Digital Image Forensic Techniques for Identification of Image Tampering

GROUP: G24

Supervised By:

Dr. G.M.R.I. Godaliyadda

Dr. M.P.B. Ekanayake

Dr. S.A.H.A. Suraweera

Group Members:

E.D.G.J.B. Senanayaka (E/14/313)

M.G.D.T. Morawaliyadda (E/14/226)

K.A.S.T. Senarath (E/14/318)

Department of Electrical and Electronic Engineering

Faculty of Engineering

University of Peradeniya

Peradeniya 20400

Sri Lanka

July 2020

ABSTRACT

Image forgery detection is the process of identifying the fake or tampered regions in images. With the advancement of high resolution digital cameras and photo editing software featuring new and advanced features, the chances of image forgery has increased. Therefore, image manipulation has become a trivial task that does not require expert know how. Since, the unprecedented involvement of digital images can be seen in various paramount fields like medical science, journalism, sports, criminal investigation, image forensic, etc., where authenticity of image is of vital importance. Various tools are available free of cost or with a negligible amount of cost for manipulating images. Some modern tools can manipulate images to such an extent that it becomes impossible to discriminate by human visual system that image is forged or genuine. Hence, image forgery detection is a challenging area of research.

The proposed methodology come up with localizing regions of forgery within an image tampered with an image forgery technique known as ***splicing***. In this case, a tampered image is generated by the spliced regions from different source images. To do this, we utilize the variation of noise levels and statistics of different segments in the tampered image. The image is segmented into non-overlapped segments using ***Adoptive Centroid Placement based Simple Linear Iterative Clustering (SLIC)*** and then the noise statistics of each segment is estimated. The estimated noise level of each segments are used to classify the segments where the segments with relatively different or deviated noise statistics are identified as forged regions. This proposed algorithm introduces a new graph based representation on the segmented image and then the clustering task is performed in the graph spectral domain using ***spectral clustering***. For this method, the very important key point is non-requirement of supervisory training phase for the algorithm. And it does not depend on prior knowledge or intrinsic models.

ACKNOWLEDGEMENTS

First, we would like to express our sincere gratitude to our project supervisors Dr. G.M.R.I. Godaliyadda, Dr. M.P.B. Ekanayake and Dr. H.A.S.A. Suraweera for the guidance given throughout the courses EE405 - Undergraduate Project I and EE406 - Undergraduate Project II. We are highly indebted to them for their guidance, inspiration and constant supervision as well as for providing necessary information regarding the project.

We also wish to thank Prof. J.B. Ekanayake, the course coordinator for the courses EE405 and Prof. S.G.Abeyratne, the course coordinator for the courses EE406 respectively, for the encouragement and cooperation extended and especially for structuring the course in an effective manner.

Moreover, we are immensely grateful to Mr. Kasun Weerakoon, who is currently a post graduate student in USA for giving initial knowledge and guidance to approach to topic.

Finally, we wish to extend our sincere gratitude to the lecturers, fellow colleagues and all other individuals who gave their ideas, support and encouragement that empowered us to make our project a success.

E.D.G.J.B. Senanayaka (E/14/313),
M.G.D.T. Morawaliyadda (E/14/226),
K.A.S.T. Senarath (E/14/318).

Department of Electrical and Electronic Engineering,
Faculty of Engineering,
University of Peradeniya,
Sri Lanka.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ANN	Artificial Neural Network
CNN	Convolutional Neural Network
EM	Expectation Maximization
GMM	Gaussian Mixture Model
LDA	Linear Discriminant Analysis
MAP	Maximum A Posterior
ML	Maximum Likelihood
MLE	Maximum Likelihood Estimator
MVN	Multivariate Normal Distribution
NLL	Negative Log-likelihood
PCA	Principle Component Analysis
PDF	Probability Density Function
SLIC	Simple Linear Iterative Clustering
SNIC	Simple Non-Iterative Clustering

CHAPTER 1 INTRODUCTION

1.1 Overview

1.1.1 Introduction to Image forgeries

What is image tampering or image forgeries?

In simple words, **"Images tampering"** means editing or modifying the real world appearances of photos through computer software or mobile application for some reason or using other physical techniques while taking the photos.

Image forgery means manipulation of the digital image to conceal some meaningful or useful information of the image. There are cases when it is difficult to identify the edited region from the original image. The detection of a forged image is driven by the need of authenticity and to maintain integrity of the image. As we are living in the todays digital world in which all type of advancement are becoming possible and at the same time the use of images have been increasing day by day in our lives, the motivation to make manipulation of images also increases simultaneously.

Eg: Common Images which were identified as forged.

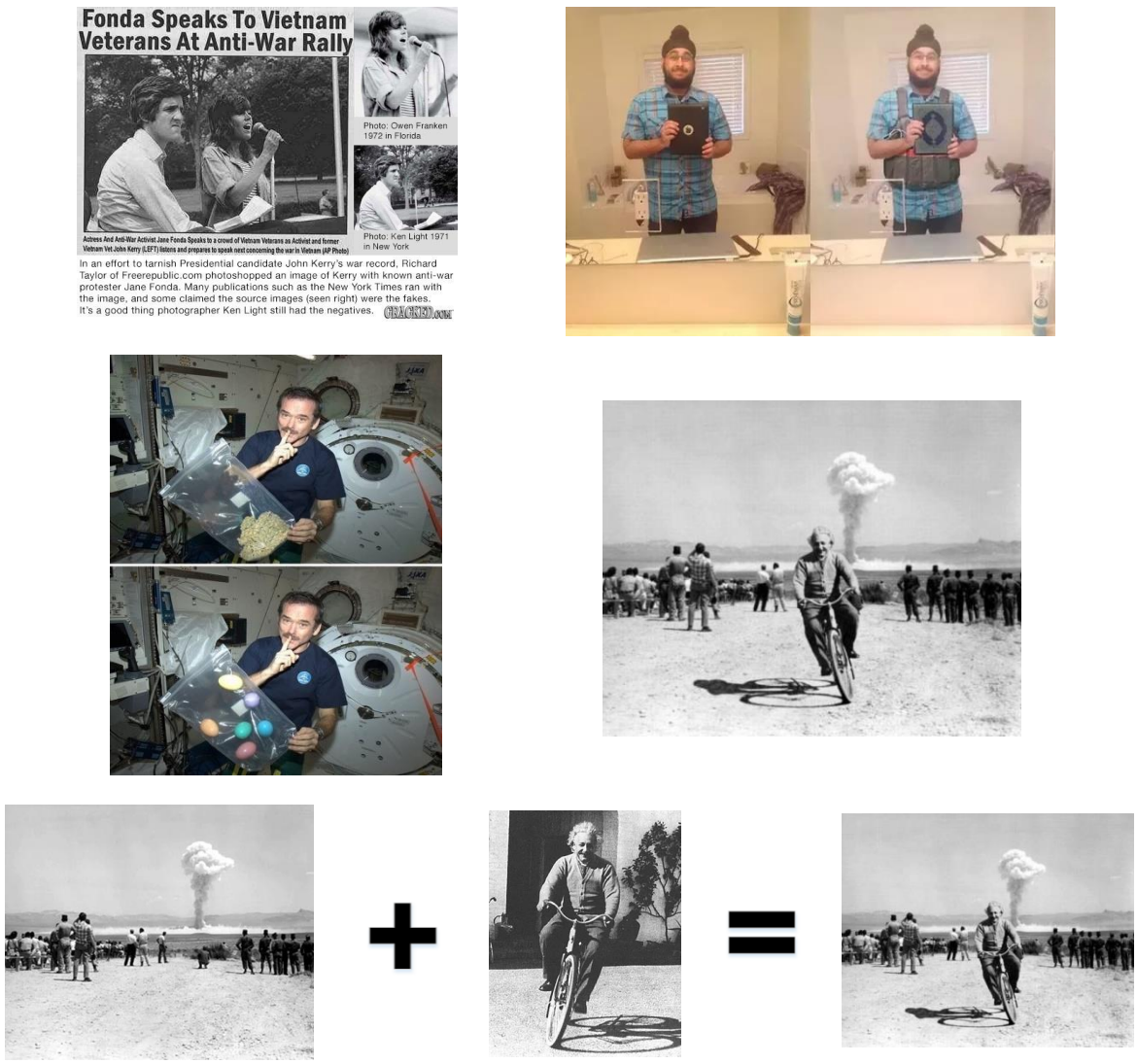


Figure 1.1: Commonly identified image forgeries

As the use of images have been increasing day by day in our lives, with the introduction of digital technology, The forgery of digital image has become more and more simple and indiscoverable. Today's digital technology had begun to erode the integrity of images and image counterfeiting and forgeries with the move to the world of Megapixels, opens a new door to the dark-side of it. We are living in an age, where anything can be manipulated or altered with the help of modern technology. With the increasing applications of digital imaging, different types of software tools are introduced for processing images and photographs. They are used to make forge images to make it look real or objects can be added or deleted. For decades, photographs have been used to document and they have used as evidence in courts. Although, photographers are able to create composites of analog pictures. But this process is very time consuming and requires expert knowledge so it is hard to implement than digital pictures. Today, however, powerful digital image editing software makes image modifications straightforward. Today's digital technology has begun to remove trust in our knowledge, as from the magazines, to fashion world and in scientific journals, political campaigns, courts and the photo that come in our e-mail. In all of these forged photographs are appearing with a more frequencies and sophistication. In the increase in the availability of multimedia data in digital form has come to a tremendous growth of tools to manipulate digital multimedia contents.

The process of creating **fake image** has been tremendously simple with the introduction of new and powerful computer graphics editing software which are freely available as Photoshop, GIMP, and Corel Paint Shop. Today, this powerful image processing software's allow people to modify photos and images conveniently and unperceivably. Now days it creates a big challenge to authenticate images. ***Image forgery*** *means manipulation of the digital image to conceal some meaningful or useful information from it*. Sometimes it is difficult to identify the edited region from the original image. The detection of a forged image is driven by the need of authenticity and to maintain integrity of the image. The survey has been done on existing techniques for forged image and it highlights various copy–move detection and splicing detection methods based on their robustness and computational complexity.

1.1.2 Classification of Image forgery techniques

When considering image forgery techniques, there are different types. But basically they can be divided into two groups. Then, there are two kinds of techniques for image forensics: one is active protection, and the other is passive detection. Which again consist of many different methods, as shown in below *Figure 1.2.*

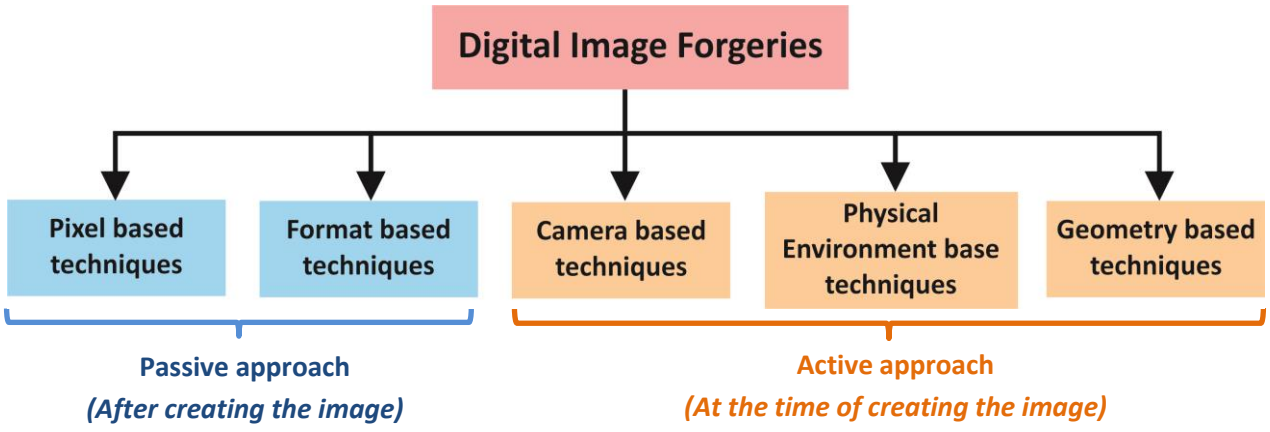


Figure 1.2: Basic classification of image forgeries

ACTIVE APPROACH

In this ***active approach***, the digital image requires some kind of pre-processing such as watermark embedded or signatures are generated at the time of creating the image. However, in practice this would limit their application. Digital watermarking and signature are two main active protection techniques, as something are embedded into images when they are obtained. We can detect the Image is tampered, if special information cannot be extracted from that obtained image. Watermarking is such a method of active tampering detection, as a security structure is embedded into the image, but most present imaging devices do not contain any watermarking or signature module and that are similar to the application of active protection. This structure is used for integrity evaluation in the sense that if any discrepancy is found with the structure then the image is tampered and an inverse analysis over the structure is done to locate tampered regions of the image. In recent times, various schemes are proposed for providing security to the image, which is analogous to the concept of watermarking like, message authentication code, image hash, image checksum and image shielding as a counterpart to it.

PASSIVE APPROACH

Passive image forensics is usually a great challenge in image processing techniques. There is not a particular method that can treat all these cases, but many methods each can detect a special forgery in its own way. The stream of passive tampering detection deals with analyzing the raw image based on various statistics and semantics of image content to localize tampering of image. Neither construct is embedded in the image and nor associated with it for security, as like active approaches and hence this method is also known as raw image analysis. The localization of tampering is solely based on image feature statistics. Hence, algorithms and methods of detection and localization of image based on passive tampering vary depending upon the type of security construct used. Nevertheless, passive tampering detection typically aims for localization of tampering on raw image.

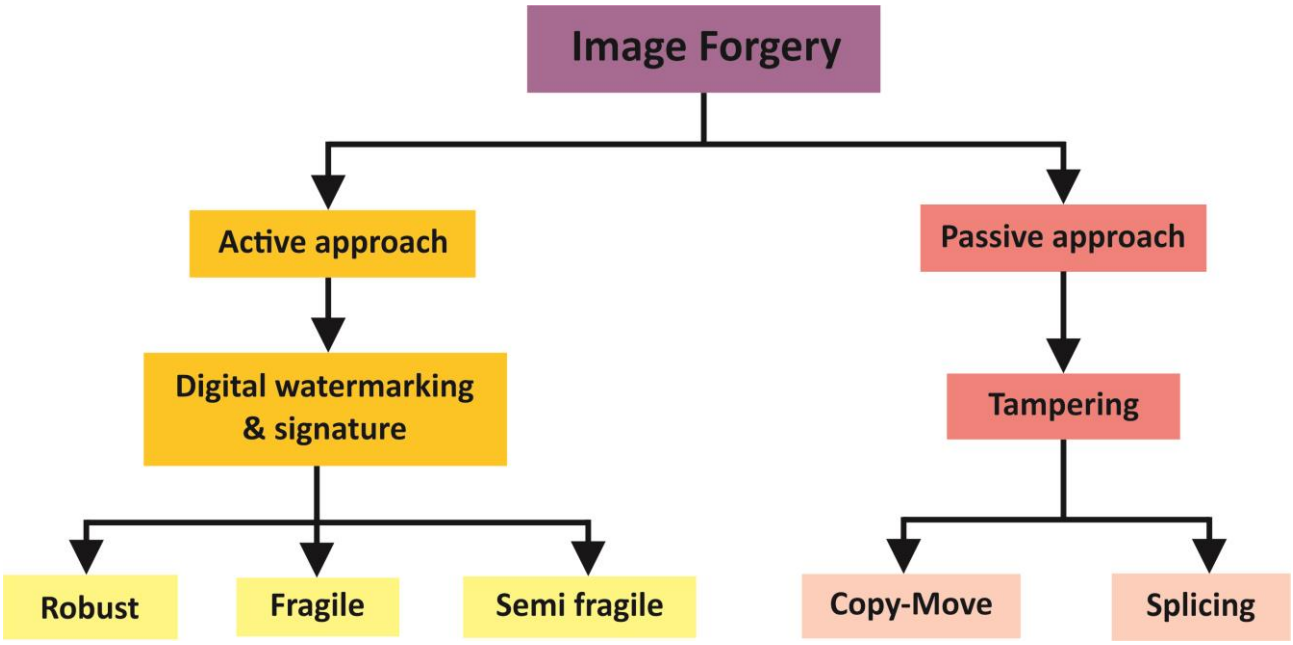


Figure 1.3: Classification of image forgery approaches

1.1.3 Types of image forgeries

IMAGE RETOUCHING

Image Retouching is considered as less harmful kind of digital image forgery than other types present. In case of image retouching original image does not significantly changes, but there is enhancement or reduces certain feature of original image. This technique is popular among magazine photo editors. This type of Image forgery is present in almost all-magazine cover that would employ this technique to enhance certain features of an image so that it is more attractive. Actually, the fact is that such enhancement is ethically wrong.

IMAGE SPLICING OR PHOTOMONTAGE

This technique for making forgery images is more aggressive than image retouching. Image splicing is fundamentally simple process and can be done as crops and pastes regions from the same or separate sources. This method refers to a paste-up produced by sticking together images using digital tools available such as Photoshop. In Image Splicing technique there is composition of two or more images, which are combined to create a fake image. Examples include several infamous news reporting cases involving the use of faked images. *Figure 1.4* below shows how to create forge Image; by copying a spliced portion from the source image into a target image, it is a composite picture of scenery which is forge image.

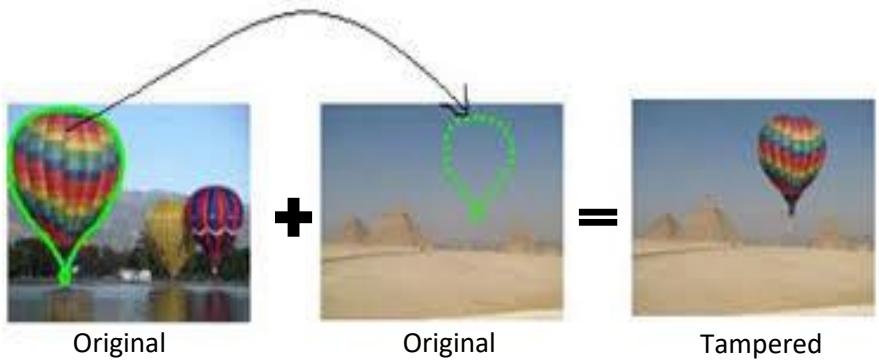


Figure 1.4: Example of image splicing

COPY-MOVE ATTACK

The ***copy move forgery*** is popular as one of the difficult and most commonly used kind of image tampering technique. In this technique, one needs to cover a part of the image in order to add or remove information. In the Copy-Move image, manipulation technique a part of the same image is copied and pasted into another part of that image itself. In a copy-move attack, the intention is to hide something in the original image with some other part of the same image. The example of Copy-Move type is as shown in below *Figure 1.5* below. The original image contains only three missiles and its Copy-Moved version on the right has four missiles.

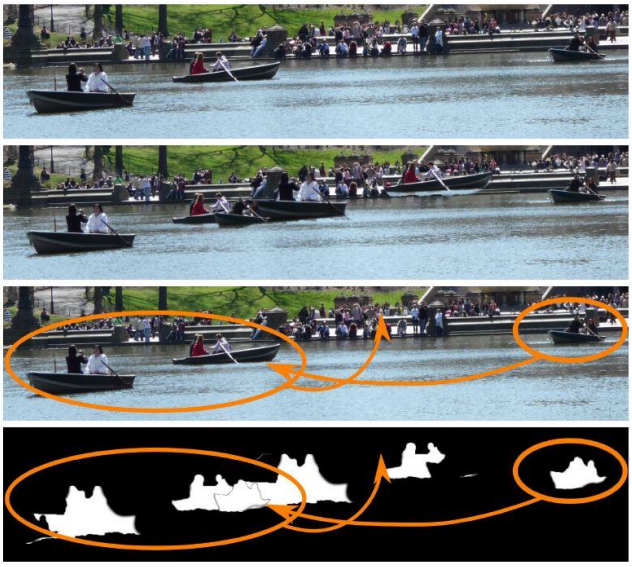


Figure 1.5: Example of image copy-move

1.2 Forgery detection

1.2.1 Idea of forgery detection

Digital Image Forgery Detection is an important field in Image Processing, because digital images are used in many social areas like in courts where they are used as evidence. In information channels like newspapers, magazines, websites and televisions, digital images are powerful tool for communication. Unfortunately, it is easy to use computer graphics, image editing software and image processing techniques to manipulate the images. These manipulated images create some unwanted problems in information channels.

In image forgery detecting techniques, it is assumed that the original image has some inherent patterns, which are introduced by the various imaging devices or processing. These patterns are always consistent in the original image and altered after some forgery operations. The image forgery detection has become complex, because of the advanced and sophisticated processing tools. In this view, researchers have proposed various techniques to detect the forgery in an image.

1.2.2 Our methodology

Out of all those image forgery types, our main focus is on ***splicing type image forgery detection.*** After lots of literature review, it was decided to use a new property of an image, which is ***The Noise Content of the Image*** There are few steps involved in splicing detection. They can be illustrated as below in *Figure 1.6.*

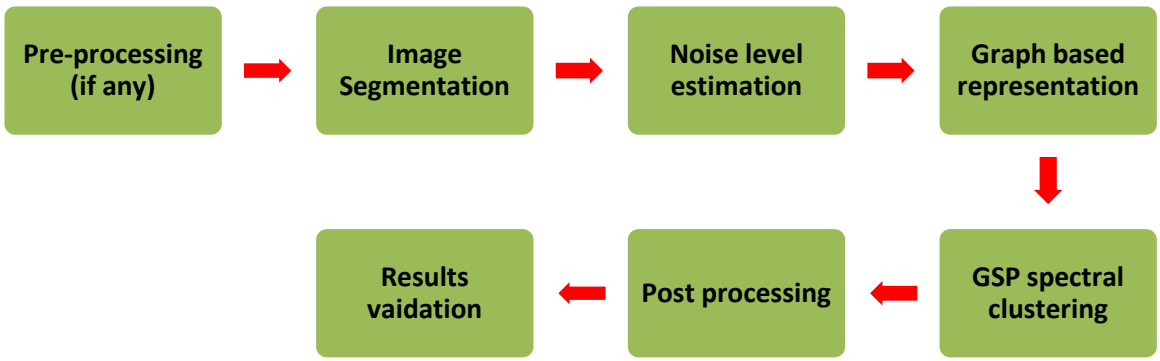


Figure 1.6: Splicing image forgery detection procedure

1.3 Data Base

Since our main focus is on ***splicing type forgery detection,*** we used one of the most commonly used data-base of tempered images called ***"Columbia Uncompressed Image Splicing Detection".*** This is the one, that, most people have used for their researches based on forgery image detection as we mentioned above. From that data-base we used the images in **splicing category.**

1.3.1 Directories

There are 2 directories in this dataset: 4cam_auth & 4cam_splc. 4cam_auth contains authentic images, and 4cam_splc contains spliced images. By the term 'authentic', they mean an image that is taken using just one camera.

