K-MEANS, MEAN SHIFT, AND SLIC CLUSTERING ALGORITHMS: A COMPARISON OF PERFORMANCE IN COLOR-BASED SKIN SEGMENTATION

by

Abdulkarim A. Alorf

B.S. in Electrical Engineering, Qassim University, SA, 2012

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Master of Science

University of Pittsburgh

2017

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This thesis was presented

by

Abdulkarim A. Alorf

It was defended on

June 1, 2017

and approved by
Yiran Chen, Ph.D., Associate Professor,
Department of Electrical and Computer Engineering

Zhihong Mao, Ph.D., Associate Professor,
Department of Electrical and Computer Engineering

Brandon Grainger, Ph.D., Assistant Professor,
Department of Electrical and Computer Engineering

Thesis Advisor: Yiran Chen, Ph.D., Associate Professor,
Department of Electrical and Computer Engineering

K-MEANS, MEAN SHIFT, AND SLIC CLUSTERING ALGORITHMS: A COMPARISON OF PERFORMANCE IN COLOR-BASED SKIN SEGMENTATION

Abdulkarim A. Alorf, M.S.

University of Pittsburgh, 2017

Commonly used in computer vision, segmentation is grouping pixels into meaningful or perceptually similar regions. In this work, we are going to evaluate the performance of three popular data-clustering algorithms, the K-means, mean shift and SLIC algorithms, in the segmentation of human skin based on color.

The K-means algorithm Iteratively aims to group data samples into K clusters, where each sample belongs to the cluster with the nearest mean. The mean shift algorithm is a non-parametric algorithm that clusters data iteratively by finding the densest regions (clusters) in a feature space. An enhanced version of the classic K-means algorithm, the SLIC limits the search region to a small area around the cluster reducing the algorithm complexity to be only dependent on the number of pixels in the image. It also provides control over the compactness of the clusters.

Color-based skin segmentation algorithms depend on both a color space at which segmentation is performed and a classification method used to determine whether a pixel is skin or non-skin. We have implemented the K-means, mean shift and SLIC algorithms in the RGB color space to detect human skin. Our method begins by clustering images using these algorithms and then segmenting the clustered regions occupied by skin. Pixels in the clusters are classified as skin or non-skin using the Kovac model.

We have evaluated the algorithms' performance on the SFA database (controlled environment) and on another database created for testing on an uncontrolled environment. The

performance has been evaluated using time complexity, F1 score, recall, and precision. We have found that on average the mean shift algorithm triumphs over the three algorithms in terms of performance while the SLIC algorithms holds an advantage being the fastest.The K-means algorithm has a good performance when the number of clusters K is between 10 and 15, whereas the mean shift algorithm has good performance when the bandwidth h is between 0.03 and 0.06. The SLIC algorithm maxes out its performance at around k = 100 and the number of clusters can be increased to K = 300 without remarkably increasing the complexity.

TABLE OF CONTENTS

PREFACE xi

1.0 INTRODUCTION 1

1.1 OVERVIEW 1

1.2 Organization 5

2.0 THEORETICAL BACKGROUND 6

2.1 K-MEANS ALGORITHM FOR DATA CLUSTERING 6

2.1.1 Overview 6

2.1.2 Algorithm 6

2.1.3 Strengths and Weaknesses 10

2.2 MEAN SHIFT ALGORITHM FOR DATA CLUSTERING 12

2.2.1 Overview 12

2.2.2 Algorithm 13

2.2.3 Strengths and Weaknesses 20

2.3 SLIC ALGORITHM FOR DATA CLUSTERING 21

2.3.1 Overview 21

2.3.2 Algorithm 23

2.3.3 Strengths and Weaknesses 25

3.0 EXPERIMENT AND RESULTS 26

3.1 EXPERIMENT AND RESULTS 26

3.1.1 Experiment Details 26

3.1.2 Qualitative Results 29

3.1.3 Quantitative Results 36

4.0 CONCLUSION AND FUTURE WORK 41
BIBLIOGRAPHY 43

LIST OF TABLES

3.1 K-means algorithm: average computation time for segmenting one image using different numbers of clusters K. 38
3.2 Mean shift algorithm: average computation time for segmenting one image using different bandwidths h. 39
3.3 SLIC algorithm: average computation time for segmenting one image using different numbers of clusters K. 40

LIST OF FIGURES

2.1 Clustering data in 2D feature space using the K-means algorithm. K here is equal to 3. 7

2.2 Shifting the center of the region of interest (window) to the center of mass using the mean shift algorithm in 2D feature space. 13

2.3 Overview on how the mean shift algorithm detects the densest region in 2D feature space. 14

2.4 Densities of the data points in this 2D feature space imply a nonparametric pdf [30]. 15

2.5 Kernel functions: (left) Epanechnikov kernel, (middle) uniform kernel, and (right) normal kernel [30]. 16

2.6 The gradient of the kernel density estimate implies the mean shift algorithm. 18

2.7 Two attraction basins. 19

2.8 search region (shaded area): (left) k-means algorithm and (right) SLIC algorithm. 22

3.1 Evaluation space. 28

3.2 K-means algorithm: clustered and segmented skin with different number of clusters K. Black regions indicate pixels that are classified as non-skin. . . . 31

3.3 Mean shift algorithm: clustered and segmented skin with different bandwidths h. Black regions indicate pixels that are classified as non-skin. 32

3.4 SLIC algorithm: clustered and segmented skin with different number of superpixels K. Black regions indicate pixels that are classified as non-skin. . . . 33

3.5 Continued on next page 34

3.6 Skin segmentation in an uncontrolled environment: clustered and segmented skin with different numbers of clusters K using both the K-means and SLIC algorithms, and different bandwidths h using the mean shift algorithm. Black regions indicate pixels that are classified as non-skin. 35
3.7 Comparing time complexity of detection process between clustered and unclustered images. 36
3.8 Comparing time complexity of the k-means, mean shift, and SLIC algorithms with respect to image resolutions. 37
3.9 Performance analysis of the K-means clustering algorithm for different numbers of clusters K under two different environments. 38
3.10 Performance analysis of the mean shift clustering algorithm for different bandwidths h under two different environments. 39
3.11 Performance analysis of the SLIC clustering algorithm for different numbers of superpixels K under two different environments. 40

PREFACE

I would like to thank the people who significantly assist me in every aspect of life making this thesis possible.

First, I would like to express my deepest appreciation to my advisor, Dr.Yiran Chen. Not only was he inspiring, but he was always available whenever I need help. His inspiration surly will have a lasting impact on my future life. I greatly appreciate sharing your knowledge and experience with me. I would also extend my sincere gratitude to Dr.Zhihong Mao and Dr.Brandon Grainger for taking the time to review my thesis and provide me with valuable feedback.

I was very fortunate to join the EI-Lab where I had the opportunity to meet very cooperative and helpful people. I would like to take the chance and thank all the lab members, especially Enes Eken and Amr Mahmoud, for being so helpful and supportive.

I would like to thank Qassim university in Saudi Arabia for the financial support. Without a doubt, the scholarship has played a key role in pursuing my graduate studies. special thanks Dr.Fahad Abdulrahman Al-Mufadi, the dean of Engineering College at Qassim University, and Dr.Abdulrahman F. Almarshoud, the head of the department of Electrical Engineering, for their continuous encouragement and support.

Finally, I must express my very profound gratitude to my family, to my father, to my mother who passed away before observing this accomplishment, and to my brother Abdulaziz for providing me with unstinting support and continuous motivation during the whole study period including the researching and writing my master thesis. The last word of acknowledgment goes to my dear wife, Reem, who has been always standing by me, helped me get through this period in the most positive way and made these years the best of my life. This accomplishment would not have been possible without them.

1.0 INTRODUCTION

1.1 OVERVIEW

Image segmentation is considered to be the most important vision task, but what makes a segment meaningful? To answer this question, we may need to look at Gestalt psychology, or gestaltism [1]. Gestalt psychology tries to define the principles (laws) behind the capability to obtain and maintain meaningful perceptions in a clearly chaotic world. In essence, Gestalt principles try to describe how we see things the way that we do. For example, when we are watching a soccer game on TV, we are actually seeing a bunch of images stacked one after another. We take these images and put them in our heads, telling ourselves that we are watching a fluid, realistic soccer game! Although some of the Gestalt principles, such as symmetry, are difficult to implement in practice, they are considered the basis for many ideas in segmentation. We can define segmentation as grouping pixels into meaningful or perceptually similar regions.

We need segmentation for multiple purposes. We use it to increase efficiency, such as in superpixel segmentation ([2],[3]), where we group similar-looking pixels together for efficiency of further processing. We sometimes need segmentation to extract some features [4] or to extract object proposals ([5],[6],[7]). Usually, we need to use segmentation as a result [8], such as segmenting the background of a face image.

There are many segmentation methods such as thresholding methods (the simplest), clustering methods, histogram-based methods, edge detection methods, and many other methods. Segmentation algorithms are classified based on these methods. The selection of a segmentation algorithm is based on the task we need it for. For example,

if we need to detect human eyes, then the wise choice would be a segmentation algorithm for edge detection [9] because human eyes have elliptical or spherical shapes.

In this work, we are interested in clustering algorithms. The purpose of clustering is to group data such that similar objects are in one cluster and objects in different clusters are dissimilar. Clustering is an unsupervised learning technique, i.e., clustering algorithms does not require training. There are two major types of clustering, which are hierarchical clustering and iterative clustering. In hierarchical clustering, we seek to create a hierarchy of related objects or clusters. One example hierarchical clustering is Google News, where articles with similar topics (such as politics and sports) are grouped together. In this work, we have used three iterative clustering algorithms widely used for the purpose of segmentation in computer vision. These algorithms are the K-means algorithm ([10, 11]), the mean shift algorithm ([12],[13]) and the SLIC algorithm[14]. They are application-independent tools, suitable to be applied on real data, and can handle arbitrary feature space analysis. We have used the k-means, mean shift and SLIC algorithms to detect human skin based on color.

The K-means clustering algorithm ([10, 11]) aims to partition n data samples into K clusters, in which the samples are assigned to the cluster with the nearest mean. In the field of data mining, the algorithm is considered as one of the top ten[15]. One of the main problems associated with the K-means clustering algorithm is the lack of the usage of spatial continuity. Then, the usage of this algorithm is restricted to input images that are defined by homogenous regions with respect to the local texture and color information. In [16], the authors have developed a space-partitioning algorithm that is able to return meaningful results even when applied to complex, natural scenes that exhibit large variation in color and texture.

The number of clusters K should be determined in advance, which is another problem of the algorithm. The speed of the algorithm or whether it converges is based on the choice of K. In [17], the authors have proposed an improved bisection K-means algorithm that segments images in the LAB color space. Their method does not need to determine the value of K in advance, improving the adaptability of the algorithm and lowering the subjectivity of segmentation.

The mean shift algorithm is another iterative clustering algorithm. It was introduced in [18] and then has been expanded and used in different fields such as computer vision, with probably the first application in [12] and [13]. It is a nonparametric algorithm that clusters data by finding modes (peaks) in the nonparametric probability density function of the data. These modes correspond to the densest regions (clusters) in the feature space. The algorithm then extracts clusters (segments) associated with each mode.

One of the problems associated with the mean shift algorithm is that the bandwidth h of the kernel density estimate must be determined in advance. In [19], the authors have developed an adaptive mean shift algorithm where the bandwidth h varies for each data point using the k-nearest neighbors (k-NN) algorithm [20].

The SLIC algorithm was developed based on the k-means algorithm[14]. The main difference between these algorithms is that in the SLIC algorithm, the search region is limited by a predefined area that encloses the superpixel (cluster) while in the k-means, the search region is defined as the whole color space. In other words, SLIC algorithm performs local clustering. Limiting the search region greatly reduces the complexity of the SLIC algorithm compared to the k-means. The SLIC algorithm also provides other advantages over the k-means algorithm such as providing control over the compactness of the superpixels through introducing a new distance measure that takes into account not only the color but also the spatial coordinates. Given these desirable features, the SLIC algorithm has been commonly used in many image processing applications, particularly in salient region/object detection[21],[22] and [23].

