Assignment 3: Segmentation

Prepared by:

Adriana Bottega

Computer Engineer Student, Senior

CAD 4410 - Computer Vision

Prof. Muhammad Abid

02/23/2025

Florida Polytechnic University

CONTENTS

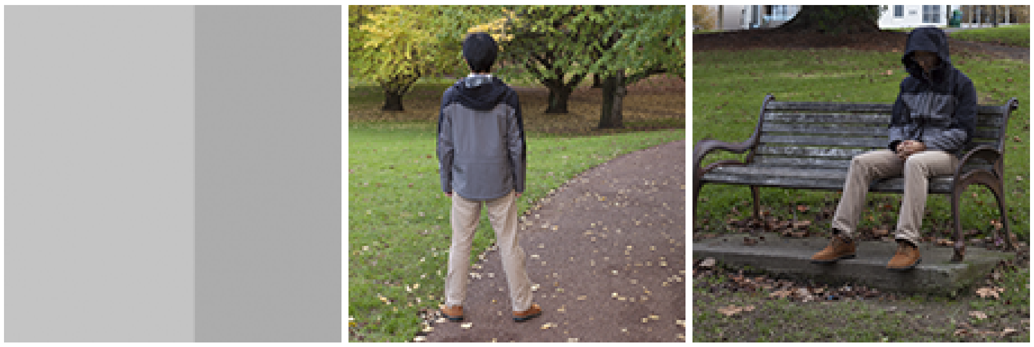
LIST OF FIGURES

LIST OF SOURCE CODES

1 INTRODUCTION

The methods analyzed in this report are Otsu Multiclass Thresholding, Binary Otsu Thresholding, and Mean Shift Filtering. Each of these methods have their own advantages and benefits as well as drawbacks. However, they are fundamental in object detection and medical imaging, among others.

- **Binary Otsu Thresholding** Binarize the image based on pixel intensities.
- **Multiclass Otsu Thresholding** Segment the image into different classes based on pixel intensities
- **Mean Shift Filtering** Groups pixels into regions of similar color and texture.



(a) Original image for Binary OTSU. **(b)** Original image for multi OTSU. **(c)** Original image for mean shift filtering.

Figure 1.1. Original images to which the aforementioned filters will be applied.

2 METHODS

2.1 BINARY OTSU THRESHOLDING

This method consists of processing the image histogram and segmentating the image according to the 2 peaks shown in the histogram (assuming that the image is bimodal). The threshold value is then computed as the middle value between the two peaks in the histogram. Pixel values greater than the threshold are replaced by black and pixels lower than the threshold are replaced by white.

1. Process the input image
2. Obtain image histogram (distribution of pixels)
3. Compute the threshold value T
4. Replace image pixels into white in those regions, where saturation is greater than T and into the black in the opposite cases.

Note: if the value determined as the threshold is incorrect, the variance of a class would be large.