In 4cam_auth, there are 183 images, and in 4cam_splc, there are 180. The image sizes range from 757x568 to 1152x768 and are uncompressed, in either TIFF or BMP formats. The spliced images are created using the authentic images, without any post processing. Full EXIF information is included in authentic images. The images are mostly indoor scenes: labs, desks, books ...etc. Only 27 images, or 15%, are taken outdoors on a cloudy day (which makes the outdoor illumination similar to indoor conditions). Several examples are shown below in *Figure 1.7.*

Figure 1.7: Some Examples of images in "Columbia Uncompressed Image Splicing Detection" data-base

1.3.2 Edge-marks

Under each directory there is a sub directory storing ***edge-masks*** for test images. These edge-masks label regions within each image, indicating them as parts that come from different cameras and are created manually. For authentic images, since there is no actual splicing boundary, we just picked a salient object boundary as the suspicious splicing boundary.

Edge-mask filenames follow the naming conventions of its corresponding image, with the string '_edge-mask' or '_edgemask_3' appended. For the former case, the image is divided into 4 regions: bright red (255,0,0), bright green (0,255,0), regular red (200,0,0), and regular green (0,200,0).

- Bright red indicates the part **near** the suspicious splicing boundary that comes from camera 1.
- Bright green indicates the part **near** the suspicious splicing boundary that comes from camera 2.
- Regular red indicates the part **far from** the suspicious splicing boundary that comes from camera 1.
- Regular green indicates the part **far from** the suspicious splicing boundary that comes from camera 2.

For '_edgemask_3', the image is divided into 3 regions, with bright red and bright green merged into a single 'splicing boundary' region in regular blue (0,0,200).

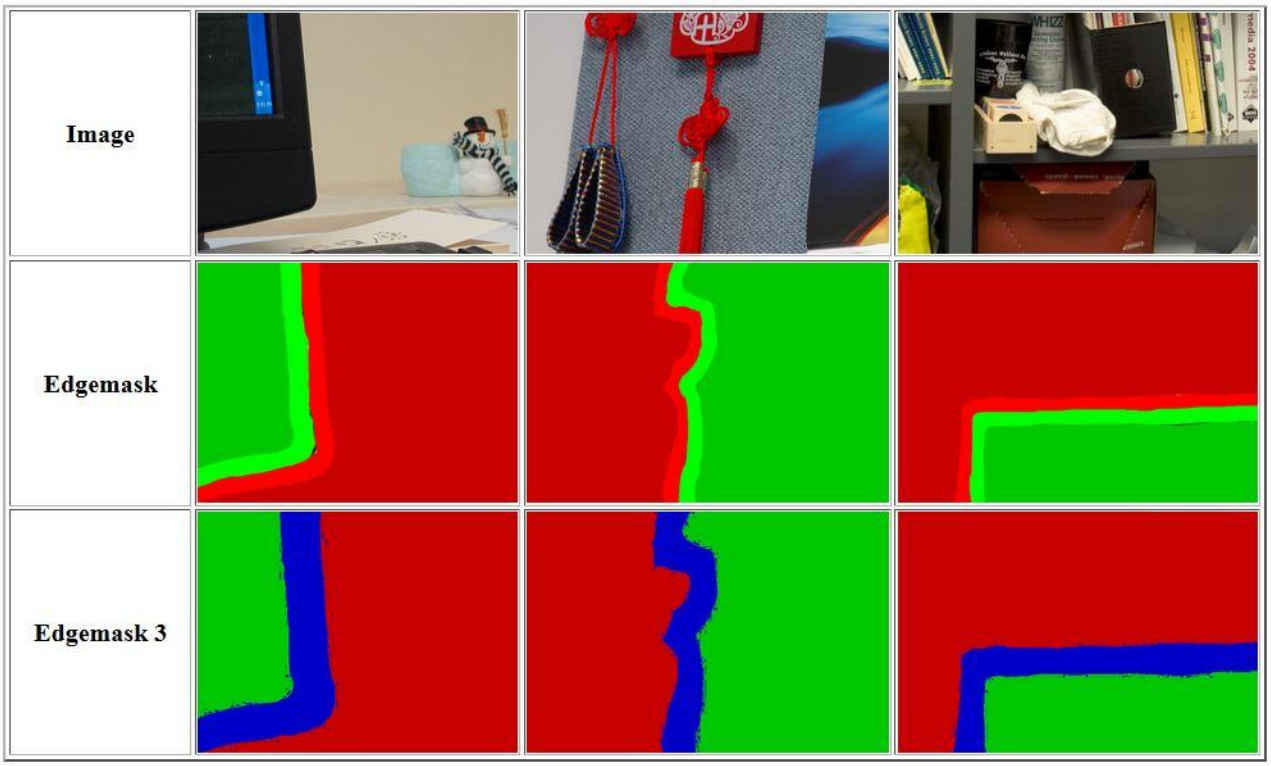


Figure 1.8: Some Examples of images and their edge-marks in "Columbia Uncompressed Image Splicing Detection" data-base (in Splicing category)

1.4 WORK CARRIED OUT

In order to meet the objectives, the following activities were carried out according to the schedule represented by the Gantt chart in *Figure 1.9.*

- Activity 1 – Literature review
- Activity 2 – Analysis of image segmentation algorithms
- Activity 3 – Implementation of an image segmentation algorithm
- Activity 4 – Analysis of image noise models and implement an algorithm
- Activity 5 – Development of an algorithm for forgery detection
- Activity 6 – Validating results
- Activity 7 – Optimization of the image segmentation algorithm developed
- Activity 8 – Validating results and comparison
- Activity 9 – Writing a research paper from the results generated

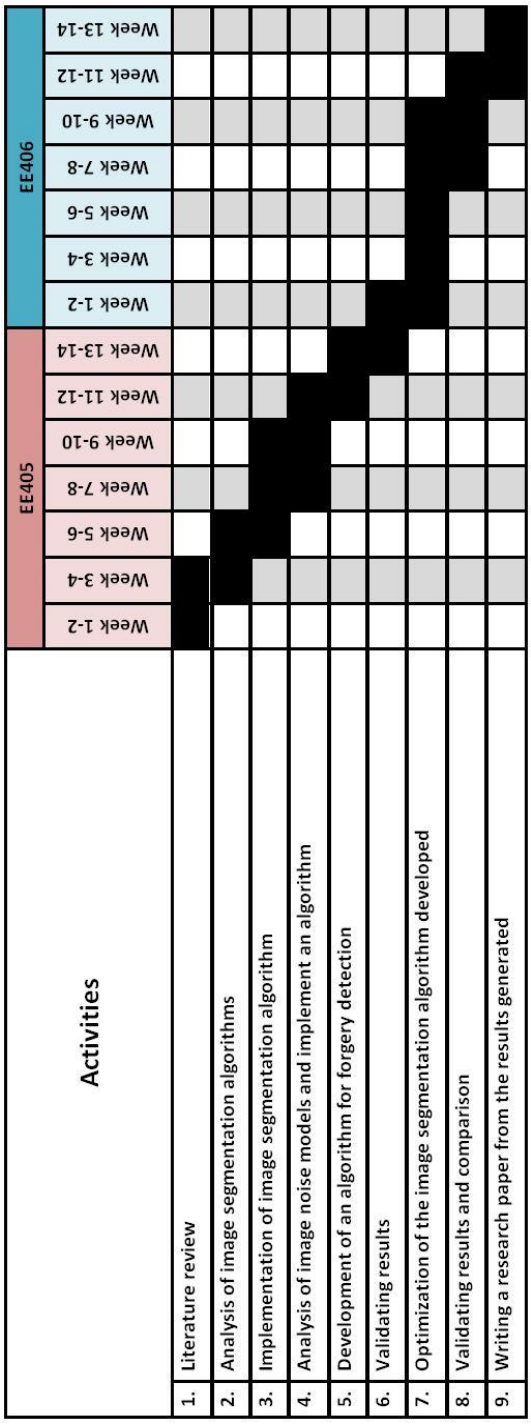


Figure 1.8: Gantt chart of the project activities

CHAPTER 2 LITERATURE REVIEW

2.1 Currently available forgery detection techniques

2.1.1 Digital signature

Digital signature is one among the active method used for detecting image forgery or tampering. Demonstrating the authenticity of digital document using a sort of mathematical scheme is called as digital signature. In digital signature a robust bits are extracted from the original image. An image is divided into blocks of 16*16 pixels. A secret key k is used to generate N random matrices with entries uniformly distributed in interval [0, 1]. A low pass filter is applied on each random matrix repeatedly to obtained N random smooth pattern. System generate digital signature by applying signing process on digital image. Image Signing process contain following steps.

- Decompose the image using parameterized wavelet feature.
- Extract the SDS
- Cryptographically hash the extracted SDS, generate the crypto signature by the image senders private key.
- Send the image and its associated crypto signature to the recipient.

Digital signature is simple and basic approach for digital image authentication.

2.1.2 Digital watermarking

Watermarking is also used for image forgery detection. Several watermarking techniques have been proposed. One uses a checksum schema in that it can add data into last most significant bit of pixels. Others add a maximal length linear shift register sequence to the pixel data and then identify the watermark by computing the spatial cross-correlation function of the sequence and the watermarked image. These watermarks are designed to be invisible, or to blend in with natural camera or scanner noise. Visible watermarks also exist. In addition to this, a visually undetectable watermarking schema is also available which can detect the change in single pixels and it can locate where the change occur. Embedding watermarks during creation of digital image it may limits its application where digital image generation mechanism have built-in watermarking capabilities. These active techniques have some limitation because they required some human intervention or specially equipped cameras. To overcome this problem a passive authentication has been proposed.

2.1.3 Copy-Move

Copy-move is the most popular and common photo tampering technique because of the ease with which it can be carried out. It involves copying of some region in an image and moving the same to some other region in the image. Since the copied region belong to the same image therefore the dynamic range and color remains compatible with the rest of the image. Along with the copy move operation, image editing related operations such as rotation, color, scaling, blurring, compression and noise addition are added to the original image. This is done in order to make the forged part unnoticed to the human vision. The detection of some parameters like noise, color from the forged is not possible to differentiate.

(a). Block based algorithms

In the ***block based methods*** surveyed till now, the input image is divided (segmented) into overlapping and regular image blocks. The tampered region is then obtained by matching the blocks of image pixels or transform coefficient.
The block based methods involve:

- Quantized Discrete Cosine Transform (DCT) coefficients of blocks matched to detect the tampered regions.
- Principal Component Analysis (PCA) to reduce the block feature dimensions.
- RGB color components and direction information as block features.
- Calculation of 24 blur invariant moments as block features.
- Fourier-Mellin Transform (FMT) for block feature calculation.
- Gray average results of each block and sub-blocks used as block features.
- Zernike moments for block feature.
- Information entropy used as block feature.

Apart from the above mentioned process the image feature calculation is also important to meet the rotation, scaling, compressions and time complexity improvements in image forgery detection. Hence the feature key-point based techniques were developed to achieve the accuracy even with the forgery subjected to rotation and scaling.

(b). Feature key-point based

The feature key-point detection involves the following method

- SIFT- Scale invariant feature transform to extract host image feature transform to match for forgery or duplication detection.
- SURF- Speed up Robust Feature for feature extraction.

Although these feature results in locating matched key-points, they fail to locate the forgery. Fast and robust copy move forgery detection methods are developed with the combination of block based and feature based algorithms. Although these techniques can improve the computational complexity and detect the forgery accurately, they have a drawback of low recall rate because of the regularity in blocking methods.

In order to overcome this problem recent technique based on an adaptive over - segmentation and feature point matching was developed. In this method segmentation of blocks are non-overlapping and irregular blocks called image blocks (IB). The recall rate was improved compared to the previous works because of the irregularity in the blocks.

2.1.4 Splicing

Image splicing is a method of combining two or more images to make it a composite (single) image. When images are spliced, resulting image shows lines, edges, regions and blur at the point where the images have been spliced. Development of the editing tools have made the lines, edges, regions and blur to merge in the image so that the human vision is not able to detect the forgery. Hence the image splicing detection has become one of the challenging topics for the researchers.

Steganography and Image Splicing have different approaches but still both the process create a new tampered image. Both the method makes an alteration in image smoothness, regularity, continuity and periodicity. Hence, statistical approaches are applied to detect these

traces. As steganalysis and Image Splicing detection make use of statistical approaches, some of the statistical natural models applied for steganalysis can be applied for Image Splicing detection.
Image splicing technique involves dimensional feature vectors. Four general methods applied for steganalysis were applied to image splicing detection with the accuracy of less than 80%.

The methods applied are:

- 72- Dimensional (72-D) feature vector composed of higher-order statistical moments of wavelet.
- 78-Dimensional (78D) feature vector- first three moments of characteristics and the prediction error applied for each four sub-bands in the three-level wavelet decomposition.
- 2-D Markov chain - for threshold prediction-error image. Features are extracted from three directions (horizontal, vertical, and main diagonal).
- Singular Value Decomposition (SVD) - SVD based 50-Dimensional feature vector merged with Discrete Cosine Transform (DCT).
- Combination of 1-D and 2-D - statistical moments of 1-D and 2-D characteristic functions extracted from the spatial domain and multi block discrete cosine transform (MB-DCT) are combined.

2.1.5 Image retouching

Image retouching is one more type of image forgery tool which is most commonly used for commercial and aesthetic applications. Retouching operation is carried out mostly to enhance or reduce the image features. Retouching is also done to create a convincing composite of two images which may require rotation, resizing or stretching of one of the image.

Image retouching detection is carried out by trying to find the blurring, enhancements, color changes and illumination changes in the forged image. Detection is easy if the original image is available. However, blind detection is a challenging task. For this type of forgery two type of modification is done either global or local. Local modification is done usually in copy-move and in splicing forgery. Contrast enhancement that is carried out in case of retouching is done at global level and for detection of tampering these is investigated. For illumination and changes in contrast global modification is carried out.

Number of methods have been proposed and discussed for retouching forgery. Limitation is that most of the methods work well if the image is greatly modified in comparison to the original image. Moreover, the human intervention required to interpret the result makes them non blind techniques.

2.1.6 Format based technique

Image alteration does not prove malicious tampering, as in the cases of color/contrast adjustment for image enhancement, and file format conversion for saving storage space. These manipulations do not fundamentally change the contents of the original image, while malicious tampering will alter the meaning of the image such as removing, adding and modifying an object in a scene. Malicious manipulations, in collaboration with subsequent operations such JPEG compression, contrast adjustment, blurring, etc. would make forgeries hard to detect. Therefore image-alteration detection can determine whether the images are original and help us with further analysis.

2.1.7 Camera based techniques

High-resolution and low-cost digital cameras have been rapidly replacing the typical film cameras. Now, most images in our daily life are acquired by various brands of digital cameras, such as Canon, Nikon, Sony, Olympus, etc. One of the main problems related to source identification is the classification of the different camera models or individuals for a given image.

The most straightforward solution for camera identification is to check the exchangeable image file (EXIF) format header of the output image. Some settings of an image are stored in the headers, and the settings are constrained by a given camera, such as the manufacturer, the model of the camera, image size, exposure time, and the quantization matrix used in JPEG compression etc. If the given image settings are out of the range of the given camera, it can be concluded that the image did not come from the camera or it was not the original one at least. However, it is difficult to distinguish among the cameras of the same or similar model whose images contain the same header information. Furthermore, the header information can be easily replaced or made consistent by JPEG recompression or other operations. And also, there are some defect pixels in the charge coupled device (CCD) inside the low-cost digital cameras. These defects pixels are at the different places of the CCD according to the different sensors, and thus can be used as the unique evidence for the cameras. However, there are some restrictions when using this method. For example, the defects in pixels are visible only in the regions that are darker or the lighter areas if a surface has the same intensity lighting. These defects pixels also depend on the temperature. Furthermore, some post-processing operations such as JPEG compression, etc., may remove or suppress the defective pixels. For the expensive cameras which have better CCDs with fewer errors, the method cannot be applied. Due to cost considerations, many manufactures employ a single sensor instead of multiple sensors to capture the color scene. Thus the color filter array (CFA) is always applied in front of the sensor to control the band of wavelengths arriving at the CCD array.

In order to reconstruct the full-resolution color scene, some interpolation algorithms will be employed. The estimations are usually carried out by interpolating neighboring pixel values using a weighting matrix around the missing pixel, which are called demosaicking techniques.
The correlations may be linear, nonlinear or adaptive. And these different techniques are employed in different models of cameras, which will inevitably introduce a different statistical correlation between the original values and the interpolation values. The simplest demosaicking methods are kernel-based ones that act on each channel independently. More sophisticated algorithms interpolate edges differently from uniform areas to avoid blurring salient image features. Regardless of the specific implementation, CFA interpolation introduces specific statistical correlations between a subset of pixels in each color channel.

To estimate the pattern of the correction between the samples, the EM (expectation/maximization) algorithm has been applied. The algorithm includes two steps: E-step and M-step. In the E-step, the probability of each pixel belonging to an original pixel or to the interpolated one is evaluated. Then in the M-step, the estimation is optimized and updated.

The pattern noise is defined as any noise component that survives frame averaging, which is another important characteristic of imaging sensors. The pattern noise include two main components: the fixed pattern noise (FPN) and the photo-response non-uniformity noise (PRNU). Fixed pattern noise (FPN) is mainly caused by the dark current on a CCD chip. The dark current is due to thermal activity in the photocathode and the dynodes. And it is present whether the shutter is open or closed. However, the magnitudes of the dark current on a CCD are always non-uniformity as different pixels may have different generation rates of dark current. The millions of non-uniformity pixels are arranged regularly on each CCD, and therefore can create the unique pattern for each sensor. In this method, people used FPN to identify the video camera from videotape

images. They recorded 100 black images with each camera by covering the lens, and then the images were accumulated to suppress the effect of the random noise. The results show that some bright dots are observed in the accumulated images, and these bright dots are at different positions for each camera. However, FPN is visible only in the dark frames. Furthermore, the noise can be alleviated at a low temperature.

Another main source of the pattern noise in imaging sensor is PRNU. Unlike FPN, which is generated thermally in the sensor even when no light arrives, PRNU is the pixel variation under illumination. FPN is an offset, while PRNU is a gain. Therefore, the primary source of pattern noise remaining in nature images may be PRNU. Two sources contribute to PRNU. The main source is pixel non-uniformity (PNU), and the other source is low frequency defects, which is caused by light refraction on dust particles and optical surfaces, etc. This source is low spatial frequency in nature. In this method, people used PNU as an inherent pattern of the imaging sensor for camera identification. To verify that a given image p was taken with a specific camera C; they first extracted the camera reference pattern P_c, which is an approximation of PNU.

2.1.8 Physically based techniques

Images that are combined during tampering are taken in different lighting conditions. It becomes difficult to match the lighting condition from combining photographs. This lighting inconsistency in the composite image can be used for detection of image tampering. Initial attempt in this regard was made by some people. They proposed a technique for estimating the direction of an illuminating light source within one degree of freedom to detect forgery. By estimating direction of light source for different objects and people in an image, inconsistencies in lighting are uncovered in the image and tampering can be detected.

Some people proposed a model based on lighting inconsistencies because of presence of multiple light sources. This model is motivated from earlier model but it generalizes this model by estimating more complex lighting and can be adapted to a single lighting source.

And also there are techniques of estimated 3-D direction to a light source by means of the lights reflection in the human eye. These reflection called specular highlights are a powerful clue as to the location and shape of the light sources. Inconsistencies in location of the light source can be used to detect tampering.

There also a method for authentication of image with infinite light source based on inconsistencies in light source direction. Hestenes-Powell multiplier method was employed to calculate the light source direction of different objects and their background in infinite light source images. Authenticity is determined on the basis of consistency between the light source direction of the object and its background with detection rate of 83.7%.

2.1.9 Geometric based techniques

In authentic images, the principal point (the projection of the camera center onto the image plane) is near the center of the image. When a person or object is translated in the image, the principal point is moved proportionally. Differences in the estimated principal point across the image can therefore be used as evidence of tampering. There are people who have described how to estimate a cameras principal point from the image of a pair of eyes (i.e. two circles) or other planar geometric shapes. They showed how translation in the image plane is equivalent to a shift of

the principal point. Inconsistencies in the principal point across an image can then be used as evidence of tampering.