Like k-means algorithm, the SLIC algorithm has a downside since the number of clusters K must be predefined. As mentioned previously, there have been some research studies conducted eliminating the necessity for defining K.

Although there are many color-based skin segmentation algorithms ([24],[25]), some difficulties still exist due to the variations in color tones and the presence of light variations, shadows, noise, etc. Thus, we need a reliable segmentation model that withstands all these difficulties. Color-based skin segmentation models obviously depend on both a color space (RGB, HSV, etc.) at which segmentation is performed and a classification method used to determine whether a pixel is skin or non-skin. We have used the K-means,

mean shift and SLIC algorithms to detect human skin based on color. The algorithms are implemented, so that they produce the clustered image in the RGB color space. Our method begins by clustering images using the three algorithms and then segmenting the clustered regions that occupy skin. Pixels in the clusters are classified to be skin or non-skin using the Kovac model [26].

We have compared the performance of the K-means, mean shift and SLIC algorithms under different settings. We have compared their performance in a controlled environment (faces with uniform backgrounds and image resolutions) and in an uncontrolled environment (images with crowded faces, various light conditions and resolutions). The qualitative results have shown that the mean shift algorithm has better segmentation performance in both environments. The SLIC algorithm comes in second regarding the performance, yet it is the fastest. Also, we have noticed that the quality of a segmented image using the K-means algorithm varies slightly after the algorithm has been run multiple times for the same number of clusters K due to the initial means.

We are interested in testing the algorithms under different environments. Thus, we have quantitatively evaluated the algorithms' performance on a newly created dataset representing the uncontrolled environment. In the dataset, we have chosen the images so that they contain many faces taking under various lighting conditions and have different resolutions. We have also tested the algorithms on another dataset, called SFA database [27]. This dataset represents the controlled environment where the images contain only one face with uniform backgrounds and resolutions. We have evaluated the K-means and SLIC algorithms for different numbers of clusters K and evaluated the mean shift for different bandwidths h. To evaluate the performance of these algorithms, we have used four measures which are time complexity, F1 score (or F-measure), recall, and precision. The results show that on average the mean shift algorithm has the best performance on all four measures compared to the others. The SLIC algorithms almost matches the performance of the mean shift, but it excels in terms of speed. We have found that the K-means algorithm has a good performance when the number of clusters K is between 10 and 15. On the other hand, the mean shift algorithm has good performance when the bandwidth h is between 0.03 and 0.06. The SLIC

algorithm maxes out its performance at around $k = 100$ and the number of clusters can be increased to $K = 300$ without introducing a substantial amount of time.

1.2 ORGANIZATION

The remainder of this thesis is organized as follows:

- Chapter 1: provides a general overview of this thesis.
- Chapter 2: covers theoretical backgrounds behind the K-means, mean shift, and SLIC algorithms. Also, it will discuss their strengths and weaknesses.
- Chapter 3: presents experimental results.
- Chapter 4: presents concluding remarks.

2.0 THEORETICAL BACKGROUND

2.1 K-MEANS ALGORITHM FOR DATA CLUSTERING

2.1.1 Overview

The K-means algorithm ([10, 11]) is an unsupervised learning algorithm used for clustering data. In the field of data mining, the algorithm is considered as one of the top ten [15]. In addition, it is heavily used in computer vision for grouping objects into K groups based on attributes or features. It has many applications, including image segmentation.

The K-means algorithm begins by randomly creating K points (called centers) in a feature space such as the color space. Then, all data points in the feature space are assigned to their nearest centers by minimizing the distances between them and the centers. The distance measure (measure of similarity and dissimilarity) used here is the squared Euclidean distance. After that, the centers are updated by computing the means of the data point locations assigned to them. In other words, each centroid is moved to be the mean of all of the points belongs to the designated center. We keep assigning points to the centers and shifting them until convergence (no further shifts are available in one iteration). Practically, the algorithm is terminated when the minimum shift is less than a certain threshold. This process is simply shown in Figure 2.1.

2.1.2 Algorithm

We are going to show how the K-means algorithm theoretically works. Given some data, $D = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_n)$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $d > 2$, we now need to cluster these data points into K clusters. Assume that there are K clusters, with "unknown" centers $(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_j, \ldots, \boldsymbol{\mu}_K)$,

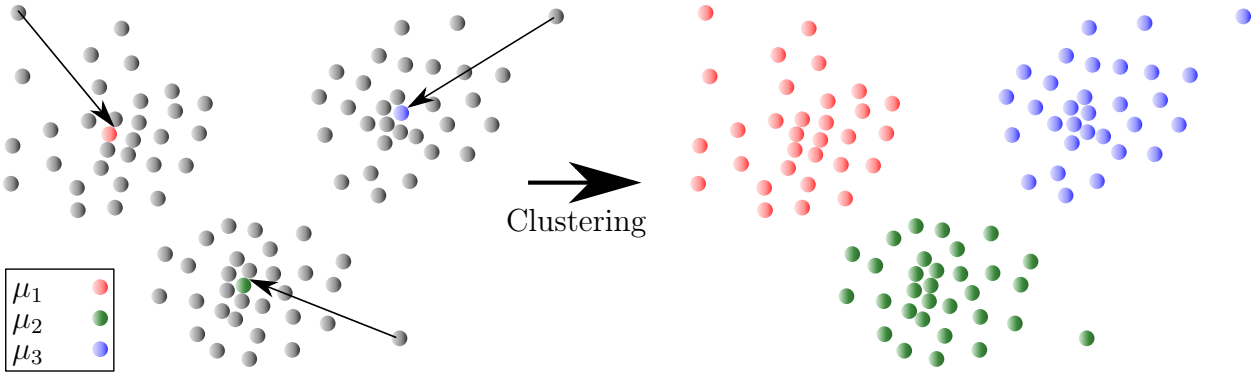


Figure 2.1: Clustering data in 2D feature space using the K-means algorithm. K here is equal to 3.

where $\boldsymbol{\mu}_j \in \mathbb{R}^d$ and $d > 2$. One measure of how accurate the clustering is would be the sum of the distances to the center $\boldsymbol{\mu}_j$, such that

$$\sum_{\substack{\text{i : } \boldsymbol{x}_i \text{ is} \\ \text{assigned to} \\ \boldsymbol{\mu}_j}} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$$

We now need to minimize L, where

$$L = \sum_{j=1}^{K} \sum_{\substack{\text{i : } \boldsymbol{x}_i \text{ is} \\ \text{assigned to} \\ \boldsymbol{\mu}_j}} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$$

Let's rewrite the previous equation as

$$\begin{aligned} L &= \sum_{j=1}^{K} \sum_{i=1}^{n} a_{ij} \|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2 \\ &= \sum_{j=1}^{K} \sum_{i=1}^{n} a_{ij} (\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T (\boldsymbol{x}_i - \boldsymbol{\mu}_j) \end{aligned} \tag{2.1}$$

$$\text{where} \qquad a_{ij} = \begin{cases} 1 & \text{if } \boldsymbol{x}_i \text{ is assigned to } \boldsymbol{\mu}_j \\ 0 & \text{else} \end{cases}$$

We are trying to minimize Equation 2.1 by choosing the assignments a_{ij}s and the centers $\boldsymbol{\mu}_j$s. It is difficult to minimize L analytically, because a_{ij} has two values, which makes an

analytical solution impossible. Instead, K-means tries to minimize L iteratively with respect to a_{ij}s and $\boldsymbol{\mu}_j$s by the following three steps:

Step 1: Choose the optimal assignments a_{ij}s for the fixed (given) centers $\boldsymbol{\mu}_j$s (created randomly). In other words, we will fix $\boldsymbol{\mu}_j$s and then optimize a_{ij}s.

Step 2: Choose the optimal $\boldsymbol{\mu}_j$s for the fixed (given) a_{ij}s. In other words, we will fix the a_{ij}s and then optimize the $\boldsymbol{\mu}_j$s.

Step 3: Iteratively, repeat step 1 and 2 until convergence.

The first step asks to find the optimal choice of the assignments a_{ij}s when the centers $\boldsymbol{\mu}_j$s are fixed. This is simple where the optimal choice of assignments is by assigning each point to the nearest center. Therefore, we are going to assign each point $\boldsymbol{x}_i$ to the nearest $\boldsymbol{\mu}_j$, as shown in Equation 2.2.

$$a_{ij} = \begin{cases} 1 & \text{if } j = \arg\min_l \|\boldsymbol{x}_i - \boldsymbol{\mu}_l\|^2 \\ 0 & \text{else} \end{cases} \tag{2.2}$$

On the other side, the second step of the K-means algorithm asks to find the optimal centers $\boldsymbol{\mu}_j$s that minimize the objective function L given the assignments a_{ij}s. Then, we will minimize L by obtaining its derivative with respect to $\boldsymbol{\mu}_j$ for fixed a_{ij} and then equalize the derivative to zero, as follows:

$$\begin{aligned} \nabla_{\boldsymbol{\mu}_j} L &= 0 \\ \sum_{i=1}^{n} \sum_{j'=1}^{K} a_{ij'} \nabla_{\boldsymbol{\mu}_{j'}} \left[(\boldsymbol{x}_i - \boldsymbol{\mu}_{j'})^T (\boldsymbol{x}_i - \boldsymbol{\mu}_{j'}) \right] &= 0 \end{aligned} \tag{2.3}$$

$\nabla_{\boldsymbol{\mu}_{j'}}[(\boldsymbol{x}_i - \boldsymbol{\mu}_{j'})^T (\boldsymbol{x}_i - \boldsymbol{\mu}_{j'})]$ is equal to zero unless $j' = j$. Then, we can rewrite Equation 2.3 as follows:

$$\begin{aligned}
0 &= \sum_{i=1}^{n} a_{ij} \nabla_{\boldsymbol{\mu}_j} \left[(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^T (\boldsymbol{x}_i - \boldsymbol{\mu}_j) \right] \\
&= \sum_{i=1}^{n} a_{ij} \nabla_{\boldsymbol{\mu}_j} \left(\boldsymbol{x}_i^T \boldsymbol{x}_i - 2\boldsymbol{\mu}_j^T \boldsymbol{x}_i + \boldsymbol{\mu}_j^T \boldsymbol{\mu}_j \right) \\
&= \sum_{i=1}^{n} a_{ij} \left(-2\boldsymbol{x}_i + 2\boldsymbol{\mu}_j \right) \\
&= -2 \sum_{i=1}^{n} a_{ij} \boldsymbol{x}_i + 2\boldsymbol{\mu}_j \sum_{i=1}^{n} a_{ij} \\
&= -\sum_{i=1}^{n} a_{ij} \boldsymbol{x}_i + \boldsymbol{\mu}_j \sum_{i=1}^{n} a_{ij}
\end{aligned}$$

Then

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} a_{ij} \boldsymbol{x}_i}{\sum_{i=1}^{n} a_{ij}} \tag{2.4}$$

We need to check if $\boldsymbol{\mu}_j$ shown in Equation 2.4 is the minimum for L by taking the second derivative of L, i.e., we need to find the Hessian matrix.

$$\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\mu}_{jk}} \left(\nabla_{\boldsymbol{\mu}_j} L \right) &= \frac{\partial}{\partial \boldsymbol{\mu}_{jk}} \left(-2 \sum_{i=1}^{n} a_{ij} \boldsymbol{x}_i + 2\boldsymbol{\mu}_j \sum_{i=1}^{n} a_{ij} \right) \\
&= \left(2 \sum_{i=1}^{n} a_{ij} \right) \frac{\partial}{\partial \boldsymbol{\mu}_{jk}} \left(\boldsymbol{\mu}_j \right)
\end{aligned} \tag{2.5}$$

$\frac{\partial}{\partial \boldsymbol{\mu}_{jk}} \left(\boldsymbol{\mu}_j \right)$ is equal to zero, except when $j = k$, and then it is going to be one. Hence, Equation 2.5 can be updated as follows.

$$\frac{\partial}{\partial \boldsymbol{\mu}_{jk}} \left(\nabla_{\boldsymbol{\mu}_j} L \right) = 2 \left(\sum_{i=1}^{n} a_{ij} \right) \boldsymbol{e}_k \quad \text{where} \quad k = 1, \ldots, K$$

$\boldsymbol{e}_k$, in the previous equation, is the standard basis such that $\boldsymbol{e}_1 = (1, 0, \ldots, 0)$ and $\boldsymbol{e}_K = (0, 0, \ldots, 1)$. Then, the Hessian matrix is given by

$$\nabla^2_{\boldsymbol{\mu}_j} L = 2 \left(\sum_{i=1}^{n} a_{ij} \right) I \tag{2.6}$$

I, in the previous equation, is the identity matrix. As long as at least one of the a_{ij}s shown in Equation 2.6 is nonzero, then the Hessian is going to be positive definite (i.e., $\nabla^2_{\boldsymbol{\mu}_j} L > 0$). Therefore, $\boldsymbol{\mu}_j$ shown in Equation 2.4 is the minimum.

