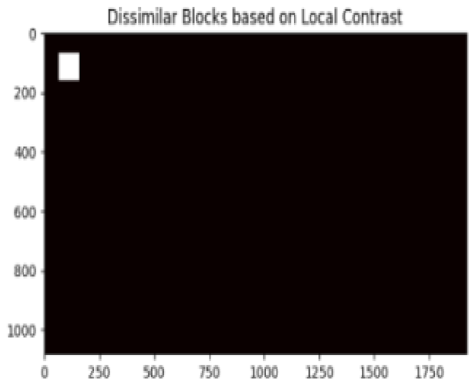


(e) Mean Luminance

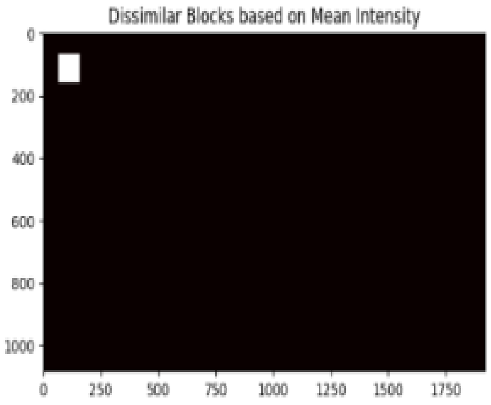
Figure 5.6: Heat-maps representing the dissimilar blocks in terms of local statistics for CASE 2 (Best Suited Block)