Some have analyzed the physical differences in generation between CG and photographic images, e.g., the sharp structures in CG images and gamma correction in photographic. They then proposed a geometry-based image model that reveals the differences. For source identification, the method extracts the geometry features based on the rigid body moments. Finally, an SVM classifier is employed. The experimental results show the effect of the proposed method with a classification accuracy of 83.5%, which outperforms the prior methods. Another contribution of their work is that they created an image benchmark for the classification problem of CG and photographic images

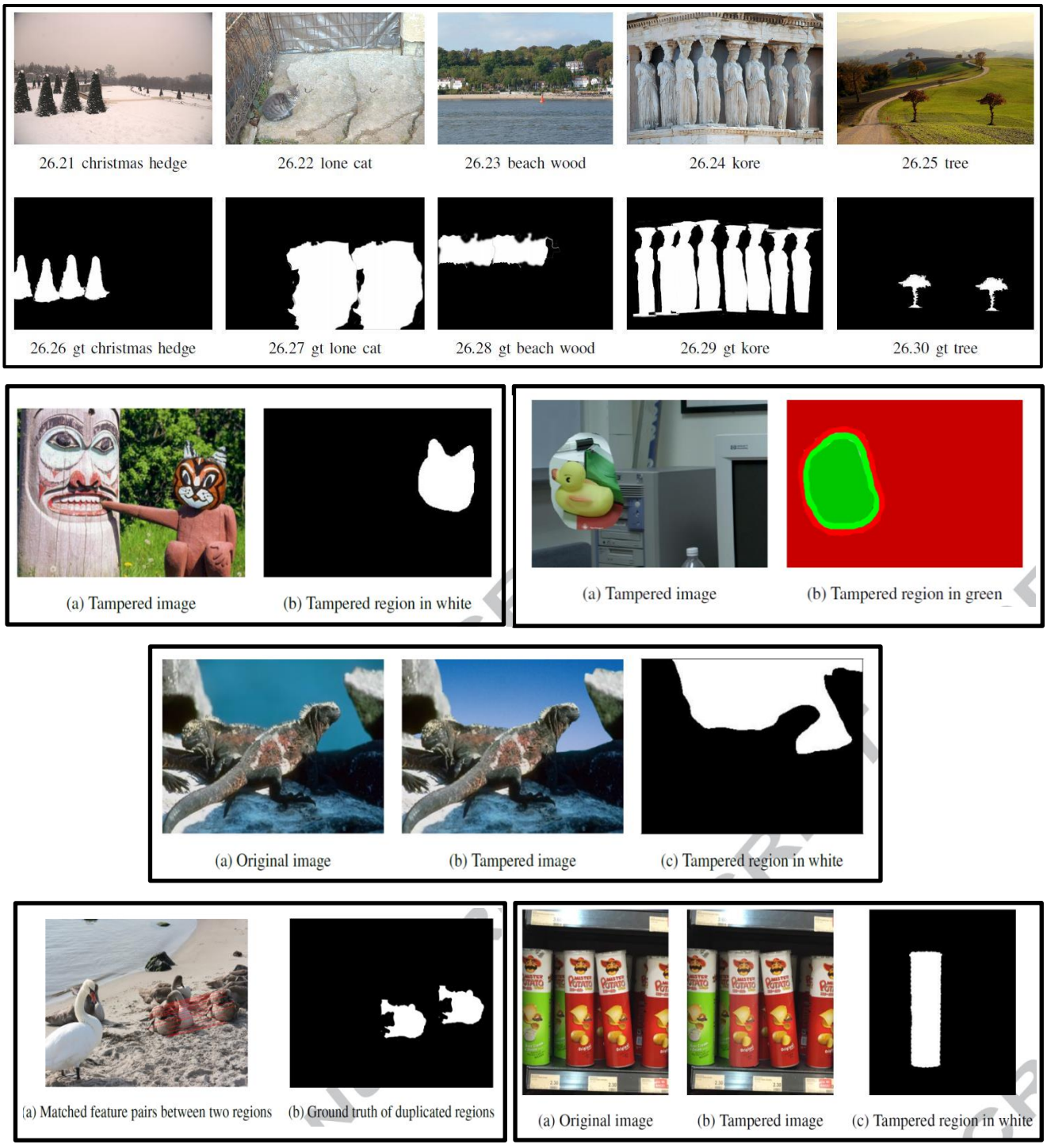


Figure 2.1: Some of detected image forgeries available techniques

2.2 Data analysis and Dimension reduction techniques

2.2.1 Principal Component Analysis (PCA)

INTRODUCTION

The central idea of ***principal component analysis (PCA)*** is to reduce the dimensionality of a data set consisting of a large number of interrelated variables while retaining as much as possible of the variation present in the data set. This is achieved by transforming to a new set of variables, the *principal components (PCs)*, which are uncorrelated, and which are ordered so that the first few retain most of the variation present in all of the original variables

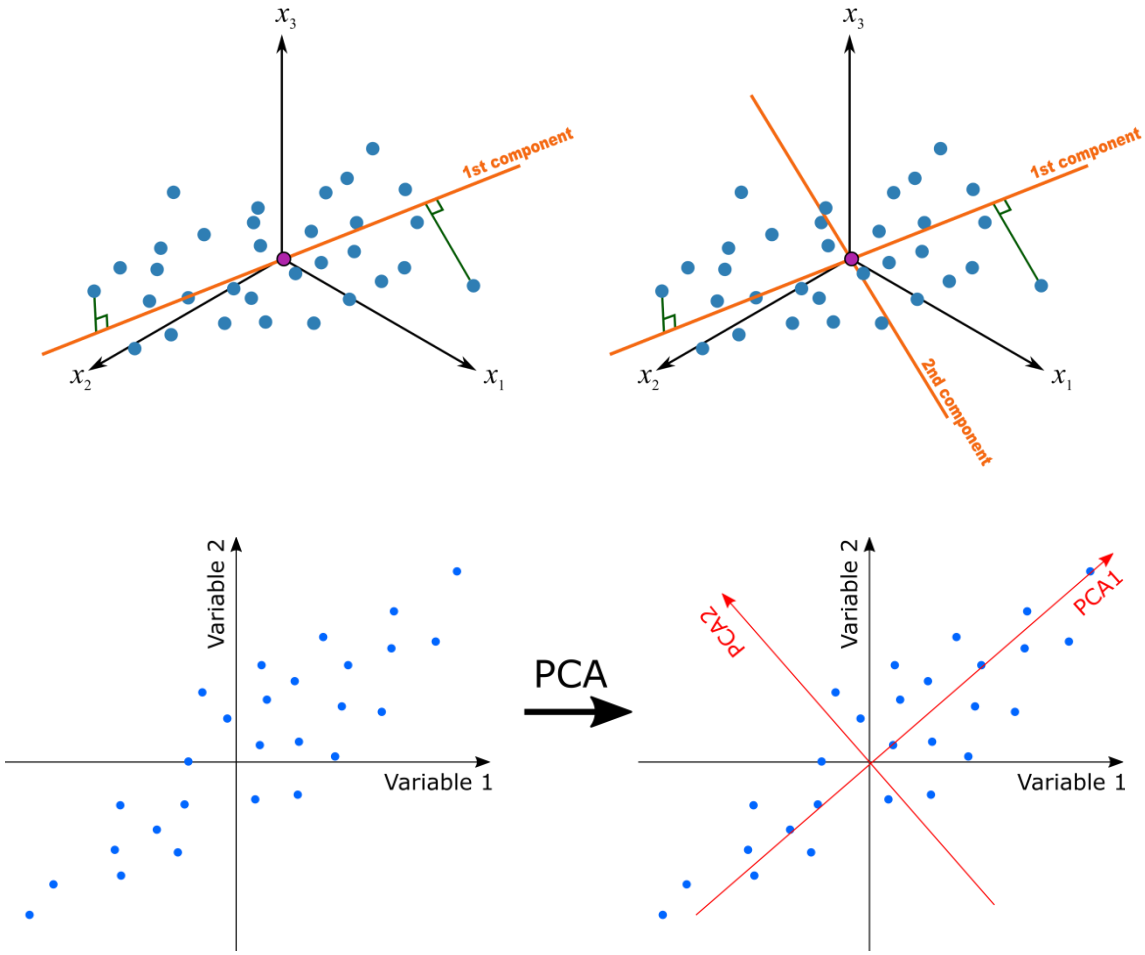


Figure 2.2: Simple illustration of PCA

STEP BY STEP PROCEDURE WITH AN EXAMPLE

Step 01: Mean subtraction

After taking the data set that we want to reduce the dimension, find the mean of the data set in each dimension. And then, from the each of the data dimensions, the corresponding mean should be subtracted. The mean subtracted is the average across each dimension. Table 2.1 shows the two-dimensional data set used in this example and Table 2.2 shows the produced data set after subtracting the mean. And the *Figures 2.3 & 2.4* illustrates those data points

Table 2.1: Initial data set

Dimension 1 (X)	Dimension 2 (Y)
0.81	0.16
0.91	0.97
0.13	0.96
0.91	0.49
0.63	0.80
0.10	0.14
0.28	0.42
0.55	0.92
0.96	0.79
0.96	0.96
4.97	5.12
3.11	3.10
5.55	3.83
5.80	3.14
5.04	3.29
5.27	5.47
5.23	5.08
4.18	3.95
4.97	5.85
3.51	3.10

Table 2.2: Mean subtracted data set

Dimension 1 (X_{new})	Dimension 2 (Y_{new})
-1.88	-2.27
-1.79	-1.46
-2.57	-1.47
-1.78	-1.94
-2.06	-1.63
-2.60	-2.28
-2.41	-2.01
-2.15	-1.51
-1.74	-1.63
-1.73	-1.47
2.27	2.69
0.41	0.67
2.85	1.40
3.11	0.71
2.34	0.86
2.58	3.04
2.54	2.66
1.48	1.52
2.27	3.42
0.82	0.68

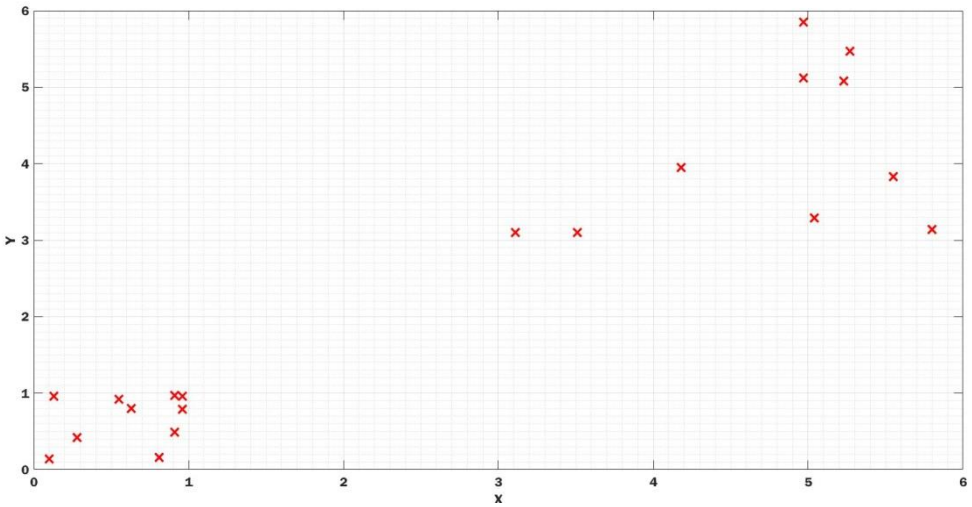


Figure 2.3: Initial data set

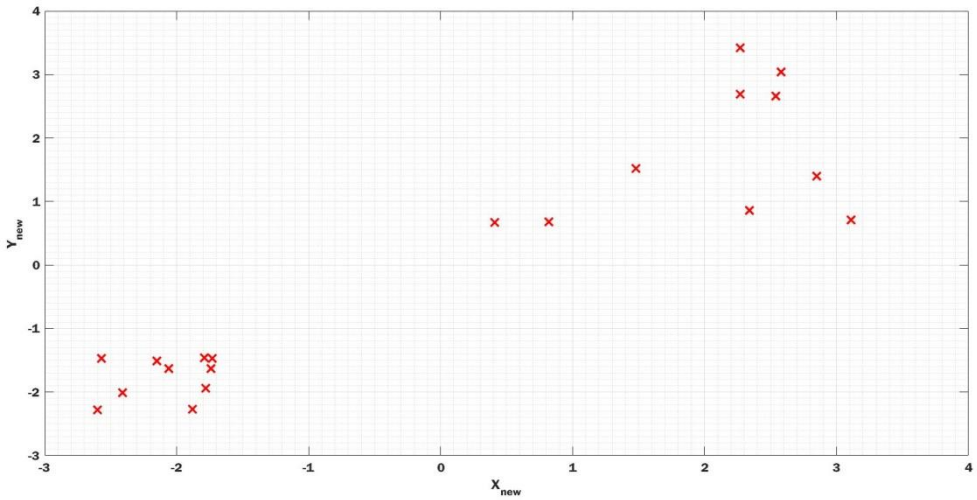


Figure 2.4: Mean subtracted data set

Step 02: Generating the Covariance Matrix

Following formula is used to construct the ***covariance matrix*** of a data set

$$\text{cov} = \frac{\sum(x_n - \mu_n)(x_n - \mu_n)^T}{\text{n}-1}$$

where,

x_n = vector containing dimension n data
μ_n = mean of dimension n data
n = dimension of the data set

In this example, n = 2. And the generated covariance matrix is,

$$\text{cov} = \begin{bmatrix} 4.9310 & 4.0352 \\ 40352 & 3.8916 \end{bmatrix}$$

Step 03: Calculation of the eigenvectors and eigenvalues of the covariance matrix

Let **C** be the ***covariance matrix*** of the data set. Then **C** will be always square & symmetric matrix. To calculate the ***eigenvalues (λ)*** and ***eigenvectors (v)*** used the following formula,

$$\text{Cv} = \lambda\text{v}$$
$$(\text{C} - \lambda\text{I})\text{v} = 0$$

To find the eigen values,

$$|C - \lambda I| = 0$$

To find the eigenvectors,

Cv = λv can be used.

In this example, the eigenvalues (λ) and the eigenvectors (v) can be obtained as below

$$\lambda = \begin{pmatrix} 8.4739 \\ 0.3424 \end{pmatrix}$$

$$\text{v} = \begin{pmatrix} -0.7509 & 0.6604 \\ -0.6604 & -0.7509 \end{pmatrix}$$

It is important to select the ***normalized eigenvectors*** in PCA (Norm of the eigenvector should be When looking at the eigenvector direction on top of the real data set, (*In Figure 2.5*) It is clearly seen that the directions given by those eigenvectors are the maximum and minimum variation direction of the data set. Similarly, for n- dimensional data set also, PCA gives the variance maximum to minimum directions as the eigenvector directions.

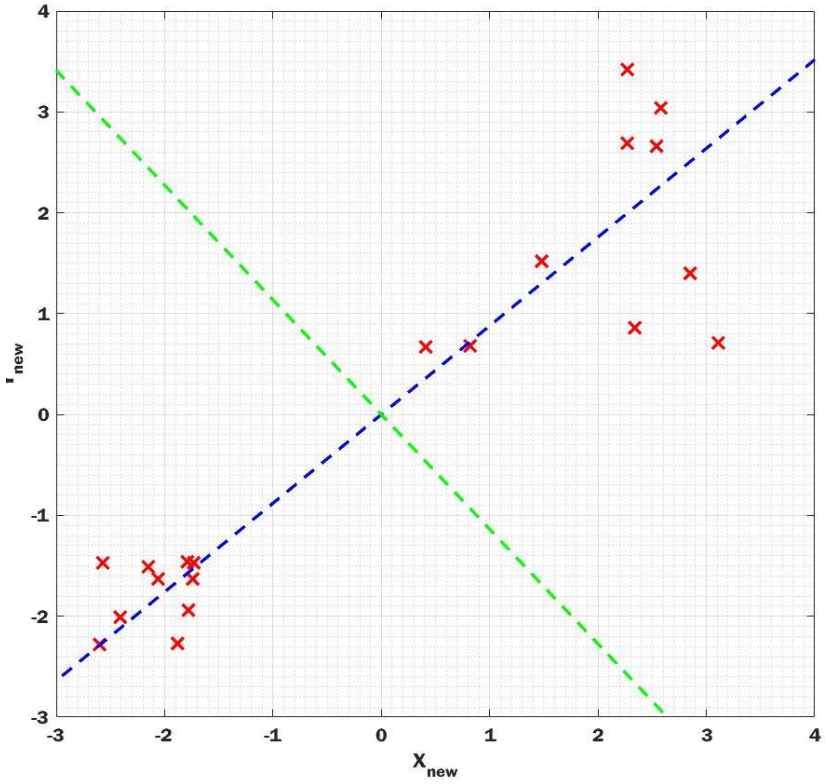


Figure 2.5: Eigenvector direction on mean subtracted data set

Step 04: Choosing principal components and forming a feature vector

Eigenvector with the highest eigenvalue is the ***principal component*** of the data set. Therefore after finding the eigenvectors from the covariance matrix, they should be rearranged in the descending order of the magnitude of the eigenvalues. This gives the components in order of significance. By leaving out some of the lesser significance components, the final data set can be transformed into a low dimensional space than the original. Therefore a ***feature vector*** can be formed using the higher significance components.

$$\text{Feature vector} = \left[v_1 \; v_2 \; \dots \; v_p \right]$$

where, v_i's are eigenvectors of the covariance matrix. ($i = 1, 2, \dots, p$)

Step 05: Transforming the data set into the new dataset which has lower dimension

Once the feature vector is formed, the original data set can be transformed using following equation.

$$\text{Transformed data} = \text{Feature vector}^T . \text{X}$$

where, X is the initial data set

Following is the transformed data set of the above example and their illustration

Table 2.3: Transformed data set into new feature space

Dimension 1 (PC_1)	Dimension 2 (PC_2)
2.91	0.46
2.31	-0.09
2.90	-0.59
2.62	0.28
2.62	-0.14
3.46	0.00
3.14	-0.08
2.61	-0.29
2.38	0.07
2.27	-0.04
-3.48	-0.52
-0.75	-0.23
-3.06	0.83
-2.80	1.52
-2.33	0.90
-3.94	-0.58
-3.66	-0.32
-2.12	-0.16
-3.96	-1.07
-1.06	0.03

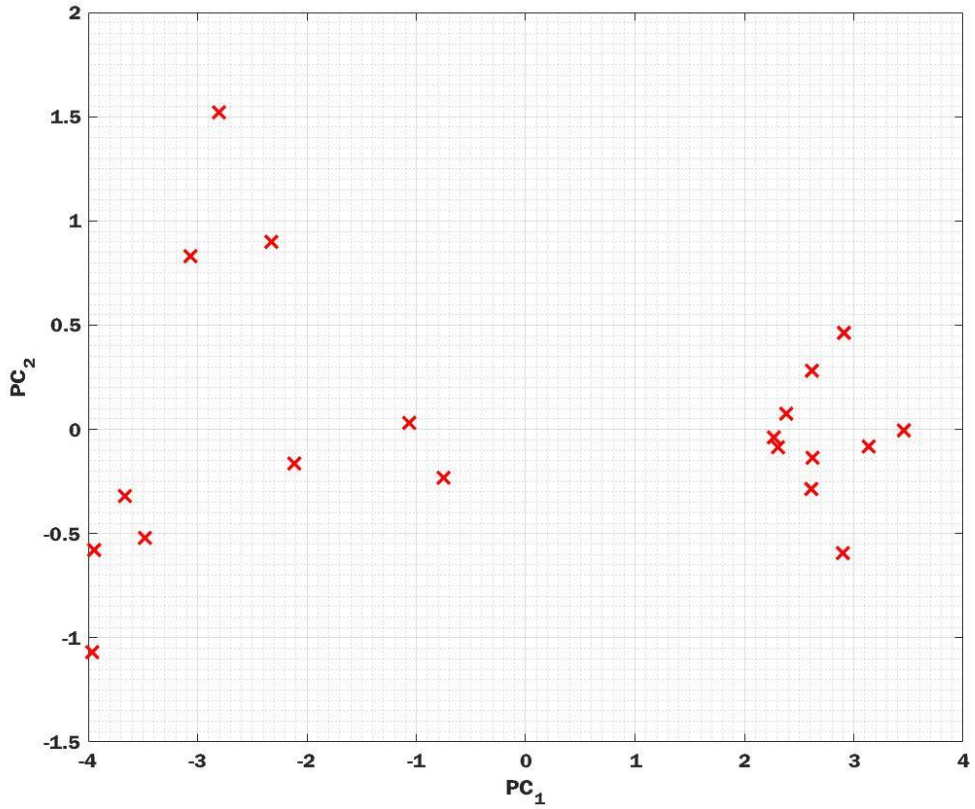


Figure 2.6: Plot of Transformed data set into new feature space

2.2.2 Linear Discriminant Analysis (LDA)

INTRODUCTION

Linear Discriminant Analysis (LDA) is most commonly used as dimensionality reduction technique in the pre-processing step for pattern-classification and machine learning applications. The goal is to project a dataset onto a lower-dimensional space with good class-separability in order avoid over-fitting ("curse of dimensionality") and also reduce computational costs. It is a simple clustering technique use to classify a large data set into few categories based on their means and the variances. And always used followed by PCA.

LDA picks a new dimension that gives,

- **Maximum separation between means of projected classes**
- **Minimum variance within projected classes**

$$\max \frac{(\mu_1 - \mu_2)^2}{({\sigma_1}^2 + {\sigma_2}^2)}$$

Uses Eigenvalues based on between-class and within-class covariance matrices.

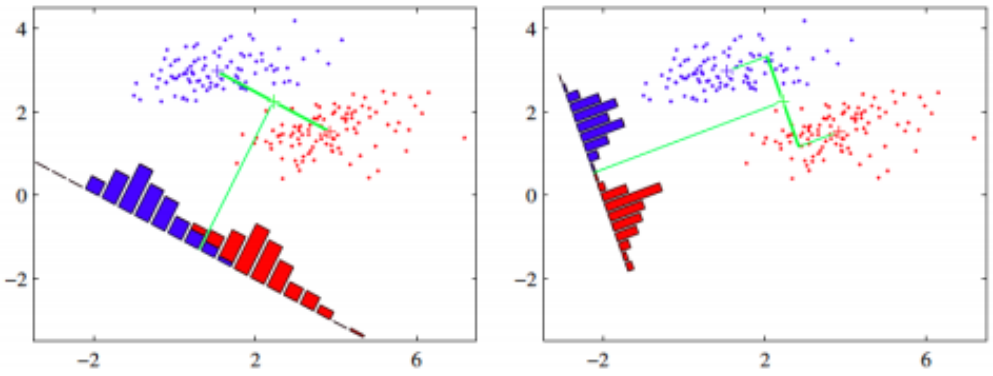


Figure 2.7: LDA illustration

PCA VS LDA COMPARISON

Both Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA) are linear transformation techniques that are commonly used for dimensionality reduction. PCA can be described as an "unsupervised" algorithm, since it "ignores" class labels and its goal is to find the directions (the so-called principal components) that maximize the variance in a dataset. In contrast to PCA, LDA is "supervised" and computes the directions ("linear discriminants") that will represent the axes that that maximize the separation between multiple classes.

Although it might sound intuitive that LDA is superior to PCA for a multi-class classification task where the class labels are known, this might not always the case. For example, comparisons between classification accuracies for image recognition after using PCA or LDA show that PCA tends to outperform LDA if the number of samples per class is relatively. In practice, it is also not uncommon to use both LDA and PCA in combination: E.g., PCA for dimensionality reduction followed by an LDA.

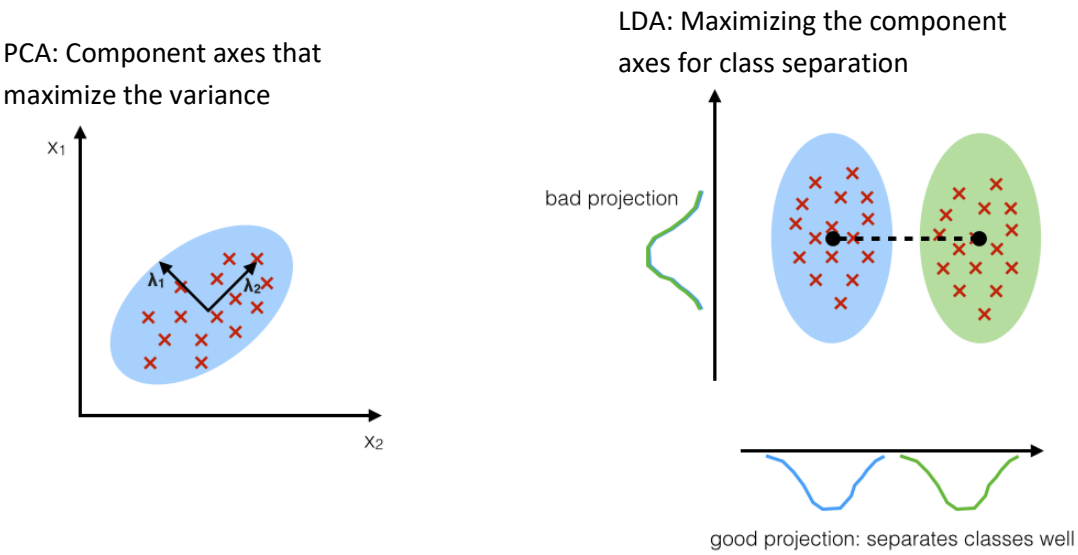


Figure 2.8: Comparison of PCA vs LDA

2.3 Mixture Models

Mixer model is a mixture of K base distributions mixed together as follows:

$$p(\boldsymbol{x}_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k p_k(\boldsymbol{x}_i|\boldsymbol{\theta})$$

This is a convex combination of the p_k's, since it is a weighted sum, where the mixing weights π_k satisfy $0 \leq \pi_k \leq 1$ and $\sum_{k=1}^{K} \pi_k = 1$.

2.3.1 Gaussian Mixture Model (GMM)

The most widely used mixture model is the **mixture of Gaussians** (MOG), also called a **Gaussian mixture model** or **GMM**. In this model, each base distribution in the mixture is a multivariate Gaussian with mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$. Thus, the model has the form

$$p(\boldsymbol{x}_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Given a sufficiently large number of mixture components, a GMM can be used to approximate any density defined on $\mathbb{R}^D$. Figure 2.8 illustrates a mixture of 3 Gaussians in 1D.

