COLOUR BASED SEGMENTATION USING K-MEANS CLUSTERING

(Using Matlab)

A COURSE PROJECT REPORT

By

SHRIDHARSHAN V.A.K
(RA2011003010180)
GUNNU JAI RAJ
(RA2011003010171)

Under the guidance of

Mr.S. SAMINATHAN

In partial fulfillment for the Course

of

18CSE353T - Digital Image Processing

in

Computer Science & Engineering

FACULTY OF ENGINEERING AND

TECHNOLOGY SRM INSTITUTE OF SCIENCE

AND TECHNOLOGY

Kattankulathur, Chengalpattu District

APRIL 2023

COLLEGE OF ENGINEERING & TECHNOLOGY
SRM INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this project report **" Colour based segmentation using k-means clustering"** is the bonafide work of **" Shridharshan v.a.k (RA2011003010180) and Gunnu jai raj (RA2011003010171)"** of III Year/VI Sem B.tech(CSE) who carried out the mini project work under my supervision for the course **18CSE353T- Digital Image Processing** in SRM Institute of Science and Technology during the academic year 2022-2023 (Even sem).

SIGNATURE

S.Saminathan
Assistant Professor (Senior Grade)
Department of Computing Technologies

CONTRIBUTION TABLE	
NAME & REGNO.	**CONTRIBUTION**
SHRIDHARSHAN V.A.K **RA2011003010180**	• Idea and suggestions for group project • Literature analysis • Preparation and planning for review presentation • Preperation and planning for oral presentation • Preperation, planning and writing of report.
GUNNU JAI RAJ **RA2011003010171**	Literature search

TABLE OF CONTENTS

PROBLEM DEFINITION :

Clustering is a way to separate groups of objects. K-means clustering treats each object as having a location in space. It finds partitions such that objects within each cluster are as close to each other as possible, and as far from objects in other clusters as possible. You can use the *imsegkmeans* function to separate image pixels by value into clusters within a colour space. This example performs k-means clustering of an image in the RGB and L*a*b* colour spaces to show how using different colour spaces can improve segmentation results.

In this project we'll be trying to separate the blue nuclei from an image of tissue stained with hematoxylin and eosin (H&E). We'll be classifying the RGB colors by K-means clustering and then classify them in a*b* color space. We then create image which is segmented into colors finally giving out the resulted blue nuclei. In this project we apply K-means clustering and segment the given image in RGB format and then convert it into a*b* segmentation then classify the image and produce the required image as output, in this case we apply k-means clustering to separate blue nuclei from an image of tissue stained with hematoxylin and eosin (H&E).

EXPLANATION OF THE PROBLEM :

In this project we'll be trying to separate the blue nuclei from an image of tissue stained with hematoxylin and eosin (H&E). We'll be classifying the RGB colors by K-means clustering and then classify them in a*b* color space. We then create image which is segmented into colors finally giving out the resulted blue nuclei. In this project we apply K-means clustering and segment the given image in RGB format and then convert it into a*b* segmentation then classify the image and produce the required image as output, in this case we apply k-means clustering to separate blue nuclei from an image of tissue stained with hematoxylin and eosin (H&E).

H&E Image

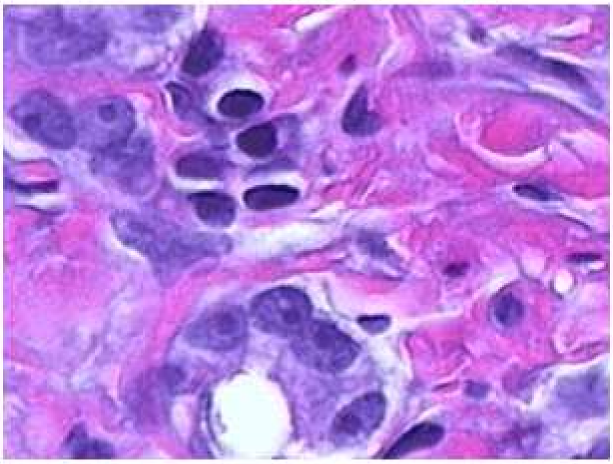
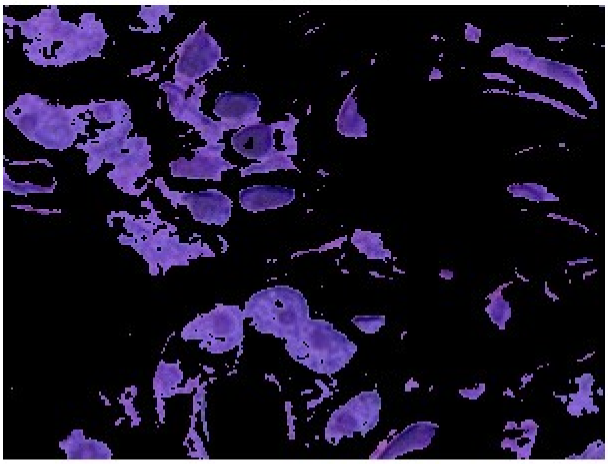