Using the definition of a_{ij} shown in Equation 2.2 and the condition that at least one of the a_{ij}s must be nonzero, we can conclude that the Hessian matrix is positive definite when there is at least one point $\boldsymbol{x}_i$ that is assigned to the mean $\boldsymbol{\mu}_j$. When all a_{ij}s are zeros (happens very rarely), then we must re-pick the centers $\boldsymbol{\mu}_j$s randomly and restart the algorithm from the beginning.

Assume that $n_j > 0$ (i.e., there is at least one point $\boldsymbol{x}_i$ that is assigned to $\boldsymbol{\mu}_j$), where

$$n_j = \sum_{i=1}^{n} a_{ij}$$

then

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n} a_{ij} \boldsymbol{x}_i}{\sum_{i=1}^{n} a_{ij}} = \frac{1}{n_j} \sum_{\substack{\text{i : } \boldsymbol{x}_i \text{ is} \\ \text{assigned to} \\ \boldsymbol{\mu}_j}} \boldsymbol{x}_i \tag{2.7}$$

The last equation is going to be the update for the first step of the K-means algorithm, and it has very intuitive interpretation. It forms the average for all points' locations inside the cluster j.

Finally, using the K-means algorithm, we iterate between two steps. In the first step, we choose the assignments a_{ij}s by assigning each point to the nearest center as shown in Equation 2.2. Then, in the second step, we update each center j by computing the average for the points' locations inside the cluster j, as shown in Equation 2.7. We have summarized the K-means algorithm as shown in Algorithm 1.

2.1.3 Strengths and Weaknesses

The strengths of the K-means algorithm:

1. It is very fast with algorithm complexity $\mathcal{O}(\#\,Iterations \times \#\,Clusters \times \#\,Instances \times \#\,Dimensions)$.

The weaknesses of the K-means algorithm:

Algorithm 1: K-means algorithm.

Input : The set of data $D = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_n)$, and K number of centers. Note that $\boldsymbol{x}_i \in \mathbb{R}^d$ and $d > 2$.

Output: K number of clusters.

Steps :

- Place the centers $(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_j, \ldots, \boldsymbol{\mu}_K)$ at random locations where $\boldsymbol{\mu}_j \in \mathbb{R}^d$ and $d > 2$.
- Repeat the following two steps until convergence (i.e., when none of the cluster assignments change):

  Step 1: For each point $\boldsymbol{x}_i$:

  -Find the nearest center $\boldsymbol{\mu}_j$ using the squared Euclidean distance.

  -Assign the point $\boldsymbol{x}_i$ to the cluster j.

  Step 2: Update the mean $\boldsymbol{\mu}_j$ for each cluster j, from the previous step, by computing the mean (average) of all points inside the cluster j.

1. We need to choose the number of clusters K ahead.
2. It is not guaranteed to converge to the global minimum of L, but usually it works quite well and fast.
3. The selection of the initial centers is very critical because the algorithm sometimes gets stuck at local minima and then returns very bad clusters (segments). Also, the selection of the initial centers influences the algorithm run time.
4. It is very sensitive to outliers.
5. The centers (whether the initial or the updated centers) need not be points in the clusters; otherwise, the algorithm would be the K-medoids algorithm [28]. The initial centers in the K-medoids algorithm are points in the clusters that have the closest distances to the initial centers. In addition, the algorithm enforces the updated centers to also be points in the clusters.
6. The clusters have spherical (or elliptical) shapes.

2.2 MEAN SHIFT ALGORITHM FOR DATA CLUSTERING

2.2.1 Overview

The mean shift algorithm is another unsupervised data clustering algorithm. It was introduced in [18] and then has been expanded and used in different fields such as computer vision, with probably the first applications in [12] and [13]. It has many applications in computer vision, such as image segmentation and object tracking [29]. The algorithm treats a feature space such as the color space as a nonparametric probability density function (pdf). In other words, it considers the points in the feature space to be sampled points from the underlying probability density function. The mean shift algorithm detects the densest regions (corresponding to clusters) in the feature space. These regions form the modes (or local maxima) in the nonparametric pdf. The algorithm then extracts clusters (segments) associated with each mode.

The mean shift starts by defining windows with predefined bandwidth at each point (or random locations) in the feature space. The centers of the windows form the initial estimates of the densest regions. For each window, we compute the center of mass that is equal to the weighted mean (average) for all data points inside the window. After that, we shift each window to its center of mass as shown in Figure 2.2. The mean shift vector is shown in Equation 2.8. We keep calculating the means and shifting the windows until convergence, i.e., the mean shift vectors ≈ 0. Figure 2.3 provides an overview about how the mean shift algorithm works.

$$\boldsymbol{M}_h\left(\boldsymbol{y}_0\right) = \left[\frac{\sum_{i=1}^{n_x} w_i\left(\boldsymbol{y}_0\right)\boldsymbol{x}_i}{\sum_{i=1}^{n_x} w_i\left(\boldsymbol{y}_0\right)}\right] - \boldsymbol{y}_0 \tag{2.8}$$

where

- n_x : number of points in the kernel (window).
- w_i : associated weights determined by the kernel type.
- $\boldsymbol{y}_0$: initial mean location (the window center).
- $\boldsymbol{x}_i$: data points inside the kernel.
- h : kernel radius (bandwidth).

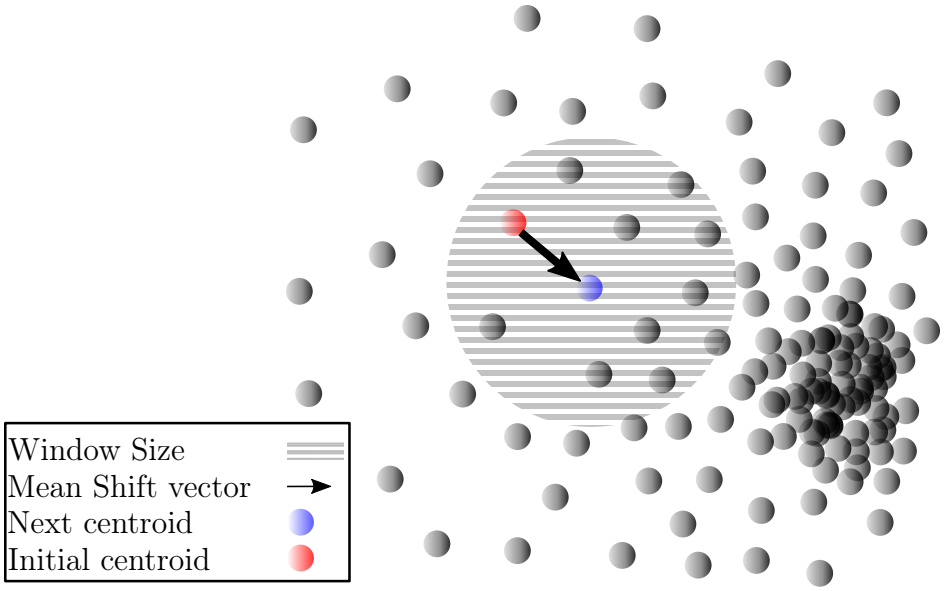


Figure 2.2: Shifting the center of the region of interest (window) to the center of mass using the mean shift algorithm in 2D feature space.

2.2.2 Algorithm

Before getting into the algorithm, we need to look at the differences between a parametric and a nonparametric probability density function (pdf). A Gaussian distribution has a parametric pdf i.e., has an explicit expression for pdf. Then, by computing the mean (peak) and standard deviation for the distribution, we can compute the probability of any value. This implies that we do not need to store all data in order to obtain a parametric pdf. On the other hand, we have to store all data in order to obtain a nonparametric pdf, which is a very time-consuming task, especially when we have big data. We need a nonparametric pdf, because sometimes a data distribution can not be fitted as a Gaussian distribution (a bell curve) or multiple Gaussian distributions (a mixture of Gaussian distributions).

The densities of data points in the distribution of data imply a nonparametric probability density function, as shown in Figure 2.4. We are now going to obtain a nonparametric probability density function for real data samples (a distribution of data) by using kernel density estimation. Then, we are going to show how the mean shift algorithm detects the local

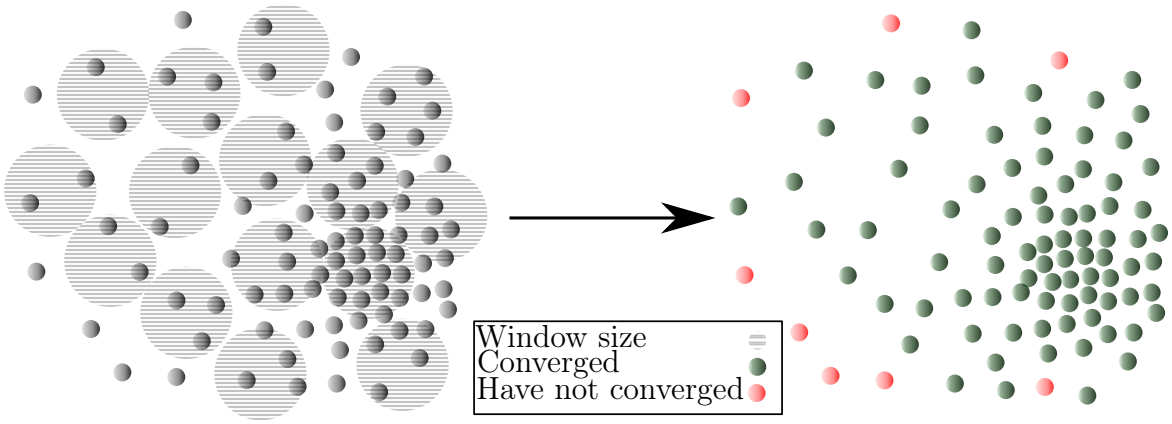


Figure 2.3: Overview on how the mean shift algorithm detects the densest region in 2D feature space.

maxima (peaks) that correspond to the densest regions (clusters) in the data distribution.

Given some data, $D = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_n)$, where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $d > 2$, the multivariate kernel density estimator computed at the point $\boldsymbol{x}$ with kernel $K(\boldsymbol{x})$ is given by

$$P(\boldsymbol{x}) = \frac{1}{n} \sum_{i=1}^{n} K(\boldsymbol{x} - \boldsymbol{x}_i) \tag{2.9}$$

In the previous equation, we can notice that all data points are contributing to obtaining the probability density function at $\boldsymbol{x}$. This is why we must store all data in order to obtain a nonparametric pdf. Also, the kernel in the previous equation adds weights to the distances based on how far the data points are from the point $\boldsymbol{x}$. If the data point $\boldsymbol{x}_i$ is very far away from the point $\boldsymbol{x}$, then the kernel will assign less weight and vice versa.