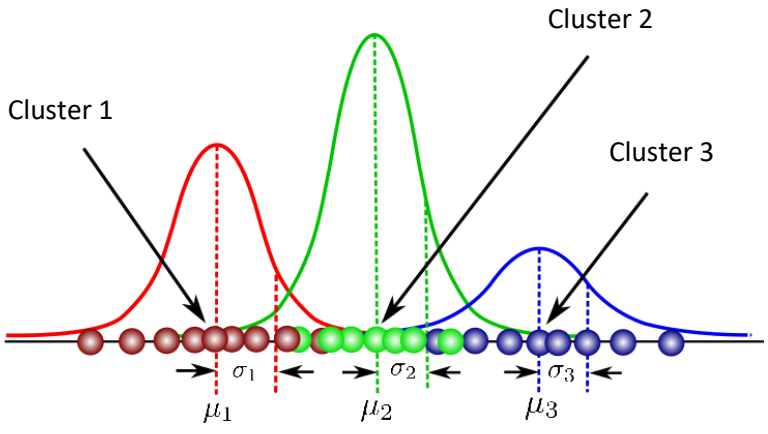


Figure 2.9: Illustration of 3 Gaussians in 1D

Figure 2.9 shows a mixture of three Gaussians in 2D. Each mixture component is represented by a different set of elliptical contours.

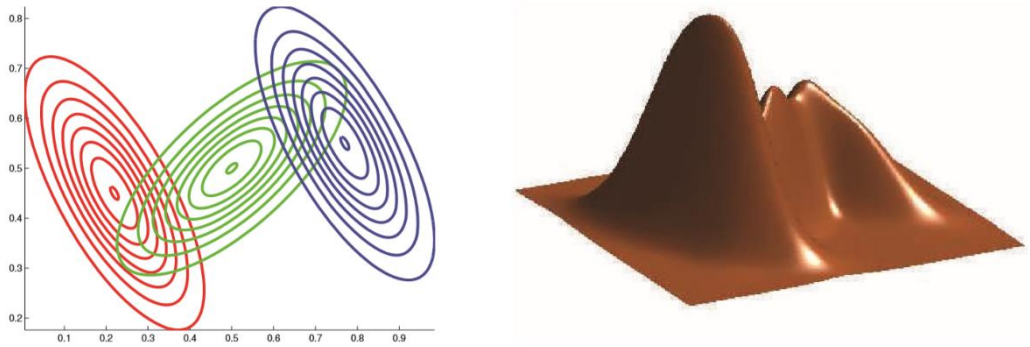


Figure 2.10: A mixture of 3 Gaussians in 2d. (a) We show the contours of constant probability for each component in the mixture. (b) A surface plot of the overall density.

2.3.2 The EM Algorithm

For many models, computing the **maximum likelihood** (**ML**) or **maximum a posterior** (**MAP**) parameter estimate is easy provided all the values of all the relevant random variables are observed, i.e., if we have complete data. However, if we have missing data and/or latent variables, then computing the ML/MAP estimate becomes hard.

One approach is to use a generic gradient-based optimizer to find a local minimum of the **negative log likelihood** or **NLL**, given by

$$NLL(\boldsymbol{\theta}) \triangleq -\frac{1}{N}\log p(\mathcal{D}|\boldsymbol{\theta})$$

However, we often have to enforce constraints, such as the fact that covariance matrices must be positive definite, mixing weights must sum to one, etc., which can be tricky. In such cases, it is often much simpler (but not always faster) to use an algorithm called **expectation maximization**, or **EM** for short. This is a simple iterative algorithm, often with closed-form updates at each step. Furthermore, the algorithm automatically enforces the required constraints.

EM exploits the fact that if the data were fully observed, then the ML/ MAP estimate would be easy to compute. In particular, EM is an iterative algorithm which alternates between inferring the missing values given the parameters (E step), and then optimizing the parameters given the "filled in" data (M step).

BASIC IDEA

Let $\boldsymbol{x}_i$ be the visible or observed variables in case i, and let $\boldsymbol{z}_i$ be the hidden or missing variables. The goal is to maximize the log likelihood of the observed data:

$$\ell(\boldsymbol{\theta}) = \sum_{i=1}^{N} p(\boldsymbol{x}_i|\boldsymbol{\theta}) = \sum_{i=1}^{N} \log \left[\sum_{z_i} p(\boldsymbol{x}_i, \boldsymbol{z}_i|\boldsymbol{\theta}) \right]$$

Unfortunately, this is hard to optimize, since the log cannot be pushed inside the sum.

EM gets around this problem as follows. Define the complete data log likelihood to be

$$\ell_c(\boldsymbol{\theta}) \triangleq \sum_{i=1}^{N} p(\boldsymbol{x}_i, \boldsymbol{z}_i|\boldsymbol{\theta})$$

This cannot be computed, since $\boldsymbol{z}_i$ is unknown. So, let us define the expected complete data log likelihood as follows:

$$Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^{t-1}) = \mathbb{E}[\ell_c(\boldsymbol{\theta})|\mathcal{D}, \boldsymbol{\theta}^{t-1}]$$

where t is the current iteration number. Q is called the **auxiliary function**. The expectation is taken wrt the old parameters, $\boldsymbol{\theta}^{t-1}$, and the observed data $\mathcal{D}$. The goal of the **E step** is to compute $Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^{t-1})$, or rather, the terms inside of it which the MLE depends on; these are known as the **expected sufficient statistics** or ESS. In the **M step**, we optimize the Q function w.r.t. $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^t = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^{t-1})$$

To perform MAP estimation, we modify the M step as follows:

$$\boldsymbol{\theta}^t = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^{t-1}) + \log p(\boldsymbol{\theta})$$

The E step remains unchanged.

EM algorithm monotonically increases the log likelihood of the observed data (plus the log prior, if doing MAP estimation), or it stays the same. So, if the objective ever goes down, there must be a bug in our math or our code. (This is a surprisingly useful debugging tool!)

Below we explain how to perform the E and M steps for several simple models, that should make things clearer.

2.3.3 EM for GMMs

In this section, we discuss how to fit a mixture of Gaussians using EM. We assume the number of mixture components, K, is known.

AUXILIARY FUNCTION

The expected complete data log likelihood is given by

$$
\begin{aligned}
Q(\boldsymbol{\theta}^t, \boldsymbol{\theta}^{t-1}) &\triangleq \mathbb{E}\left[\sum_i p(\boldsymbol{x}_i, z_i|\boldsymbol{\theta})\right] \\
&= \sum_i \mathbb{E}\left[\log\left[\prod_{k=1}^{K}\left(\pi_k p(\boldsymbol{x}_i|\boldsymbol{\theta}_k)\right)^{\mathbb{I}(z_i=k)}\right]\right] \\
&= \sum_i \sum_k \mathbb{E}[\mathbb{I}(z_i = k)] \log[\pi_k p(\boldsymbol{x}_i|\boldsymbol{\theta}_k)] \\
&= \sum_i \sum_k p(z_i = k|\boldsymbol{x}_i, \boldsymbol{\theta}^{t-1}) \log[\pi_k p(\boldsymbol{x}_i|\boldsymbol{\theta}_k)] \\
&= \sum_i \sum_k r_{ik} \log \pi_k + \sum_i \sum_k r_{ik} \log p(\boldsymbol{x}_i|\boldsymbol{\theta}_k)
\end{aligned}
$$

where $r_{ik} \triangleq p(z_i = k|\boldsymbol{x}_i, \boldsymbol{\theta}^{t-1})$ is the **responsibility** that cluster k takes for data point i. This is computed in the E step, described below.

E STEP

The E step has the following simple form, which is the same for any mixture model:

$$r_{ik} = \frac{\pi_k p(\boldsymbol{x}_i|\boldsymbol{\theta}_k^{t-1})}{\sum_{k'} \pi_{k'} p(\boldsymbol{x}_i|\boldsymbol{\theta}_{k'}^{t-1})}$$

M Step

In the M step, we optimize Q wrt π and the $\boldsymbol{\theta}_k$. For π, we obviously have

$$\pi_k = \frac{1}{N}\sum_i r_{ik} = \frac{r_k}{N}$$

where $r_k \triangleq \sum_i r_{ik}$ is the weighted number of points assigned to cluster k.

To derive the M step for the μ_k and Σ_k terms, we look at the parts of Q that depend on μ_k and Σ_k. We see that the result is

$$
\begin{aligned}
\ell(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) &= \sum_i \sum_k r_{ik} \log p(\boldsymbol{x}_i|\boldsymbol{\theta}_k) \\
&= -\frac{1}{2}\sum_i r_{ik}[\log|\boldsymbol{\Sigma}_k| + (\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)]
\end{aligned}
$$

This is just a weighted version of the standard problem of computing the **maximum likelihood estimator** (**MLE**) of an **multi variate normal** (**MVN**). The new parameter estimates are given by

$$
\begin{aligned}
\boldsymbol{\mu}_k &= \frac{\sum_i r_{ik} \boldsymbol{x}_i}{r_k} \\
\boldsymbol{\Sigma}_k &= \frac{\sum_i r_{ik} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T}{r_k} = \frac{\sum_i r_{ik} \boldsymbol{x}_i \boldsymbol{x}_i^T}{r_k} - \boldsymbol{\mu}_k \boldsymbol{\mu}_k^T
\end{aligned}
$$

These equations make intuitive sense: the mean of cluster k is just the weighted average of all points assigned to cluster k, and the covariance is proportional to the weighted empirical scatter matrix.

After computing the new estimates, we set $\boldsymbol{\theta}^t = (\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ for $k = 1{:}K$, and go to the next E step.

Algorithm: EM for GMM

---

initialize $\boldsymbol{\theta_k}{:}\boldsymbol{\pi_k}, \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}$ *for* $\boldsymbol{k = 1{:}K}$

repeat

E step (do for all i, k)

$$r_{ik} \leftarrow \frac{\pi_k p(\boldsymbol{x}_i|\boldsymbol{\theta}_k)}{\sum_{k'} \pi_{k'} p(\boldsymbol{x}_i|\boldsymbol{\theta}_{k'})}$$

$$r_k \leftarrow \sum_i r_{ik}$$

M step (do for all k)

$$\pi_k \leftarrow \frac{r_k}{N}$$

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_i r_{ik} \boldsymbol{x}_i}{r_k}$$

$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_i r_{ik} \boldsymbol{x}_i \boldsymbol{x}_i^T}{r_k} - \boldsymbol{\mu}_k \boldsymbol{\mu}_k^T$$

do for all k

$$\boldsymbol{\theta}_k \leftarrow (\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Log-likelihood

$$\ell(\boldsymbol{\theta}) \leftarrow \sum_{i=1}^{N} p(\boldsymbol{x}_i|\boldsymbol{\theta})$$

until *converged (Check using log-likelihood* $\boldsymbol{\ell(\theta)}$*)*

2.4 Different noise models in digital images

Noise is random signal. It is used to destroy most of the part of image information. Image distortion is most pleasance problems in image processing. Image distorted due to various types of noise such as Gaussian noise, Poisson noise, Speckle noise, Salt and Pepper noise and many more are fundamental noise types in case of digital images. These noises may be came from a noise sources present in the vicinity of image capturing devices, faulty memory location or may be introduced due to imperfection/inaccuracy in the image capturing devices like cameras, misaligned lenses, weak focal length, scattering and other adverse conditions may be present in the atmosphere.

A brief introduction of some of those noise models are described below.

2.4.1 Gaussian Noise

It is also called as electronic noise because it arises in amplifiers or detectors. Gaussian noise caused by natural sources such as thermal vibration of atoms and discrete nature of radiation of warm objects. Gaussian noise generally disturbs the gray values in digital images. That is why Gaussian noise model essentially designed and characteristics by its PDF or normalizes histogram with respect to gray value. This is given as,

$$P(g) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(g-\mu)^2}{2\sigma^2}}$$

Where,

$g - gray\ value$
$\mu - mean$
$\sigma - standard\ deviation$

Generally Gaussian noise mathematical model represents the correct approximation of real world scenarios. In this noise model, the mean value is zero, variance is 0.1 and 256 gray levels in terms of its PDF which can be illustrated as follows.

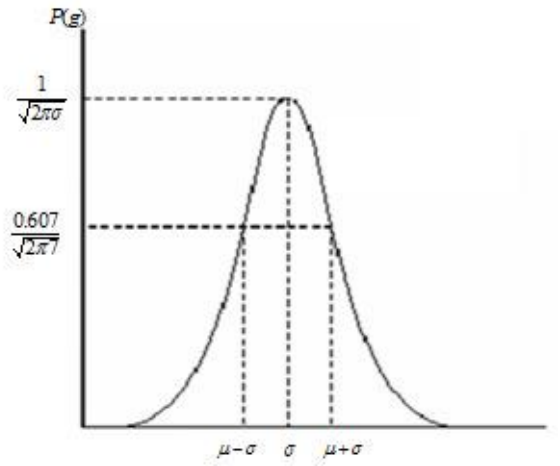


Figure 2.11: PDF of Gaussian noise

2.4.2 White Noise

Noise is essentially identified by the noise power. Noise power spectrum is constant in white noise. This noise power is equivalent to power spectral density function.

However neither Gaussian property implies the white sense. The range of total noise power is $-\infty$ to $+\infty$ available in white noise in frequency domain. That means ideally noise power is infinite in white noise. This fact is fully true because the light emits from the sun has all the frequency components. In white noise, correlation is not possible because of every pixel values are different from their neighbors. That is why autocorrelation is zero. So that image pixel values are normally disturb positively due to white noise.

2.4.3 Impulse Valued Noise (Salt and Pepper Noise)

This is also called data drop noise because statistically its drop the original data values. This noise is also referred as salt and pepper noise. However the image is not fully corrupted by salt and pepper noise instead of some pixel values are changed in the image. Although in noisy image, there is a possibilities of some neighbors does not changed. This noise is seen in data transmission. Image pixel values are replaced by corrupted pixel values either maximum 'or' minimum pixel value i.e., 255 'or' 0 respectively, if number of bits are 8 for transmission.

Let us consider 3x3 image matrices which are shown in the following figure. Suppose the central value of matrices is corrupted by **Pepper noise**. Therefore, this central value i.e. 212 is given in following figure is replaced by value zero. In this connection, we can say that, this noise is inserted dead pixels either dark or bright. So in a salt and pepper noise, progressively dark pixel values are present in bright region and vice versa.

254	207	210
68	169	35
101	174	23

→

254	207	210
68	0	35
101	174	23

Figure 2.12: The central pixel value corrupted by Pepper noise

Inserted dead pixel in the picture is due to errors in analog to digital conversion and errors in bit transmission. The percentagewise estimation of noisy pixels, directly determine from pixel metrics. The PDF of this noise is shown in the following figure.

If mean is zero and variance is 0.05. Here we will meet two spike one is for bright region (where gray level is less) called ''region a'' and another one is dark region (where gray level is large) called ''region b'', we have clearly seen here the PDF values are minimum and maximum in ''region a'' and ''region b'', respectively. Salt and Pepper noise generally corrupted the digital image by malfunctioning of pixel elements in camera sensors, faulty memory space in storage, errors in digitization process and many more.

$$P(g) = \begin{cases} P_a \; for \; g = a \\ P_b \; for \; g = b \\ 0 \;\; otherwise \end{cases}$$

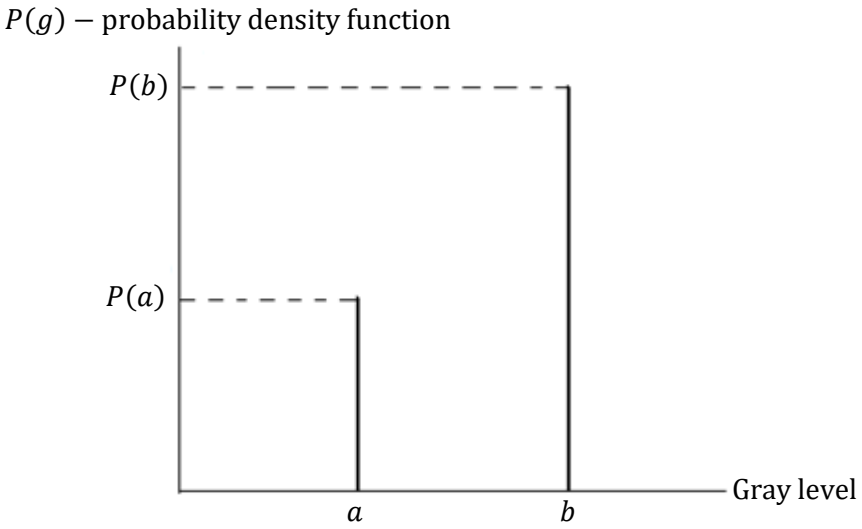


Figure 2.13: PDF of Salt pepper noise

2.4.4 Periodic Noise

This noise is generated from electronics interferences, especially in power signal during image acquisition. This noise has special characteristics like spatially dependent and sinusoidal in nature at multiples of specific frequency. It's appears in form of conjugate spots in frequency domain. It can be conveniently removed by using a narrow band reject filter or notch filter.

2.4.5 Quantization Noise

Quantization noise appearance is inherent in amplitude quantization process. It is generally presents due to analog data converted into digital data. In this noise model, the signal to noise ratio (SNR) is limited by minimum and maximum pixel value, P_{min} and P_{max} respectively.

The SNR is given by the equation,

$$SNR_{dB} = 20\log_{10}\left(\frac{P_{max}-P_{min}}{\sigma_n}\right)$$

Where, $\sigma_n - Standard\ deviation\ of\ noise$

When the input is full amplitude sine wave, this SNR becomes,

$$SNR_{dB} = 6n + 1.76\ dB$$

Where, $n - Number\ of\ bits$

Quantization noise obeys the uniform distribution. That is why it is referred as **uniform noise**. Its PDF is shown in following figure.

$$P(g) = \begin{cases} \frac{1}{b-a} & if\ a \le g \le b \\ 0 & otherwise \end{cases}$$

Where, $\mu = \frac{a+b}{2}$ and $\sigma^2 = \frac{(b-a)^2}{12}$

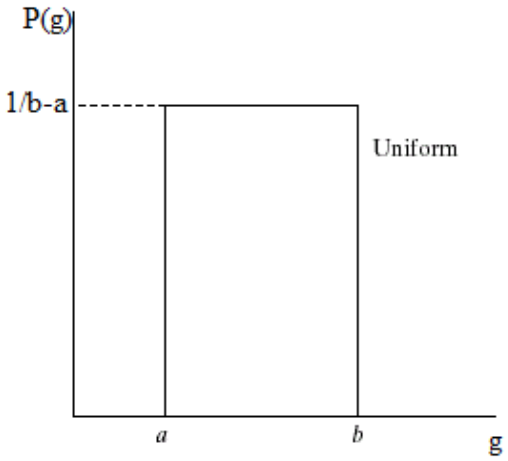


Figure 2.14: PDF of uniform noise

2.4.6 Speckle Noise

This noise is multiplicative noise. Their appearance is seen in coherent imaging system such as laser, radar and acoustics etc. Speckle noise can exist similar in an image as Gaussian noise. Its probability density function follows gamma distribution and given as below.

$$F(g) = \frac{g^{\alpha-1}e^{\frac{-g}{a}}}{(\alpha-1)!a^{\alpha}}$$

Figure 2.15: Image with speckle noise with variance 0.04

2.4.7 Photon Noise (Poisson Noise)

The appearance of this noise is seen due to the statistical nature of electromagnetic waves such as x-rays, visible lights and gamma rays. The x-ray and gamma ray sources emitted number of photons per unit time. These rays are injected in patient's body from its source, in medical x rays and gamma rays imaging systems. These sources are having random fluctuation of photons. Result gathered image has spatial and temporal randomness. This noise is also called as quantum (photon) noise or shot noise. This noise obeys the Poisson distribution.

2.5 Spectral Clustering

2.5.1 Similarity Graphs

Given a set of data points $x_1, \ldots, x_n$ and some notion of similarity $s_{ij} \geq 0$ between all pairs of data points x_i and x_j, the intuitive goal of clustering is to divide the data points into several groups such that points in the same group are similar and points in different groups are dissimilar to each other. If we do not have more information than similarities between data points, a nice way of representing the data is in form of the **similarity graph** $G = (V, E)$. Each vertex v_i in this graph represents a data point x_i. Two vertices are connected if the similarity s_{ij} between the corresponding data points x_i and x_j is positive or larger than a certain threshold, and the edge is weighted by s_{ij}. The problem of clustering can now be reformulated using the similarity graph: we want to find a partition of the graph such that the edges between different groups have very low weights (which means that points in different clusters are dissimilar from each other) and the edges within a group have high weights (which means that points within the same cluster are similar to each other).

2.5.2 Graph Notation

Let $G = (V, E)$ be an undirected graph with vertex set $V = \{v_1, \ldots, v_n\}$. In the following we assume that the graph G is weighted, that is each edge between two vertices v_i and v_j carries a non-negative weight $w_{ij} \geq 0$. The weighted adjacency matrix of the graph is the matrix $W = (w_{ij})\,; i, j = 1, \ldots, n$. If $w_{ij} = 0$ this means that the vertices v_i and v_j are not connected by an edge. As G is undirected we require $w_{ij} = w_{ji}$. The degree of a vertex $v_i \in V$ is defined as,

$$d_i = \sum_{j=1}^{n} w_{ij}$$

Note that, in fact, this sum only runs over all vertices adjacent to v_i, as for all other vertices v_j the weight w_{ij} is 0. The degree matrix is defined as the diagonal matrix with the degrees $d_1, \ldots, d_n$ on the diagonal.

$$D = diag(d_1, \ldots, d_n)$$

2.5.3 Different Similarity Graphs

There are several popular constructions to transform a given set $x_1, \ldots, x_n$ of data points with pairwise similarities s_{ij} or pairwise distances d_{ij} into a graph. When constructing similarity graphs the goal is to model the local neighborhood relationships between the data points.

The ε-neighborhood graph: Here we connect all points whose pairwise distances are smaller than ε. As the distances between all connected points are roughly of the same scale (at most ε), weighting the edges would not incorporate more information about the data to the graph. Hence, the ε-neighborhood graph is usually considered as an un-weighted graph. Figure shows an example of an ε-neighborhood graph.

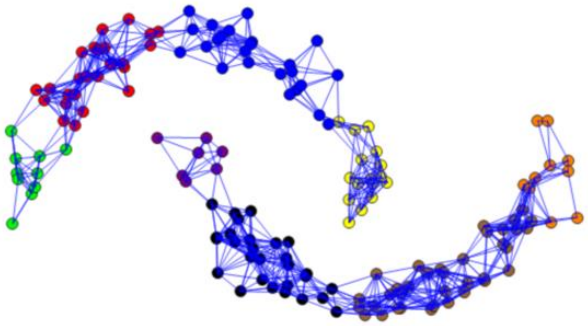


Figure 2.16: ε-neighborhood graph ε = 0.28 and 8 clusters

k-nearest neighbor graphs: Here the goal is to connect vertex v_i with vertex v_j if v_j is among the k-nearest neighbors of v_i. However, this definition leads to a directed graph, as the neighborhood relationship is not symmetric. There are two ways of making this graph undirected. The first way is to simply ignore the directions of the edges, that is we connect v_i and v_j with an undirected edge if v_i is among the k-nearest neighbors of v_j or if v_j is among the k-nearest neighbors of v_i. The resulting graph is what is usually called the k-nearest neighbor graph. Figure shows an example of a k-nearest neighbor graph.

