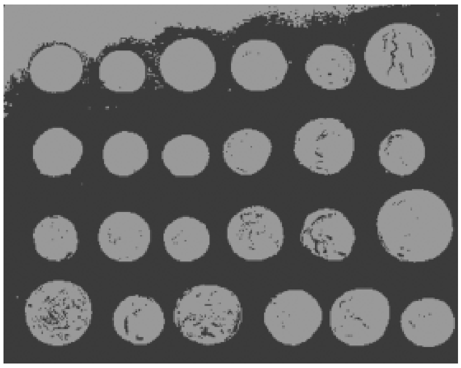
Homework Assignment 5

Yi Liu djl174
ZhaoYang Xu chm564
Jingzheng Li bnd559

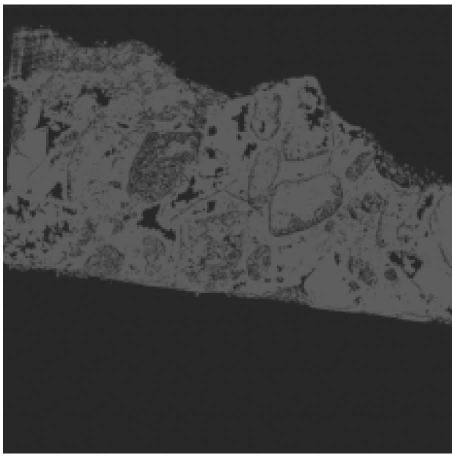
January 22, 2021

1 K-means algorithm

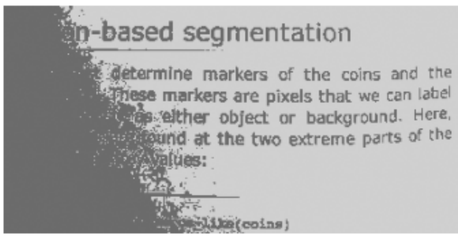
(a) *camera*



(b) *coins*



(c) *rocksample*



(d) *page*

Figure 1: Kmeans algorithm on four images

First we briefly introduce image segmentation, we know that the objective of segmentation is to simplify or transform the representation of an image to something more meaningful or easier to analyse. The image segmentation is commonly used to locate objects as well as lines or curves in an image. To be more precise, image segmentation refers to the process of assigning a label to each pixel in an image to make pixels with the same label share some features.