The kernel is a function that satisfies the following requirements:

1. Normalized: $\int_{R^d} K(\boldsymbol{x})\, d\boldsymbol{x} = 1$
2. $K(\boldsymbol{x}) \geq 0$
3. Symmetric: $\int_{R^d} \boldsymbol{x}\, K(\boldsymbol{x})\, d\boldsymbol{x} = 0$
4. Exponential weight decay: $\lim_{\|\boldsymbol{x}\| \to \infty} \|\boldsymbol{x}\|^d K(\boldsymbol{x}) = 0$

There are three popular kernel functions, which are shown in Figure 2.5:

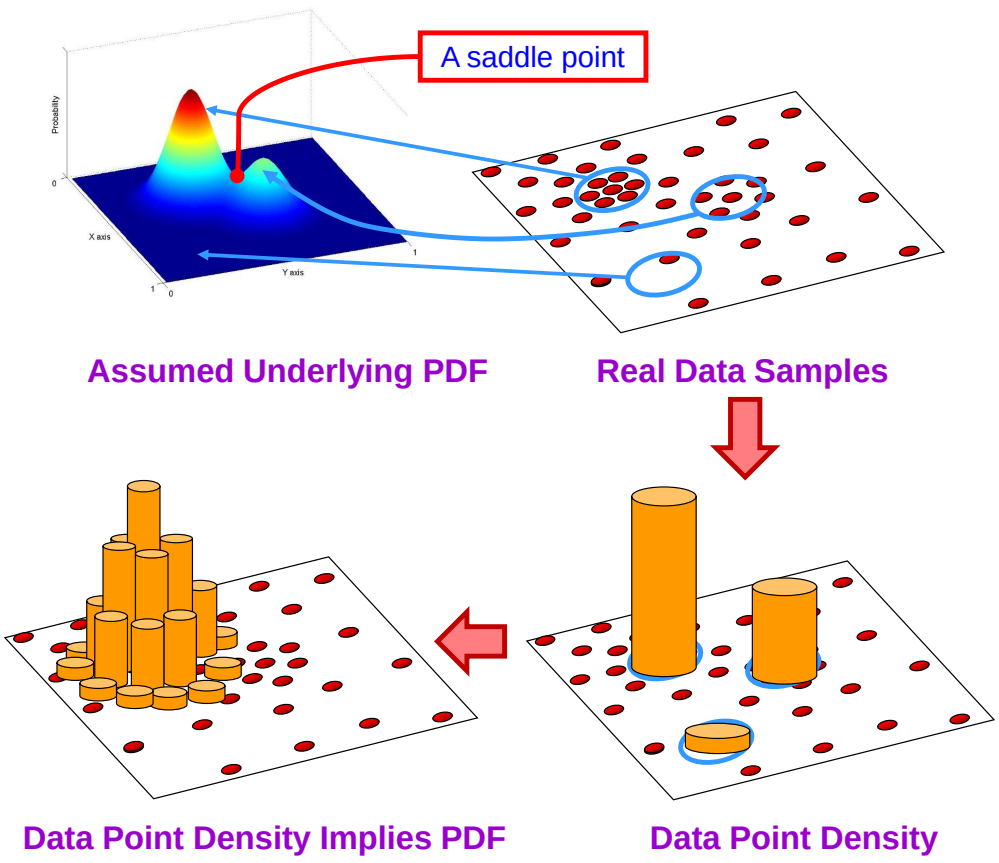


Figure 2.4: Densities of the data points in this 2D feature space imply a nonparametric pdf [30].

1. Epanechnikov kernel:

$$K_E(\boldsymbol{x}) = \begin{cases} c\left(1 - \|\boldsymbol{x}\|^2\right) & \|\boldsymbol{x}\| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

2. Uniform kernel:

$$K_U(\boldsymbol{x}) = \begin{cases} c & \|\boldsymbol{x}\| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

3. Normal kernel:

$$K_N(\boldsymbol{x}) = c.\exp\left(-\frac{1}{2}\|\boldsymbol{x}\|^2\right)$$

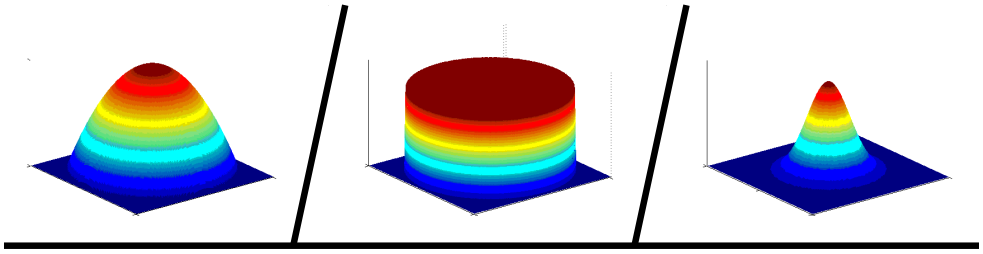


Figure 2.5: Kernel functions: (left) Epanechnikov kernel, (middle) uniform kernel, and (right) normal kernel [30].

We are looking for the radially symmetric kernels that only satisfy Equation 2.10. Then, we will update Equation 2.9 based on Equation 2.10, as shown in Equation 2.11.

$$K\left(\boldsymbol{x}\right) = \frac{c}{h^d}\, k\left(\left\|\frac{\boldsymbol{x}}{h}\right\|^2\right) \tag{2.10}$$

where

c : a normalization constant.

h : the kernel radius.

$k(x)$: the profile of the kernel.

$$P\left(\boldsymbol{x}\right) = \frac{c}{n\, h^d} \sum_{i=1}^{n} k\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|^2\right) \tag{2.11}$$

In order to find the relationship between the kernel density estimate shown in Equation 2.11 and the mean shift, we need to differentiate both sides of Equation 2.11 as follows.

$$\nabla_{\boldsymbol{x}} P\left(\boldsymbol{x}\right) = \frac{c}{n\, h^d} \sum_{i=1}^{n} \nabla_{\boldsymbol{x}} k\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|^2\right)$$

By using the chain rule, we get

$$\nabla_{\boldsymbol{x}} P\left(\boldsymbol{x}\right) = \frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} \left(\boldsymbol{x} - \boldsymbol{x}_i\right) k'\left(\left\|\frac{\boldsymbol{x} - \boldsymbol{x}_i}{h}\right\|^2\right)$$

Assume that $g(\boldsymbol{x}) = -k'(\boldsymbol{x})$, then

$$\nabla_{\boldsymbol{x}} P(\boldsymbol{x}) = \frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{x})\, g\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right)$$

After expanding the previous equation, we obtain

$$\begin{aligned} \nabla_{\boldsymbol{x}} P(\boldsymbol{x}) =& \frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} \boldsymbol{x}_i\, g\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) - \\ & - \frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} \boldsymbol{x}\, g\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \end{aligned}$$

We are going to manipulate the previous equation in order to suit it to the mean shift as follows.

$$\begin{aligned} \nabla_{\boldsymbol{x}} P(\boldsymbol{x}) =& \frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} g\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right) \cdot \\ & \cdot \left[\frac{\sum_{i=1}^{n} \boldsymbol{x}_i\, g\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right)}{\sum_{i=1}^{n} g\left(\left\| \frac{\boldsymbol{x} - \boldsymbol{x}_i}{h} \right\|^2 \right)} - \boldsymbol{x} \right] \end{aligned}$$

For simplicity, we will rewrite the previous equation as

$$\nabla_{\boldsymbol{x}} P(\boldsymbol{x}) = \left[\frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} g_i \right] \times \left[\frac{\sum_{i=1}^{n} \boldsymbol{x}_i\, g_i}{\sum_{i=1}^{n} g_i} - \boldsymbol{x} \right] \tag{2.12}$$

The gradient of the kernel density estimate shown in Equation 2.12 implies the mean shift algorithm as shown in Figure 2.6. Therefore, the mean shift is the gradient of the nonparametric probability density function. For more clarification, we will rewrite Equation 2.12, as in Equation 2.13, where $\boldsymbol{m}(\boldsymbol{x})$, in the equation, represents the mean shift vector.

$$\nabla_{\boldsymbol{x}} P(\boldsymbol{x}) = \frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} g_i \times \boldsymbol{m}(\boldsymbol{x}) \tag{2.13}$$

where

$$\boldsymbol{m}(\boldsymbol{x}) = \frac{\nabla_{\boldsymbol{x}} P(\boldsymbol{x})}{\frac{2c}{n\, h^{d+2}} \sum_{i=1}^{n} g_i}$$

Equation 2.12 and Equation 2.13 show two important conclusions: (1) The vector of the mean shift algorithm always indicates the direction where the density experiences the

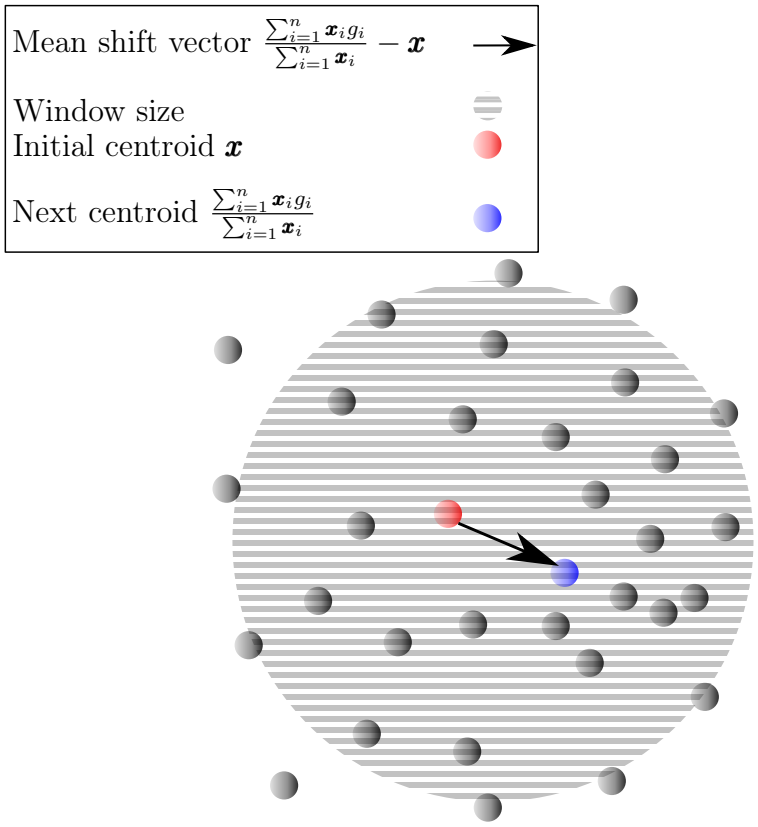


Figure 2.6: The gradient of the kernel density estimate implies the mean shift algorithm.

maximum increase, and (2) the direction of the gradient of the non-parametric pdf estimate is the same as the direction of the mean shift vector.

We need to mention that a cluster (a segment) represents a mode in which the attraction basin encloses all data points. The attraction basin is the region where all windows' trajectories are attracted to the same mode, as shown in Figure 2.7. We have summarized the mean shift algorithm as shown in Algorithm 2.

When the algorithm is stuck at saddle points, as shown in Figure 2.4, we need to do small, random perturbation (change) to mode positions and then check if we return back. Also, to improve the convergence speed of the algorithm, it is recommended to use the adaptive mean shift algorithm. In the adaptive algorithm, the window size h

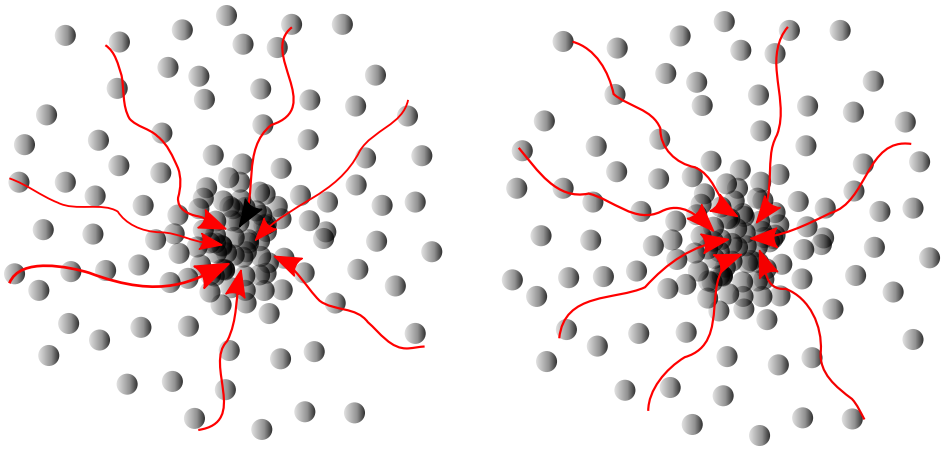


Figure 2.7: Two attraction basins.

varies for each data point using the k-nearest neighbors (k-NN) algorithm [20]. Let's assume that $\boldsymbol{x}_{i,k}$ is the k-nearest neighbor of $\boldsymbol{x}_i$; then, the window size is computed as

$$h = \|\boldsymbol{x}_i - \boldsymbol{x}_{i,k}\|$$

If the k-NN algorithm is used to determine h, then the choice of k will impact the value of h. For good performance, k should increase when the dimension of data increases.