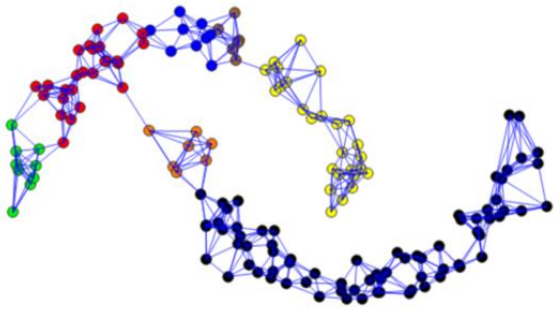


Figure 2.17: k-nearest neighbor graph k = 6 and 7 clusters

The second choice is to connect vertices v_i and v_j if both v_i is among the k-nearest neighbors of v_j and v_j is among the k-nearest neighbors of v_i. The resulting graph is called the mutual k-nearest neighbor graph. In both cases, after connecting the appropriate vertices we weight the edges by the similarity of their endpoints. Figure shows an example of a mutual k-nearest neighbor graph.

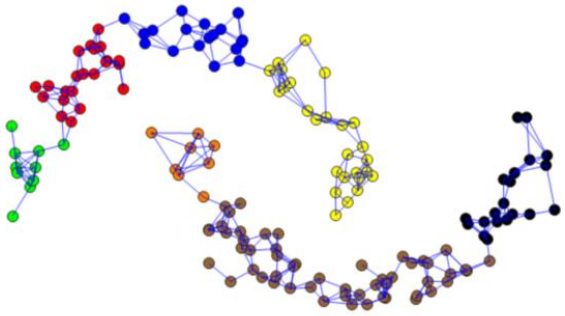


Figure 2.18: mutual k-nearest neighbor graph k = 6 and 7 clusters

The fully connected graph: Here we simply connect all points with positive similarity with each other, and we weight all edges by s_{ij}. As the graph should represent the local neighborhood

relationships, this construction is only useful if the similarity function itself models local neighborhoods. An example for such a similarity function is the Gaussian similarity function $s(x_i, x_j) = \exp\left(-\|x_i - x_j\|^2 / 2\sigma\right)$, where the parameter σ controls the width of the neighborhoods. This parameter plays a similar role as the parameter ε in case of the ε-neighborhood graph.

2.5.4 Graph Laplacians

2.5.4.1 The unnormalized graph Laplacian

The unnormalized graph Laplacian matrix is defined as

$$L = D - W$$

2.5.4.2 The normalized graph Laplacians

There are two matrices which are called normalized graph Laplacians in the literature. Both matrices are closely related to each other and are defined as

$$L_{sym} \triangleq D^{\frac{1}{2}} L D^{\frac{1}{2}} = I - D^{\frac{1}{2}} W D^{\frac{1}{2}}$$

$$L_{rw} \triangleq D^{-1} L = I - D^{-1} W.$$

We denote the first matrix by L_{sym} as it is a symmetric matrix, and the second one by L_{rw} as it is closely related to a random walk.

2.5.5 Spectral Clustering Algorithm

First took our graph and built a weight matrix. We then created the Graph Laplacian by subtracting the weight matrix from the degree matrix. The eigenvalues of the Laplacian are indicator of clusters. The vectors associated with those eigenvalues contain information on how to segment the nodes.

Algorithm: Spectral Clustering with Symmetric Graph Laplacian

Input: *Similarity matrix* $\boldsymbol{S} \in \boldsymbol{n} \times \boldsymbol{n}$*, number* $\boldsymbol{k}$ *of clusters to construct.*

- *Construct a similarity graph by one of the ways described. Let W be its weighted adjacency matrix.*
- *Compute the unnormalized Laplacian L.*
- *Compute the first k generalized eigenvectors* $\boldsymbol{u_1, \ldots, u_k}$ *of the generalized eigenproblem* $\boldsymbol{Lu = \lambda Du}$
- *Let* $\boldsymbol{U} \in \boldsymbol{n} \times \boldsymbol{k}$ *be the matrix containing the vectors* $\boldsymbol{u_1, \ldots, u_k}$ *as columns.*
- *For* $\boldsymbol{i = 1, \ldots, n}$ *let* $\boldsymbol{y_i} \in \boldsymbol{k}$ *be the vector corresponding to the* $\boldsymbol{i}$*-th row of* $\boldsymbol{U}$.
- *Cluster the points* $(\boldsymbol{y_i})_{\boldsymbol{i=1,\ldots,n}}$ *in* $\mathbb{R}^{\boldsymbol{k}}$ *with the k-means algorithm into clusters* $\boldsymbol{C_1, \ldots, C_k}$.

Output: *Clusters* $\boldsymbol{A_1, \ldots, A_k}$ *with* $\boldsymbol{A_i} = \{\boldsymbol{j} | \boldsymbol{y_j} \in \boldsymbol{C_i}\}$.

CHAPTER 3 SEGMENTATION

3.1 Introduction

Segmentation is a first step of most of the algorithms which were studied in literature served. As previously mentioned, our goal is to identify the spliced forgery region. It is a pixel-based forgery type. Because of that a proper method is required to analyses the image pixel. Also, images have lots of pixels. Therefore, segmenting the image is a significant part of the algorithm. Simplest way to address this problem is segmenting as blocks. The blocks which are at the boundary of the spliced region, contains both the original image pixels and forged image pixel by considerable percentage of amount. Because of that those blocks can't be categorized precisely. Because proper segmentation method was required and few segmentation methods were studied.

Region based segmentation methods

- Thresholding method
- Clustering based segmentation

Super pixel-based method

- Simple linear iterative clustering (SLIC)
- Simple non iterative clustering (SNIC)

3.2 Threshold segmentation

Threshold segmentation is the simplest method of image segmentation. It is a common segmentation algorithm which directly divides the image gray scale information processing based on the gray value of different targets. Threshold segmentation can be divided into local threshold method and global threshold method. The global threshold method divides the image into two regions of the target and the background by a single threshold. The local threshold method needs to select multiple segmentation thresholds and divides the image into multiple target regions and backgrounds by multiple thresholds. The most commonly used threshold segmentation algorithm is the largest interclass variance method, which selects a globally optimal threshold by maximizing the variance between classes. In addition to this, there are entropy-based threshold segmentation method, minimum error method, co-occurrence matrix method, moment preserving method, simple statistical method, probability relaxation method, fuzzy set method and threshold methods combined with other methods.

The advantage of the threshold method is that the calculation is simple and the operation speed is faster. In particular, when the target and the background have high contrast, the segmentation effect can be obtained. The disadvantage is that it is difficult to obtain accurate results for image segmentation problems where there is no significant gray scale difference or a large overlap of the gray scale values in the image. Since it only takes into account the gray information of the image without considering the spatial information of the image, it is sensitive to noise and grayscale unevenness, leading it often combined with other methods.

So, when it was studied, lots of drawbacks were identified regarding to our required segmentation tasks because this is not considering the spatial information of the image pixels. Also edge of the spliced region is smoothing and contrast level is changed. Therefore, thresholding segmentation is not good for our application and further studied was not done regarding to thresholding segmentation.

3.3 Clustering based segmentation

Segmentation is based on the application, that is used this segmentation for fulfill a certain task. Overall idea of image segmentation is dividing the image into small group of pixels with compared to total image pixels. Therefore, clustering base segmentation is based on that idea. Image pixels are taken as a data set and then it is clustered according to clustering technique. Most common clustering technique is K-means clustering. Lots of works were done related to clustering-based segmentation through the K-means clustering. After refer those methods which are already developed, clustering based segmentation method was implemented to identify how it performs and how much capable for our task.

Steps of the K-means clustering base segmentation

I. RGB color values of the image pixels and their spatial position (x, y co-ordinate of the pixel) are taken as five-dimensional data set.

II. Then K number of centroids are defined inside the bounded region of the data set. K is a pre-defined number and it is equal to number of segments required in the image .it should be feed to the algorithm.

III. Then distance is calculated between a data point and every centroid which are defined in last step. After that data point is assigned to nearest centroid. All the data points are assigned to nearest centroid.

IV. After assigned all the data points, centroid of the cluster is updated by taking mean of the data points in the cluster.

V. Then step III is done with updated centroids as next iteration. That iterative procedure is done until stopping criteria is met

Result for the method is in figure 3.1. Image was taken from Columbia data set. Number of clusters were taken as 4 and weighting factor was 0.05.

(a) *(b)* *(c)*

(d) *(e)*

Figure 3.1 Results of K-means clustering based segmentation for images in Colombia data set (K=4 and weight factor =0.05) (a) original image (b)segment 1 (c) segment2 (d) segment 3 (e) segment 4

3.3.1 Distance calculation in K-mean clustering base segmentation

Distance calculation can be done using variance methods in mathematics. In K-means clustering, 2-norm of the vector difference is the most commonly used method. It is also called as Euclidean distance.

$$d_{ik} = \|C_k - P_i\|_2$$

Where,

d_{ik} -distance between kth centroid and ith pixel
C_k-kth centroid
P_i-ith pixel

But taking Euclidean distance with five dimensional as it is, special position factor is more dominant than the RGB color values. There Euclidean distance is modified by introduced a weighting factor in front of the special position part as follows.

$$d_{ik} = \sqrt{(R_k - r_i)^2 + (G_k - g_i)^2 + (B_k - b_i)^2 + W[(X_k - x_i)^2 + (Y_k - y_i)^2]}$$

Where,

d_{ik} -distance between kth centroid and ith pixel
C_k=$[R_k, G_k, B_k, X_k, Y_k]$
P_i=$[r_i, g_i, b_i, x_i, y_i]$
W-weighting factor

In above equation weighting factor is played a very important role in final output. According to the figure 3.2 when W is increased segmentation is deviated towards position-based segments. However, weighting factor is also defined according to the requirement of the application.

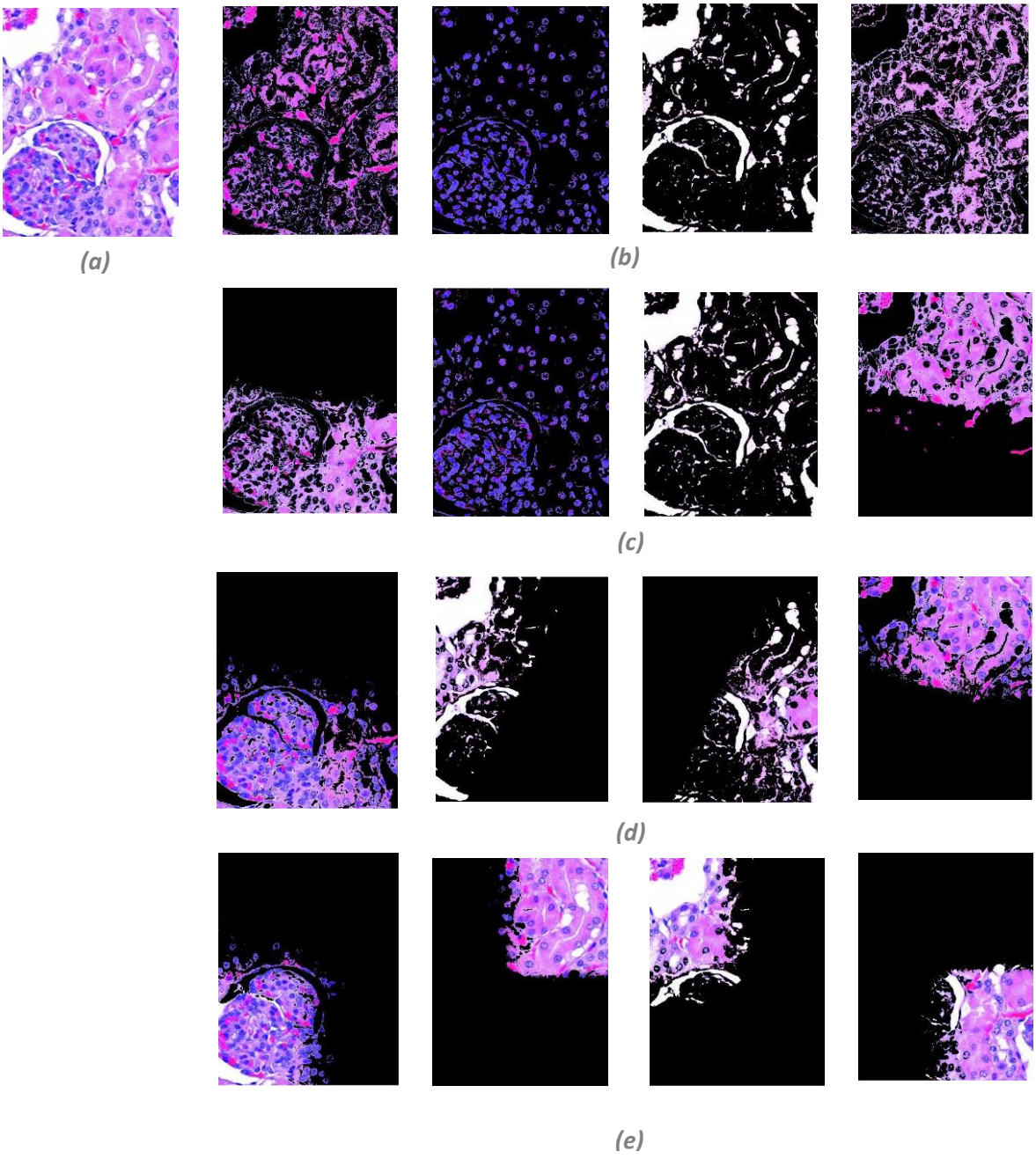


Figure 3.2 Results of K-means clustering based segmentation for different weighting factors (a) original image (b)weighting factor 0.01 (c) weighting factor 0.1 (d) weighting factor 0.5 (e) weighting factor 1

3.3.2 stopping criteria

Number of iterations can be given as an input to the algorithm. Then algorithm is run until complete those number of iterations. Another method is minimizing the mean square error in each cluster. In this method summation of mean square error is calculated between cluster centroid and each pixel in cluster. Then total summation is calculated by adding all cluster based mean square errors.

$$Total\ MSE = \sum_{k=1}^{K} \sum_{i=i}^{n_k} \|C_k - P_i\|_2$$

C_k-k[th] updated centroid
P_i-i[th] pixel in the K cluster
K-number of clusters
n_k-number of pixels in the cluster

Then algorithm is run until total summation become less than given tolerance. Tolerance is selected according to the task which is used the segmentation.

3.3.3. Drawbacks of K-means clustering based segmentation method

Selecting require number of clusters are very tough to introduced to an algorithm as well as it is also difficult in human sense. Because details of an image are different by image tot image. Therefore, pre-defined number is very tough to select. It is the main drawback of the method. Also, segments are not simply connected. In our application, spliced region is simply connected in most of the cases. Sometimes it may not be. But if it is simply connected, method is used to classify spliced and original image region will fail. Specially our case noise-based approach was used. Noise parameters were taken to a segment. Then segments were classified according to the noise parameters. Therefore, both spliced and original image pixels are in same amount will not get better noise parameter variation. Although segmentation was done before noise model creation, with that intuition those effects were tried to minimize with selecting appropriate segmentation method. Although K-means based segmentation is good method, some other methods were studied to reduce those drawbacks. Therefore, couple of super pixels-based segmentation methods were studied.

3.4 Super pixel-based methods

Super pixel means a group of pixels with same color. This kind of segmentation makes fast computation in image processing algorithms. Also selecting number of super-pixels are easier than k-means clustering. Because number of super pixels are in range of hundreds of those super pixel-based methods which are not focus to object detection as k-means clustering. Therefore, number of super pixels can be selecting according to the size of the image and also it can be introduced to an algorithm. However, couple of super pixel-based methods were studied and implemented in MATLAB to identify a better method for our objective.

3.4.1 Simple linear iterative clustering (SLIC)

Following steps are mainly done in approach.

I. Selecting number of super pixels is the first step of this method. It is decided according to the objective and size of the image. In our project, Columbia Dataset is used for testing the algorithm and all the images are same size in the data set. Number of super pixels (K) is made as an input to the algorithm.

II. RGB color space is changed to CIELAB color space. Because CEILAB color space is widely considered as perceptually uniform for small color distances.

CEILAB color space: In this color space there are three parameter a^*, $b^*$and L^*. a^* and b^* are in range -100 to +100 or -127 to +127 and L^* is in range 0 to +100. L^* is represent the lightness value. L^*=0 is corresponding to darkest black and L^*=100 is corresponding to brightness white. The color channels, *a** and *b**, represent true neutral gray values at *a** = 0 and *b** = 0. The *a** axis represents the green–red component, with green in the negative direction and red in the positive direction. The *b** axis represents the blue–yellow component, with blue in the negative direction and yellow in the positive direction.

III. Then calculate rough size of a super pixel is $\frac{N}{K}$ $where\ N = total\ number\ of\ pixels$. Initial centroids are placed in regular grid by maximize the spatial distance between two centroids. Roughly that is $\sqrt{S}$.

IV. Every pixel is taken as 5D vector with CEILAB color components and spatial position.

$$C_k = [a_k^*, b_k^*, l_k^* x_k, y_k]$$

Then every centroid is adjusted to minimum gradient point of their $n \times n$ neighborhood. When n is increases centroid placement is better. But it takes lot of computations. In our case n was taken as 3.

V. After that every pixel in $2s \times 2s$ neighborhood of each centroid is checked for assign to centroids. For that Euclidean distance is taken between each pixel and centroid as follows.

$$d_{lab} = \sqrt{(a_k - a_i)^2 + (b_k - b_i)^2 + (l_k - l_i)^2}$$
$$d_{xy} = \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2}$$
$$d \quad = d_{lab} + \frac{m}{s} d_{xy}$$

m is taken within 1 to 40.

VI. When considering $2s \times 2s$ neighborhood every pixel is compared with 4 centroids at initially and assigned to the nearest centroid.

VII. After assign all the pixels each pixel has group of pixels and centroid is updated by take the mean values of each vector component.

VIII. Then same procedure is redone to obtain a better result. As mentioned in the name this is an iterative process. Stopping point is decided according to the application that introduced this method. In the algorithm error E measured as mean square error between centroid and pixel assign to cluster. Then threshold point should introduce according to the application.

When the iterations are increased error is reduced. But the computational time is increased almost linearly. Because of that in our case, iterations were limited to 5.

3.4.1.1 Pseudo code of the SLIC algorithm

1: Initialize cluster centers $C_{k=[a_k^*,b_k^*,l_k^*x_k,y_k]}$ by sampling pixels at regular grid steps S.
2: Perturb cluster centers in an $n \times n$ neighborhood, to the lowest gradient position.
3: **Repeat**
4: **For** each cluster center C_k **do**
5: Assign the best matching pixels from a $2s \times 2s$ square neighborhood around the cluster center according to the distance measure
6: **end for**
7: Compute new cluster centers and residual error E {d distance between previous centers and recomputed centers}
8: **Until** E ≤ threshold
9: Enforce connectivity.

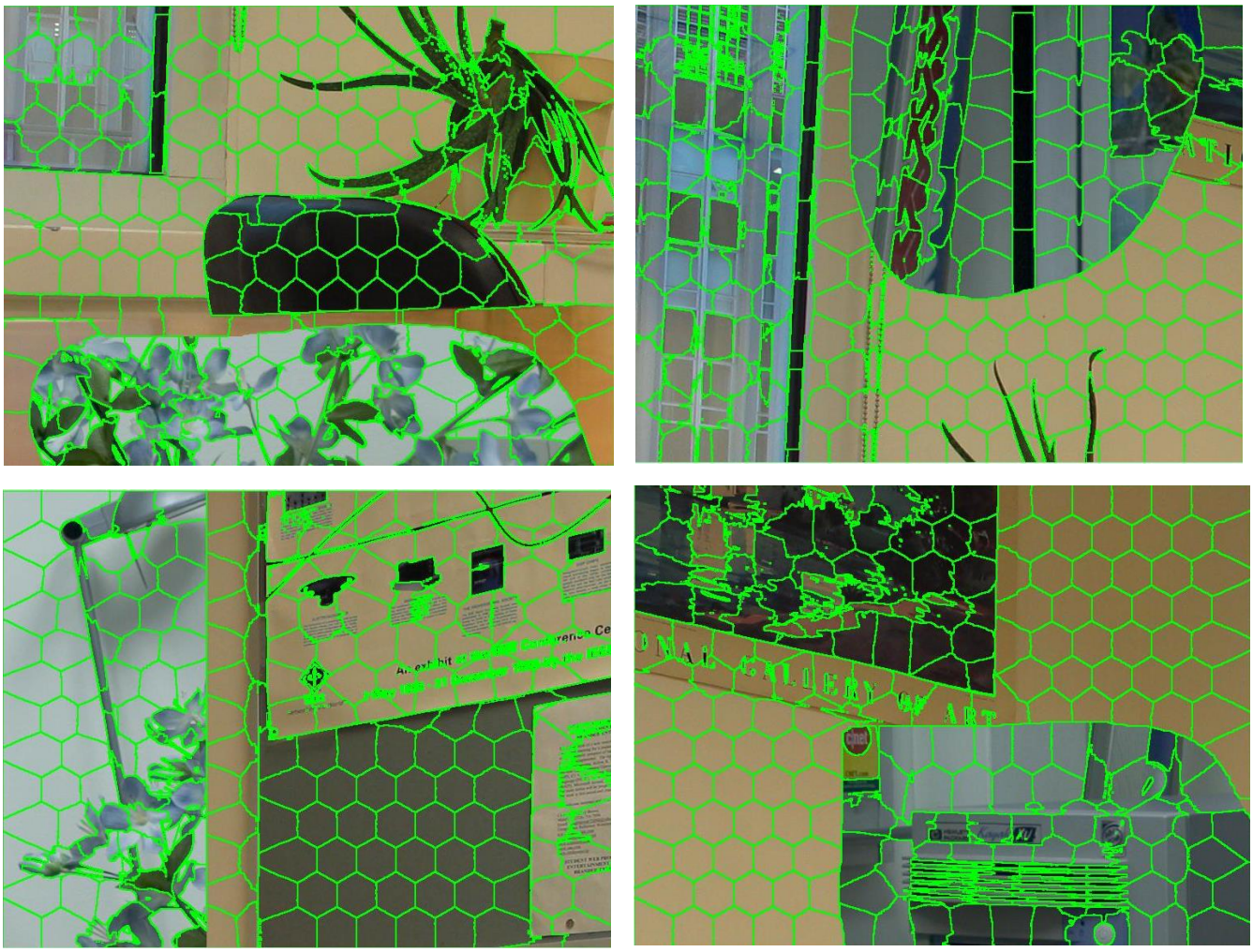


Figure 3.3 Results of SLIC base segmentation for images in Colombia data set

3.4.1.2 Draw Backs of the SLIC Segmentation Method

- Main drawback is super pixels are not simply connected. As shown in figure 3.3, there are some super pixels which are not simply connected. Those super pixels are highly textured. At the last point of the pseudo code is expressed enforce connectivity. But the inventors did not mention a method to do that connectivity in the paper they published.
- Time consumption is high in this method. For an example segmented image in figure 3.3 was taken 4minites and 34 seconds to make the output.

