Region and Edge Based Segmentation

Segmentation

Segmentation is the separation of one or more regions or objects in an image based on a discontinuity or a similarity criterion. A region in an image can be defined by its border (edge) or its interior, and the two representations are equal. There are prominently three methods of performing segmentation:

- Pixel Based Segmentation
- Region-Based Segmentation
- Edges based segmentation

Edges based segmentation

Edge-based segmentation contains 2 steps:

- **Edge Detection:** In edge detection, we need to find the pixels that are edge pixels of an object. There are many object detection methods such as Sobel operator, Laplace operator, Canny, etc.

1	**0**	**-1**
2	**0**	**-2**
1	**0**	**-1**

Sobel vertical Operator

+1	**2**	**1**
0	**0**	**0**
-1	**-2**	**-1**

Sobel Horizontal Operator

0	**-1**	**0**
-1	**4**	**-1**
0	**-1**	**0**

Negative Laplace Operator

- **Edge Linking:** In this step, we try to refine the edge detection by linking the adjacent edges and combine to form the whole object. The edge linking can be performed using any of the two methods below:
    - Local Processing: In this method, we used gradient and direction to link the neighborhood edges. If two edges have a similar direction vector then they can be linked.
    - Global processing: This method can be done using HOG transformation
- **Pros** :
    - This approach is similar to how the humans brain approaches the segmentation task.
    - Works well in images with good contrast between object and background.
- **Limitations:**
    - Does not work well on images with smooth transitions and low contrast.
    - Sensitive to noise.
    - Robust edge linking is not trivial and easy to perform.

Region-Based Segmentation

In this segmentation, we grow regions by recursively including the neighboring pixels that are similar and connected to the seed pixel. We use similarity measures such as differences in gray levels for regions with homogeneous gray levels. We use connectivity to prevent connecting different parts of the image.

There are two variants of region-based segmentation:

- **Top-down approach**
    - First, we need to define the predefined seed pixel. Either we can define all pixels as seed pixels or randomly chosen pixels. Grow regions until all pixels in the image belongs to the region.

- **Bottom-Up approach**
    - Select seed only from objects of interest. Grow regions only if the similarity criterion is fulfilled.
- **Similarity Measures:**
    - Similarity measures can be of different types: For the grayscale image the similarity measure can be the different textures and other spatial properties, intensity difference within a region or the distance b/w mean value of the region.
- **Region merging techniques:**
    - In the region merging technique, we try to combine the regions that contain the single object and separate it from the background.. There are many regions merging techniques such as Watershed algorithm, Split and merge algorithm, etc.
- **Pros:**
    - Since it performs simple threshold calculation, it is faster to perform.
    - Region-based segmentation works better when the object and background have high contrast.
- **Limitations:**
    - It did not produce many accurate segmentation results when there are no significant differences b/w pixel values of the object and the background.

Implementation:

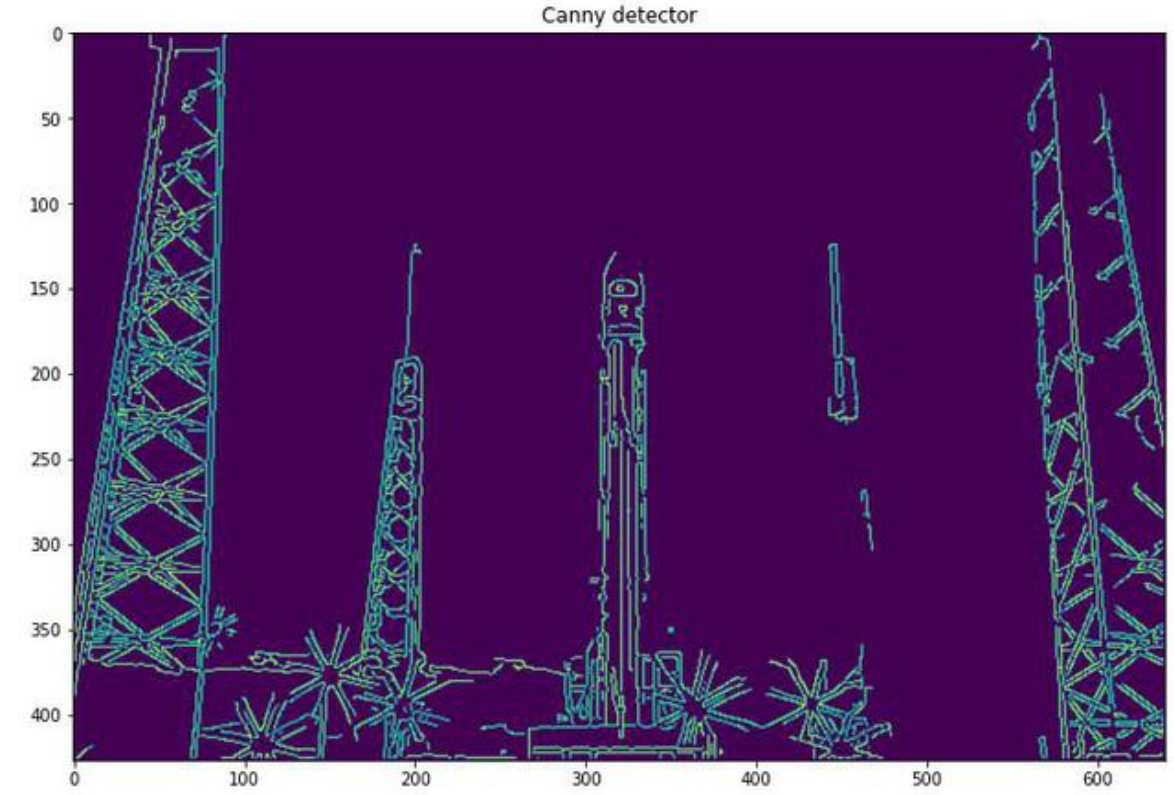
- In this implementation, we will be performing edge and region-based segmentation. We will be using scikit image module for that and an image from its dataset provided.