Image courtesy of Alan Partin, Johns Hopkins University

Blue Nuclei



STEPS INVOLVED :

1. Classify Colors in RBG Color Space Using K-Means Clustering.
2. Convert Image from RGB Color Space to L*a*b* Color Space.
3. Classify Colors in a*b* Space Using K-Means Clustering.
4. Create Images that Segment H&E Image by Color.
5. Segment Nuclei.

DESIGN TECHNIQUE :

RGB -

The RGB color space represents images as an m-by-n-by-3 numeric array whose elements specify the intensity values of the red, green, and blue color channels. The range of numeric values depends on the data type of the image.

For single or double arrays, RGB values range from [0, 1].

For uint8 arrays, RGB values range from [0, 255].

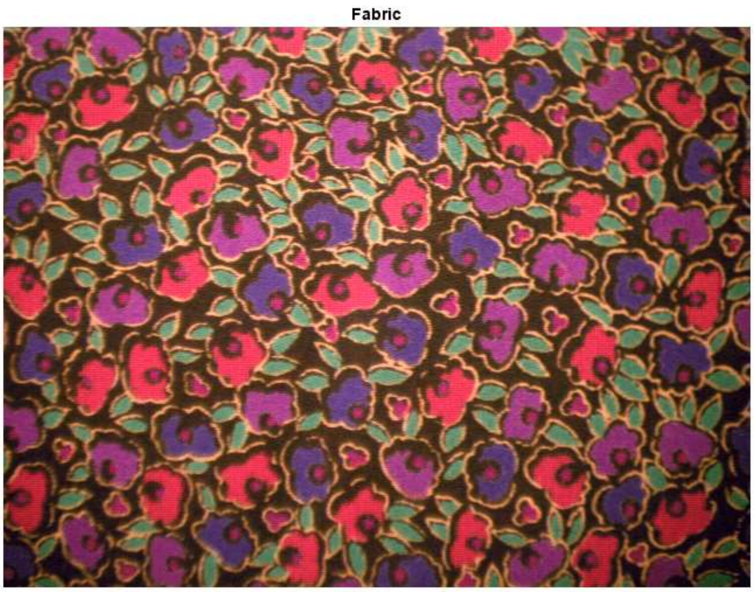
For uint16 arrays, RGB values range from [0, 65535].

The toolbox supports variations of the RGB color space.