3.4.2 Simple Non-Iterative Clustering (SNIC)

This is an improved version of the simple linear iterative clustering (SLIC) method which was discussed earlier. As mention in the name this is a non-iterative process. Also, it has several features which are included to overcome the problems and weaknesses in the SLIC method. Following things are the important improvements of the method.

- SNIC method is the non-iterative process. Therefore, time consumption is less than SLIC method. Also, iterations are given a measure to accuracy and fineness of the final output. If the process is non-iterative then output is not depending on the no of iterations and because of that, fine tune for no of iterations are not required.
- Resulting super pixels are simply connected.

Those two key points are included in the SNIC method. Following steps are the SNIC algorithm. Also, at the beginning of the SNIC method which was mentioned as improved version of the SLIC method. Because of that some steps are similar to the steps in SLIC. Those points are not discussed with details again. The steps of the method can be stated as follows.

I. First step is selecting number of super pixels with the dimension of the image and application.
II. RGB color space is changed to CIELAB color space. Reason for the transformation is same as CEILAB color space is widely considered as perceptually uniform for small color distances.
III. Every pixel is taken as 3D vector with CEILAB color components and 2D vector with spatial position.

$$C_k = [a_k^*, b_k^*, l_k^*]$$
$$X_k = [x_k, y_k]$$

IV. Then calculate rough size of a super pixel is $\frac{N}{K}$ $where\ N = total\ number\ of\ pixels.$ Initial centroids are placed in regular grid by maximize the spatial distance between two centroids. Roughly that is $\sqrt{S}$.
V. The initial K seeds $C[k] = [X_k, c_k]$ are obtained on regular grid over the image. Using seed pixels, K $e_i = \left[X_i, c_i, k, d_{i,k}\right]$ are created, wherein each label k is set to one unique super pixel label from 1 to K, and each distance value $d_{i,k}$, representing the distance of the pixel from the k^{th} centroid, is set to zero.
VI. Then every centroid is adjusted to minimum gradient point of their $n \times n$ neighborhood. When n is increases centroid placement is better. But it takes lot of computations. In our case n was taken as 3.

VII. A priority queue Q is initialized with these K elements. When popped, Q always returns the element e_i whose distance value $d_{i,k}$to the kth centroid is the smallest.

VIII. Distance is calculated using following formula.

$$d_{i,j} = \sqrt{\frac{\|X_j - X_k\|_2^2}{S} + \frac{\|c_j - c_k\|_2^2}{m}}$$

where s and m are the normalizing factors for spatial and color distances, respectively.

IX. While Q is not empty, the top-most element is popped. If the pixel position on the label map L pointed to by the element is unlabeled, it is given the label of the centroid.

X. The centroid value, which is the average of all the pixels in the super pixel, is updated with this pixel. In addition, for each of its 4 or 8 neighbors that have not been labeled yet, a new element is created, assigning to it the distance from the connected centroid and the label of the centroid. These new elements are pushed on the queue.

XI. As the algorithm executes, the priority queue is emptied to assign labels at one end and populated with new candidates at the other. When there are no remaining unlabeled pixels to add new elements to the queue and the queue has been emptied, the algorithm terminates.

3.4.2.1 Pseudo Code of the SNIC Algorithm

Input: Input image *I, K* initial centroids $C[k] = [X_k, c_k]$ sampled on a regular grid, color normalization factor m
Output: Assigned label map *L*

```
Initialize L[:] ← 0
for kϵ[1,2, ….. K]do
    Element e ← [X_k, c_k, k, 0]
    Push e on priority queue Q
end for
while Q is not empty do
    Pop Q to get e_i
    if L[x_i] is 0 then
        L[x_i] = k_i
        Update centroid L[x_i] online with x_i and c_i
        for Each connected neighbor x_j of x_i do
            Create element e_i = [X_k, c_k, k_i, d_{j,k_i}]
            if L[x_i] is 0 then
                Push e_j on Q
            end if
        end for
    end if
end while
return L
```

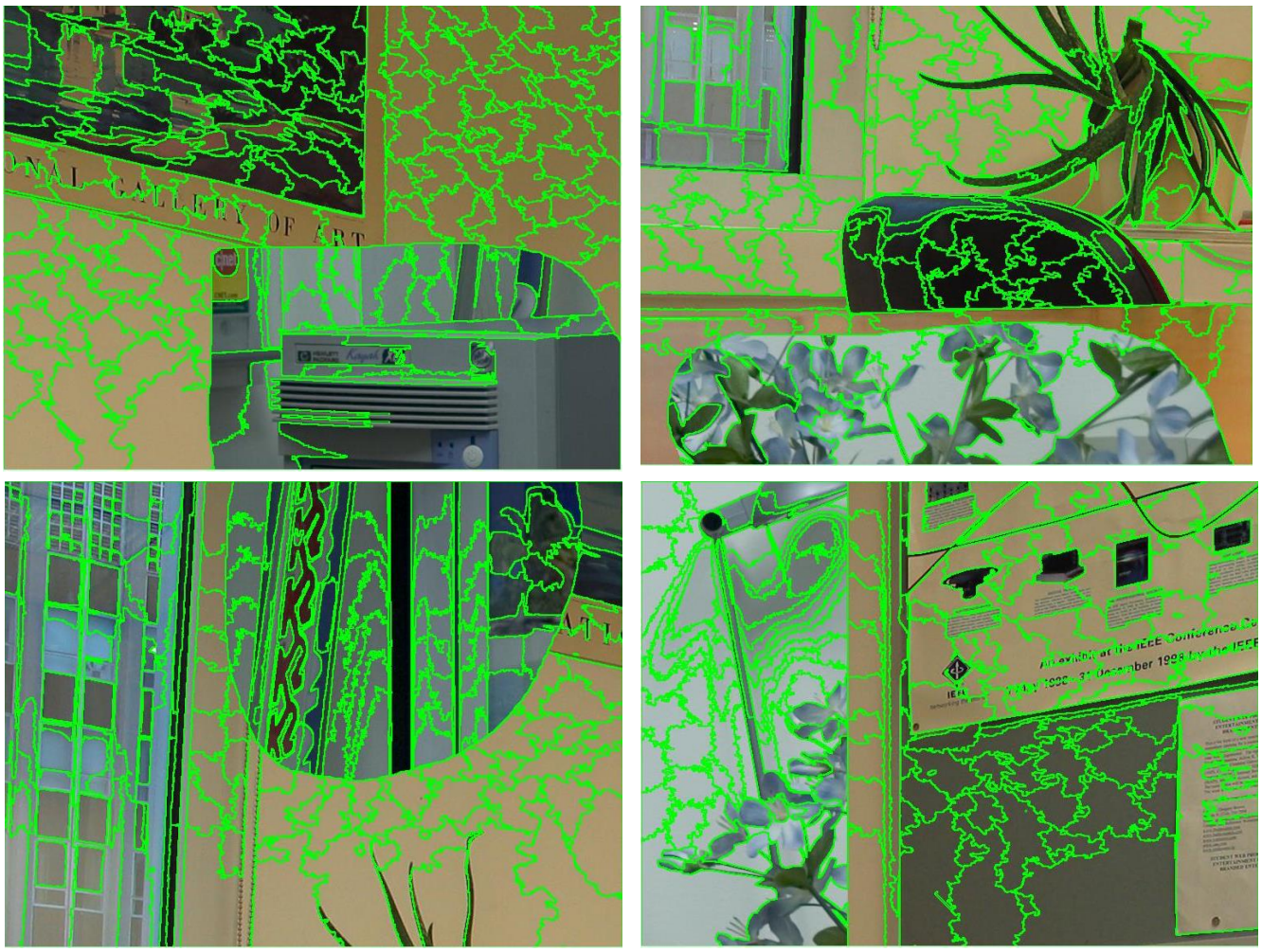


Figure 3.4 Results of SNIC Base Segmentation for Images in Colombia Data Set

3.4.3. Drawbacks of existing segmentation algorithm

After studding all the segmentation methods, super pixel-based methods were identified as most appropriate for our task with compared to other methods. SNIC method was the best from all of them. Few reasons were affected to our selection.

- In K-means or thresholding selection number of clusters were difficult. In super pixel-based method selection can be done according to the image size. Because in K-means clusters are very small number compared to the number of super pixels. In K-means, algorithm is tried to identify object wise region. But in super pixel-based method, algorithm gives attention to small localized area and therefore there are number of super pixels in same object in the picture. Therefore, in super pixel-based method is not try to read the image as its details and only gathered small group of same color pixel together.
- In super pixel segmentation, numbers of pixels are in same range than K-means clustering.
- In SNIC method super pixels are simply connected and all other methods were not simply connected.
- Super pixel-based methods are satisfied boundary preserving condition more than other methods.

However, couples of points were identified as drawbacks of superpixel based methods as well. Due to more uniform spread of super pixels, information less regions has lot of super pixels with almost same properties. Main reason for that is initial centroid placement is done in regular grid. Also, some super pixels have details which can be further segmented. Therefore, to overcome those problems adaptive centroid initialization was suggested to make the segmentation as more adaptive. Therefore, Adaptive centroid placement Based SNIC for superpixel segmentation method was proposed and detail of the proposed method is in the next section.

3.5. The proposed Adaptive Centroid Placement Based SNIC for Superpixel Segmentation

The proposed segmentation method can be mainly categorized into two stages.

- The first stage is the adaptive initial centroid placement with a higher density of centroids to the information-rich regions.
- The second stage is the enhancement of the segment by assigning pixels to the centroids according to a similarity measure between the pixels and the initial centroids.

The pixel assignment process is done as proposed in the SNIC method. The initial centroid placement of the SNIC method is uniform and it does not consider the information distribution of the image. In the proposed method, the initial centroids are placed adaptively with the information distribution of the image. The overall procedure is illustrated in the flowchart given in figure 3.5.

3.5.1 Adaptive Initial Centroids Placement

The main idea of the adaptive centroid placement is to initialize more centroids in regions with high detail content. Initially, the image was segmented into an array of non-overlapping blocks. Then different techniques were applied to place the centroids into those blocks adaptively by using the entropy of the image blocks and the histogram of the image block. In the beginning, the main focus was given to the information embedded in the histogram. Initial idea was to count the number of peaks (modes) in the histogram corresponding to the image blocks and to place the number of centroids randomly in the corresponding image blocks. For the execution of the suggested method, the histograms were smoothed using a moving average filter in order to remove the unnecessary spikes and counted the number of peaks by checking the gradient. The problem arose was that the number of peaks is dependent on how much the histogram was smoothened. Since the mode count is very sensitive to the filter size, selection of a suitable filter size was critical. A smaller filter is adding an unnecessary number of modes while a larger filter is cleaning up the expected modes. Therefore, the expectation-maximization (EM) algorithm was employed to estimate the distribution instead of the smoothing process. The distribution was assumed to be a mixture of Gaussians. In order to estimate the distribution using expectation maximization, the number of Gaussians has to be provided and the final estimate is very sensitive to the initial parameters. There was no proper way to determine the number of Gaussians and those initial parameters. Additionally, it was identified that even if the density estimation problem is solved, mode counting requires proper thresholding which identifies dominant modes. Introducing a threshold to identify dominant modes was not successful.

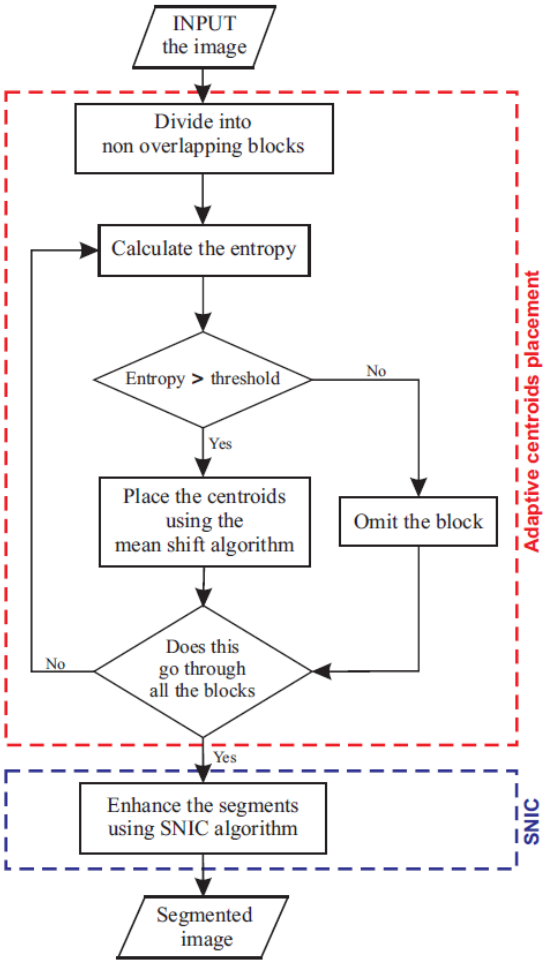


Figure 3.5: Flow Chart for Overall Procedure of Adaptive Centroid Based SNIC for Super pixel Segmentation

With the aforementioned difficulties, mode counting of the distribution of gray levels of pixels in an image block was done using kernel density estimation and mean shift. Initially, entropy measure was taken for the image block and image blocks were separated as low entropy blocks and high entropy blocks. Entropy was calculated as

$$Entrophy = -\sum_{i} P_i \, log_2 \, P_i,$$

where pi is the i^{th} element of the vector p consisting, the normalized histogram counts of the color values with bin size of five.

Then kernel density estimation was utilized to estimate the distribution and mean shift was used to find the modes. Kernel density estimation produced an accurate distribution and more importantly mean shift was successful in finding the modes accurately. But finding the corresponding locations to place centroids was difficult using gray level distribution. Accordingly, a five-dimensional joint feature space was constructed consisting of the spatial coordinates and the CIELAB color values of the pixels. In this five-dimensional joint space, the modes were found successfully by the mean shift algorithm and the centroids were placed according to the spatial coordinates of these modes.

3.5.2 Mode Estimation Using Mean-Shift

Let us see how the mean-shift procedure can be utilized to estimate the modes of a distribution. Given n data points $x_i;\ i = 1,2,..,n$ embedded in the d-dimensional Euclidean space $\mathbb{R}^d$, the multivariate kernel density estimator computed at $X \in \mathbb{R}^d$ with a d-variate kernel $K(x)$ and a symmetric positive definite $d \times d$ bandwidth matrix H is given by ,

$$\hat{p}_K, H(x) = \frac{1}{n}\sum_{i=1}^{n} K_H(x - x_i), \ldots\ldots\ldots\ldots \textcircled{1}$$

Where,

$$K_H(x) = |H|^{-1/2} K(H^{-1/2}x).$$

In the proposed method, a radially symmetric kernel called normal kernel defined as

$$K_N(x) = (2\pi)^{-1/2} exp\left(-\frac{1}{2}\|x\|^2\right),$$

is utilized for the density estimation and mean-shift. Substituting the normal kernel, the multivariate kernel density estimator in (1) can be modified as

$$\hat{p}_N, H(x) = \frac{1}{n|H|^{-1/2}}\sum_{i=1}^{n} K_N |H|^{-1/2}(x - x_i),$$

If the bandwidth matrix H is defined to be the diagonal matrix $H = diag[h_1^2, h_2^2, \ldots h_d^2]$, the parameters $h_1^2, h_2^2, \ldots h_d^2$ control the resolution of the estimated density in respective dimensions in $\mathbb{R}^d$.

Let's define $m(x)$ to be the weighted mean calculated using the normal kernel centered at the point x for weights and the bandwidth matrix H as

$$m(x) = \frac{\sum_{i=1}^{n} x_i K_N |H|^{-1/2}(x - x_i)}{\sum_{i=1}^{n} K_N |H|^{-1/2}(x - x_i)}.$$

Then, the difference $m(x) - x$, called the mean shift is proportional to the normalized density gradient estimate obtained with the normal kernel.

i.e.,

$$m(x) - x \propto \frac{\hat{\nabla} p_N, H(x)}{\hat{p}_N, H(x)} \ldots\ldots\ldots\ldots \textcircled{2}.$$

In the regions of low density, the mean shift steps are large while near the regions of relatively high density, the steps are small. Hence the mean-shift procedure is an adaptive gradient ascent method.

Since a mode is a local maximum of the density where the gradient is zero, the mean shift vector calculated at a mode of a density is zero due to (2). Hence, the sequence of successive locations of the points $\{y_j\}_{j=1,2,\ldots}$ given by

$$y_{j+1} = m(y_j)$$

converges to the corresponding mode if it is a unique stationary point when y_1 is initialized sufficiently close to that mode.

The primary goal of the proposed method is to estimate the modes rather than the density. For this, the mean shift was sufficient as it serves as a mode estimator. Kernel density estimation is not necessary for the mode estimation, nevertheless, it was worth mentioning to understand mean shift algorithm.

3.5.3 Initialization Points Selection for Mean Shift

In order to find the modes of a distribution using the mean shift, the entire data space has to be searched. Accordingly, a sufficient number of initial points should be taken to cover the entire distribution. There are several ways of selecting these initial points. One method is initializing the mean shift with samples taken from a regular grid covering the entire feature space. However, this is not an effective method since the data points are usually distributed in a small portion of the feature space when the dimensionality is high (the curse of dimensionality) and therefore it is useless to search empty areas in the feature space. Hence, it is a waste of computational cost and time. Another method is to initialize the mean shift with all the data points. Although this generates all the required modes, it is an exhaustive task where the iteration of the same calculation causes a sort of redundancy. Therefore, random sampling technique was used.

In random sampling, the number of samples which proportional to the entropy of the image block are taken randomly from the data set as the initial points for the mean shift algorithm. The reason is higher the entropy value, a better tendency to have a larger number of modes in the feature space. To encapsulate all possible modes that appear in this distribution, relatively, a large number of initialization points are required for the mean shift algorithm. Sampling reduces the bulk of data and reduces the computational cost and time. Also, in the random sampling, the tendency to get the samples close to the modes is high and therefore there is a high probability to capture all possible modes that cover the entire data space from a small number of initialization points. Accordingly, there are two advantages of this random sampling technique to run the mean shift. First, the number of mean-shift runs is small and second, since these samples are more likely to be selected from the data points in the vicinity of the modes of the distribution, the mean shift will converge to the corresponding modes efficiently. These two will save computational time considerably.

3.5.4 Enhancement of the Segment

As mentioned at the beginning of the methodology, enhancement of the segment is done according to the pixel assignment method in SNIC algorithm which was done by minimizing the distance between the pixel and initial centroid with a weighting factor which is called compactness factor for the square term of the color difference. Distance between the i^{th} pixel and the k^{th} centroid is defined as

$$d_{i,k} = \sqrt{\left\|X_j - X_k\right\|_2^2 + \frac{\left\|c_j - c_k\right\|_2^2}{m}}$$

where x and c are the two-dimensional vector consisting the spatial position (i.e., $x = [x, y]^T$) and the three dimensional vector of CIELAB color values (i.e., $c = [l, a, b]^T$) of the i^{th} pixel and the k^{th} centroid respectively. m is the compactness factor.

3.5.5 Algorithm

Initially, the image is separated into non-overlapping blocks. The number of blocks is calculated as follows due to the size variation of different images. The width and the height of the image in pixels are taken as the vector $[width\ height]^T$. Then it is divided by the block size parameter which selected as 100 and rounded off the values creating a vector that contains the height and the width of a block. Thereby an array of image blocks is created where the height and width of a block are given by the two elements of that vector. Then the image blocks are taken as one by one and centroid placement is done.

The entropy is calculated and the value of the entropy is compared with the pre-defined threshold of 5.0. If the value is less than the threshold, the block is omitted; otherwise, the mean shift algorithm is executed to find the modes. To do that, a five-dimensional data point set is formed consisting of the CIELAB color values and spatial coordinates of the pixels. Then the mean shift algorithm is employed to find the modes of the distribution made up of this data set. $H = diag[h_1^2, h_2^2, \dots h_d^2]$, is used to capture the different nature of the spatial position and CIELAB color values in the joint space. $hx = 15$ and $hc = 15$ are the values which are used to generate the results in the paper. After that, the spatial coordinates of these modes are rounded off to the nearest integer (within the possible image coordinates) and only the unique modes with respect to the spatial coordinates are retained. Then the centroids correspond to that image block are placed in the locations pointed by the spatial coordinates of these modes. Finally, SNIC algorithm is executed using these centroids to accomplish the segmentation. Algorithm of the proposed method of adaptive initial centroid placement is given in Algorithm below.

In the enhancement of the segments using SNIC method, elements $e_i = \{x_i, c_i, i, d_{i,k}\}$ are created using the initial centroids produced by the proposed method where xi and ci being the spatial coordinates and the CIELAB color values of thei^{th}centroid respectively as before whereas k is the label and $d_{i,k}$ representing the distance from k^{th} centroid. With the use of this elements a priority queue Q is initiated. When popped, Q always returns the element that has the smallest distance from a centroid. While Q is not empty the top element is popped. It is given the label of the centroid if the pixel position on the label map L pointed by the element is unlabeled. After that corresponding centroid is updated with this pixel by considering the centroid to be the average of all the pixels that have the same label. Additionally, for each of its 8 connected neighbors that have not been labeled yet, a new element is created, assigning to it the distance from the connected centroid and the label of the centroid. These new elements are pushed in to the queue.

As the algorithm executes, the priority queue is emptied due to label assignment at one end and populated with new candidates at the other. When there are no remaining unlabeled pixels to add new elements to the queue and the queue has been emptied, the algorithm terminates.

