**HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY  
SCHOOL OF ELECTRICAL & ELECTRONIC ENGINEERING**

**Ảnh có chứa biểu tượng

Mô tả được tạo tự động**

**ET4591E – Digital Image Processing**

**ASSIGNMENT**

|  |  |  |
| --- | --- | --- |
| **Instructor :** | Prof. Tran Thi Thanh Hai Prof. Le Thi Lan | |
| **Course :** | ET4591E | |
| **Class :** | 152446 | |
| **Name :** | Nguyen Quang Anh | 20213564 |

**HÀ NỘI –** **2025**

Table of Contents

[List of references: 2](#_Toc187765828)

[1. Introduction 4](#_Toc187765829)

[1.1. What is image blur and its unnecessity 4](#_Toc187765830)

[1.2. Description of our problem 4](#_Toc187765831)

[2. Explanation of Our Method 4](#_Toc187765832)

[2.1. Background 4](#_Toc187765833)

[2.2. Implementation 5](#_Toc187765834)

[2.2.1. Image Loading - Dataset 5](#_Toc187765835)

[2.2.2. Gaussian Mixture Model (GMM) Segmentation 6](#_Toc187765836)

[Maximization Step with EM algorithm 7](#_Toc187765837)

[Expectation Step with Two-Color GMM 8](#_Toc187765838)

[2.2.3. Optimization Setup by Tensorflow 8](#_Toc187765839)

[2.2.4. IRLS Optimization 9](#_Toc187765840)

[2.2.5. Final Visualization 10](#_Toc187765841)

[3. Why we choose those method 12](#_Toc187765842)

[4. Discussion and Future Work 13](#_Toc187765843)

### List of references:

1. <https://www.microsoft.com/en-us/research/wpcontent/uploads/2009/06/Joshi-CVPR09.pdf>
2. <https://www.cronj.com/blog/gaussian-mixture-model-in-image-processing-explained/>
3. <https://towardsdatascience.com/iterated-reweighted-least-squares-and-glms-explained-9c0cc0063526>

# 1. Introduction

## 1.1. What is image blur and its unnecessity

Image blur is an unfavorable loss of bandwidth that reduces image quality and is difficult to avoid. Blur is produced by both atmospheric instability and improper camera settings. Noise alters the recorded image in addition to blur effects. Whether arising from motion, focus errors, or other factors, blur inhibits precise interpretation and analysis of visual data. Blur can take many different forms, including Gaussian blur, motion blur, and so on.

Blur is a common occurrence affecting images and videos, presenting notable challenges for computer vision systems. The capability to extract meaningful information from blurry images holds immense importance for diverse applications, including object detection, image recognition, and autonomous navigation.

Blur in computer vision refers to the loss of sharpness or clarity in images or videos. It can occur due to different factors, including camera motion, object motion, incorrect focus, or optical limitations. Understanding and handling blur is essential because it directly impacts the quality and reliability of computer vision algorithms.

There are distinct ways for eliminating various forms of blurring image restoration has applications in a wide range of fields, including medical imaging, crime analysis, and astronomy. We based on the suggested article for developing the project.

## 1.2. Description of our problem

Image restoration is the process of reducing blur from a deteriorated image and returning it to its original state. Image deblurring is a sort of image restoration in which distortions are removed to restore clear images. One sort of distortion is blurring artefacts. Deblurring is the process of recovering a clear picture from a blurred input. Because it is inherently an inverse problem, there is no one unique solution; it is thus an ill-posed problem.

# 2. Explanation of Our Method

## 2.1. Background

The image processing algorithm centers on color distribution, incorporating the Bayes theorem and Prior distribution. The Prior, representing the original distribution to be updated, is crucial, starting with the Bayes theorem. There are distinct ways for eliminating various forms of blurring Image restoration has applications in a wide range of fields, including medical imaging, crime analysis, and astronomy. We based on the suggested article for developing the project.

The algorithm decomposes color components using Gaussian distributions, considering neighboring pixel context. A likelihood function, minimizing quadratic distance and gradient terms, finds the Alpha parameter for interpolation between distributions. Exploring intricacies, it addresses pixel color distribution by decomposing components. The K-means algorithm identifies these based on pixel context, providing a defined color.

Implemented with TensorFlow and gradient descent, the algorithm is cautioned for numerical problems and inefficiency with large images due to quadratic growth. TensorFlow and gradient descent contribute to obtaining a noise-free image. Challenges arose from the paper's lack of detail, requiring a deep dive. In conclusion, the algorithm adeptly handles color distribution, emphasizing adaptation in the face of sparse details.