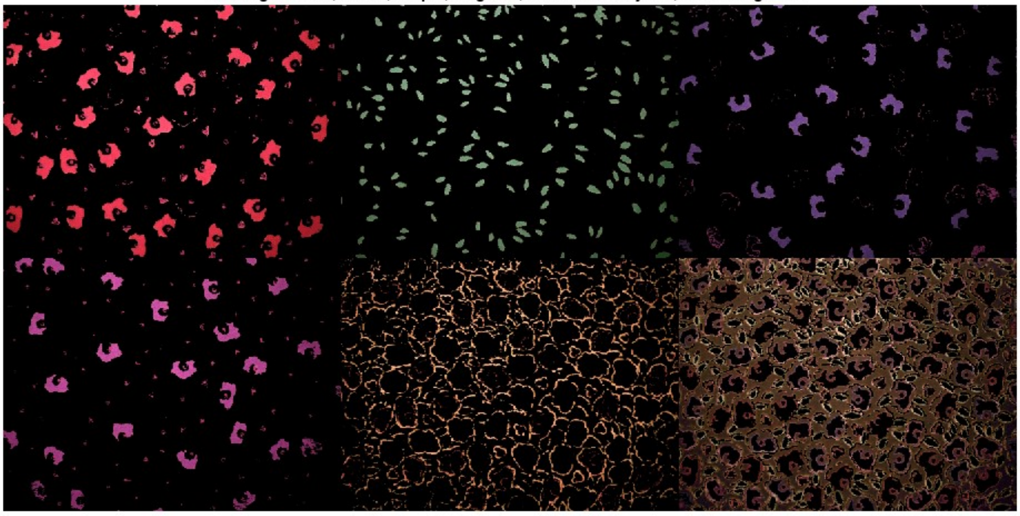
L*a*b* Color Space -

The L*a*b* colorspace (also known as CIELAB or CIE L*a*b*) enables you to quantify these visual differences. The L*a*b* color space is derived from the CIE XYZ tristimulus values. The L*a*b* space consists of a luminosity 'L*' or brightness layer, chromaticity layer 'a*' indicating where color falls along the red-green axis, and chromaticity layer 'b*' indicating where the color falls along the blue-yellow axis. Your approach is to choose a small sample region for each color and to calculate each sample region's average color in 'a*b*' space. You will use these color markers to classify each pixel.

EXAMPLE -



Montage of Red, Green, Purple, Magenta, and Yellow Objects, and Background



IMSEGKMEANS :

K-means clustering based image segmentation

Syntax -

L = imsegkmeans(I,k)

[L,centers] = imsegkmeans(I,k)

L = imsegkmeans(I,k,Name,Value)

Description

L = imsegkmeans(I,k) segments image I into k clusters by performing k-means clustering and returns the segmented labeled output in L.

[L,centers] = imsegkmeans(I,k) also returns the cluster centroid locations, centers.

L = imsegkmeans(I,k,Name,Value) uses name-value arguments to control aspects of the k-means clustering algorithm.

The function yields reproducible results. The output does not vary across multiple runs given the same input arguments.

The imsegkmeans function accepts input images in all supported color spaces. Using a different color space generates different results. If you do not receive satisfactory results for an input image, consider trying an alternative color space. For more information about color spaces in MATLAB®, see Understanding Color Spaces and Color Space Conversion.

To perform k-means clustering on images of data type double, convert the image to data type single by using the im2single function. For applications requiring input data of type double, see the kmeans (Statistics and Machine Learning Toolbox) function.

ALGORITHM :

1. Read in the histology image "hestain.png" using the "imread" function.

2. Display the image using the "imshow" function and set the title of the image using the "title" function.

3. Add text to the top-right corner of the image using the "text" function to give credit to the source of the image.

4. Perform k-means clustering on the image using the "imsegkmeans" function with the specified number of colors.

5. Display the resulting labeled image using the "labeloverlay" function and set the title of the image to "Labeled Image RGB".

6. Convert the image from RGB to the CIELAB color space using the "rgb2lab" function.

7. Extract the "a" and "b" color channels from the image and convert them to single precision using the "im2single" function.

8. Perform k-means clustering on the "a*b*" channels using the "imsegkmeans" function with the specified number of colors and number of attempts.

9. Display the resulting labeled image using the "labeloverlay" function and set the title of the image to "Labeled Image a*b*".

10. Create binary masks for each cluster of pixels in the labeled image.

11. Use the binary masks to extract the pixels in each cluster and display them in separate images with titles indicating which cluster they belong to.

12. Extract the blue nuclei from the image by applying a binary mask to the "L" channel of the CIELAB image.

13. Generate a binary mask for the light blue pixels in the "L" channel using the "imbinarize" function.

14. Apply the binary mask for the light blue pixels to the blue cluster mask to exclude them.

15. Use the resulting mask to extract the blue nuclei from the original image.

16. Display the blue nuclei in an image with the title "Blue Nuclei".

Syntax -

L = imsegkmeans(I,k)

[L,centers] = imsegkmeans(I,k)