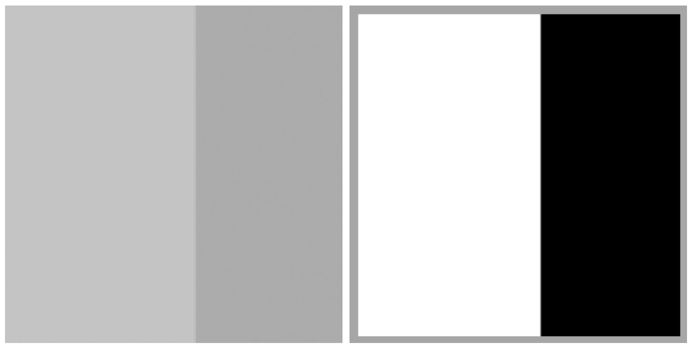

(a) Original image. **(b)** Binary OTSU Example

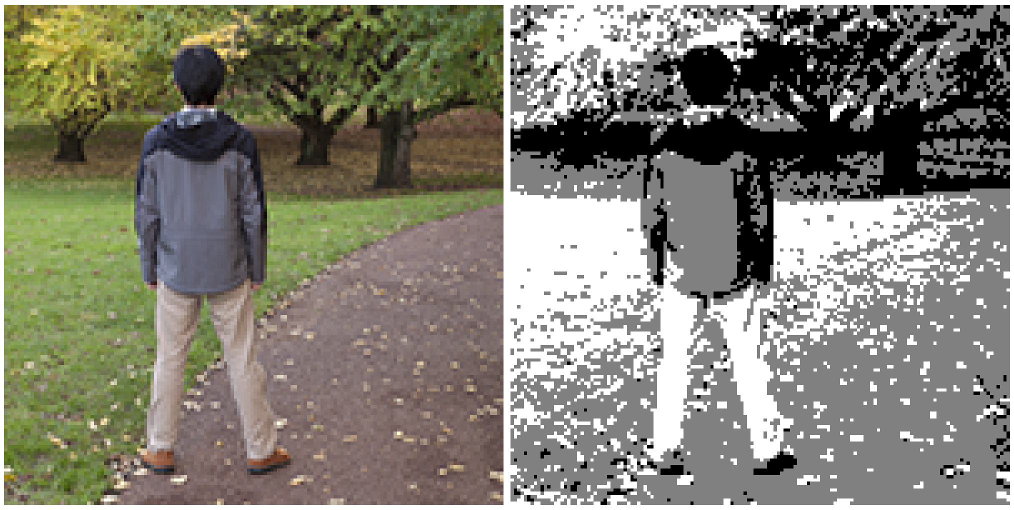
Figure 2.1. In Binary OTSU pixels are either foreground (white) or background (black).

*In Figure 2.1(b), a grey border is shown for displaying purposes

2.2 MULTI OTSU THRESHOLDING

Binary OTSU is effective when working with a bimodal image. However, when more peaks are displayed in the histogram it looses it effectiveness. Furthermore, simply using a binary threshold on the grayscale image, will not yield any good result since everything will be gray by then. Multi-OTSU Thresholding is an extension of Binary Otsu that allows various peaks in the histogram (segments) to be accounted for. Therefore, multiple threshold values are determined, respectively. The pixels of the input image are separated into several different classes due to the intensity of their gray levels.

1. Just like in Binary Otsu, Multi-Otsu examines the histogram
2. Separates pixels into three or more segments or classes based on their intensities.
3. Finds the optimal threshold values among the selected segments or classes.
4. Pixels are classified into an assigned segment based on the calculated thresholds.



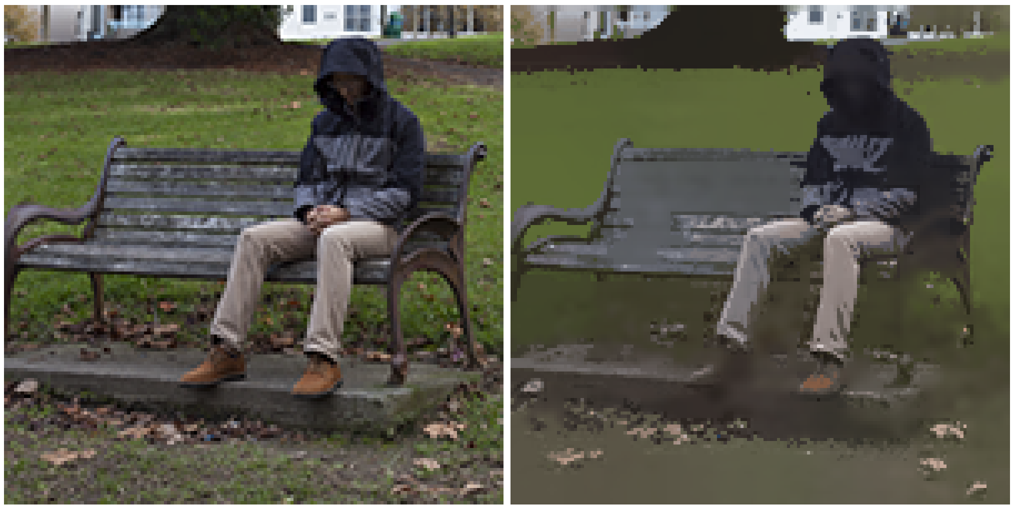
(a) Original image. **(b)** Multi-OTSU Example

Figure 2.2. In Multi-OTSU the image is segmented into a number of regions.

2.3 MEAN SHIFT SEGMENTATION

Mean shift segmentation, in contrast with both the aforementioned methods, segments images using clusters without specifying the number of clusters a priori.