Finally, for better clustering performance when we use the algorithm for image segmentation, in the feature space of pixels, we should consider the spatial proximity. Then, the feature space will be defined as a joint of two domains (the spatial domain and the range domain). The joint domain can be incorporated using the multivariate kernel K_{h_s,h_r} shown in the following equation.

$$K_{h_s,h_r} = \frac{c}{h_s^2 h_r^d} k\left(\left\|\frac{\boldsymbol{x}^s}{h_s}\right\|^2\right) k\left(\left\|\frac{\boldsymbol{x}^r}{h_r}\right\|^2\right) \tag{2.14}$$

where

$\boldsymbol{x}^s$	: the spatial part of a feature vector.
$\boldsymbol{x}^r$	: the range part (such as color information) of a feature vector.
h_s and h_r	: the kernel bandwidths.

Algorithm 2: Mean shift algorithm.

Input : The bandwidth h (or the two bandwidths h_s and h_r), and the set of data $D = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_n)$ where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $d > 2$.

Output: A number of clusters.

Steps :

- Define windows with the predefined bandwidth h at each data point (or random locations).
- For each window in the space, repeat the following two steps until convergence (i.e., the mean shift vectors ≈ 0):

  Step 1: Calculate the mean shift vector $\boldsymbol{m}(\boldsymbol{x})$.

  Step 2: Move each window from $\boldsymbol{x} \rightarrow \boldsymbol{x} + \boldsymbol{m}(\boldsymbol{x})$.
- Clusters are defined by the attraction basins of the detected modes.

The spatial resolution parameter h_s affects the smoothing and connectivity of segments. It is determined according to the image size and objects. The range resolution parameter h_r affects the number of segments. It should be kept low if contrast is low.

2.2.3 Strengths and Weaknesses

The strengths of the mean shift algorithm:

1. We need only to specify the bandwidth (window size) h or the two bandwidths h_s and h_r.
2. The bandwidth h has a physical meaning, not like the parameter K in the K-Means algorithm.
3. It does not assume any prior shape (spherical or elliptical shape) about clusters. It handles arbitrarily shaped clusters because it is based on density estimation.
4. Adaptive gradient ascent: It has automatic convergence speed and converges to near maxima with small and refined steps.
5. Convergence is guaranteed.

6. It is not very sensitive to outliers.

The weaknesses of the mean shift algorithm:

1. The choice of the bandwidth h is not trivial, because a large h might lead to improper clustering (merge distinct clusters), and a very small h might lead to too many clusters, i.e., very slow convergence.
2. We need to use adaptive window size because improper window size can lead modes to be merged, which results in bad clusters (segments).
3. Although the algorithm is nonparametric, the window size h needs to be tuned using the k-NN algorithm!
4. When the dimension of data increases, the number of local maxima increases too. Then, the algorithm might not work efficiently in higher dimensions because it might converge quickly to local optima.
5. It is computationally very expensive compared to the K-means algorithm, with time complexity $\mathcal{O}(Tn^2)$, where T is the number of iterations and n is the number of data points.
6. The convergence speed and quality depend on the kernel type. For example, the algorithm converges quickly with a limited number of steps when we use the uniform kernel. On the other hand, the algorithm is slow when utilizing the normal kernel.

2.3 SLIC ALGORITHM FOR DATA CLUSTERING

2.3.1 Overview

Standing for Simple Linear Iterative Clustering , SLIC is similar to the k-means and mean shift algorithms in regard to being both gradient-ascent-based algorithms and unsupervised data clustering algorithms. However, many state-of-the-art algorithms, including k-means and mean shift, were compared with SLIC in terms of two standard measures, boundary recall and under-segmentation error, and SLIC was shown to be very superior to such

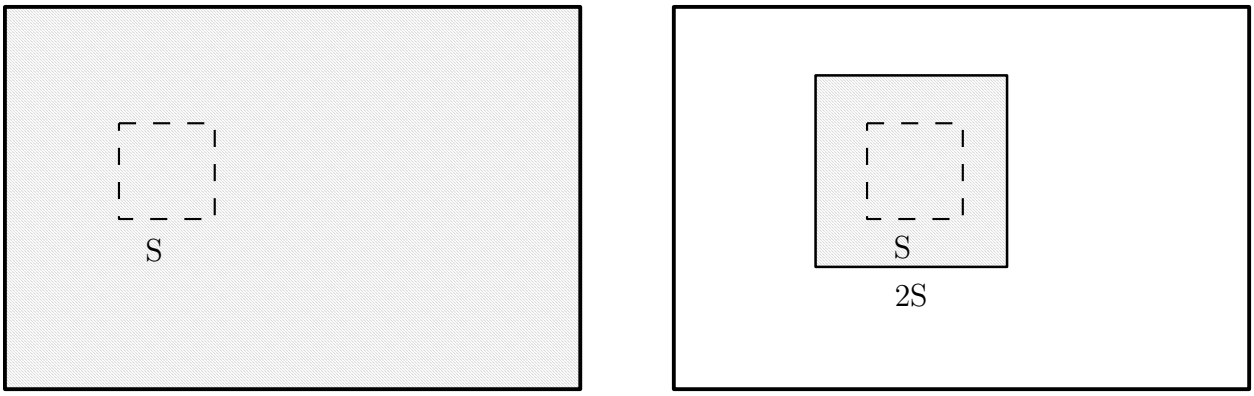


Figure 2.8: search region (shaded area): (left) k-means algorithm and (right) SLIC algorithm.

algorithms[14]. Besides producing a high segmentation quality, SLIC creates dense and homogenous superpixles with a minimal effort and complexity. It also provides control over the number and compactness of superpixles, features that are highly preferable. Thus, it is one of the most commonly used algorithms for image segmentation. It has been widely utilized in many image processing applications, one of which is salient region/object detection[21],[22] and [23].

We can consider SLIC as a special version of the k-means with two crucial differences, which provide major enhancements in the performance. First, in the k-means algorithm, the search region is the whole feature space,Figure 2.8, meaning every centroid k is compared to every point/pixel in the space resulting in a relatively higher complexity $\mathcal{O}(\#\,Iterations \times \#\,Clusters \times \#\,Instances \times \#\,Dimensions)$. In contrast, SLIC performs local clustering in which a limited region proportional to the size of the superpixels is defined as the search space contributing to lowering the computational cost $\mathcal{O}(\#\,Instances)$. Second, SLIC introduces a new distance measure that takes into account not only the spatial but also the color proximity offering at the same time the feature of controlling the number and compactness of the superpixels.

2.3.2 Algorithm

Given an image with a total number of pixels N, we need to segment the image into a number of superpixels K. Assuming superpixels having square shapes, the number of pixels per superpixel or the size of the superpixel is $S^2 = N/K$. Hence, every centroid ci of a superpixel is placed $S = \sqrt{N/K}$ apart on the xy plane of the image, which defines the grid interval. The pixels that could be similar to the pixels inside the superpixel are assumed to lie inside a $2S \times 2S$ search area enclosing the superpixel region. It is possible to find the similarities between the centroids and the pixels inside the search region by introducing the new distance measure which combines the color and spatial proximity.

The feature space in SLIC includes the color and the spatial (xy plane) proximity, so the total distance, similarity measure, Ds should contain both. The algorithm takes the input image in CIELAB color space which consists of l (luminance), a (color channel), and b (color channel) values. It is perceptually uniform color space in which a small change in any direction along any axis is clearly detectable by human eyes. Taking the simple Euclidean distance of this 5D space is inconsistent without normalizing the spatial distance to be relevant to the color distance. Therefore, to measure the distance between the centers of each superpixel and the pixels within the search region, a new distance measure is given by:

$$
\begin{aligned}
d_{lab} &= \sqrt{(l_k - l_i)^2 + (a_k - a_i)^2 + (b_k - b_i)^2} \\
d_{xy} &= \sqrt{(x_k - x_i)^2 + (y_k - y_i)^2} \\
D_s &= d_{lab} + \frac{m}{S} d_{xy}
\end{aligned}
\tag{2.15}
$$

From Equation Equation 2.15, we can see that the total distance D_s is the sum of the Euclidean distances of the colore d_{lab} and xy coordinates d_{xy} of the image. The spatial distance is normalized by the grid interval S. The constant m provides a control over the compactness of the superpixels. When m is large, D_s will be biased towards the spatial distance d_{xy} resulting in denser superpixels and vice versa.

We have summarized SLIC algorithm in Algorithm 3. The algorithm first begins by defining the cluster centers C_k in the 5D space. These are the centroids of the pixels taking

at every grid step S. Next, some of the centroids may end up at an edge of the image resulting in inappropriate segmentation. To avoid this situation, the centers are moved to a position in 3×3 neighborhood which has the lowest gradient. Then, three steps are repeated until convergence, the error is less than or equal a predefined threshold. First, within the search region $2S \times 2S$, every pixel is assigned to the nearest centroid. Once the whole image is segmented into different clusters and pixels are labeled, the new centroids of each cluster are recomputed to be the mean $[labxy]^T$ of all the pixels in the designated cluster. Next, the residual error should be calculated to decide whether to break the loop. Finally, a few pixels may rarely end up disjoint, so another step should be taken to reconnect them to the nearby superpixels.

Algorithm 3: SLIC algorithm.

Input : The number of superpixels K, the image in the CIELAB color space that needs to be segmented, and, optionally, the constant m which controls the compactness of the superpixels

Output: The labels of the superpixels/clusters.

Steps :

- Define cluster centers $C_k = [l_k, a_k, b_k, x_k, y_k]^T$ by taking pixels at every grid step S.
- readjust centroids in a 3×3 neighborhood, by moving them to positions with the lowest gradients.
- Repeat the following three steps until $E \leq threshold$:

  Step 1: For each centroid C_k : assign the pixels with the minimum distance using Equation2.15 between the pixels and the centroid within a $2S \times 2S$ search region to the superpixel region S.

  Step 2: Recompute the cluster centers.

  Step 3: Calculate the residual error E.
- Connect pixels that left unconnected to the nearby superpixels.

2.3.3 Strengths and Weaknesses

The strengths of the SLIC algorithm:

1. It is easy to implement with the number of the superpixels K being the only required parameter.
2. It provides control over the compactness of the superpixels by utilizing the constant m.
3. It is very fast with a linear algorithm complexity $\mathcal{O}(\#\,Instances)$, a result of limiting the search region.

The weaknesses of the SLIC algorithm:

1. We need to specify the number of superpixels K ahead, similar to the k-means algorithm.
2. We need to readjust the locations of the centroids by moving them to the positions with the lowest gradients.
3. It does not enforce connectivity, so an extra step needs to be taking to make sure every pixel is reassigned.

3.0 EXPERIMENT AND RESULTS

3.1 EXPERIMENT AND RESULTS

This section describes the experiments that have been carried out and discusses the qualitative and quantitative results.