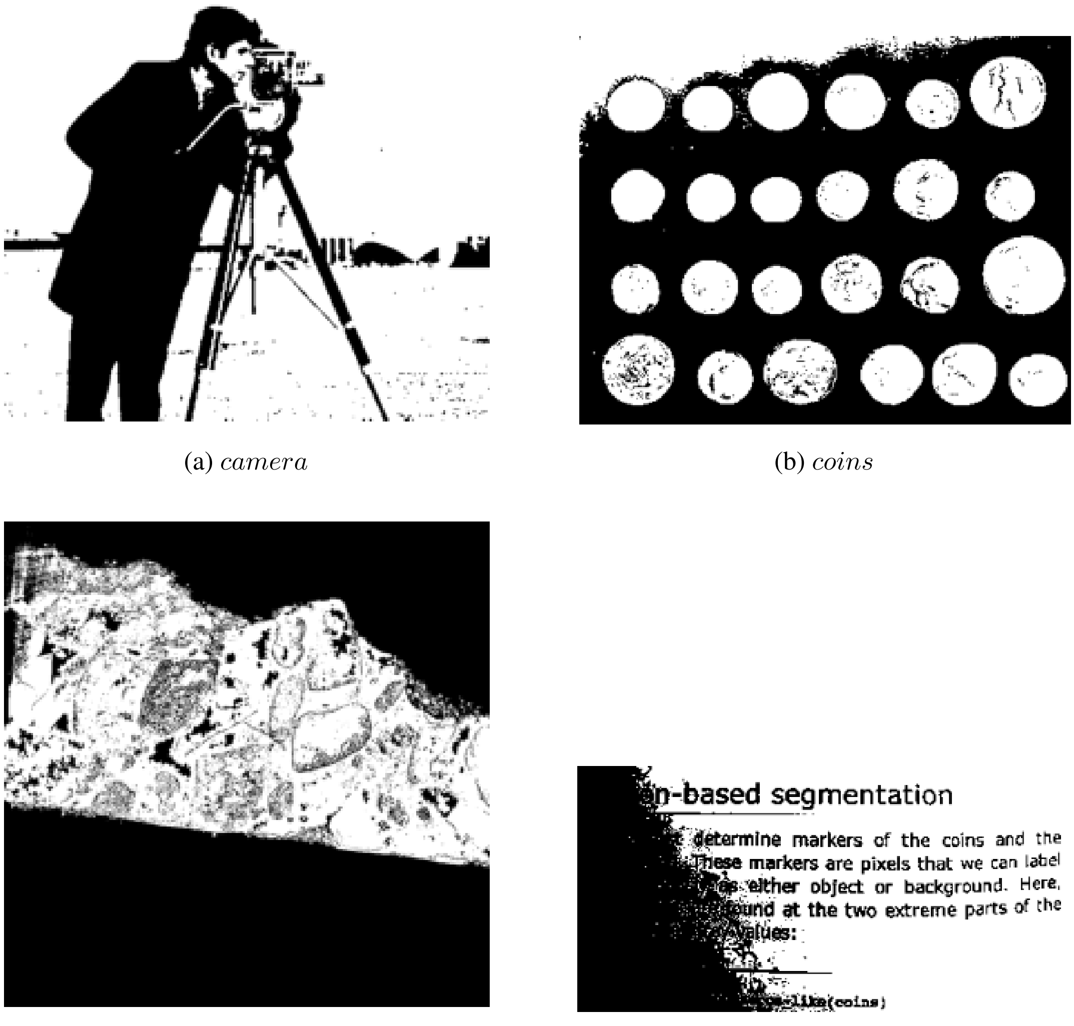
K means clustering use the Euclidean distance for the similarity measure. In our case, the pixel's intensity value is used to determine the clusters to be generated.

As can be clearly seen in Figure a, the image is segmented by colour into two groups, black and grey.

We can also see in Figure b that the image is segmented into two colour types, but since the coins themselves have a mixture of colour intensity, some of the coins on the bottom left clearly have some noise.

Due to the nature of rocksample images itself, it's mainly divided into the black background and white region in the middle. However. the middle part is a mixture of different intensity regions.

2 Otsu's ding algorithm(a single threshold)



(a) *camera* (b) *coins*

(c) *rocksample* (d) *page*

Figure 2: Otsu algorithm on four images

The aim of Otsu's algorithm is to find the threshold values that minimises the sum of foreground and background diffusion.

Through the above pictures, the only colours we can see are black and white. That is because in Otsu, we obtain the binary image, where 0 means the low intensities region and 1 means the high intensity region.

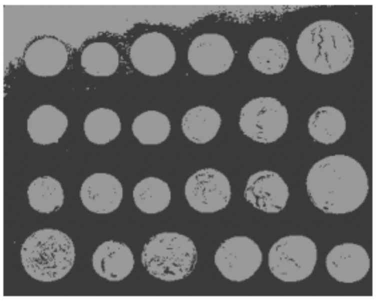
3 Comment on their similarity, dissimilarities?


(a) Camera original


(b) Camera K-means


(c) Camera Otsu


(d) Coins original


(e) Coins K-means

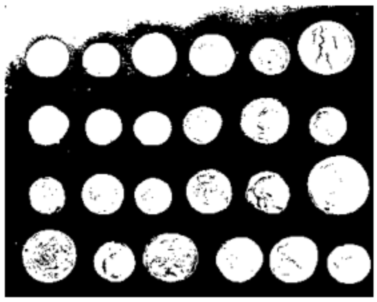
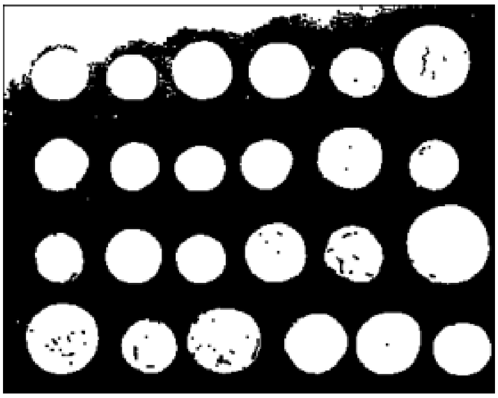