For a randomly selected central point in the image, the neighbors of this point are analyzed and a new central point is determined based on the highest density point of the neighbors. It then slides the cental point onto this high density point and repeats the analysis. It iteratively does this until the spatial and the color (or grayscale) mean stops changing.



(a) Original image. **(b)** Mean Shift Filtering

Figure 2.3. In Mean Shift Filtering the cluster with greater mean values is found iteratively until there is no greater mean value.

3 SUMMARY OF EFFECTS

Method	Segmentation Type	Advantages	Drawbacks
Multi-Otsu	Intensity based	Separates regions on grayscale images	Limited to intensity differences, struggles with textures
Binary Otsu	Intensity based	Effective on bimodal images	Struggles with RGB, can not handle multi-class segmentation
Mean Shift Segmentation	Color-based clustering	Effective on color segmentation, denoising	Computationally expensive, loss of details

4 CODE RUN-THROUGH

Importing required libraries for use in the image pre-processing and segmentation methods.

- **cv2 (OpenCV):** Used for image reading, writing, and filtering.
- **numpy:** Used for numerical computations (e.g., convolution operations).
- **matplotlib.pyplot:** Used for plotting images.
- **skimage.filters:** Used for multi-Otsu thresholding.

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
from skimage.filters import threshold_multiotsu
```

Listing 1. Importing required libraries

Next we define what file names of the images to be processed, load them into our program with cv.imread(), and send them to the processing function.

```python
if __name__ == "__main__":
    image_files = ['OTSU Multiple Class-S01-150x150.png',
    ↪  'OTSU2class-edge_L-150x150.png', 'meanshift S00-150x150.png']
    image_list = []
    for img_file in image_files:
        img_path = "Assigment3/Images/" + img_file
        if "OTSU2" in img_file:
            image = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
        else:
            image = cv2.imread(img_path)
        image_list.append(image)
    processing(image_list)
```

Listing 2. main

In processing, we send each image to their respective method, gather the results, convert to RGB if needed, attach their respective titles, and display them.

```python
def processing(images):
    processed_images = [
        multiotsu(images[0]),  # Multi-Otsu
        otsu2class(images[1]),  # Otsu 2-Class
        mean_shift(images[2]),  # Mean Shift Filtering
    ]

    results = images + processed_images
    titles = ["Otsu Multiclass", "Binary Otsu", "Mean Shift"]

    for i in range(3):
        results[i] = cv2.cvtColor(image_list[i], cv2.COLOR_BGR2RGB)
        if i==2:
            results[i+3] = cv2.cvtColor(results[i+3], cv2.COLOR_BGR2RGB)
        plt.subplot(2,3,i+1), plt.imshow(results[i])
        plt.title("Original"), plt.xticks([]), plt.yticks([])
        plt.subplot(2,3,i+4), plt.imshow(results[i+3], cmap= 'gray')
        plt.imsave(f"{titles[i]}.png", results[i+3], cmap = 'gray')
        plt.title(titles[i]), plt.xticks([]), plt.yticks([])
    plt.show()
```

Listing 3. processing()

4.1 BINARY OTSU FUNCTION

Inside the Binary Otsu function, the cv2 function .threshold() is called. Its parameters are the image to be processed, threshold value, maximum value to use with the thresholding type, and the thresholding type. Then returns the computed threshold value.

```python
def otsu2class(image):
    """Applies Otsu's thresholding for binary (2-class) segmentation."""
    _, binary_image = cv2.threshold(image, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)
    return binary_image

```

Listing 4. otsu2class() function

4.2 MULTI-OTSU FUNCTION

In the Multi-Otsu function (multiotsu()), first the image is converted into grayscale. Then, the multiple thresholds are determined using the function threshold_multiotsu() inside the skimage.filter module. Finally, the segmented image is computed using the digitize function inside the numpy package, passing the thresholds as the bins. This segmented_image is returned.

```python
    def multiotsu(image):
    """Applies Multi-Otsu thresholding for multi-class segmentation."""
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    thresholds = threshold_multiotsu(gray_image)
    segmented_image = np.digitize(gray_image, bins=thresholds)
    return segmented_image
```

Listing 5. multiotsu() function

4.3 MEAN SHIFT METHOD

The mean shift function implemented the OpenCV function pyrMeanShiftFiltering() which returns the segmented image.