3.1.1 Experiment Details

Due to lack of datasets with ground truths for skin detection evaluation, we created a new dataset representing the uncontrolled environment. Our dataset contains 300 open source images with different resolutions. Each image represents more than one face taking under different lighting conditions. We evaluated the performance of the K-means, mean shift and SLIC algorithms on this dataset and another dataset representing the controlled environment, called the SFA database [27]. SFA dataset contains 1118 face images with ground truths for skin detection evaluation. We randomly selected 100 face images to perform our experiments from both database. To increase the processing speed, we reduced the resolution of the images from 512×768 to 384×256 (SFA dataset), i.e., we reduced the dimensions to half of their sizes.

We have used the algorithms to detect human skin based on the color space. In the beginning, we clustered images using the three algorithms and then segmented the clustered regions that occupy skin. Pixels in the clusters are classified to be skin or non-skin using the Kovac model [26]. Based on the Kovac model, a pixel is classified as skin if it satisfies four rules. Since we are using RGB color images, then the rules of the Kovac model are based on the red (R), green (G), and blue (B) color intensity values of pixels of the clustered images.

The rules are assigned to filter out the non-skin pixels that are close to skin color tone. The Kovac rules are given as follows:

Rule 1: R > 95 and G > 40 and B > 20 and

Rule 2: Max(R,G,B) − Min(R,G,B) > 15 and

Rule 3: |R − G| > 15 and

Rule 4: R > G and R > B.

The rules were chosen to reduce false detections without compromising the ability to detect a wide range of skin color tones. The threshold values used in the above rules are fixed after checking its efficiency in detecting skin tones over a small dataset of face images.

The main advantage of applying the Kovac rules on the clustered images and not on the original ones is that during the clustering process, a pixel is grouped into a skin cluster if it is more similar to the centroid of that particular cluster than the centroids of other clusters. When the clustering process gets finished, the pixels in a particular cluster will have a unique value that represents the mean or centroid value in that particular cluster. Obviously, a skin cluster will have a value close to the majority of skin pixels. Hence, the Kovac rules can be easily applied without much error. On the other hand, if we directly apply the rules on original images without clustering, then the rules should be broadened; otherwise, it may lead to false detections. In other words, using clustering, we can make sure that a skin cluster will be represented by a pixel within the range of the Kovac rules.

For the sake of comparison, all pixels that occupy skin in the ground truths are set to 1, and others 0. Also, all pixels detected as skin are set to 1, and non-skin to 0. This is to simplify the process of comparison with the ground truths. Then, detection is performed on the basis of a pixel-to-pixel similarity measurement between a segmented image and its ground truth image. That means that every detected pixel is compared with the corresponding pixel value in the ground truth image. The quality of segmentation between each image and its ground truth has been evaluated using four measures, which are time complexity, F1 score (or F-measure), recall, and precision.

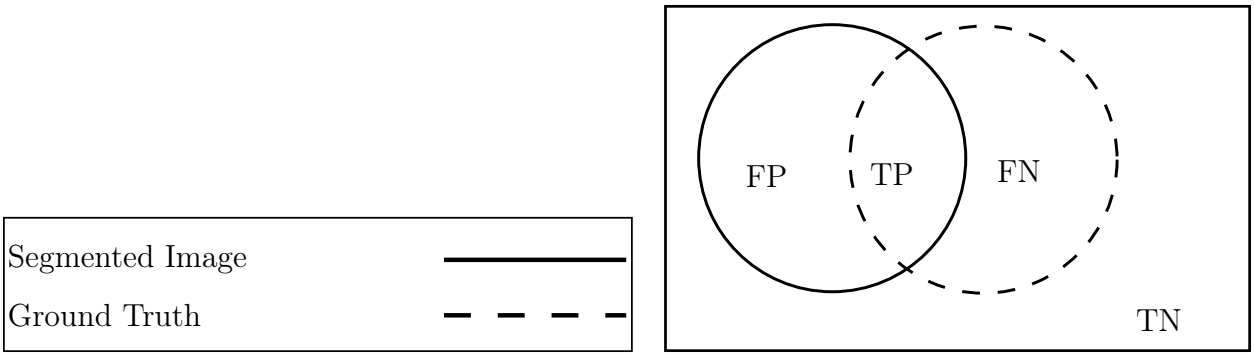


Figure 3.1: Evaluation space.

Let P_g and N_g respectively be the positive and negative pixel classes in a ground truth image. Similarly, let P_s and N_s respectively be the positive and negative pixel classes in a segmented image. Then, we can define the following four parameters, Figure 3.1.

True Positive (TP): the number of pixels that are detected as skin and also labelled as skin in the ground truth image. Then

$$TP = P_s \cap P_g$$

True Negative (TN): the number of pixels that are detected as non-skin and also labelled as non-skin in the ground truth image. Then

$$TN = N_s \cap N_g$$

False Positive (FP): the number of pixels that are detected as skin and labelled as non-skin in the ground truth image. Then

$$FP = P_s \cap N_g$$

False Negative (FN): the number of pixels that are detected as non-skin and labelled as skin in the ground truth image. Then

$$FN = N_s \cap P_g$$

Using the previous parameters, the recall and precision measures are respectively shown in Equation 3.1 and Equation 3.2. Although the recall and precision are robust measures to evaluate performance, the harmonic mean that gives a general trade-off between them is also important. This measure is called the F1 score (or F-measure), and it is shown in Equation 3.3.

$$Recall = \frac{TP}{TP + FN} \tag{3.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

$$F = 2 \cdot \frac{Recall \cdot Precision}{Recall + Precision} \tag{3.3}$$

We have used MATLAB R2015b to do our experiments. We have carried out our experiments on a standard laptop with Intel Core i7, 2.5 GHz CPU, and 8 GB RAM. Finally, we need to mention that we have used the uniform kernel in the implementation of the mean shift algorithm.

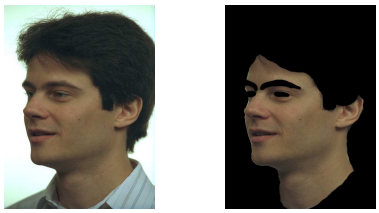
3.1.2 Qualitative Results

Using the K-means algorithm, we have segmented skin from images for different numbers of clusters K, and some results are shown in Figure 3.2. In addition, we have segmented images for different values of bandwidths h using the mean shift algorithm, and some results are shown in Figure 3.3. We have also used the same procedure to test the SLIC algorithm trying different numbers of superpixels K as shown in Figure 3.4. These experiments were executed in a controlled environment, i.e., faces with uniform backgrounds and image sizes. We have performed some experiments in an uncontrolled environment (i.e., images with crowded faces and different lighting conditions), and the results of both algorithms are shown in Figure 3.6.

From Figure 3.2, Figure 3.3, Figure 3.4, and Figure 3.6, it can be seen that all algorithms perform a reasonably good job. Nevertheless, it can be seen that the mean shift algorithm, since it takes into account the spatial variation of data during clustering, tends to outperform both K-means and SLIC algorithms with the SLIC being very close in performance to the

mean shift. We have also observed that the clustered images using the SLIC algorithm are usually visually more pleasant than the ones clustered by either the mean shift or the k-means algorithms, but for the task of skin segmentation, we prefer the performance of the mean-shift algorithm. Finally, we have noticed that the quality of a segmented image using the K-means and SLIC algorithms varies slightly after the algorithms are run multiple times for the same number of clusters K due to the initial means.

(Left) Original image.
(Right) Ground truth.



(Left) Clustered with $K = 4$.
(Right) Segmented with $K = 4$.

(Left) Clustered with $K = 8$.
(Right) Segmented with $K = 8$.

(Left) Clustered with $K = 10$.
(Right) Segmented with $K = 10$.

(Left) Clustered with $K = 12$.
(Right) Segmented with $K = 12$.

(Left) Clustered with $K = 16$.
(Right) Segmented with $K = 16$.

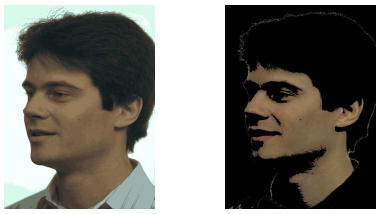


Figure 3.2: K-means algorithm: clustered and segmented skin with different number of clusters K. Black regions indicate pixels that are classified as non-skin.

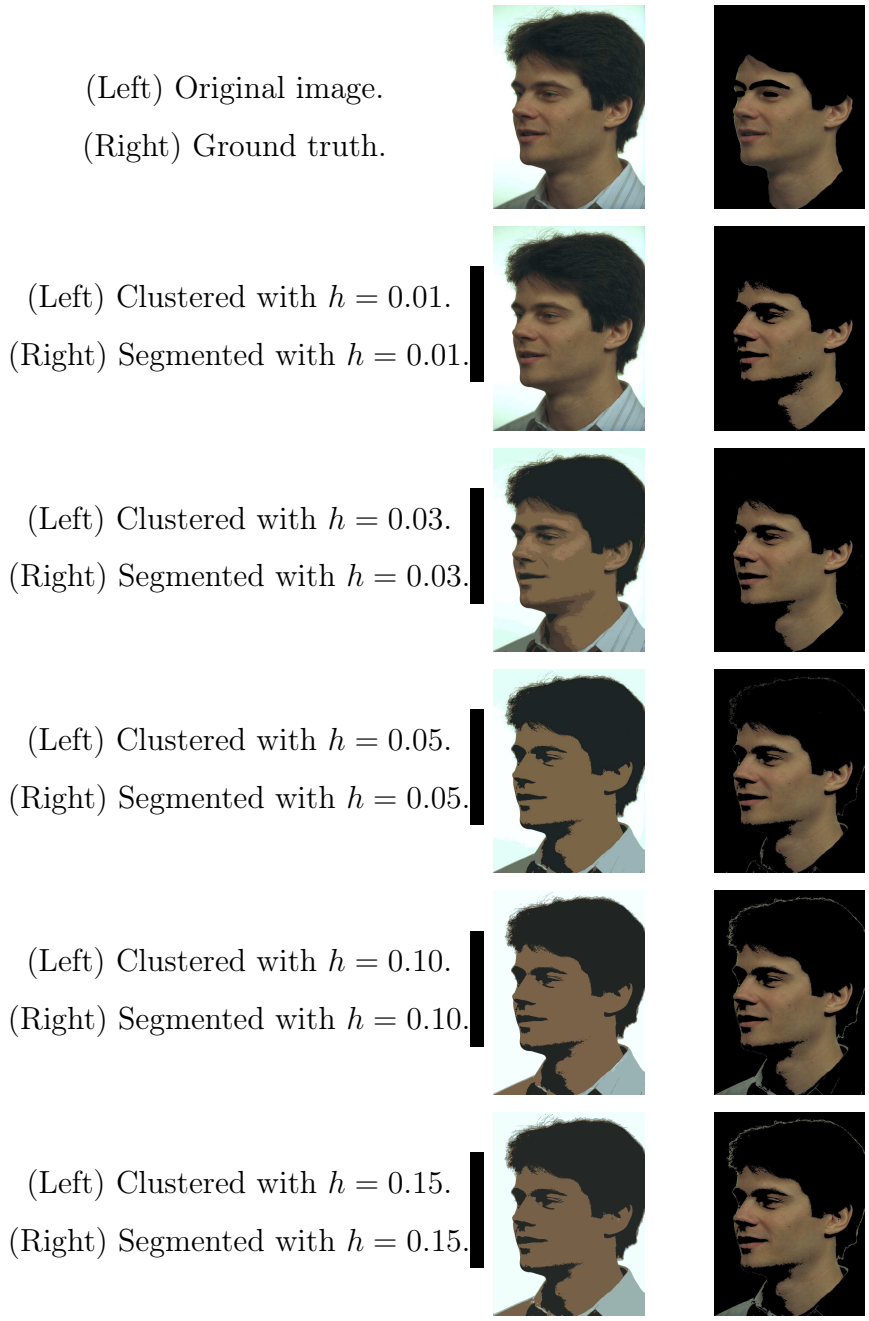


Figure 3.3: Mean shift algorithm: clustered and segmented skin with different bandwidths h. Black regions indicate pixels that are classified as non-skin.