(f) Coins Otsu

Figure 3: Similarity and dissimilarities of K-means and Otsu

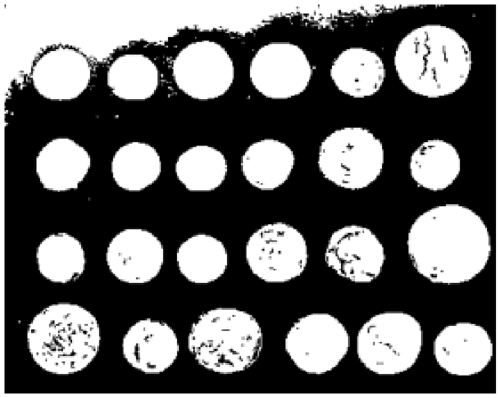
Use the camera image and coins image as two examples. Compared to the original images, we can see that both the K-means algorithm and the Otsu algorithm filter out well and extract the main content of the image.

However, these two algorithm have different systems to segment the images. For Otsu's algorithm, we should calculate the histogram to find a global optimal threshold, which is used to separate pixels into two classes. But For K-means algorithm, we used pixel's intensive value to determine the clusters to be generated. It is more like finding two local optimal centroids.

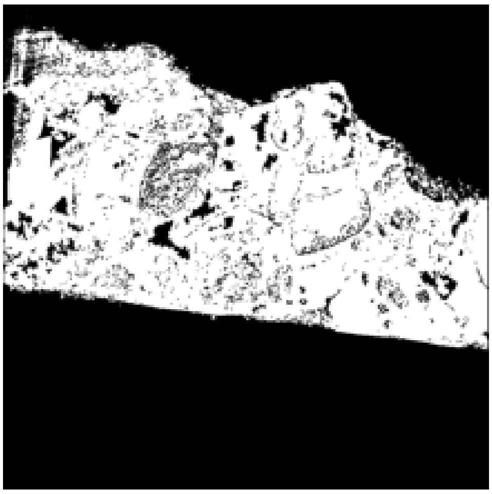
4 Run several iterations of the segmentation denoising algorithm with varying voting thresholds values. Comment on the results.



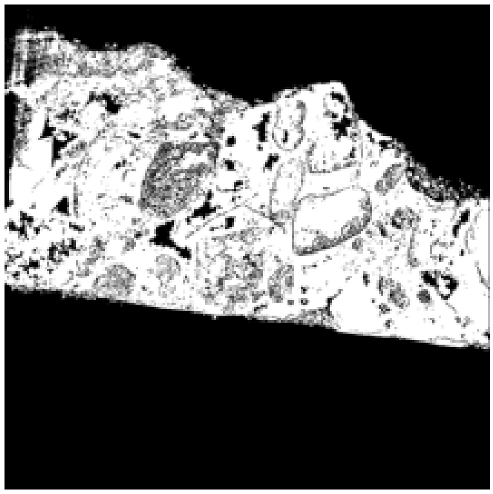
(a) Coins:Vote by all neighbours



(b) Coins:Vote by most neighbours



(c) Rocksample:Vote by all neighbours



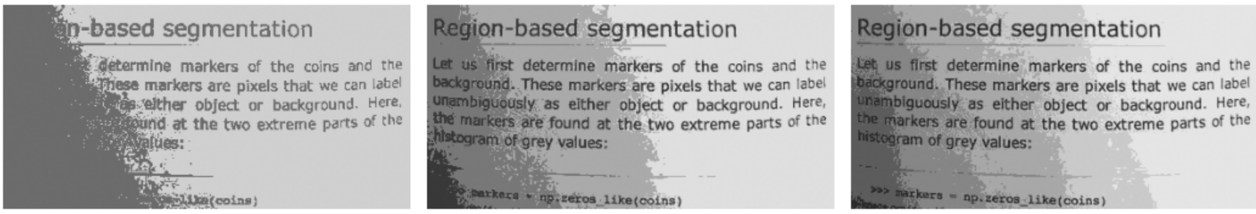
(d) Rocksample:Vote by most neighbours

Figure 4: Denoising with two voting thresholds

We used 4-pixels neighbourhoods system with 10 iterations to clean the noise. We chose two voting thresholds values. The first threshold is 3/4 of neighbourhoods. If more than two neighbours have the same value, then the center pixel is equal to its neighbour value. If there are none, then the pixel remains unchanged. The second one uses all neighbours, the pixel's value changes only if all surrounding neighbours have the same value.