```python
def mean_shift(image, spatial_radius=15, color_radius=30):
    """Applies Mean Shift filtering to an image."""
    return cv2.pyrMeanShiftFiltering(image, sp=spatial_radius, sr=color_radius)
```

Listing 6. mean_shift() function

4.4 OUTPUT

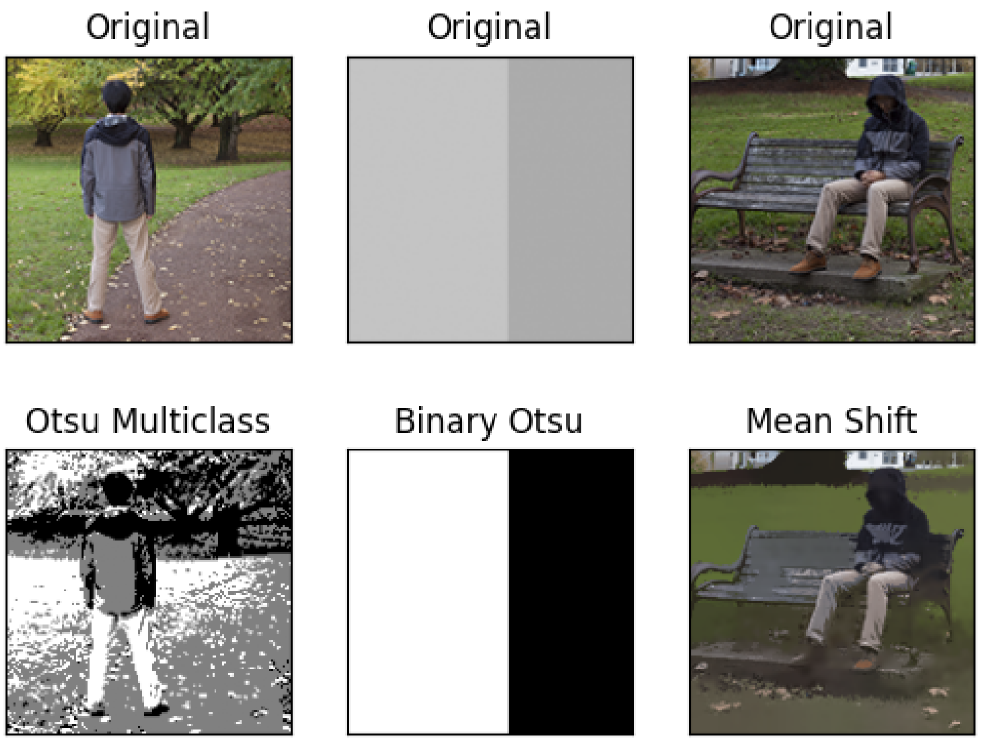


Figure 4.1. Output of the three methods

5 CONCLUSION

Each segmentation method has its strengths and weaknesses, making them suitable for different applications and uses:

- Otsu's Binary Thresholding works well for images with clear foreground and background but struggles with multi-class segmentation.
- Multi-Otsu Thresholding provides better segmentation for grayscale images with different intensity levels but is limited to grayscale.
- Mean Shift Filtering is effective for color segmentation and noise removal, but there is loss of details and is very computationally expensive.

The choice of segmentation purely depends on the input image and the desired output. There are multiple libraries and methods of implementing each segmentation method in python which allows for better and clearer results. The documentation, even though vague, was helpful in redacting this report.

REFERENCES

“Image thresholding.

“Multi-otsu thresholding.” *scikit-image*.

(2016). “Meanshift algorithm for the rest of us (python).

(2023). “Python — thresholding techniques using opencv — set-3 (otsu thresholding).” *GeeksforGeeks*.

Madrigal, R. (2022). “Image segmentation using thresholding and otsu’s method.” *Medium*.

S. Sakshi, M. A. (2020). “Otsu’s thresholding with opencv.” *LearnOpenCV*.

Uwe, K. (2007). “Mean shift filter.

Madrigal (2022) gee (2023) log (2016) (mul) S. Sakshi (2020) (thr) Uwe (2007)