## 2.2. Implementation

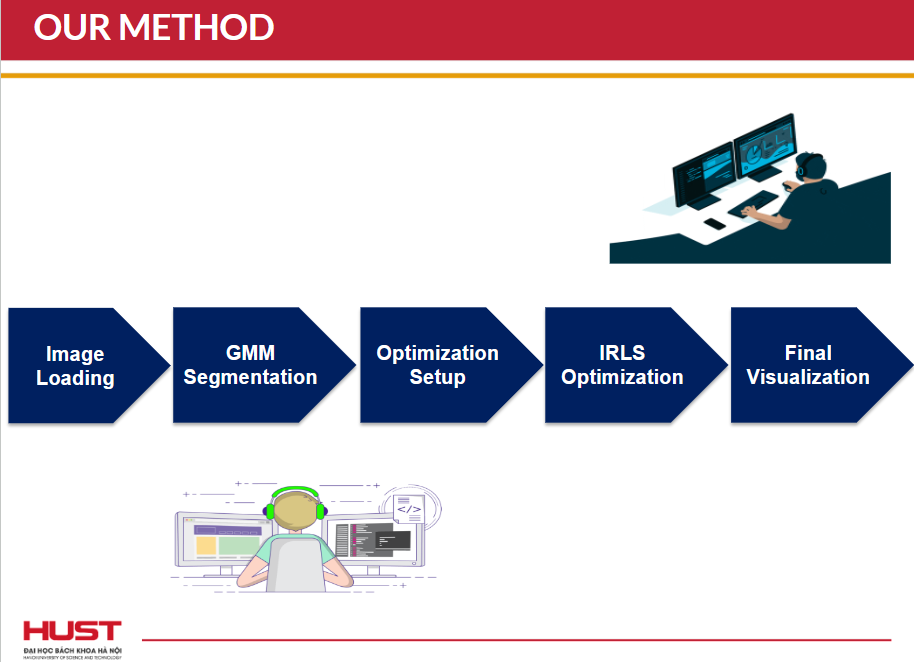


Figure 1: Our method as shown, using GMM segmentation and setup optimization via IRLS function

### 2.2.1. Image Loading - Dataset

Since there’s no actual list of collective dataset, we use distinct images to train our model.

Here’s the code for loading the image:

A computer screen with text and images

Description automatically generated

### 2.2.2. Gaussian Mixture Model (GMM) Segmentation

A brief introduction, Gaussian Mixture is a function that includes multiple Gaussians equal to the total number of clusters formed. Each Gaussian in the mixture carries some parameters which are:

- A mean, that defines the center.

- A covariance, that defines the width.

- A probability.

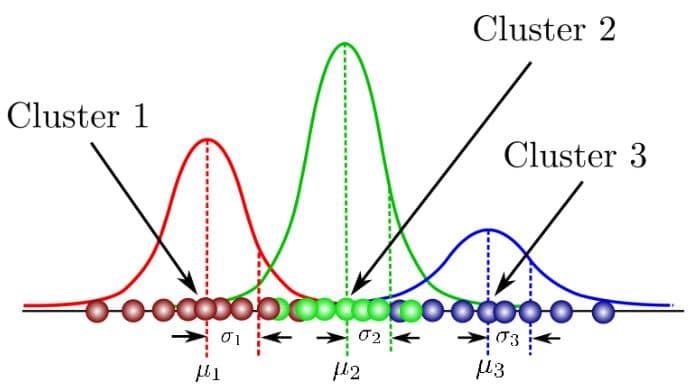
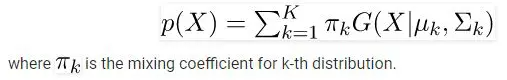


Figure 2: Distribution of GMM clusters

Since there are three( k=3) clusters and the probability density is defined as a linear function of densities of all these k distributions.



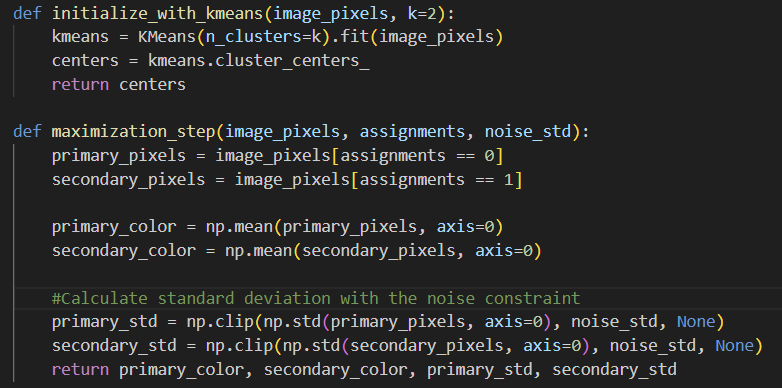
As there will be some n number of sample points in k(th) cluster and the parameters cannot be estimated in closed form. Now question is that how will you find out the missing or hidden data points?