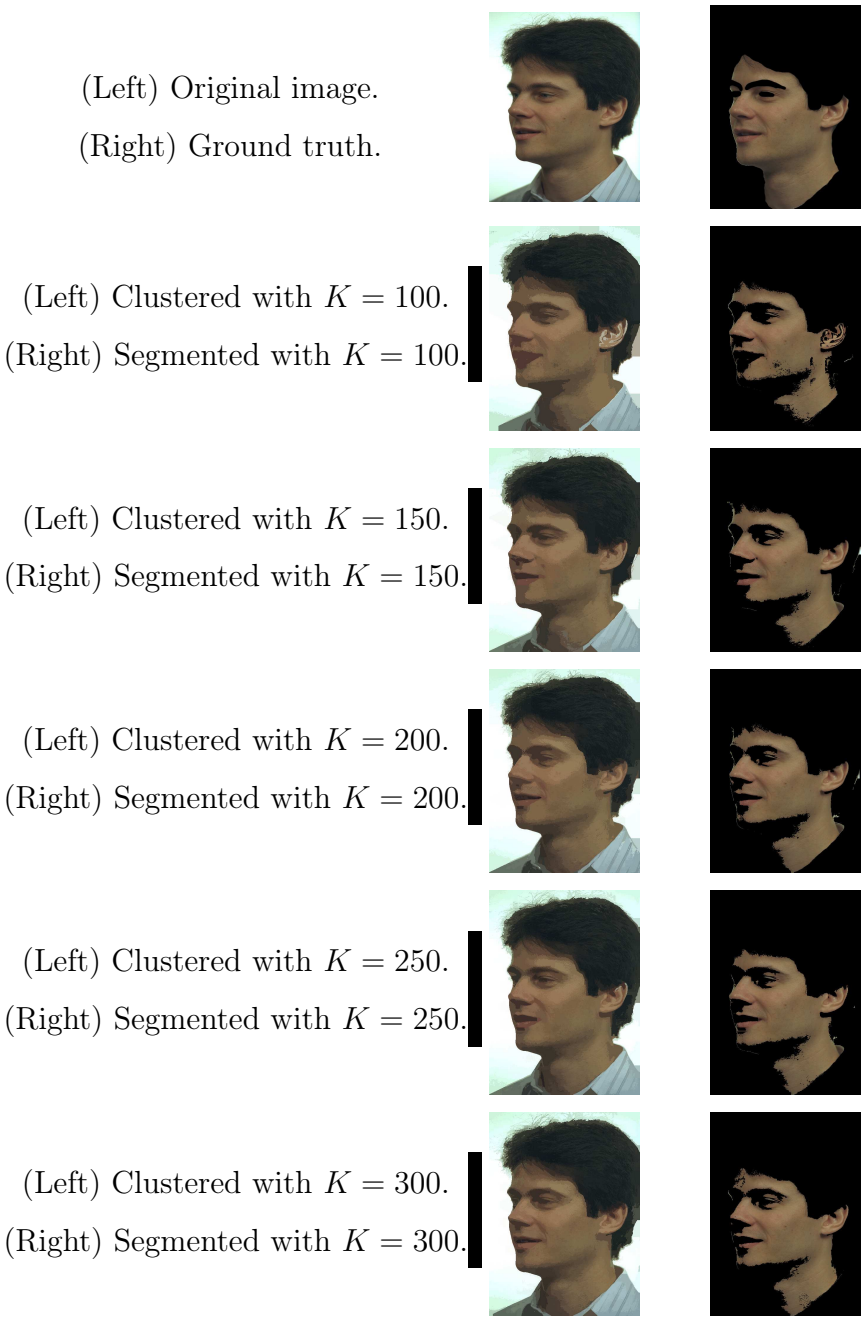


Figure 3.4: SLIC algorithm: clustered and segmented skin with different number of super-pixels K. Black regions indicate pixels that are classified as non-skin.

(a) Original image.

(b) Ground truth.

(c) Clustered using the K-means algorithm with $K = 19$.

(d) Segmented using the K-means algorithm with $K = 19$.

(e) Clustered using the mean shift algorithm with $h = 0.05$.

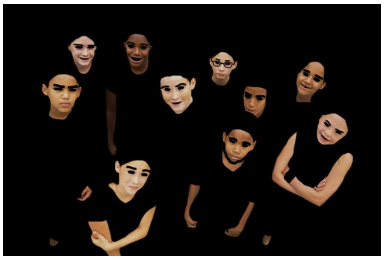
(f) Segmented using the mean shift algorithm with $h = 0.05$.

(g) Clustered using the SLIC algorithm with $k = 300$.

(h) Segmented using the SLIC algorithm with $k = 300$.

Figure 3.5: Continued on next page ...

(a) Original image.



(b) Ground truth.

(c) Clustered using the K-means algorithm with $K = 19$.

(d) Segmented using the K-means algorithm with $K = 19$.

(e) Clustered using the mean shift algorithm with $h = 0.05$.

(f) Segmented using the mean shift algorithm with $h = 0.05$.

(g) Clustered using the SLIC algorithm with $k = 300$.

(h) Segmented using the SLIC algorithm with $k = 300$.

Figure 3.6: Skin segmentation in an uncontrolled environment: clustered and segmented skin with different numbers of clusters K using both the K-means and SLIC algorithms, and different bandwidths h using the mean shift algorithm. Black regions indicate pixels that are classified as non-skin.

3.1.3 Quantitative Results

The time complexity for the segmentation process can be remarkably reduced, especially for high resolution images, by making use of the already clustered images. This can be done by applying the Kovac rules on the centroids of the clusters rather than evaluating every individual pixel in the image. For instance, evaluating the image of a pixel-by-pixel basis, if we have a 5 Megapixel image, we would need to evaluate 5 million pixels which is usually larger compared to the number of clusters. Figure 3.7 shows that as the resolution of the image increases, the time gap between the clustered and unclustered images significantly increases with the clustered image consuming very little time.

Prior to analyzing the performance of the k-means, mean shift and SLIC algorithms, we need to show how the time complexity of each algorithm changes when changing the image

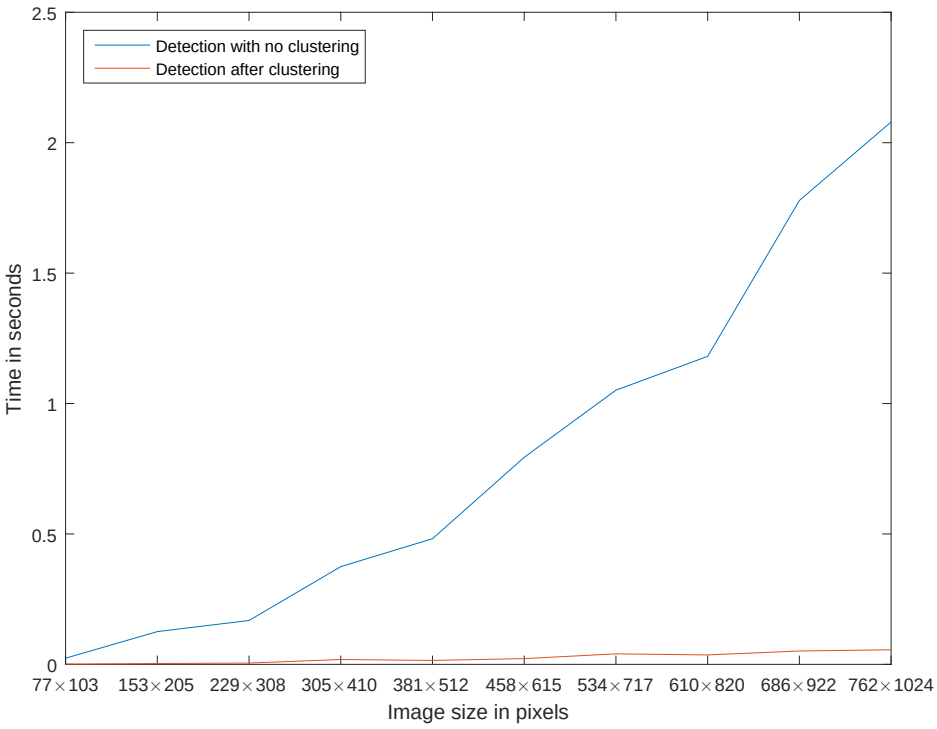


Figure 3.7: Comparing time complexity of detection process between clustered and unclustered images.

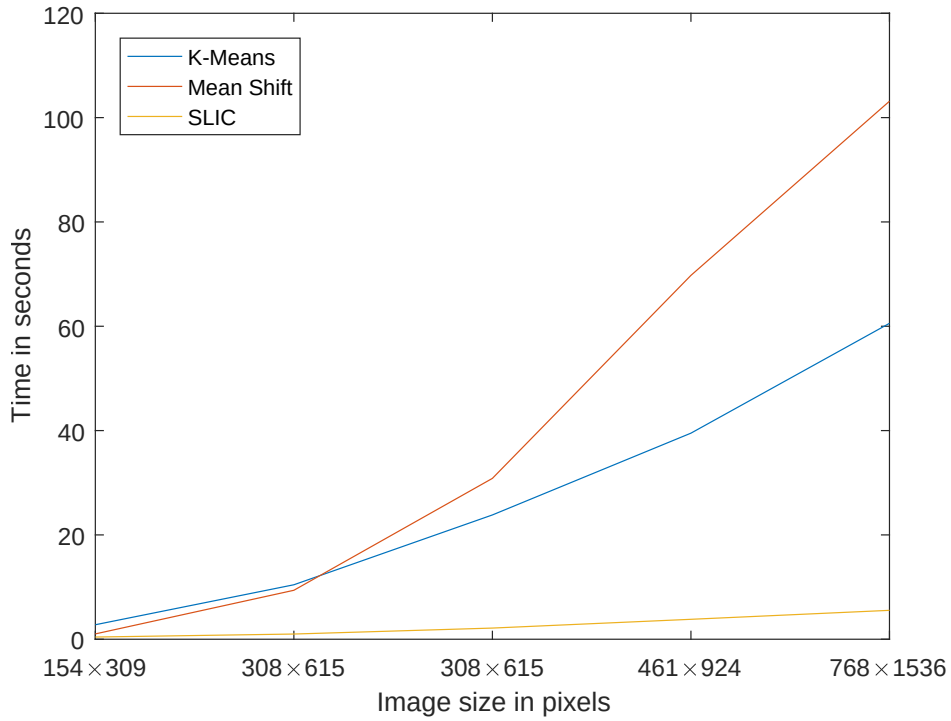
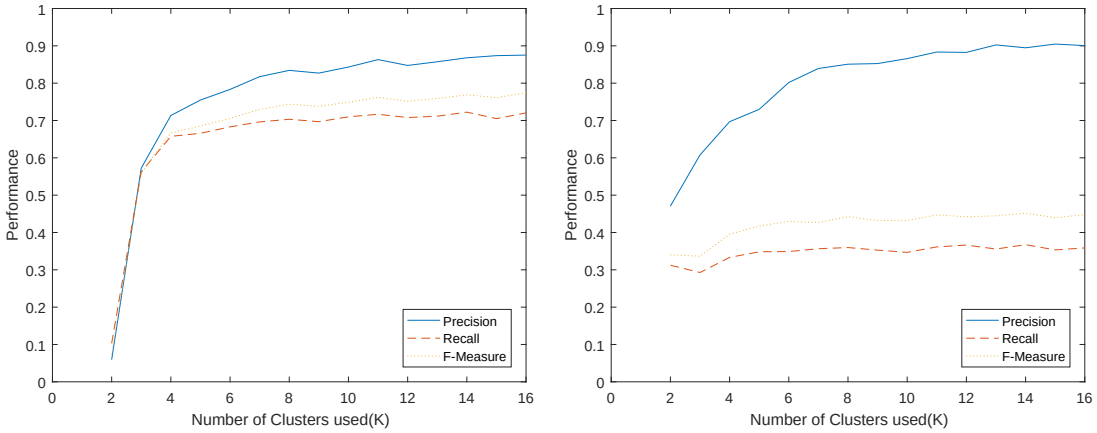


Figure 3.8: Comparing time complexity of the k-means, mean shift, and SLIC algorithms with respect to image resolutions.

resolution (the number of pixels N). We have tested these algorithms on an image scaled into different resolutions. As can be seen from Figure 3.8, the time complexity of the all algorithms increases as the image size increases. However, the complexity of SLIC algorithm slightly increases with N (smallest slop) compared to the k-means and mean shift with the mean shift being the most time consuming algorithm. This shows that the SLIC algorithm has a linear complexity depending only on the number of pixels N.