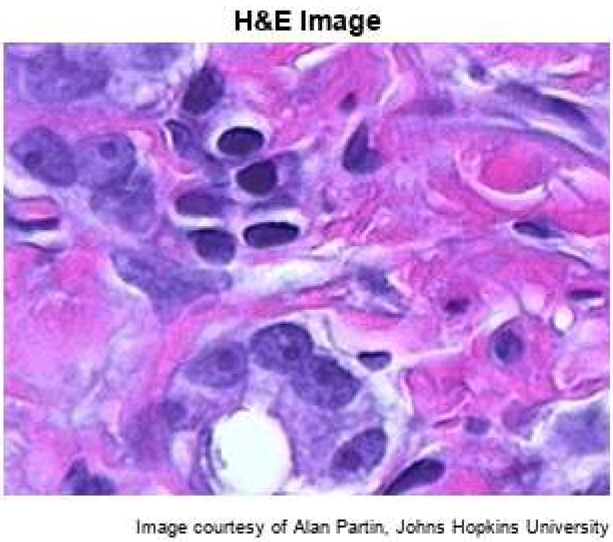
L = imsegkmeans(I,k,Name,Value)

IMPLEMENTATION :

Step 1: Read Image

Read in hestain.png, which is an image of tissue stained with hemotoxylin and eosin (H&E). This staining method helps pathologists distinguish between tissue types that are stained blue-purple and pink.

```
he = imread("hestain.png");
imshow(he)
title("H&E Image")
text(size(he,2),size(he,1)+15, ...
     "Image courtesy of Alan Partin, Johns Hopkins University", ...
     FontSize=7,HorizontalAlignment="right")
```



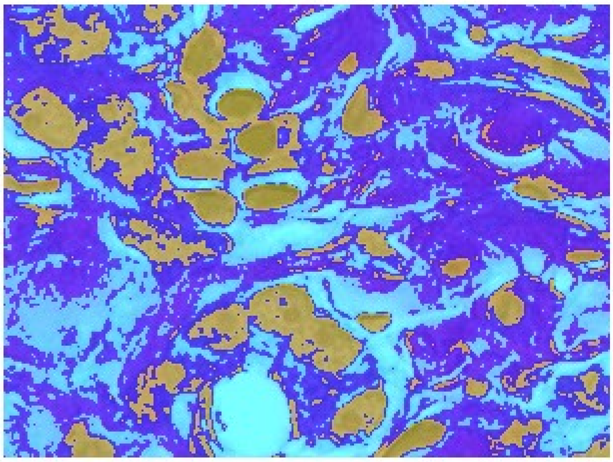
Step 2: Classify Colors in RBG Color Space Using K-Means Clustering

Segment the image into three regions using k-means clustering in the RGB color space. For each pixel in the input image, the imsegkmeans function returns a label corresponding to a cluster.

Display the label image as an overlay on the original image. The label image incorrectly groups white, light blue-purple, and light pink regions together. Because the RGB color space combines brightness and color information within each channel (red, green, blue), lighter versions of two different colors are closer together and more challenging to segment than darker versions of the same two colors.

```
numColors = 3;
L = imsegkmeans(he,numColors);
B = labeloverlay(he,L);
imshow(B)
title("Labeled Image RGB")
```

Labeled Image RGB



Step 3: Convert Image from RGB Color Space to L*a*b* Color Space

The L*a*b* color space separates image luminosity and color. This makes it easier to segment regions by color, independent of lightness. The color space is also more consistent with human visual perception of the distinct white, blue-purple, and pink regions in the image.

The L*a*b* color space is derived from the CIE XYZ tristimulus values. The L*a*b* space consists of the luminosity layer L*, the chromaticity layer a* that indicates where a color falls along the red-green axis, and the chromaticity layer b* that indicates where a color falls along the blue-yellow axis. All of the color information is in the a* and b* layers.