So the answer is the **Expectation-Maximization** algorithm.

EM algorithm finds the maximum-likelihood estimates for model parameters when the data is missing or incomplete or has some variables hidden. This algorithm chooses some random values for the missing data points and calculates a new set of data. These values are used to filling the missing points until the values get first.

### Maximization Step with EM algorithm

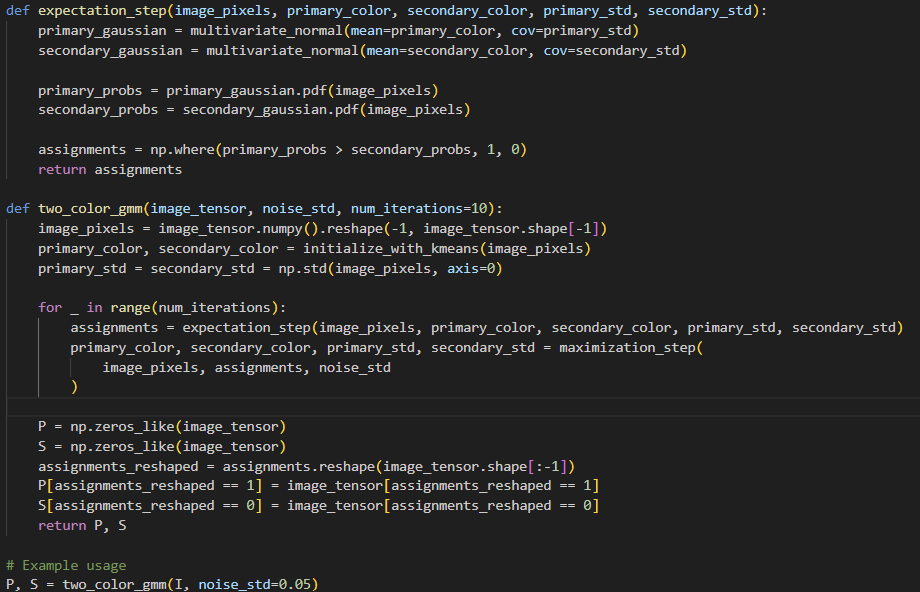
After the image is loaded, it is passed to the Gaussian Mixture Model (GMM) segmentation process. This part of the code performs image segmentation to identify primary and secondary components of the image.



This code appears to be related to the Expectation-Maximization (EM) algorithm for Gaussian Mixture Models (GMMs) in the context of image segmentation. The initialize\_with\_kmeans function utilizes K-Means clustering to initialize GMM parameters, returning cluster centers. The maximization\_step function performs the maximization step in the EM algorithm, calculating mean colors and standard deviations for pixel clusters while ensuring a minimum standard deviation specified by noise\_std.

### Expectation Step with Two-Color GMM

The two color model provides a significant constraint for deblurring; there are two ways such a model can be used for deconvolution. The first is to use the model as a hard constraint, where the sharp image I must always be a linear combination of the primary and secondary colors P and S. The second is to use a soft-constraint to encourage I to lie on the line connecting P and S in RGB space.



The provided code defines functions for the expectation step (expectation\_step) and the iterative EM algorithm for a two-color Gaussian Mixture Model (GMM) segmentation (two\_color\_gmm). The expectation\_step calculates probabilities of pixels belonging to primary and secondary clusters based on Gaussian distributions with given means and standard deviations. It assigns cluster labels (1 for primary and 0 for secondary) based on these probabilities. The two\_color\_gmm function initializes the GMM using K-Means clustering, iteratively performs the expectation and maximization steps for a specified number of iterations, and finally initializes tensors P and S for primary and secondary components, respectively.

### 2.2.3. Optimization Setup by Tensorflow

This section sets up the necessary components for the optimization process, including initialization and defining regularization parameters.