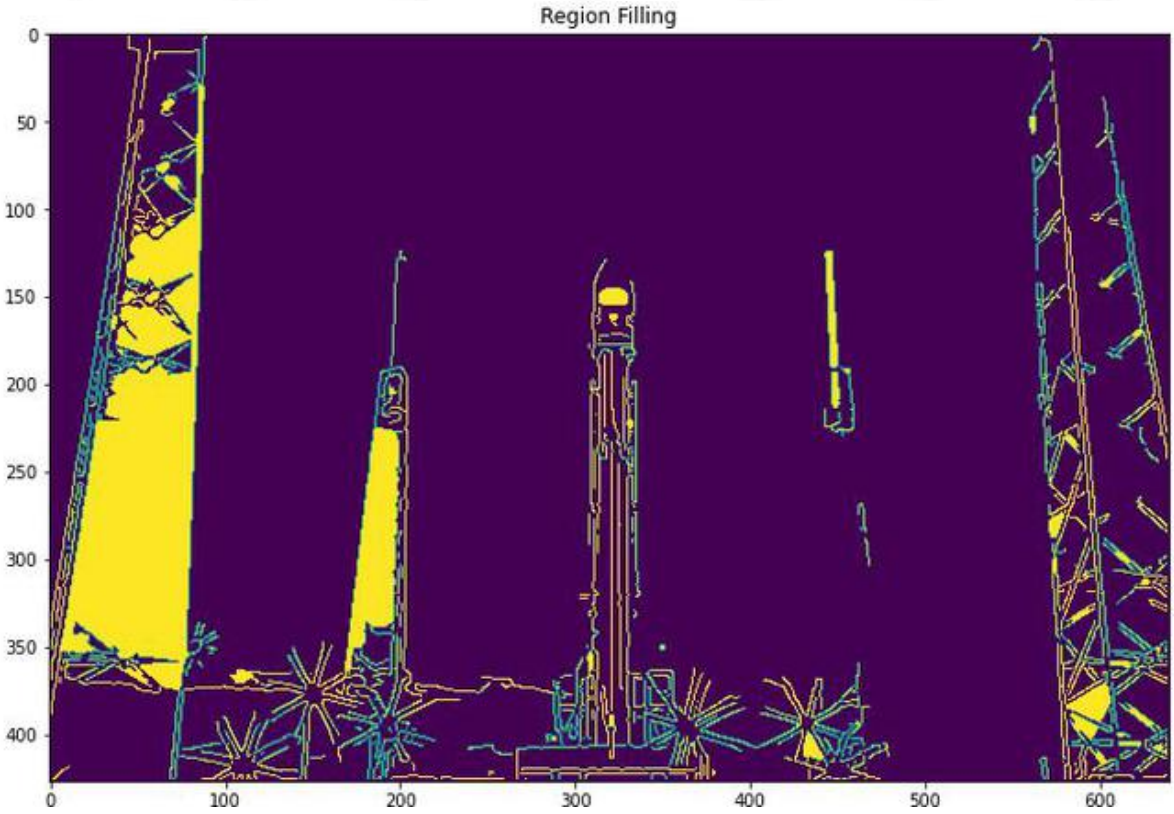
- Python3

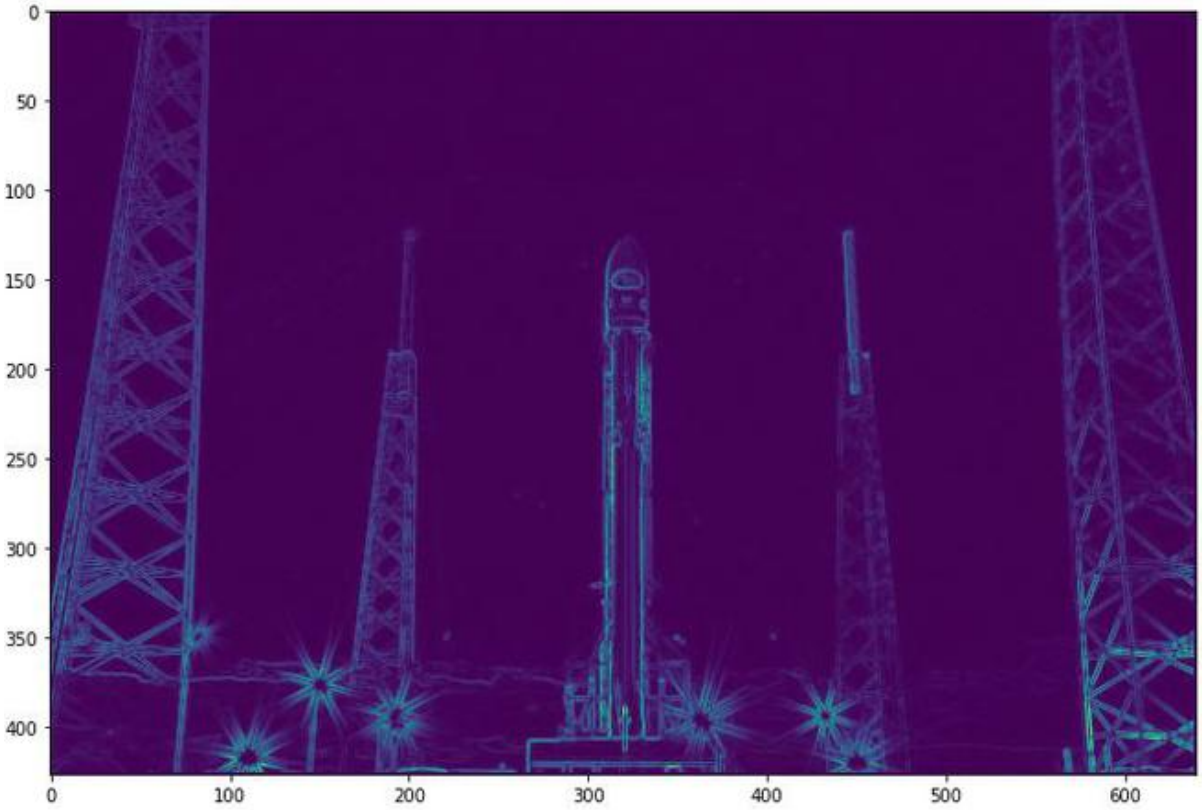
```
# code
import numpy as np
import matplotlib.pyplot as plt
from skimage.feature import canny
from skimage import data,morphology
from skimage.color import rgb2gray
import scipy.ndimage as nd
plt.rcParams["figure.figsize"] = (12,8)
%matplotlib inline

# load images and convert grayscale
rocket = data.rocket()
```

```
rocket_wh = rgb2gray(rocket)

# apply edge segmentation
# plot canny edge detection
edges = canny(rocket_wh)
plt.imshow(edges, interpolation='gaussian')
plt.title('Canny detector')

# fill regions to perform edge segmentation
fill_im = nd.binary_fill_holes(edges)
plt.imshow(fill_im)
plt.title('Region Filling')

# Region Segmentation
# First we print the elevation map
elevation_map = sobel(rocket_wh)
plt.imshow(elevation_map)

# Since, the contrast difference is not much. Anyways we will perform
it
markers = np.zeros_like(rocket_wh)
markers[rocket_wh < 0.1171875] = 1 # 30/255
markers[rocket_wh > 0.5859375] = 2 # 150/255

plt.imshow(markers)
plt.title('markers')

# Perform watershed region segmentation
segmentation = morphology.watershed(elevation_map, markers)

plt.imshow(segmentation)
plt.title('Watershed segmentation')

# plot overlays and contour
segmentation = nd.binary_fill_holes(segmentation - 1)
label_rock, _ = nd.label(segmentation)
# overlay image with different labels
image_label_overlay = label2rgb(label_rock, image=rocket_wh)

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(24, 16), sharey=True)
ax1.imshow(rocket_wh)
ax1.contour(segmentation, [0.8], linewidths=1.8, colors='w')
ax2.imshow(image_label_overlay)

fig.subplots_adjust(**margins)
```

