

## 1. A brief description of the program

### Mean-Shift Segmentation

The program uses `cv2.pyrMeanShiftFiltering()` as the mean-shift segmentor. Since we are using LAB color space for mean-shift, the program first converts input images to lab color space using `cv2.cvtColor()` and runs the mean-shift segmentation. Once it is done, the “posterized” images are reverted back to RGB space. The code for mean-shift is very straightforward.

### Watershed Segmentation

For watershed segmentation, we first start by binarizing the images. In this step, we are using Otsu’s method to determine the threshold. To remove noises, we use morphological opening. To remove small holes in objects, we use morphological closing. We are pretty sure that the region near to center of objects are foreground and region far away from the object are background. We have unsure regions, and will use distance transform to remove boundary pixels. Next we find the area which we are sure they are not coins. We dilate the result. Dilation increases object boundary to background. This enables us to be sure that whatever region in the background is really a background (since non-background regions are dilated).

The remaining regions are “borders” where foreground and background usually meets. We obtain this area by subtracting sure foreground area from sure background area. The code will label regions we know for sure (with positive integers) and unsure (with 0) using `cv2.connectedComponents()`. This is done by our adding 1 to all labels so that unknown region has a label of 0. Now, we can finally apply watershed.

Three images below are used as inputs for segmentation tests.

For mean-shift, I have tried different combinations of spatial and color radius values.

For watershed, I have tried different threshold for binarization when finding sure areas.


















## 2. Test results

I have attached the full result images at “results” folder.













### Mean-Shift Segmentation

(I have tested different spatial radius values ranging from 1 to 5, but I am only showing the results for images segmented with spatial radius of 2 since it didn't show much difference)

| Spatial Radius: 2<br>Color Radius: 10   | Spatial Radius: 2<br>Color Radius: 20   | Spatial Radius: 2<br>Color Radius: 30   | Spatial Radius: 2<br>Color Radius: 40  | Spatial Radius: 2<br>Color Radius: 50   |
|---|---|---|--|---|
|   |   |   |   |   |
|  |  |  |  |  |
|  |  |  |  |  |

## Watershed Segmentation

(Parameter value for binarization of determining whether a pixel is in a sure region from distance transform is altered in each iteration)

| Threshold Factor: 0.2   | Threshold Factor: 0.4   | Threshold Factor: 0.6  | Threshold Factor: 0.8   |
|---|---|--|---|
|    |    |    |    |
|   |   |   |   |
|  |  |  |  |

### 3. Analysis

- Because the three images contain many regions that are non-uniform (grass, zebra's texture, water particles in waves), both segmentation methods end up with poor results at different parameters combined.
- I've manually blurred images using Gaussian Blur and ran tests but I wasn't able to get any better results.
- For mean-shift segmentation, altering spatial radius value did not change results significantly. However, it seemed to show the best results at 4.0 or 5.0.
- For mean-shift segmentation, altering color radius did show meaningful difference. The "best" values are different for each image.

Stone statues image - looks best at 40.0

Zebra image - looks best at 20.0 or 30.0 (when it gets too large, grass region and zebra's outer regions begin to get mixed up)

Surfer image - looks best at 40.0 or 50.0 (if smaller, wave particles are grouped separately)

- For watershed segmentation, parameter for thresholding to determine sure region from distance transform showed notable differences.

Although the sample code from OpenCV's documentation used 0.7 for "coin example", 0.7 did not work well for statue image.

Stone statues image - looks best at 0.3

Zebra image - looks best at 0.7

Surfer image - looks best at 0.1

**Note that all results using watershed did not work well.**

- Overall, the results look very distant from the ground truth boundaries from Berkeley's site.

For the stone statues image, the shades in statues make segmentation difficult.

For the zebra image, the white and black textures make segmentation difficult.

For the surfers image, the wave water particles that are close to white make segmentation difficult.

- **In practice, it is difficult to use both methods for the supplied images.**

Mean-shift - We must know the kernel size in advance, but it is very difficult to determine it due to the textures/shades/particles in the images

Watershed - The results are only as good as good boundaries. The boundaries in the supplied images are relatively clear, but there are other obstacles that prevents watershed algorithm to correctly find the for sure region (zebra texture, shades, and water particles).