Convert the image to the L*a*b* color space by using the rgb2lab function.

```
lab_he = rgb2lab(he);
```

Step 4: Classify Colors in a*b* Space Using K-Means Clustering

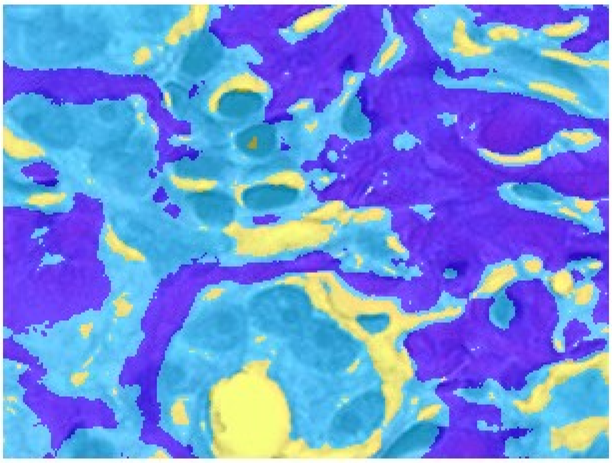
To segment the image using only color information, limit the image to the a* and b* values in lab_he. Convert the image to data type single for use with the imsegkmeans function. Use the imsegkmeans function to separate the image pixels into three clusters. Set the value of the NumAttempts name-value argument to repeat clustering three times with different initial cluster centroid positions to avoid fitting to a local minimum.

```
ab = lab_he(:,:,2:3);
ab = im2single(ab);
pixel_labels = imsegkmeans(ab,numColors,NumAttempts=3);
```

Display the label image as an overlay on the original image. The new label image more clearly separates the white, blue-purple, and pink stained tissue regions.

```
B2 = labeloverlay(he,pixel_labels);
imshow(B2)
title("Labeled Image a*b*")
```

Labeled Image a*b*

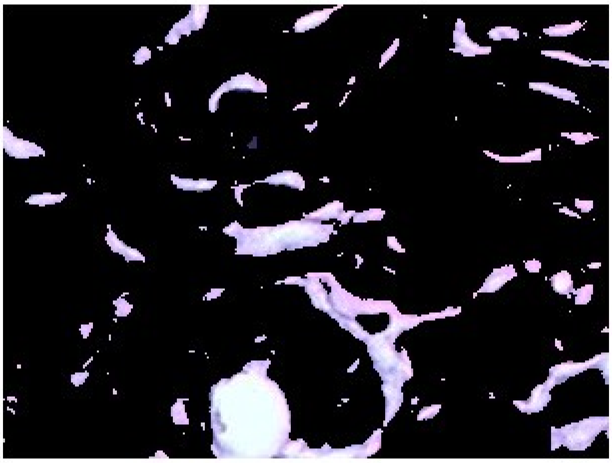


Step 5: Create Images that Segment H&E Image by Color

Using pixel_labels, you can separate objects in the original image hestain.png by color, resulting in three masked images.

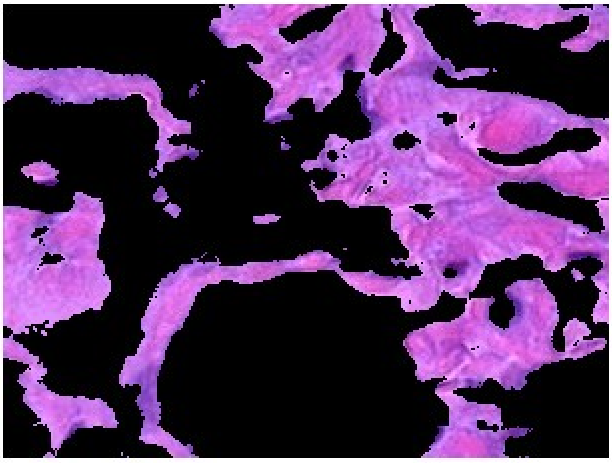
```
mask1 = pixel_labels == 1;
cluster1 = he.*uint8(mask1);
imshow(cluster1)
title("Objects in Cluster 1");
```

Objects in Cluster 1



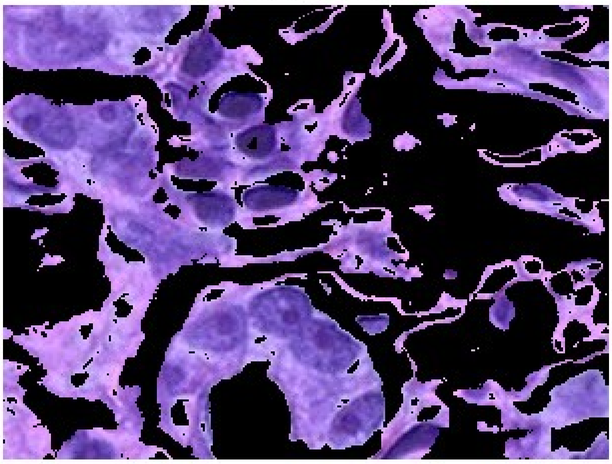
```
mask2 = pixel_labels == 2;
cluster2 = he.*uint8(mask2);
imshow(cluster2)
title("Objects in Cluster 2");
```