Algorithm : Adaptive Centroids Placement

Input : Image in CIELAB color space Im, Resolution $\mathrm{h} = [h_s \;\; h_r]$, Treshold for entropy t_E

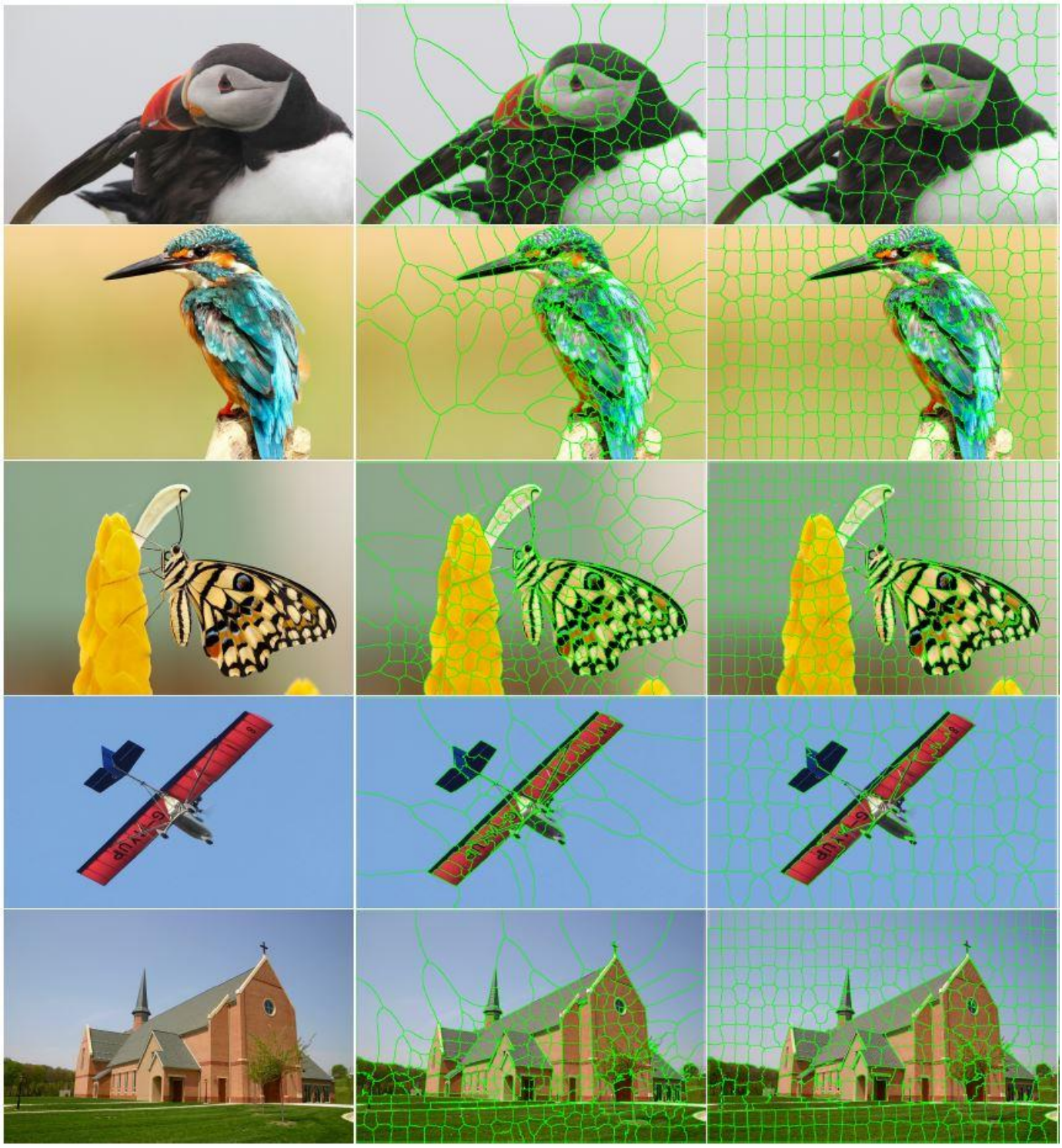
Output: Initial centroids C

```
Segment the image Im into array of non-overlapping blocks
for each image block do
    Calculate the entropy of the image block
    if entropy > t_E then
        N ← Number of pixels in the block
        for k ∈ {1, 2, ..., N} do
            D[k] ← {x_k, c_k}
        end
        S ← (2 × entropy) number of random samples from the data set D
        for all s ∈ S do
            Run the mean shift iterations until convergence for s
        end
        Create the data set M consisting converged data points m (modes)
        Round off the spatial coordinates of all m ∈ M and retain only the unique modes with respect to the spatial coordinates
        Place centroids in the locations pointed by the spatial coordinates of the modes ∀m ∈ M and append them to C
    end
end
return C
```

3.5.6 Result of The Proposed Segmentation Method

The results of the proposed segmentation method are shown in figure 3.6 with comparison of existing SNIC algorithm. There are three significant differences which can be identified in the proposed method.

- The proposed method gives more concentrated segmentation in information rich-regions and more sparse segmentation in low information regions as compared to the other two methods.
- Significant features of the images are identified accurately by the proposed method.
- No need for pre-defining the number of segments as an input parameter for the algorithm.



(a) *(b)* *(c)*

Figure 3.6: Qualitative comparison of the results; (a) original image, (b) segmented image using the proposed method, (c) segmented image using SNIC

CHAPTER 4 PCA NOISE LEVEL ESTIMATION

4.1 Basic Idea of PCA Noise Level Estimation

Blind noise level estimation is an important image processing step, since the noise level is not always known beforehand, but many image de-noising, compression and segmentation, algorithms take it as an input parameter; and their performance depends heavily on the accuracy of the noise level estimate. The most widely used noise model, which is assumed in this work as well, is signal-independent ***additive white Gaussian noise***. Noise variance estimation algorithms were being developed over the last two decades; and most of them include one or several common steps.

1. Separation of the signal from the noise.
2. Analysis of the local variance estimate distribution

To apply the PCA noise estimation technique to images, Initially, let us demonstrate the ability of image block PCA to estimate the noise variance on a simple 1D example. Consider noise free signal $\mathbf{x_k = (2 + (-1)k) = (1, 3, 1, 3, . . .)}$ and noisy signal $\mathbf{y_k = x_k + n_k}$, where $\mathbf{n_k}$ are realizations of a random variable with normal distribution $\mathbf{N(0; 0.5^2)}$. The processing of these signals using a sliding window, with the width equal to 2, results in two point sets: $\mathbf{x_k = (x_k; x_{k+1})}$ for the noise-free signal and $\mathbf{y_k = (y_k; y_{k+1}) = (x_k; x_{k+1})+(n_k ; n_{k+1})}$ for the noisy signal. By construction, points $\mathbf{x}_k$ can have only two values: (1; 3) or (3; 1). Points $\mathbf{y}_k$ are presented in following *Figure 4.1*.

Applying PCA to point set $\mathbf{y}_k$ gives new coordinate system $\mathbf{(u_1: u_2)}$ (also shown in Figure 4.1), in which $\mathbf{u_1}$ is associated with both the noise-free signal and the noise, and $\boldsymbol{u_2}$ is associated only with the noise. This allows the computation of the noise variance estimate as the variance of the points along $\mathbf{u_2}$.

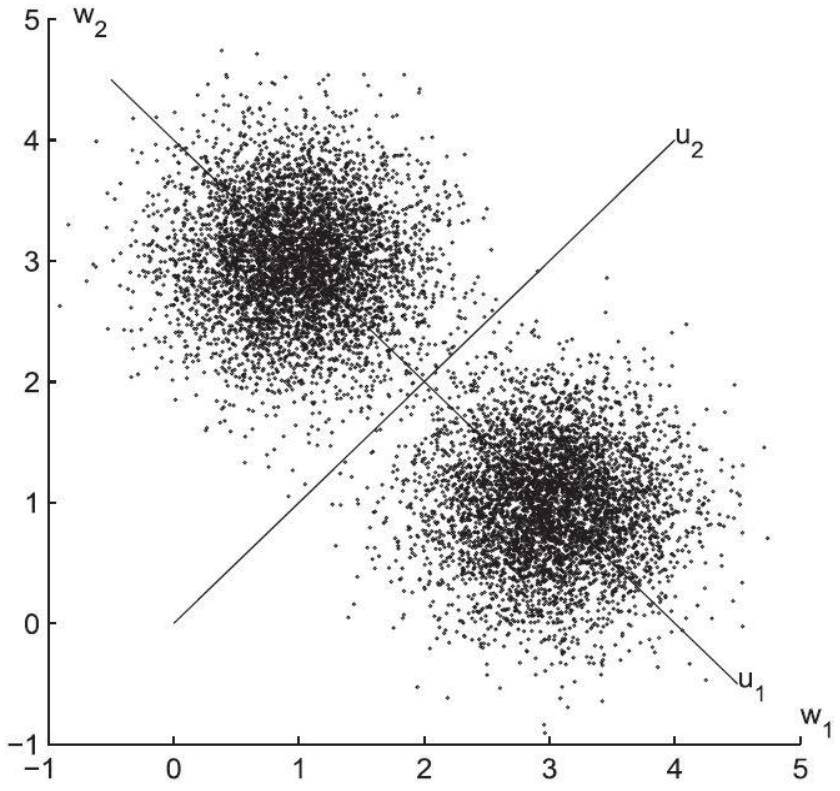


Figure 4.1: Data points y_k in original coordinate system (w_1: w_2) and the new coordinate system (u_1:u_2) computed by PCA

This example shows some properties of the proposed method:

- The method can be applied if the blocks computed from the noise-free signal can be represented by a number of dimensions smaller than the block size. In the example above, 2-dimensional points x_k with coordinates (−√2; 0) or (√2; 0) in the new coordinate system are purely situated on $\mathbf{u_1}$. Therefore, they can be represented by 1-dimensional values in the new coordinate system.
- If the blocks computed from the noise-free signal cannot be represented by a number of dimensions smaller than the block size, we cannot apply PCA directly in order to get the noise variance. In the example above, if the block set had three centroids, which did not lie on one line, then PCA would not provide a coordinate associated only with the noise.
- No assumption about signal constancy is required. Indeed, in the example above, noise-free signal $\mathbf{x_k}$ contains no constant parts.

4.2 Noise Level Estimation of Images

The proposed algorithm utilizes noise statistics disparity between the original and spliced regions to localized forged regions. Hence, each super-pixel is analyzed to approximate the noise parameters. The noise estimation should be accurate enough to approximate noise parameters of the spliced and original regions even under small noise level differences. To this end, the noise level estimation methods based on PCA are the state-of-the-art and have shown comparatively better estimation on smaller regions. Therefore, in the proposed algorithm, a PCA based noise level estimation approach similar to the one mentioned above is utilized. Let **x** be a noise free image (or an image block) of size $\mathbf{(H_1 \times H_2)}$, and $\mathbf{y = x + n}$ be the noisy image corrupted with ***Gaussian noise*** **n**. Images **x** and **y** both contain $\mathbf{N = (H_1 - M + 1)\_x\ (H_2 - M + 1)}$ overlapping patches of size **M x M**. These patches are vectorized as $\mathbf{x_i, y_i : i = 1,\ldots..,N}$ and their co variance matrices are denoted as $\sum_{xi}$ and $\sum_{yi}$ which have $\mathbf{M_2 \times M_2}$ entries.

Assumption 1 : **x** can be sparsely represented by applying PCA, (i.e., all $\mathbf{x_i}$ lie in the subspace $V_{M2\ -m}$). When the above assumption holds, it is proved that,

$$\lim_{N\to\infty} E(\left|\lambda_{y,min} - \sigma^2\right|) = 0$$

where $\boldsymbol{\lambda_{y,min}}$ is the ***minimum eigenvalue*** of $\sum_{yi}$ and σ^2 is the ***variance of the added noise***. Therefore, the noise level $\boldsymbol{\sigma}$ can be estimated as $\sqrt{\boldsymbol{\lambda_{y,min}}}$ and the estimation accuracy is proportional to N. The noise level estimation algorithm can be summarized as follows:

- **y** is decomposed into overlapping patches $\mathbf{y_i : i = 1,\ldots..,N}$. The default patch size is 5 x 5 pixels (i.e., M = 5).
- An initial estimate is computed based on the image patch variance distribution, which also serves as the upper bound of the whole estimation. Let $\mathbf{s^2(y_i)}$ be the sample variance of $\mathbf{y_i}$, and **Q(p)** be the p-quantile of $\mathbf{s^2(y_i)}$; i = 1,……,N. The initial noise level estimate is computed as $\mathbf{C_0Q(p_0)}$. Experimentally set $\mathbf{C_0 = 3{:}1}$ and $\mathbf{p_0 = 0{:}0005}$.

- A subset of the image patches $\mathbf{Y_p}$ is selected by recursively discarding the patches with the largest variance until Assumption 1 is satisfied.

$$Y_p = \{y_i |\, s^2(y_i) \leq Q_p\,, \qquad i = 1, \ldots, N\}$$

Assumption 1 is checked by the following condition:

$$\lambda_{y_p,m} - \lambda_{y_p,min} < \frac{49\,\sigma^2}{M}$$

where $\lambda_{y_p,m}$, $\lambda_{y_p,min}$ are the minimum and the mth smallest eigenvalue of $\sum_{y_p}$ respectively, and $\boldsymbol{\sigma}$ is the estimated noise variance in the previous iteration. In this context, m is set to 7 and p is decreased from 1 to 0.1 with a step of 0.05.

- $\sqrt{\boldsymbol{\lambda_{y_p,min}}}$ is regarded as the noise level estimation of current iteration. Steps 3 and 4 are iterated until convergence is achieved.

Since noise from an actual camera can be considered as a mixture of multiple noise sources, noise estimation algorithm is developed for additive Gaussian noise estimation. The estimated noise statistics vector $\boldsymbol{\sigma} = [\boldsymbol{\sigma_R}; \boldsymbol{\sigma_G}; \boldsymbol{\sigma_B}]$ of the each super-pixel is utilized in the next step to construct the graph of the forged image.

Algorithm 1 EstimateNoiseVariance

Input: Image **y**
Output: Noise variance estimate σ^2_{est}

1) $\sigma^2_{ub} \leftarrow$ GetUpperBound(**y**)
2) $\sigma^2_{est} \leftarrow \sigma^2_{ub}$
3) **for** $i = 1$ to $i_{\max}$ **do**
4) $\quad\sigma^2_{next} \leftarrow$ GetNextEstimate(**y**, σ^2_{est}, σ^2_{ub})
5) $\quad$**if** $\sigma^2_{est} = \sigma^2_{next}$ **then**
6) $\quad\quad$**return** σ^2_{est}
7) $\quad$**end if**
8) $\quad\sigma^2_{est} \leftarrow \sigma^2_{next}$
9) **end for**
10) **return** σ^2_{est}

Algorithm 2 GetNextEstimate

Input: Image **y**, previous estimate σ^2_{est}, upper bound σ^2_{ub}
Output: Next estimate σ^2_{next}

1) $p \leftarrow 1$
2) $\sigma^2_{next} \leftarrow 0$
3) **while** $p \geq p_{\min}$
4) $\quad\tilde{\lambda}_{\mathbf{Y},1}, \ldots, \tilde{\lambda}_{\mathbf{Y},M} \leftarrow$ ApplyPCA($B(p)$)
5) $\quad\sigma^2_{next} \leftarrow \tilde{\lambda}_{\mathbf{Y},M}$
6) $\quad$**if** $\tilde{\lambda}_{\mathbf{Y},M-m+1} - \tilde{\lambda}_{\mathbf{Y},M} < T\sigma^2_{est}/\sqrt{|B(p)|}$ **and** $\sigma^2_{next} \leq \sigma^2_{ub}$
7) $\quad\quad$**return** σ^2_{next}
8) $\quad$**end if**
9) $\quad p \leftarrow p - \Delta p$
10) **end while**
11) **return** σ^2_{next}

CHAPTER 5 CLASSIFICATION USING SPECTRAL CLUSTERING

5.1 Representing the Super-Pixels on a Graph

By embedding the structure of a signal onto a graph, GSP extends classical signal processing to analyze and process data samples residing on irregular domains. As graphs can accommodate relations between data samples effectively with higher degrees of connectivity, GSP has become a scalable and versatile signal processing approach. Any vector $\boldsymbol{\sigma} = [\boldsymbol{\sigma_1}, \ldots., \boldsymbol{\sigma_n}]^{\boldsymbol{T}}$ whose elements rest on nodes of a graph G can be referred to as a graph signal on **G**. A graph $\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{\varepsilon}, \boldsymbol{W})$ contains a set $\boldsymbol{V}$ of **N** nodes and a set **ε** of **M** edges. In this approach, a fully connected undirected graph $\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{\varepsilon}, \boldsymbol{W})$ is formed, where ***V is the set of all superpixels*** (Vertices/Nodes of the graph) and ***ε is the set of edges***. The column $\boldsymbol{\sigma_i}$ of the matrix $\boldsymbol{\sigma}$ resides on the corresponding node i. Then, each row of $\boldsymbol{\sigma}$ can be viewed as a signal on the same graph, yielding three independent graph signals. Each existing edge $(\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{\varepsilon}$ is undirected and contains an edge weight $\boldsymbol{W_{i,j}}$, which is typically positive; a large positive $\boldsymbol{W_{i,j}}$ would mean that samples at nodes i and j are expected to be correlated. The weight $\boldsymbol{W_{i,j}}$ of an edge connecting node (super-pixel) i and j is computed using,

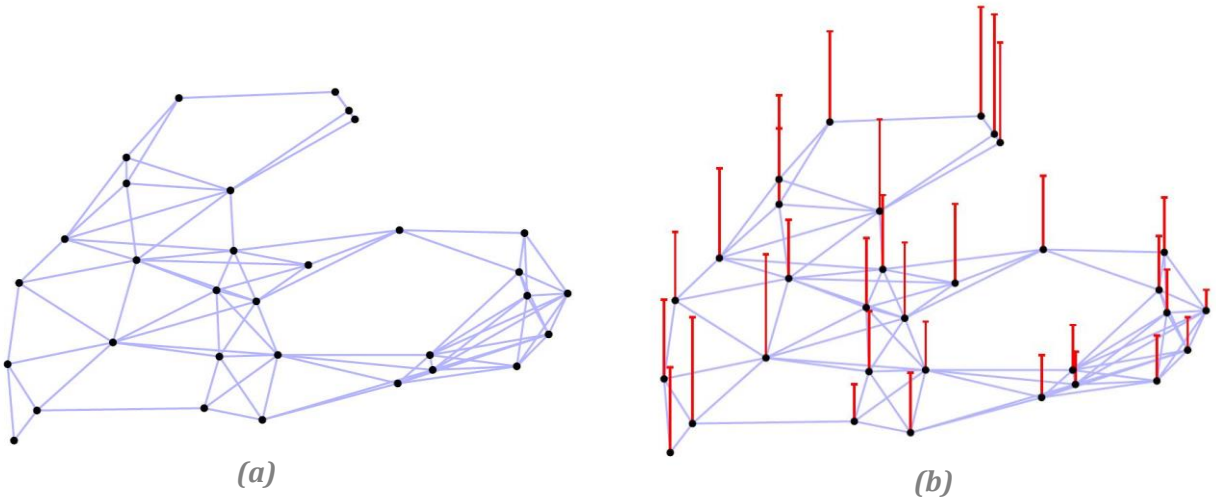
$$W_{i,j} = e^{\left(-\frac{\left\|l_i - l_j\right\|^2}{{\sigma_i}^2}\right)} \times e^{\left(-\frac{\left\|\sigma_i - \sigma_j\right\|^2}{{\sigma_s}^2}\right)}$$

where l_i is the center location of the super-pixel i on the 2-D image, σ_i is the noise parameter vector of the super-pixel i, and ${\sigma_l}^2$ l and ${\sigma_s}^2$ are two scaling parameters. Hence, $0 \leq W_{i,j} \leq 1$. Larger geometric and/or noise parameter distances between super-pixels i and j would mean a smaller weight $\boldsymbol{W_{i,j}}$. The **Degree matrix** and the **Laplacian matrix** associated with a graph **G** are defined as follows. The degree matrix $\boldsymbol{D} \in \mathcal{R}^n \times \mathcal{R}^n$ is a diagonal matrix with its i^{th}diagonal element given by $D_{i,j} = \sum_{j=1}^{N} W_{i,j}$. The graph Laplacian matrix $\boldsymbol{L}$ is defined as

$$L = D - W \in \mathcal{R}^n \times \mathcal{R}^n$$

whose elements are directly given by $L_{i,j} = -W_{i,j}$ for $i \neq j$ and $L_{i,j} = \sum_{j=1}^{N} W_{i,j}$. This makes L real and symmetric with strictly positive diagonal elements. Later, the eigenvalue decomposition of L will be utilized to perform graph spectral clustering.

Figure 5.1: Segment which is used as a node on the graph



(a) *(b)*

Figure 5.2: (a). Graph representation of super-pixels of the image (b). Graph signal constructed on top of the graph

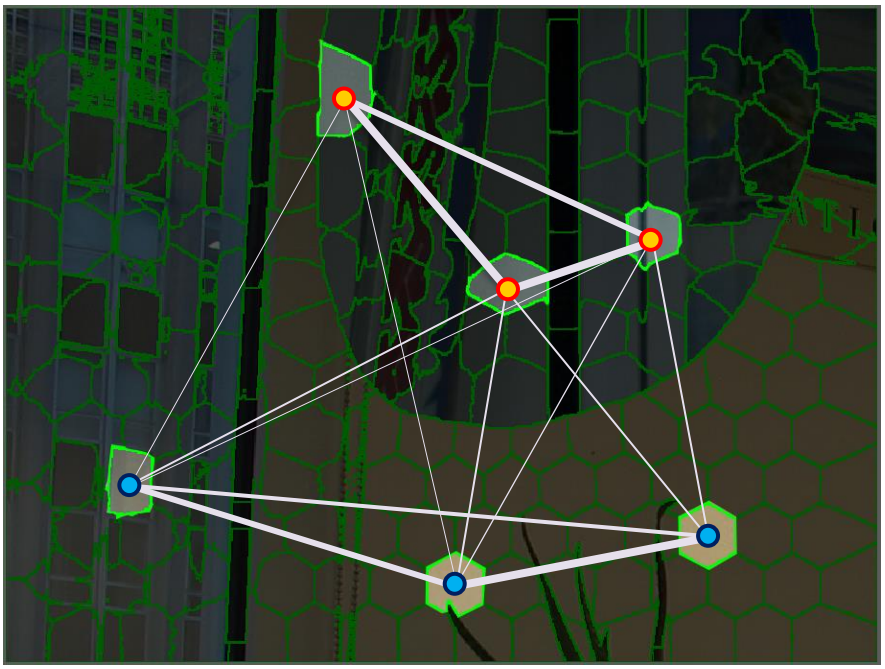


Figure 5.3: Graph Defined on Image

5.2 Graph Based Spectral Clustering

After constructing the undirected graph $\boldsymbol{G}$, spectral representation of each superpixel is obtained from the graph **Laplacian matrix** ($\boldsymbol{L}$). Since $\boldsymbol{L}$ is symmetric, the $\boldsymbol{L}$ can be eigendecomposed as,

$$L = U\Lambda U^T$$

where $\boldsymbol{\Lambda}$ is a ***diagonal matrix*** containing real eigenvalues $\boldsymbol{\lambda_k}$ along the diagonal, and $\boldsymbol{U}$ is an eigenmatrix composed of orthogonal eigenvectors u_i as columns. Since all edge weights $\boldsymbol{W_{i,j}}$ are restricted to be positive, then graph Laplacian $\boldsymbol{L}$ can be proved to be positive semi-definite (PSD). Non-negative eigenvalues $\boldsymbol{\lambda_k}$ can be interpreted as ***graph frequencies***, and eigenvectors $\boldsymbol{U}$ interpreted as corresponding ***graph frequency components***. Together they define the graph spectrum for graph $\boldsymbol{G}$. Then, eigenvectors $u_1, u_2, \ldots, u_m$ corresponding to the m largest eigenvalues of $\boldsymbol{L}$ are stacked as columns to form the matrix $\boldsymbol{C}$,

$$C = [c_1, c_2, \ldots, c_m] \in \mathcal{R}^{n \times m}$$

Each row of the matrix $\boldsymbol{C}$ is normalized to have a unit length normalized matrix $\boldsymbol{Y}$ given by,

$$Y_{i,j} = \frac{c_{i,j}}{\|c_{i,j}\|}$$

where $\boldsymbol{Y_i}$ is the i^{th} row of the matrix $\boldsymbol{Y}$ and $\| \ . \ \|$ is the Euclidean norm. Matrix $\boldsymbol{Y}$ is a spectral representation of the graph $\boldsymbol{G}$, where i^{th} super-pixel is represented as a point in $\mathcal{R}^m$ by the i^{th} row of $\boldsymbol{Y}$. The rows of $\boldsymbol{Y}$ are then clustered using ***K-means clustering***. Each super-pixel i is assigned to cluster m, only if the i^{th} row of the matrix Y_t was assigned to cluster m.