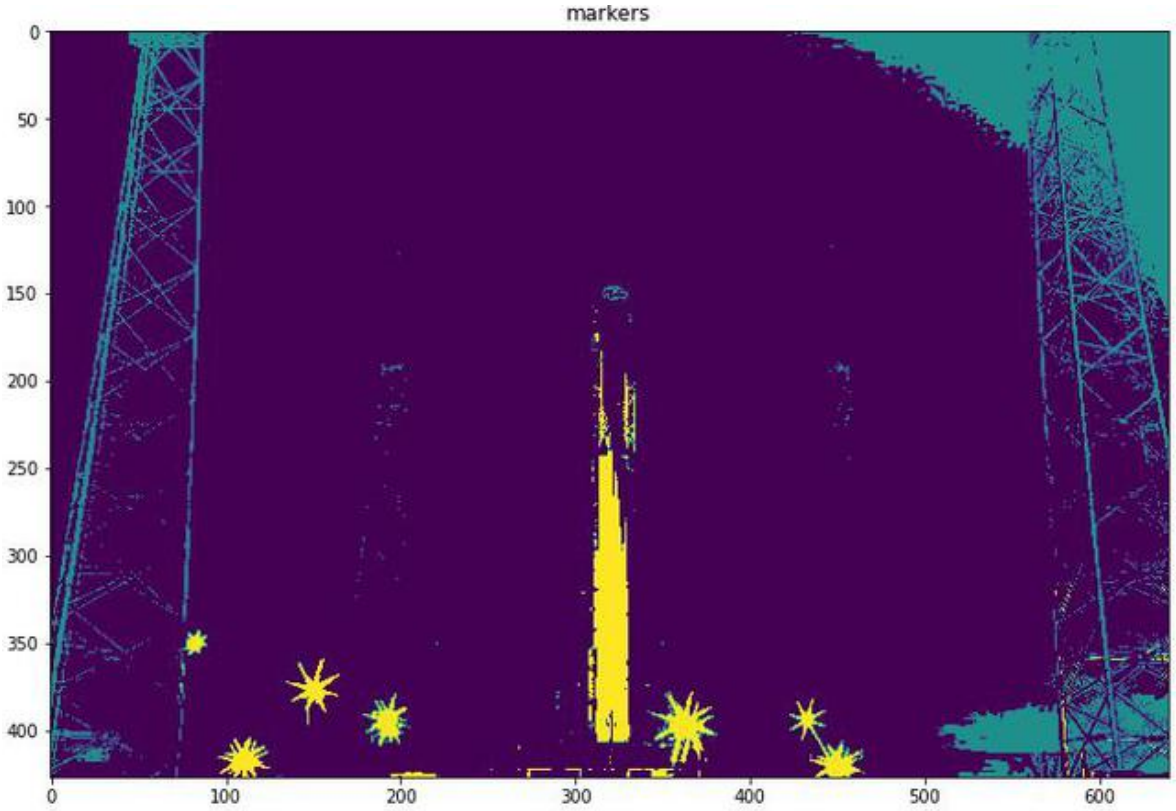
Output:

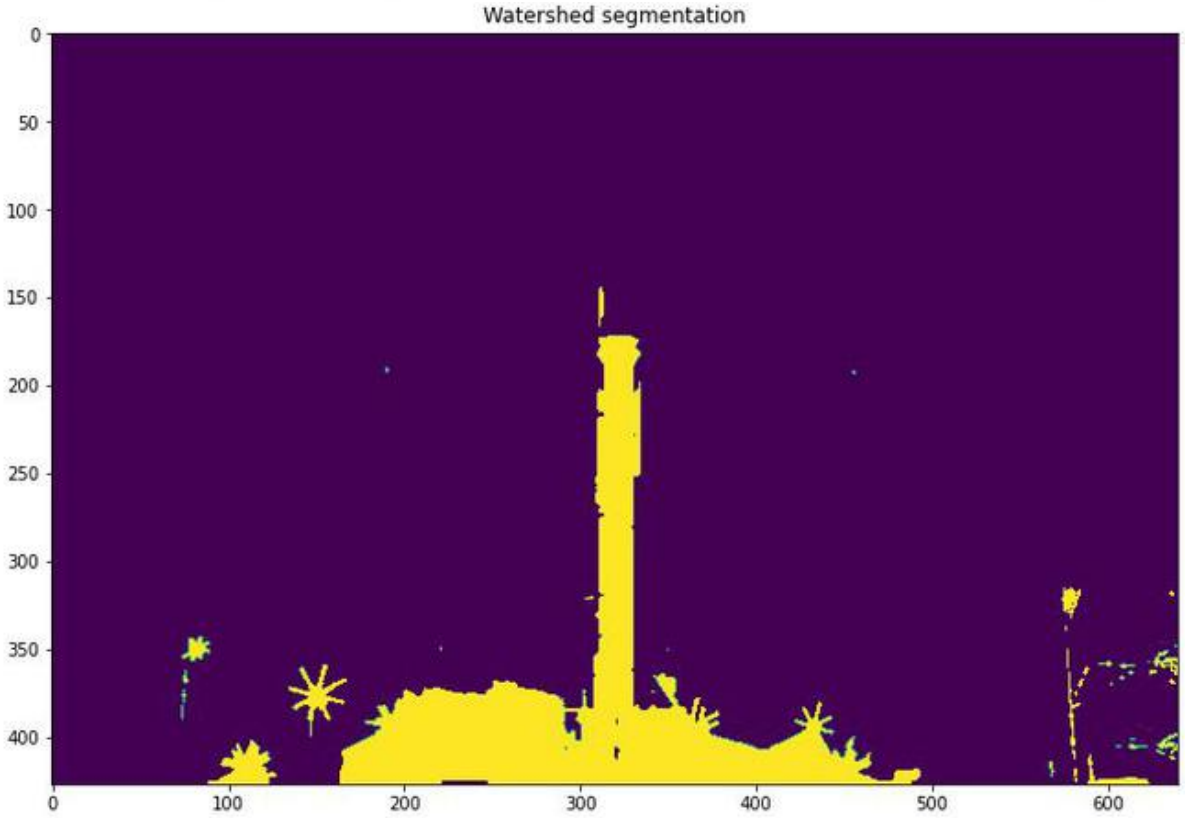

Canny detector
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600

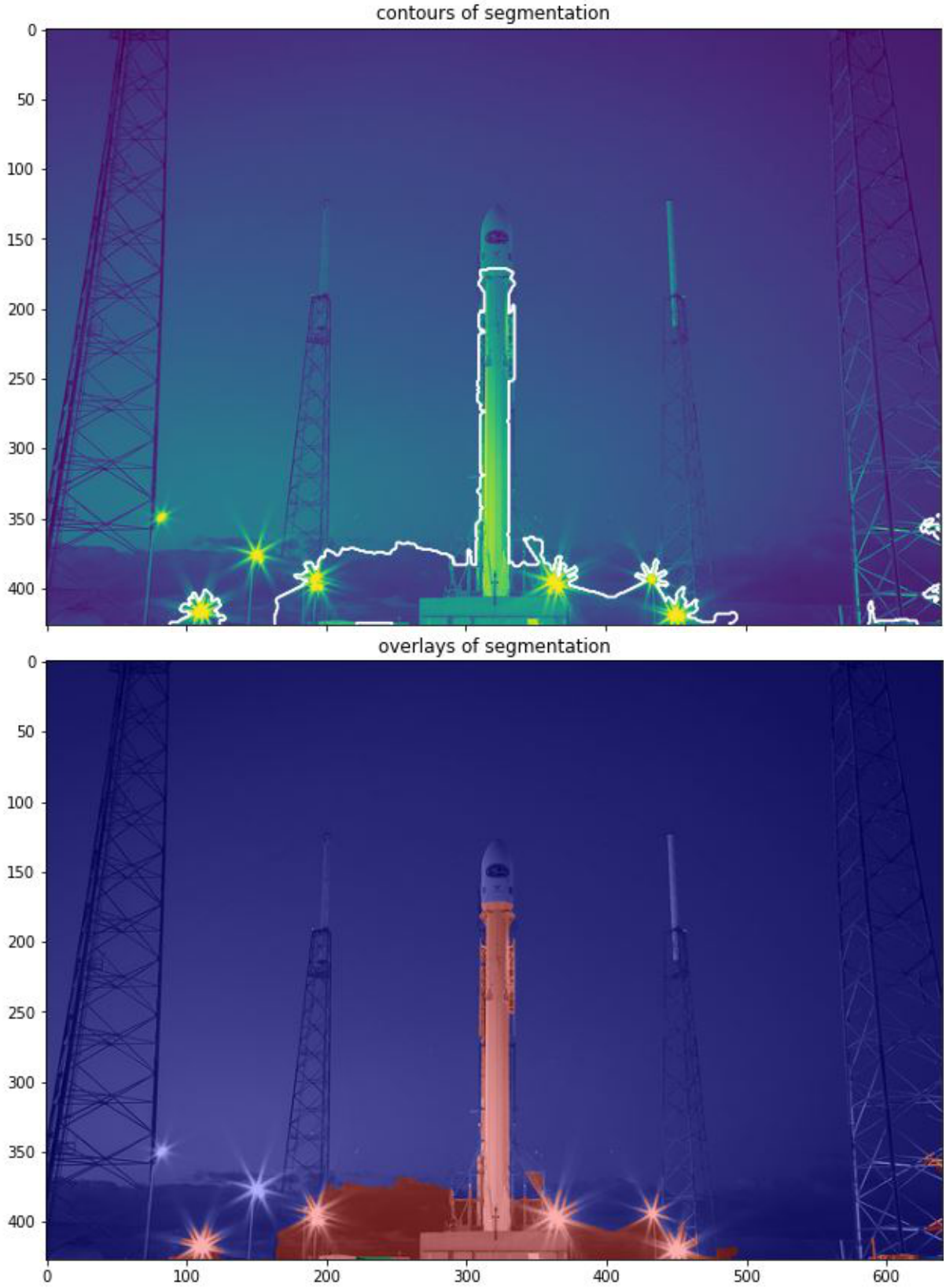

Region Filling
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600



Elevation maps


markers
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600


Watershed segmentation
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600


contours of segmentation
0
50
100
150
200
250
300
350
400
overlays of segmentation
0
50
100
150
200
250
300
350
400
0
100
200
300
400
500
600

The task is to go to this website https://app.mdb.p.lodz.pl/

And download current content of melanocytic lesion ad try to apply edge and region segmentation performing watershed region overlays and contour on several images.