Objects in Cluster 2



```
mask3 = pixel_labels == 3;
cluster3 = he.*uint8(mask3);
imshow(cluster3)
title("Objects in Cluster 3");
```

Objects in Cluster 3



Step 6: Segment Nuclei

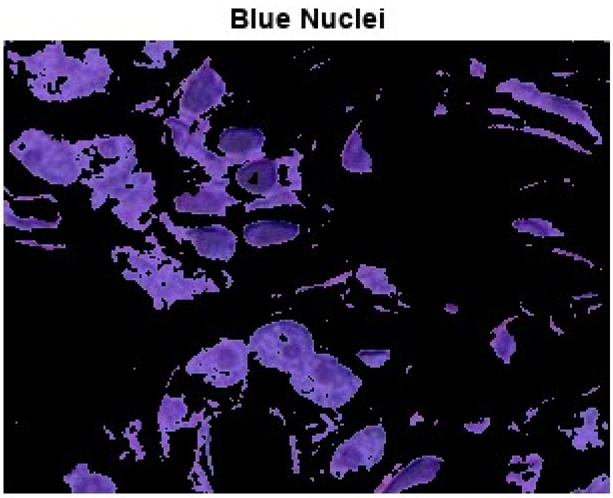
Cluster 3 contains only the blue objects. Notice that there are dark and light blue objects. You can separate dark blue from light blue using the L* layer in the L*a*b* color space. The cell nuclei are dark blue.

The L* layer contains the brightness value of each pixel. Extract the brightness values of the pixels in this cluster and threshold them with a global threshold by using the imbinarize function. The mask idx_light_blue gives the indices of light blue pixels.

```
L = lab_he(:,:,1);
L_blue = L.*double(mask3);
L_blue = rescale(L_blue);
idx_light_blue = imbinarize(nonzeros(L_blue));
```

Copy the mask of blue objects, mask3, and then remove the light blue pixels from the mask. Apply the new mask to the original image and display the result. Only dark blue cell nuclei are visible.