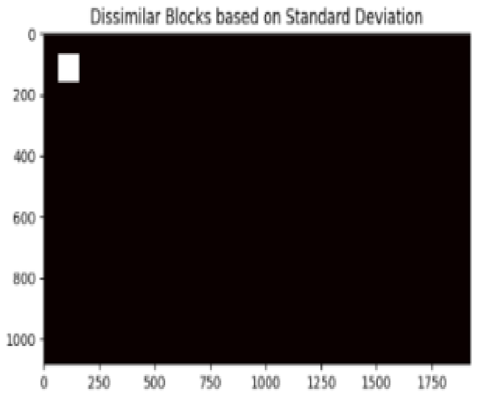
(a) Local Histogram



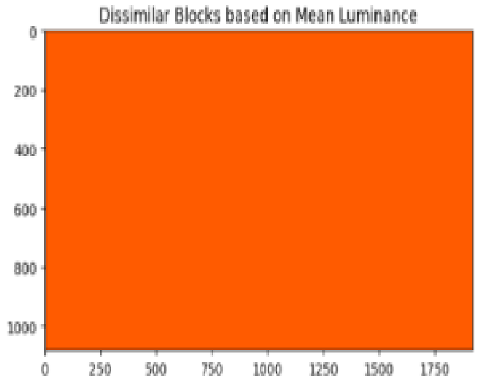
(b) Contrast



(c) Mean Intensity

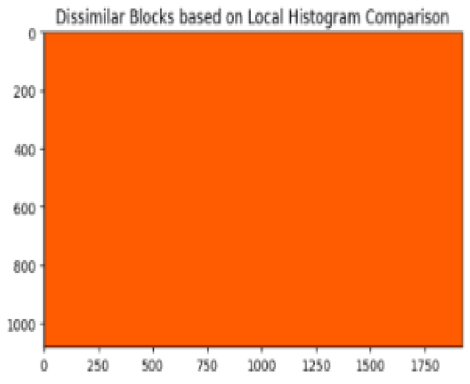


(d) Standard Deviation

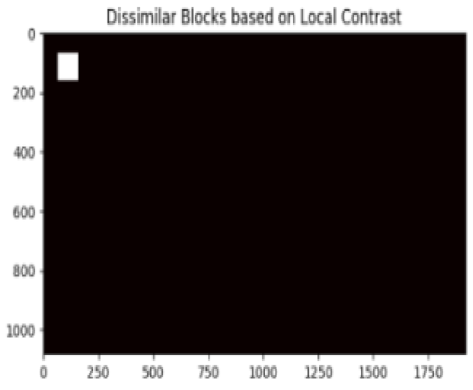


(e) Mean Luminance

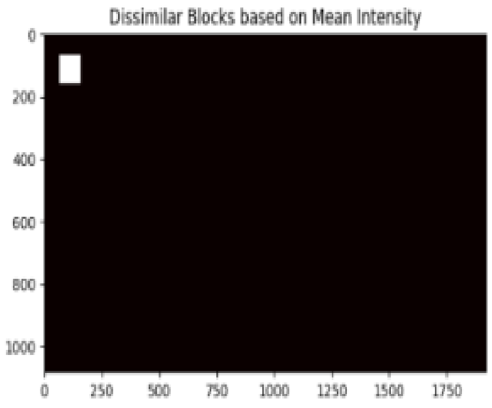
Figure 5.7: Heat-maps representing the dissimilar blocks in terms of local statistics for CASE 3 (Match the Contrast)



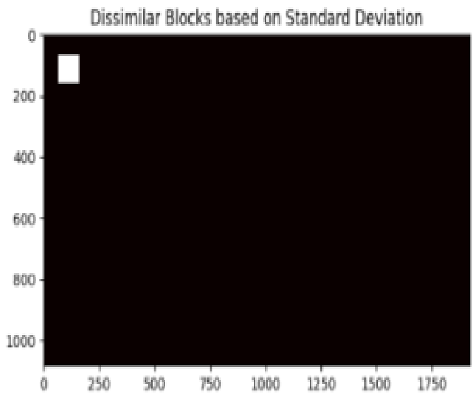
(a) Local Histogram



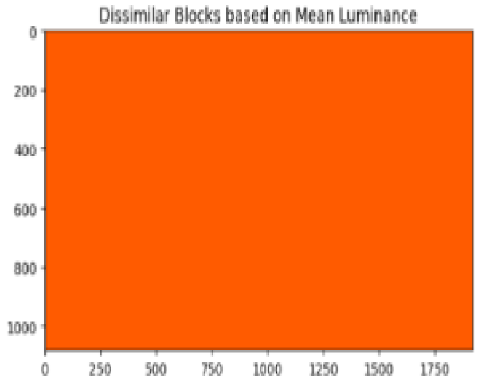
(b) Contrast



(c) Mean Intensity



(d) Standard Deviation



(e) Mean Luminance

Figure 5.8: Heat-maps representing the dissimilar blocks in terms of local statistics for CASE 4 (Match the Intensity)

Chapter 6

Conclusions and future scope

6.1 Conclusion

The technique provides a GUI for interactive picture modification and provides a variety of options for embedding artificial patterns into RAW photographs.Users select insertion methods—simple positioning, contrast matching, or preserving desired image features—to embed patterns.The method creates modified RAW photos and handles the embedding. Heatmaps are created for visual comparison by local statistic measures, which evaluate intensity differences, mean luminance, contrast, and histogram. Using ARGUS APIs, both RAW photos are converted to JPEG. To aid in comprehensive examination and assessment of the similarities between the original and changed pictures, the method also computes SSIM scores for global statistics comparison. This allows for well-informed decision-making in image processing jobs.

6.2 Future scope

Exciting prospects are ahead for artificial pattern embedding onto pictures. Advances might consist of improving customisation, investigating flexible embedding techniques that modify patterns in real-time according to user choices or the content of images.We can also create AI systems that can recognize the best embedding locations by taking into account the properties of the images in order to integrate them seamlessly.It may also facilitate cross-modal information fusion, extend embedding beyond visual patterns to encompass a variety of data kinds, such as audio, text, or 3D objects.

We can design user-friendly graphical user interfaces (GUIs) or augmented reality (AR) interfaces that enable users to interactively see and control embedded patterns in real-world situations.

Both quantitative and qualitative assessment measures should be developed in order to gain a better knowledge of the impacts of pattern embedding on pictures, going beyond SSIM. they may help in the integration of apps in several domains such as augmented reality, data concealing, healthcare (medical imaging markers), and authentication.

New frontiers across industries and user experiences will be opened by future developments in synthetic pattern embedding, which have the potential to revolutionize information integration, content augmentation, and visual modification.

Bibliography

[1] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[2] Tong Li, Hansen Feng, Lizhi Wang, Zhiwei Xiong, and Hua Huang. Stimulating the diffusion model for image denoising via adaptive embedding and ensembling. *arXiv preprint arXiv:2307.03992*, 2023.

[3] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.