Here, the ***normalized graph Laplacian*** $\boldsymbol{L}$ is eigendecomposed to create a spectral decomposition basis. According to the GSP interpretation, the Laplacian basis yields the basis of high frequency values (Large eigen values) which is utilized to reform a subspace with high graph total variation (scatter) that encourages the formation of m clusters. The proposed algorithm is evaluated by using the ***Columbia dataset*** where the spliced images contain only one type of splicing regions with different noise statistics compared to the noise of the original image regions. However, the algorithm is capable of localizing multiple splicing forgeries by simply increasing the number of clusters.

CHAPTER 6 SPLICED BOUNDARY CLASSIFICATION USING NEURAL NETWORKS AND DEEP LEARNING

6.1 Introduction

Neural networks, a beautiful biologically-inspired programming paradigm which enables a computer to learn from observational data. Deep learning is a powerful set of techniques for learning in neural networks. Neural networks and deep learning currently provide the best solutions to many problems in image recognition, speech recognition, and natural language processing. Especially convolutional neural networks are very familiar with the area of image classification problems. Deep learning is a class of machine learning algorithms that uses multiple layers to progressively extract higher level features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.

Convolutional is a class of deep neural networks, most commonly applied to analyzing visual imagery. CNNs are regularized versions of multilayer perceptions. Multilayer perceptron usually means fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to over fitting data. Typical ways of regularization include adding some form of magnitude measurement of weights to the loss function. CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme. General architecture of a CNN is shown in following figure.

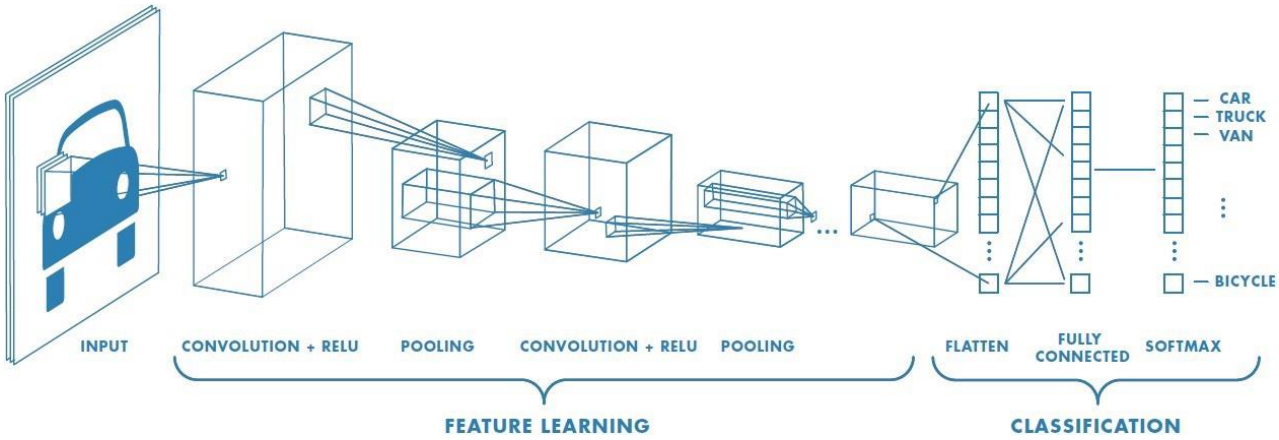


Figure 6.1 General Architecture of Convolutional Neural Network

6.2 Motivation for Spliced Boundary Detection Using CNNs

Spliced region identification in forgery image, boundary of the splice region is critical part. Because in spliced image both original image and splice image are two different images. Noise, contrast, elimination can be different in both of them. But at the boundary of two images in spliced image, there is a property change and due to that variation of above mentioned properties are high. Therefore, convolution neural network was suggested to experiment for identifying the spliced boundary.

6.3 Proposed Method Spliced Boundary Detection Using CNNs

6.3.1 Developing Training Data Set

Using the "Columbia Uncompressed Image Splicing Detection" dataset, classified training dataset was prepared for train the neural network. Seven images were taken from the data set and they were segmented into non overlapping boxes. Size of the image block was taken as 50x50 pixels. Then comparing with the edge masks of the dataset, those small blocks were separated into blocks with spliced boundary and without spliced boundary. There were 268 images with spliced boundary included and 887 images without spliced boundary.

6.3.1 Convolutional Neural Network Architecture

Convolution neural network was initiated and it was updated by changing the number of layers, number of filters, strides, etc. data set was splitted as 70% training images and 30% test images. It was done in order to maximize the test prediction accuracy. Following network model which is in table 6.1 was developed for classification.

Table 6.1 Layers and Parameters of the Developed Network Architecture

Layer	Layer specifications
Input layer	Size:40x40x3
Convolution layer 1	Number of filters :32 Filter size:16x16 Strides: 1x1 Padding: 3
ReLU layer	ReLU activation function is used in this layer
Max pooling layer	Pool size:2x2 Strides: 2
Convolution layer 2	Number of filters :128 Filter size:5x5 Strides: 1x1 Padding: 2
ReLU layer	ReLU activation function is used in this layer
Max pooling layer	Pool size:2x2 Strides: 2
Convolution layer 3	Number of filters :192 Filter size:3x3

	Strides: 1x1 Padding: 1
ReLU layer	ReLU activation function is used in this layer
Max pooling layer	Pool size:2x2 Strides: 2
Fully connected layer	Output Size: 256
Fully connected layer	Output Size: 2
SoftMax layer	Applies a SoftMax function to the input.
Classification layer	Computes the cross-entropy loss for multi-class classification problems with mutually exclusive classes.

Training parameters:

- Max epochs: 30
- Initial Learning rate: 0.001
- Learn rate drop period: 6
- Learn rate drop factor 0.2
- Minimum batch size:128

6.3.2 Results of the Classifier

Table 6.2 Results of the Developed Network Architecture

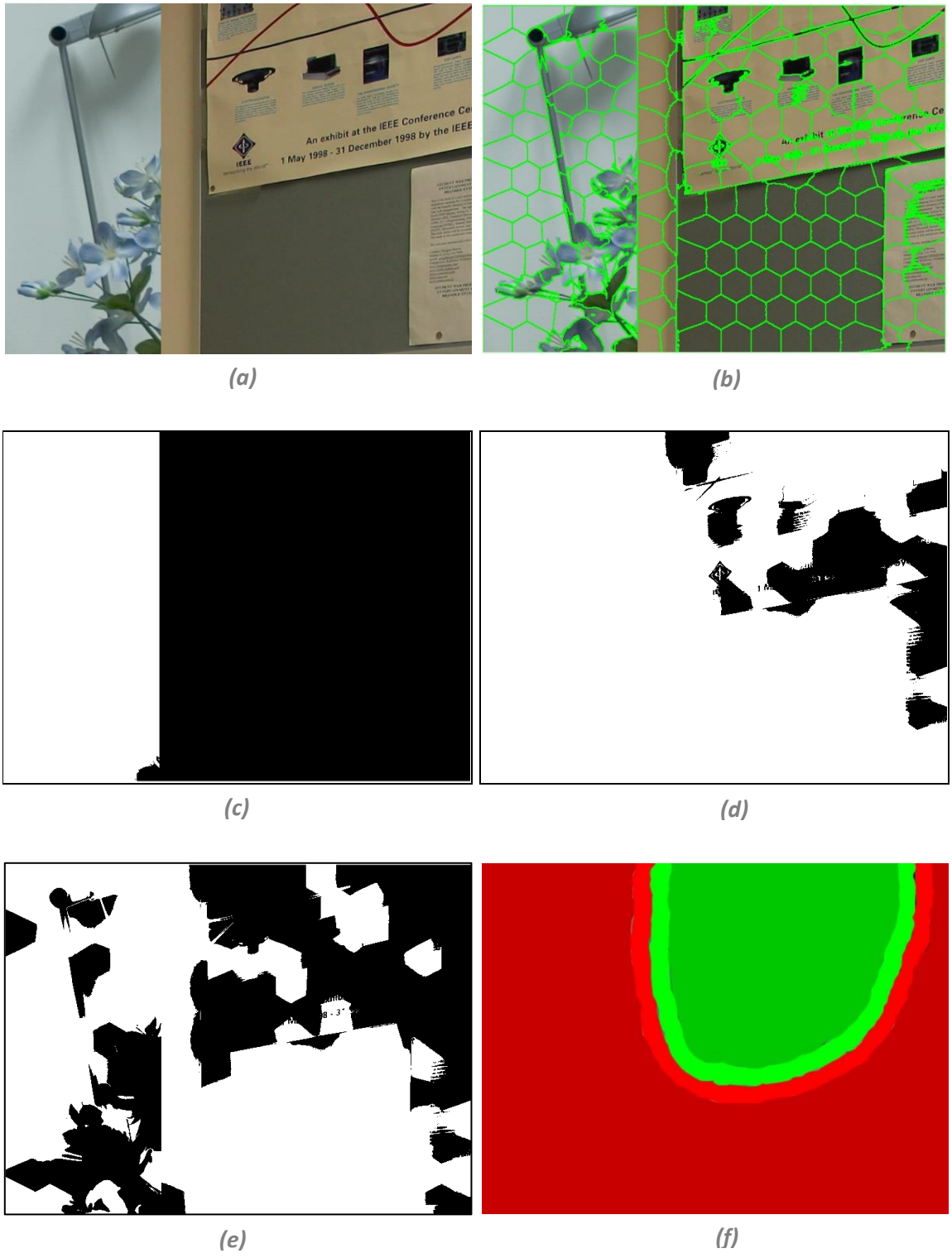
Training attempt	Training time	Training accuracy	Test accuracy
01	1 min & 55 sec	76.76%	76.87%
02	1 min & 47 sec	76.76%	76.87%
03	1 min & 44 sec	76.76%	76.87%

6.3.3 Comment on the Results

According to the prediction accuracy, method looks like successful in some distance. But there was a huge problem. All the outputs are given as non-boundary region for every input. Since number of training data of the non-boundary regions are more than three time larger. Since that even if all the results are given as non-boundary regions, even though prediction percentages are looks good. That is the reason for equal training and testing accuracy in all the three cases. This problem is aroused due to highly imbalance number of data in data set. So changed the number of data images as balanced the two categories of data images. But results look likes same. Another reason which is affect to result is color differences are dominated and other parameters like noise, elimination level, contrast will not be able to capture by the convolutional layer. Therefore, CNN base classification is not going to work for identifying the spliced boundary of a forgery image.

CHAPTER 7 RESULTS

Forgery detection results can be interpreted as follows with different noise estimation methods. Results are taken for images in ***"Columbia Uncompressed Image Splicing Detection"*** data set.



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 7.1: (a) Original Spliced Image 1 (b)Segment Image (c) Localized Spliced Region using PCA based noise estimation (d) Localized Spliced Region using AN Efficient Statistical method (e) Localized Spliced Region using Natural Scene Statistical Method (f) Ground Truth

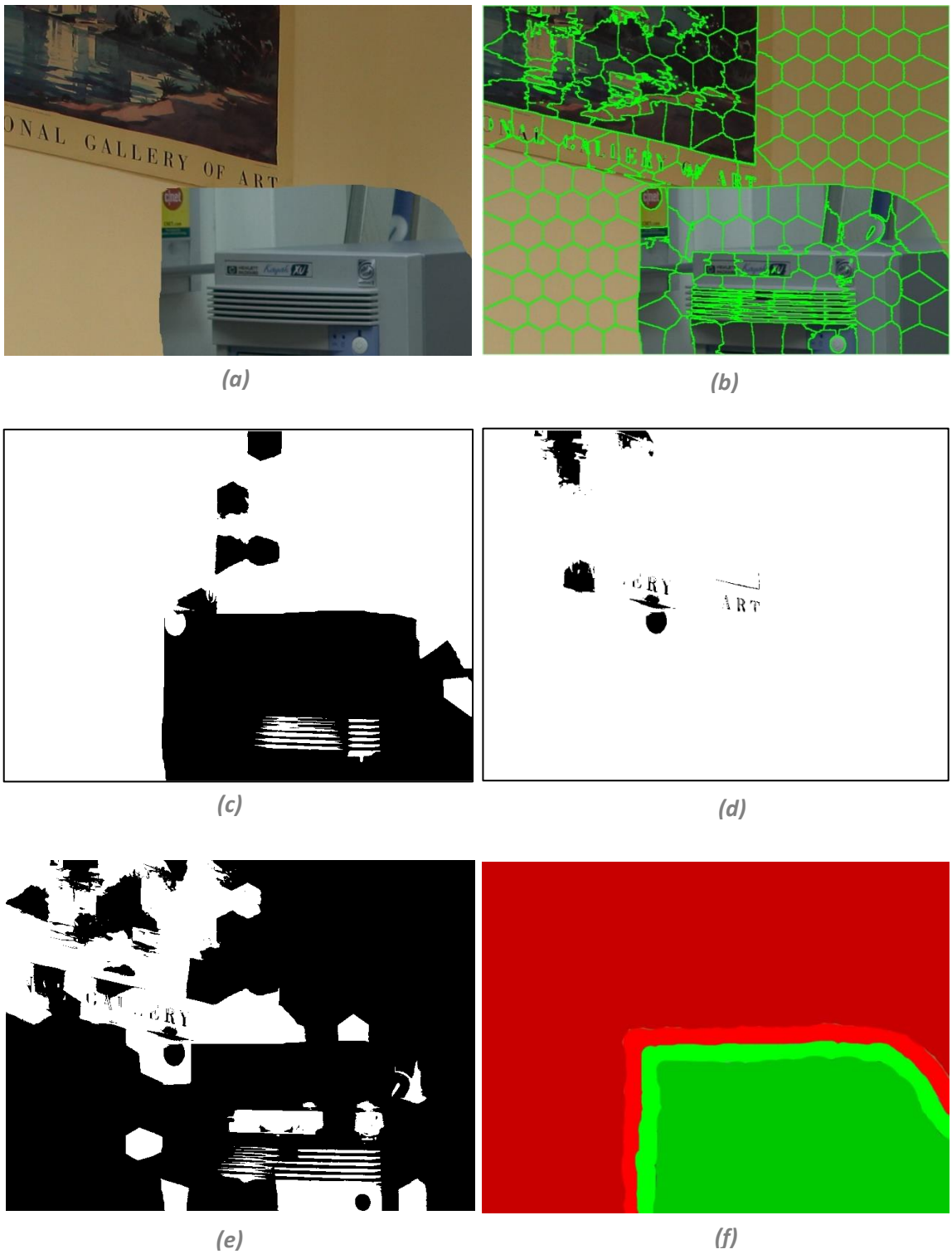


Figure 7.2: (a) Original Spliced Image 1 (b)Segment Image (c) Localized Spliced Region using PCA based noise estimation (d) Localized Spliced Region using AN Efficient Statistical method (e) Localized Spliced Region using Natural Scene Statistical Method (f) Ground Truth

According to the above results in figure 7.1 and figure 7.2, “Image Noise Level Estimation by Principal Component Analysis” is best considered to other noise model. When considering maximum eigen gap between second and third eigenvalues, above results were obtained. Table 7.1 is shown the details about the eigenvalues, execution time of the results as a comparison.

Table 7.1: Eigen value details of normalized graph Laplacian, sigma value and executions times for each result in figure 7.1 and figure 7.2

Image	Noise Estimation Method	Second Eigen Value	Third Eigen Value	Eigen Gap	Total Execution time
Image 1	PCA based noise estimation	0.0169	0.5965	0.5796	5 min & 31 sec
	AN Efficient Statistical method	0.0101	0.0133	0.0032	5 min & 25 sec
	Natural Scene Statistical Method	0.0086	0.0189	0.0103	5 min & 51 sec
Image 2	PCA based noise estimation	0.0150	0.3293	0.3143	5 min & 52 sec
	AN Efficient Statistical method	0.0145	0.0369	0.0224	5 min & 45 sec
	Natural Scene Statistical Method	0.0052	0.0289	0.0237	5 min & 42 sec

According to the above result we can observe that eigen gap between second eigenvalue and third eigenvalue is large in PCA based noise estimation with compared to other two methods. That can be also concluded that there is not good noise separation in two images using those two methods. Therefore, PCA based noise model was the better solution than others.

CONCLUSION

Image tempering and image tempering identification is broad topic in modern world because, images are very important and straightforward method for transferring information. At the same time, if it can be easily tempered, it is not a good source. Therefore, image tempering identification is very significant to make information more accurate and value for receivers. Therefore, people were suggested lot of method to fake image identification in the modern world and it is a rapidly improving section at the present. Some of the tempering method and already available methods were discussed at the beginning of the report. In our project, we focused to spliced image identification and localizing the spliced region. There are several properties of the image were used by different people for their suggested methods. In our project, noise level was used as the property. Basically, our approach had mainly three sections. There are image segmentation, noise level estimation and classification the noise data.

Image segmentation was focused as the first step. Segmentation requirements are different from application to application. In our case, few important facts were identified and several methods were studied to find out a proper solution for us. Boundary preserving condition, simply connected condition and selecting number of segments without human intuition were considered as main facts for our application. K-means clustering based segmentation, Simple Linear Iterative Clustering (SLIC) and Simple Non-Iterative Clustering (SNIC) were studied, implemented and compared for recognized a better one. K-means clustering based segmentation was not full filled our requirements and therefore it was not accepted for our project. SLIC and SNIC are super pixel-based segmentation methods and SNIC method is improvement of the SLIC method. Both of them were implemented and SNIC method was identified as the most appropriate solution for our objective. However, two improvements were suggested for the SNIC algorithm as well. Sub segmenting the super pixel and adaptive centroid initialization were the suggested improvements. Coupled of methods were suggested for each improvement and both of them were based on the RGB color value histogram. The common task was identifying the mode count of the histogram. Simple smoothen based technique was used to overcome the task. But smoothen filter selection and thresholding point selection were problems for that simple approach. Therefore, Gaussian Mixture Models were studied and fitting process was done using it. But no better solution is found out for mode counting yet.

With the SLIC and SNIC segmentation methods, noise estimation was done using PCA noise level estimation method. Noise level was estimated for RGB color components for each super pixel as three-dimensional vector. Then graph spectral clustering was used to classify the spliced image segments and original image segments. But, the results for SNIC segmentation were not as acceptable level. However, SLIC segmentation was given better results for the problem. Even though SNIC was our first choice, it was not appropriate noise level estimation because, in SNIC segmentation simply connected condition is considered at the beginning. Therefore, some super pixels have other color values than SLIC method. However, sub segmentation was solution for that problem as well. Final results were implied that approach is better for some distance. In addition, proposed segmentation improvements are improved the method.

REFERENCES

Kumar, Satender, Brij Mohan Singh, and Kuldeep Yadav. "A REVIEW ON IMAGE SEGMENTATION TECHNIQUES" 4, no. 10 (2017): 7.

Sivakumar, P, and Dr S Meenakshi. "A REVIEW ON IMAGE SEGMENTATION TECHNIQUES" 5, no. 3 (2016): 7.

Amza, Catalin. "A REVIEW ON NEURAL NETWORK-BASED IMAGE SEGMENTATION TECHNIQUES," n.d., 23.

Kaur, Manjot, and Pratibha Goyal. "A Review on Region Based Segmentation" 4, no. 4 (2013): 4.

Padmapriya, B., T. Kesavamurthi, and H. Wassim Ferose. "Edge Based Image Segmentation Technique for Detection and Estimation of the Bladder Wall Thickness." *Procedia Engineering* 30 (2012): 828–35. https://doi.org/10.1016/j.proeng.2012.01.934.

Liu, Ming-Yu, Oncel Tuzel, Srikumar Ramalingam, and Rama Chellappa. "Entropy Rate Superpixel Segmentation." In *CVPR 2011*, 2097–2104. Colorado Springs, CO, USA: IEEE, 2011. https://doi.org/10.1109/CVPR.2011.5995323.

Gurusamy, Vairaprakash, Subbu Kannan, and G.Nalini. "REVIEW ON IMAGE SEGMENTATION TECHNIQUES," 2014.

Achanta, R., A. Shaji, K. Smith, A. Lucchi, P. Fua, and Sabine Süsstrunk. "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, no. 11 (November 2012): 2274–82. https://doi.org/10.1109/TPAMI.2012.120.

Achanta, Radhakrishna, and Sabine Susstrunk. "Superpixels and Polygons Using Simple Non-Iterative Clustering." In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4895–4904. Honolulu, HI: IEEE, 2017. https://doi.org/10.1109/CVPR.2017.520.

Kaur, Dilpreet, and Yadwinder Kaur. "Various Image Segmentation Techniques: A Review," 2014, 6.

Boyat, Ajay Kumar, and Brijendra Kumar Joshi. "A Review Paper : Noise Models in Digital Image Processing." *Signal & Image Processing : An International Journal* 6, no. 2 (April 30, 2015): 63–75. https://doi.org/10.5121/sipij.2015.6206.

Gupta, Praful, Christos G. Bampis, Yize Jin, and Alan C. Bovik. "Natural Scene Statistics for Noise Estimation." In *2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, 85–88. Las Vegas, NV: IEEE, 2018. https://doi.org/10.1109/SSIAI.2018.8470313.

Chen, Guangyong, Fengyuan Zhu, and Pheng Ann Heng. "An Efficient Statistical Method for Image Noise Level Estimation." In *2015 IEEE International Conference on Computer Vision (ICCV)*, 477–85. Santiago, Chile: IEEE, 2015. https://doi.org/10.1109/ICCV.2015.62.

Christlein, Vincent, Christian Riess, Johannes Jordan, Corinna Riess, and Elli Angelopoulou. "An Evaluation of Popular Copy-Move Forgery Detection Approaches." *IEEE Transactions on Information Forensics and Security* 7, no. 6 (December 2012): 1841–54. https://doi.org/10.1109/TIFS.2012.2218597.

Pyatykh, S., J. Hesser, and Lei Zheng. “Image Noise Level Estimation by Principal Component Analysis.” *IEEE Transactions on Image Processing* 22, no. 2 (February 2013): 687–99. https://doi.org/10.1109/TIP.2012.2221728.

Cheung, Gene, Enrico Magli, Yuichi Tanaka, and Michael K. Ng. “Graph Spectral Image Processing.” *Proceedings of the IEEE* 106, no. 5 (May 2018): 907–30. https://doi.org/10.1109/JPROC.2018.2799702.

Weerakoon, Kasun. “Graph Based Image Splicing Localization,” n.d., 10.

Soni, Badal, Pradip K. Das, and Dalton Meitei Thounaojam. “CMFD: A Detailed Review of Block Based and Key Feature Based Techniques in Image Copy-Move Forgery Detection.” *IET Image Processing* 12, no. 2 (February 1, 2018): 167–78. https://doi.org/10.1049/iet-ipr.2017.0441.

Zheng, Lilei, Ying Zhang, and Vrizlynn L.L. Thing. “A Survey on Image Tampering and Its Detection in Real-World Photos.” *Journal of Visual Communication and Image Representation* 58 (January 2019): 380–99. https://doi.org/10.1016/j.jvcir.2018.12.022.

Zeng, Hui, Yifeng Zhan, Xiangui Kang, and Xiaodan Lin. “Image Splicing Localization Using PCA-Based Noise Level Estimation.” *Multimedia Tools and Applications* 76, no. 4 (February 2017): 4783–99. https://doi.org/10.1007/s11042-016-3712-8.