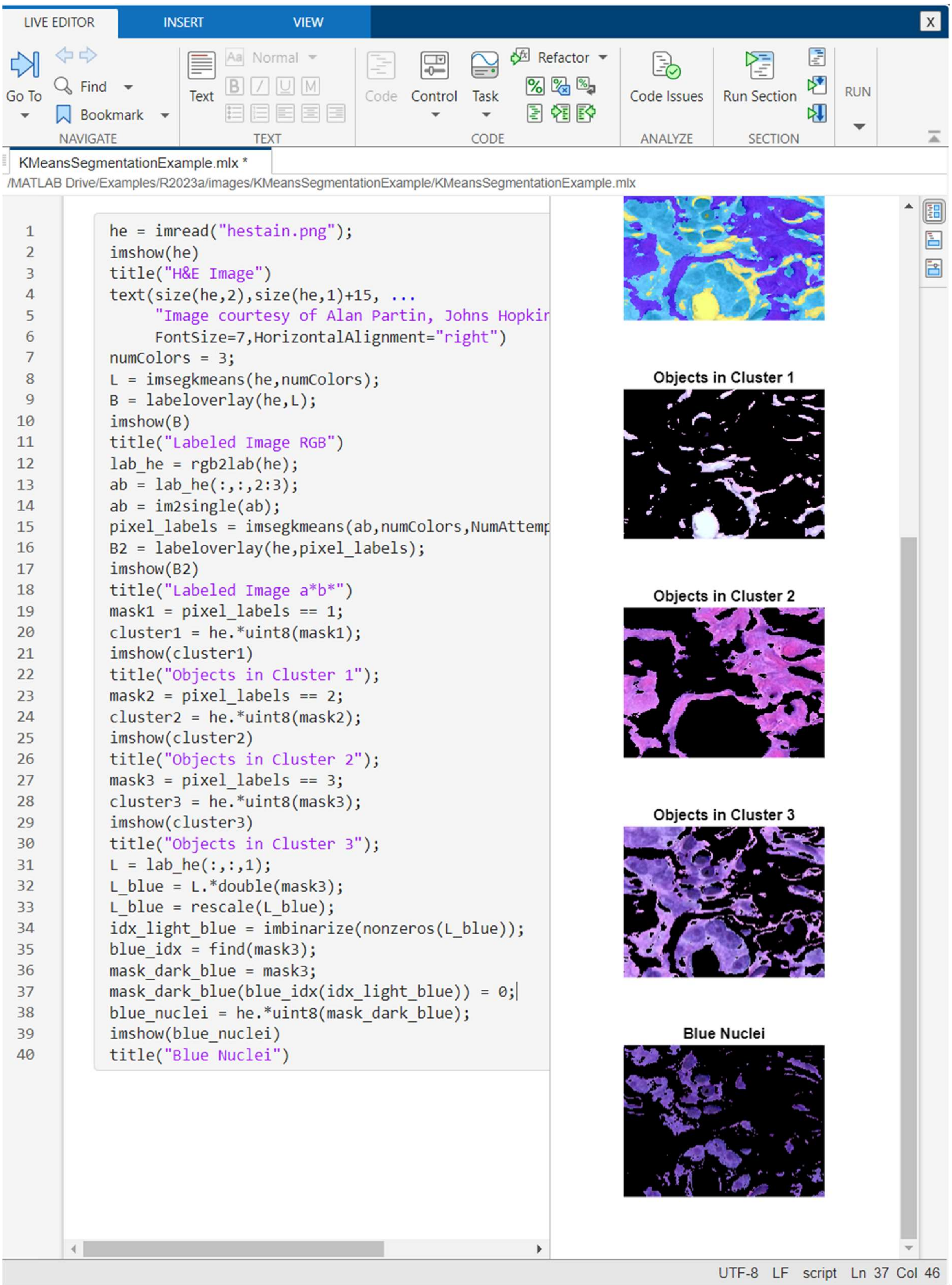
```
blue_idx = find(mask3);
mask_dark_blue = mask3;
mask_dark_blue(blue_idx(idx_light_blue)) = 0;

blue_nuclei = he.*uint8(mask_dark_blue);
imshow(blue_nuclei)
title("Blue Nuclei")
```



CODE :

```
he = imread("hestain.png");
imshow(he)
title("H&E Image")
text(size(he,2),size(he,1)+15, ...
     "Image courtesy of Alan Partin, Johns Hopkins University", ...
     FontSize=7,HorizontalAlignment="right")
numColors = 3;
L = imsegkmeans(he,numColors);
B = labeloverlay(he,L);
imshow(B)
title("Labeled Image RGB")
lab_he = rgb2lab(he);
ab = lab_he(:,:,2:3);
ab = im2single(ab);
pixel_labels = imsegkmeans(ab,numColors,NumAttempts=3);
B2 = labeloverlay(he,pixel_labels);
imshow(B2)
title("Labeled Image a*b*")
mask1 = pixel_labels == 1;
cluster1 = he.*uint8(mask1);
imshow(cluster1)
title("Objects in Cluster 1");
mask2 = pixel_labels == 2;
cluster2 = he.*uint8(mask2);
imshow(cluster2)
title("Objects in Cluster 2");
mask3 = pixel_labels == 3;
cluster3 = he.*uint8(mask3);
imshow(cluster3)
title("Objects in Cluster 3");
L = lab_he(:,:,1);
L_blue = L.*double(mask3);
L_blue = rescale(L_blue);
idx_light_blue = imbinarize(nonzeros(L_blue));
blue_idx = find(mask3);
mask_dark_blue = mask3;
mask_dark_blue(blue_idx(idx_light_blue)) = 0;
blue_nuclei = he.*uint8(mask_dark_blue);
imshow(blue_nuclei)
title("Blue Nuclei")
```

OUTPUT :


