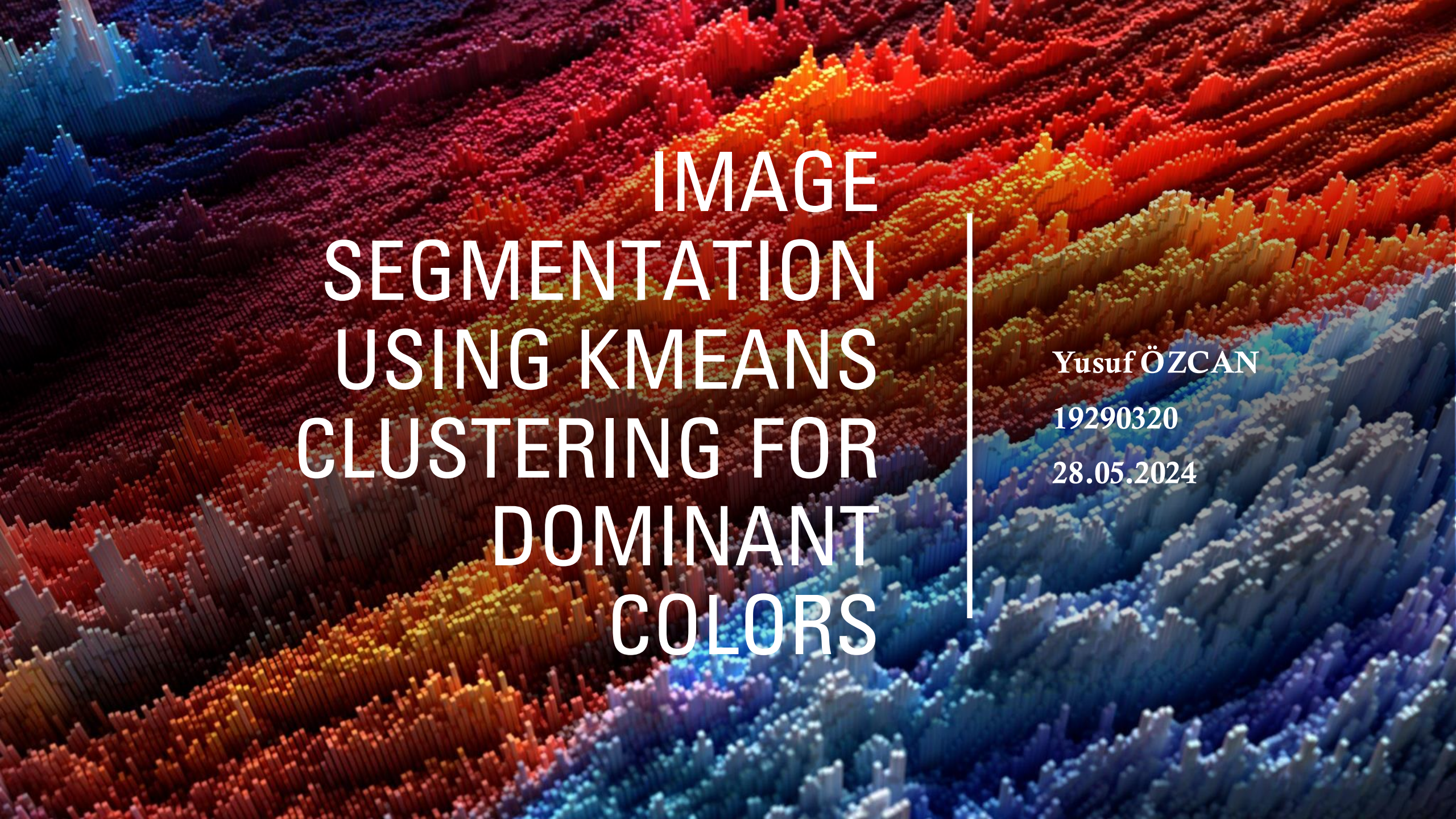

IMAGE SEGMENTATION USING KMEANS CLUSTERING FOR DOMINANT COLORS

Yusuf ÖZCAN

19290320

28.05.2024

INTORDUCTION

- **Objective :** To demonstrate image segmentation using KMeans clustering.
- **Process :** Load an image, convert it to RGB format, cluster the pixels, and generate a segmented image.
- **Tools Used :** Python, OpenCV, Numpy, Matplotlib, Scikit-learn

IMPORTING NECESSARY LIBRARIES

```
import matplotlib.pyplot as plt
import numpy as np
import cv2
from sklearn.cluster import KMeans
```

- **Matplotlib :** For visualization
- **Numpy :** For data manipulation
- **OpenCV :** For image processing
- **Scikit_learn :** For clustering

LOADING AND CONVERTING THE IMAGE

```
im = cv2.imread('resim.jpg')
im = cv2.cvtColor(im, cv2.COLOR_BGR2RGB)
original_shape = im.shape
print(im.shape)
plt.imshow(im)
plt.show()
```

- **Loading : 'cv2.imread('resim.jpg')'**
- **Converting : 'cv2.cvtColor(im, cv2.COLOR_BGR2RGB)'**
- **Displaying : 'plt.imshow(im)'**

RESHAPING THE IMAGE PIXELS

```
all_pixels = im.reshape((-1, 3))
print(all_pixels.shape)
```

- **Reshaping :** Converts image to a 2D array of pixels.
- **Purpose :** Prepare data for KMeans clustering.

DETERMINING DOMINANT COLORS

```
dominant_colors = 4
km = KMeans(n_clusters=dominant_colors)
km.fit(all_pixels)
centers = km.cluster_centers_
print(centers)
centers = np.array(centers, dtype='uint8')
print(centers)
```

- **KMeans Clustering : 'n_cluster=4'**
- **Fitting Data : 'km.fit(all_pixels)'**
- **Cluster Centers : 'km.cluster_centers_'**

VISUALIZING DOMINANT COLORS

```
i = 1
plt.figure(0, figsize=(8, 2))
colors = []
for each_col in centers:
    plt.subplot(1, 4, i)
    plt.axis("off")
    i += 1
    colors.append(each_col)
    a = np.zeros((100, 100, 3), dtype='uint8')
    a[:, :, :] = each_col
    plt.imshow(a)
plt.show()
```

- **Subplots :** Displayed each dominant color in a subplot.
- **Colors Array :** Visual representation of dominant colors.

CREATING THE NEW SEGMENTED IMAGE

```
new_img = np.zeros((original_shape[0] * original_shape[1], 3), dtype='uint8')
for ix in range(new_img.shape[0]):
    new_img[ix] = colors[km.labels_[ix]]
new_img = new_img.reshape((original_shape[0], original_shape[1], 3))
plt.imshow(new_img)
plt.show()
```

- **New Imsge Array : 'np.zeros((original_shape[0] * original_shape[1], 3), dtype="uint8')'**
- **Assigning Colors :** Each pixel is assigned a dominant color.
- **Reshape and Display : 'new_img.reshape((original_shape[0], original_shape[1], 3))'**

CONCLUSION

- **Summary :** Performed image segmentation using KMeans clustering to identfy dominant colors.
- **Result :** Generated a segmented image where each segment is represented by one of the dominant colors.
- **Application :** Effective method for simple image segmentation and dominant color identification.