A computer screen shot of a code

Description automatically generated

The image deblurring algorithm is implemented through two essential TensorFlow functions. The first function, compute\_alpha, calculates the alpha matte that determines the optimal blending between the predicted sharp image (P) and the blurred image (S). This is accomplished using the formula α = (P-S)(I-S)/(|P-S|²+ε), where ε=0.001 serves as a stability term to prevent division by zero.

The second function, update\_I, performs iterative optimization of the latent image through gradient descent. It minimizes a composite loss function comprising three terms: an L2 reconstruction loss between the latent image and the blended sharp-blur pair, a regularization term on alpha values (rho\_alpha), and a negative gradient norm term for edge preservation. The optimization process utilizes TensorFlow's automatic differentiation via GradientTape and updates the latent image with a learning rate of 0.1, effectively recovering sharp images from their blurred counterparts.

### 2.2.4. IRLS Optimization

Iterative reweighted least squares (IRLS) is a widely used algorithm for estimating regression coefficients. In the algorithm, weighted least squares estimates are computed at each iteration step so that weights are updated at each iteration. The algorithm can be applied to various regression problems like generalized linear regression or robust regression.

This step iteratively updates the image based on the optimization criteria set in the previous step. Because the previous code does not include the iteration loop for IRLS, here is the function that executes an iteration of IRLS:

A computer screen with white text

Description automatically generated

The irls function takes input tensors I, P, and S, along with three regularization parameters (lambda\_1, lambda\_2, and lambda\_3), a convergence tolerance, and a maximum number of iterations. The iterative process involves updating the variable I using the update\_I function until convergence or the maximum number of iterations is reached. The convergence check is based on the maximum absolute difference between the current and previous values of I.

Example of the use of the IRLS iteration loop: Initialize the tensors I, P, S, and the parameters lambda\_1, lambda\_2, lambda\_3. The variable I must be initialized with some initial value, e.g., an image with noise or an initial estimate P and S should be set according to the specific problem you are trying to solve.

### 2.2.5. Final Visualization

Finally, the optimized image is visualized alongside the original image for comparison by this code:

A computer screen shot of a program

Description automatically generated

The first matrix, termed "Calculated Matrix," is obtained by computing alpha values using the compute\_alpha function and combining tensors P and S. The second matrix, labeled "Original Matrix," is represented by the tensor I0. The visualization is organized into a single row with two columns of subplots using plt.subplot(1, 2, 1) and plt.subplot(1, 2, 2). The plt.imshow function is employed to display the matrices as images, and plt.axis('off') is used to hide the axes in both subplots.

A close-up of a tree

Description automatically generated

Figure 3.1: Final visualization 1

A blurry image of a dog running

Description automatically generated

Figure 3.2: Final visualization 2

A blurry image of a person walking down a sidewalk

Description automatically generated

Figure 3.3: Final visualization 3

# 3. Why we choose those method

We use GMM to segment the image into different color components. This step separates the image into primary (P) and secondary (S) components based on the color distributions. Using IRLS we can further process and optimize these components. Using IRLS, you can iteratively refine the image, adjusting the balance between P and S, reducing noise, and enhancing color. The combined use of GMM for initial segmentation and IRLS for subsequent optimization allows for a more sophisticated approach to image processing. GMM provides a statistical basis for initial color separation, and IRLS refines this separation based on specific optimization criteria, leading to a potentially more nuanced and high-quality result.

Basically, GMM first segments the image into meaningful components, then IRLS optimizes these components to achieve a specific objective (enhancement and noise reduction), finally deblurring the image.

IRLS is used to optimize the image tensor I based on the primary (P) and secondary (S) color components extracted from the image.

In each iteration, it calculates a new version of I that minimizes a specific loss function. This loss function is designed to measure the difference between the current state of I and a target state defined in terms of P, S, and additional constraints or regularization terms.

IRLS continues to update I in each iteration until a convergence criterion is met. This could be a maximum number of iterations or a threshold below which further updates do not significantly change I.

The optimized I at the end of the IRLS process is the final result. It should ideally be an image that has been enhanced or optimized according to the criteria defined in the loss function.

In summary, IRLS is a mechanism for refining the initial image I by iteratively adjusting it based on the primary and secondary color components (P and S).

# 4. Discussion and Future Work

Through this project, we have gained valuable insights and acquired a wealth of knowledge of image deblurring, GMM segmentation, IRLS optimization and many methods related to digital image processing techniques. This experience has allowed us to deepen my understanding of the subject matter and hone various essential skills. As a result, we have not only expanded our knowledge base but also developed critical skills crucial for academic and practical applications.