LIVE EDITOR
INSERT
VIEW
Go To
Find
Bookmark
NAVIGATE
Normal
Text
TEXT
Code
Control
Task
Refactor
CODE
Code Issues
ANALYZE
Run Section
SECTION
RUN
KMeansSegmentationExample.mlx *
/MATLAB Drive/Examples/R2023a/images/KMeansSegmentationExample/KMeansSegmentationExample.mlx
he = imread("hestain.png");
imshow(he)
title("H&E Image")
text(size(he,2),size(he,1)+15, ...
"Image courtesy of Alan Partin, Johns Hopkir
FontSize=7,HorizontalAlignment="right")
numColors = 3;
L = imsegkmeans(he,numColors);
B = labeloverlay(he,L);
imshow(B)
title("Labeled Image RGB")
lab_he = rgb2lab(he);
ab = lab_he(:,:,2:3);
ab = im2single(ab);
pixel_labels = imsegkmeans(ab,numColors,NumAttemp
B2 = labeloverlay(he,pixel_labels);
imshow(B2)
title("Labeled Image a*b*")
mask1 = pixel_labels == 1;
cluster1 = he.*uint8(mask1);
imshow(cluster1)
title("Objects in Cluster 1");
mask2 = pixel_labels == 2;
cluster2 = he.*uint8(mask2);
imshow(cluster2)
title("Objects in Cluster 2");
mask3 = pixel_labels == 3;
cluster3 = he.*uint8(mask3);
imshow(cluster3)
title("Objects in Cluster 3");
L = lab_he(:,:,1);
L_blue = L.*double(mask3);
L_blue = rescale(L_blue);
idx_light_blue = imbinarize(nonzeros(L_blue));
blue_idx = find(mask3);
mask_dark_blue = mask3;
mask_dark_blue(blue_idx(idx_light_blue)) = 0;
blue_nuclei = he.*uint8(mask_dark_blue);
imshow(blue_nuclei)
title("Blue Nuclei")
Objects in Cluster 1
Objects in Cluster 2
Objects in Cluster 3
Blue Nuclei
UTF-8 LF script Ln 37 Col 46

RESULT :

Thus we have implemented the k-means algorithm and using which we have successfully separated the blue nuclei from an image of tissue stained with hematoxylin and eosin (H&E) by segmenting the given image in RGB format and then converted it into a*b* segmentation.

DISCUSSION :

This code demonstrates the application of k-means clustering in image segmentation, which is a common technique used in image processing and computer vision. In particular, it shows how k-means clustering can be used to segment an image into distinct clusters based on the colors or intensities of the pixels.

The code also illustrates the use of different color spaces for image analysis, such as RGB and LAB color spaces. The RGB color space is commonly used for displaying images on digital screens, while the LAB color space is designed to be more perceptually uniform and is often used for image analysis and manipulation.

Furthermore, the code demonstrates how to use binary masks to extract specific objects or features from an image, such as the blue nuclei in a histological sample. This can be useful for quantifying different features in an image and for identifying patterns or relationships that may not be apparent through visual inspection alone.

Overall, this code provides a useful example of how image processing techniques can be applied to histological images to aid in the analysis of tissue samples. However, it should be noted that the performance of the k-means clustering algorithm may be sensitive to the specific parameters chosen, and may require tuning for optimal results on different datasets.

CONCLUSION :

Thus we have implemented the k-means algorithm and using which we have successfully separated the blue nuclei from an image of tissue stained with hematoxylin and eosin (H&E) by segmenting the given image in RGB format and then converted it into a*b* segmentation. By this project we have learnt about colour based segmentation using k-means algorithm and we have successfully practiced using the *imsegkmeans* function.

This code can be useful for researchers and medical professionals who need to analyze histological images to identify and quantify specific features of interest, such as cells or tissue structures. By using image processing techniques like those demonstrated in this code, researchers can obtain more accurate and consistent results compared to manual analysis methods, and can potentially identify new features or patterns that may be missed by human observers.

REFERENCES :

[1] Arthur, David, and Sergei Vassilvitskii. "K-Means++: The Advantages of Careful Seeding." In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 1027–35. SODA '07. USA: Society for Industrial and Applied Mathematics, 2007.

[2] Understanding Color Spaces and Color Space Conversion by *matlab.*

[3] Color-Based Segmentation Using the L*a*b* Color Space by *matlab.*