The performance curves in Figure 3.9 show changes in recall, precision, and F-measure with respect to different numbers of clusters K when the K-means algorithm is used for skin segmentation. From the figure, it can be observed that the performance degrades when a smaller number of clusters is used and vice versa. The performance seems fine when $K > 10$. There is not much improvement when we increase the number of clusters to a number greater than 15. In addition, as the number of clusters increases, the time complexity increases and



(a) Performance under controlled environment.

(b) Performance under uncontrolled environment.

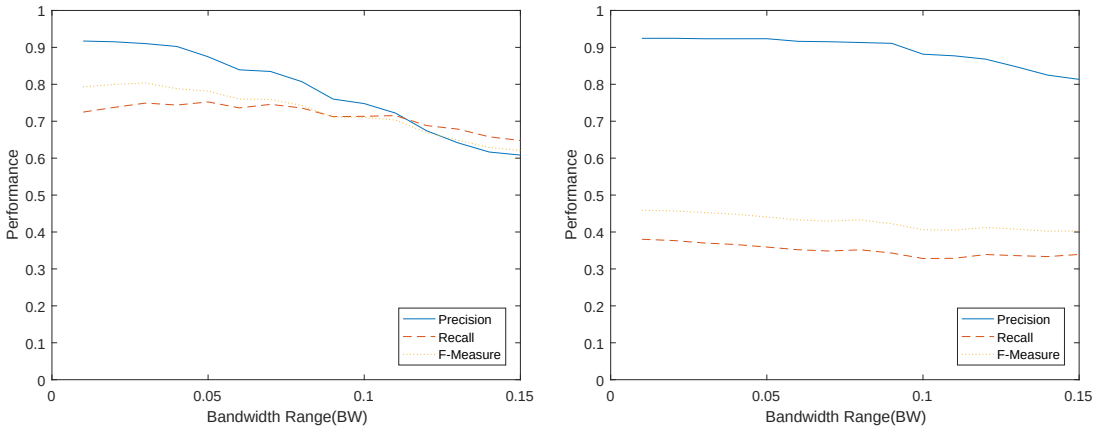
Figure 3.9: Performance analysis of the K-means clustering algorithm for different numbers of clusters K under two different environments.

vice versa, as shown in Table 3.1. This will slow down the algorithm when segmenting a large number of images. Therefore, the best choice for the number of clusters K in our case is for it to be selected in the range of 10 to 15.

On the other side, Figure 3.10 give a clear picture about the dependence of the mean shift algorithm performance on the bandwidth h. The performance is fair for lower bandwidths, as expected, but the computation time will be very high and vice versa, as shown in Table

Table 3.1: K-means algorithm: average computation time for segmenting one image using different numbers of clusters K.

K	4	8	10	12	16
Average computation time in seconds	2.56	3.64	4.24	4.78	5.98



(a) Performance under controlled environment.

(b) Performance under uncontrolled environment.

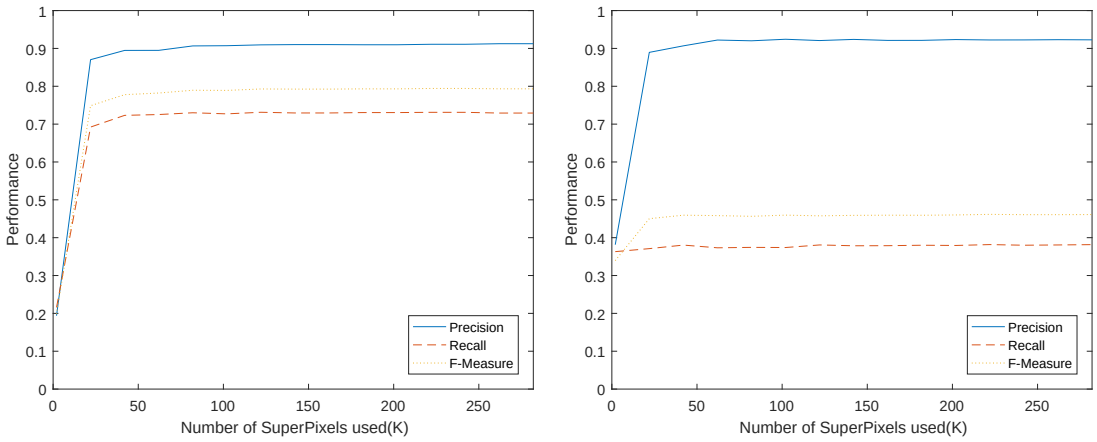
Figure 3.10: Performance analysis of the mean shift clustering algorithm for different bandwidths h under two different environments.

3.2. The bandwidth can be selected in the range of 0.03 to 0.06 to obtain good results with reasonable time complexity. Overall, the results have shown that the mean shift algorithm works better, in color-based skin segmentation, than the K-means algorithm in terms of both accuracy and time complexity.

Similar to the k-means algorithms, the performance of the SLIC increases as the number of superpixels K increases until it reaches a constant value around $K = 100$, as shown in

Table 3.2: Mean shift algorithm: average computation time for segmenting one image using different bandwidths h.

h	0.01	0.03	0.05	0.10	0.15
Average computation time in seconds	20.61	3.73	3.16	1.21	0.41



(a) Performance under controlled environment.

(b) Performance under uncontrolled environment.

Figure 3.11: Performance analysis of the SLIC clustering algorithm for different numbers of superpixels K under two different environments.

Figure 3.11. As can be seen from the figure, the performance is very close to the mean shift performance, yet consuming very little time even though we have used very large numbers of clusters compared to the numbers used in the case of the k-means, as shown in Table 3.3. This give the SLIC algorithm a great advantage as we can increase the number of clusters up to $K = 300$ with a minimal time consumption and very comparable performance.

Table 3.3: SLIC algorithm: average computation time for segmenting one image using different numbers of clusters K.

K	100	150	200	250	300
Average computation time in seconds	0.45	0.86	0.54	0.95	0.52

4.0 CONCLUSION AND FUTURE WORK

In this work, we have discussed the K-means, the mean shift, and the SLIC data clustering algorithms theoretically and experimentally. Using four measures, we have evaluated their performance in the segmentation of human skin based on color. Our method begins by clustering images using these algorithms and then segmenting the clustered regions that occupy skin. Pixels in the clusters are classified as skin or non-skin using the Kovac model.

We have used two databases the SFA database (controlled environment) and our database (uncontrolled environment) for the sake of evaluation. We have found that on average the mean shift algorithm performs batter that the other two algorithms on all four measures. In terms of computational cost, we have observed that the SLIC algorithm is the least complex algorithm resulting in faster clustering. We have found that the K-means algorithm has a good performance when the number of clusters K is between 10 and 15. On the contrary, we have found that the mean shift algorithm has good performance when the bandwidth h is between 0.03 and 0.06. The SLIC algorithm reaches its maximum performance at around $k = 100$ and the number of clusters can be increased to $K = 300$ without introducing a substantial amount of time.

Usually, the K-means and the mean shift algorithms are used to improve other algorithms, such as PCA [31]. The performance of the three algorithms can be improved by selecting another skin classification model such as the one shown in [32] and by trying other color spaces like YCbCr. In addition, the performance of the K-means algorithm can be improved by considering the color of each pixel and its position. Then, a color image is represented in a 5D space, which could be represented as (R, G, B, X, Y).

Also, it can be shown that the performance of the K-means and SLIC algorithms are quantitatively very sensitive to the initial means, by tuning the number of clusters K in the

K-means algorithm and the bandwidth h in the mean shift algorithm to make them equal. That can be achieved by running the mean shift algorithm for a certain h and selecting K to be equal to the number of clusters in the mean shift algorithm. Then, the performance of the mean shift algorithm can be compared with multiple runs of the K-means algorithm. In conclusion, we should mention that our codes are not optimized for the sake of speed.

BIBLIOGRAPHY

[1] D. B. King and M. Wertheimer, *Max Wertheimer and gestalt theory*. Transaction Publishers, 2005.

[2] X. Ren and J. Malik, "Learning a classification model for segmentation," in *International Conference on Computer Vision (ICCV)*, IEEE, 2003.

[3] Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," in *The Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June 2015.

[4] T. R. Reed and J. H. Dubuf, "A review of recent texture segmentation and feature extraction techniques," *CVGIP: Image Understanding*, vol. 57, no. 3, pp. 359–372, 1993.

[5] K. E. Van de Sande, J. R. Uijlings, T. Gevers, and A. W. Smeulders, "Segmentation as selective search for object recognition," in *International Conference on Computer Vision (ICCV)*, IEEE, 2011.

[6] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[7] I. Endres and D. Hoiem, "Category independent object proposals," in *European Conference on Computer Vision (ECCV)*, Springer, 2010.

[8] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004.

[9] S. Asteriadis, N. Nikolaidis, A. Hajdu, and I. Pitas, "An eye detection algorithm using pixel to edge information," in *Second International Symposium on Communications, Control and Signal Processing*, 2006.

[10] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Oakland, CA., 1967.

[11] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A K-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[12] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[13] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[14] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. SÃijsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[15] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, S. Y. Philip, *et al.*, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, 2008.

[16] D. E. Ilea and P. F. Whelan, "Color image segmentation using a spatial K-means clustering algorithm," 2006.

[17] Q. Zhang, Y. Chi, and N. He, "Color image segmentation based on a modified K-means algorithm," in *Proceedings of the 7th International Conference on Internet Multimedia Computing and Service*, ACM, 2015.

[18] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32–40, 1975.

[19] R. P. Duin, A. L. Fred, M. Loog, and E. Pękalska, "Mode seeking clustering by KNN and mean shift evaluated," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, Springer, 2012.

[20] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[21] L. Zhou and Z. Yang, "Salient region detection based on spatial and background priors," in *2014 IEEE International Conference on Information and Automation (ICIA)*, pp. 262–266, 2014.

[22] J. Sun, H. Lu, and X. Liu, "Saliency region detection based on markov absorption probabilities," *IEEE Transactions on Image Processing*, vol. 24, pp. 1639–1649, May 2015.

[23] Y. Ozasa, N. Enami, and Y. Ariki, "Color saliency for object identification," in *2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*, pp. 1–5, Jan 2015.

[24] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin segmentation using color pixel classification: analysis and comparison," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 1, pp. 148–154, 2005.

[25] Y. qiang Chen, C. Wu, and B. Yu, "The skin color segmentation algorithm based on the Cr* A* B color space," *International Journal of Control and Automation*, vol. 9, no. 5, pp. 255–262, 2016.

[26] J. Kovac, P. Peer, and F. Solina, "Human skin color clustering for face detection," vol. 2, 2003.

[27] J. P. B. Casati, D. R. Moraes, and E. L. L. Rodrigues, "SFA: a human skin image database based on FERET and AR facial images," in *IX workshop de Visao Computational, Rio de Janeiro*, 2013.

[28] L. Kaufman and P. Rousseeuw, *Clustering by means of medoids*. North-Holland, 1987.

[29] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[30] Y. Ukrainitz and B. Sarel, "Mean shift theory and applications."

[31] L. I. Smith *et al.*, "A tutorial on principal components analysis," *Cornell University, USA*, vol. 51, no. 52, p. 65, 2002.

[32] G. Osman, M. S. Hitam, and M. N. Ismail, "Enhanced skin colour classifier using RGB ratio model," *arXiv Preprint arXiv:1212.2692*, 2012.