It can be seen that the condition of the second threshold is more strict. So corresponding to the images, denoising with the second threshold is not as effective as the first one. We chose coins and rocksample to show the difference, as they have more and scattered noise after Otsu segmentation. It is clear that the first threshold reduced more noise.

5 Run a k-means segmentation with k > 2. Can it help with the page image?



(a) K=2 (b) K=4 (c) K=6

Figure 5: K-means segmentation with different k on pages

As the value of k increases, we can clearly see that the occlusions in the bottom left corner of the image is fading away, the print on the image is clearer, and the image can be displayed with more color segmentation.

6 Can you imagine a way to generalise the segmentation denoising algorithm to more than 2 segments? Comment on it.

For each interior pixel, we can compute the mean of all its neighbours' value, and give the mean to this pixel.

However there are some issues with this method. It destroys the details of the image while denoising the it, making the image blurred and not removing the noise points well. Also, it has to be computed for each pixel, and for some large images, there may be efficiency issues to consider.

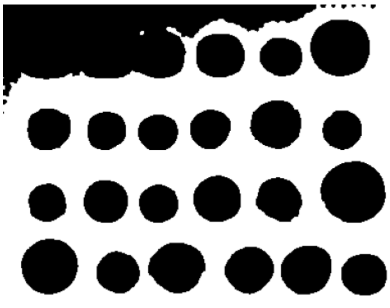
7 You may want to try some of the algorithms available in scikit-learn, such as the Chan-Vese segmentation.

7.1 Chan-Vese segmentation algorithm

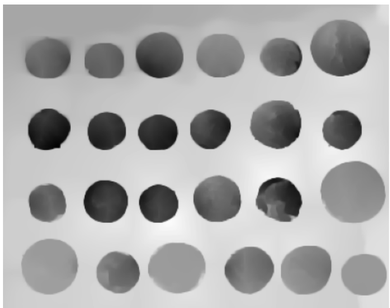
Chan-Vese segmentation algorithm is an active contour model which suitable for segementing objects without clearly boundaries. Chan-Vese algorithm is based on an energy minimization problem, which can be reformulated in the level set formulation to better solve the problem.

For Chan-Vese algorithm, we mainly use the following code

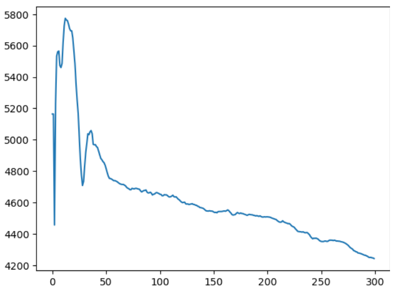
```
chan_vese(image, mu=0.1, lambda1=1.0, lambda2=1.0, tol=1e-3,
   max_iter=500,dt=0.5, init_level_set="checkerboard",
   extended_output=True)
```



(a) Coins: Chan-Vese segmentation



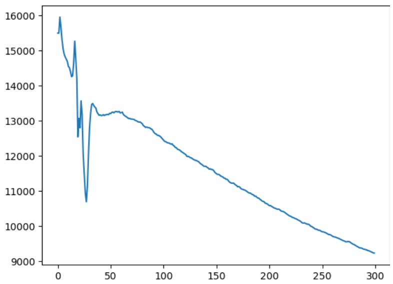
(b) Coins: Final level set



(c) Coins: Energy over iterations

(d) Camera: Chan-Vese segmentation

(e) Camera: Final level set



(f) Camera: Energy over iterations

Figure 6: Chan-Vese segmentation algorithm

The left column of pictures are Chan-Vese segmentation, the middle ones are their final level set, and right column are their evolution energy over iterations. From the above results, we can see that if taken proper parameters, CV algorithm can get better results than k-means and Otsu algorithm, which may because CV algorithm relies on global properties, instead of local information.

In chan-vese function, μ is mainly used for different type of contours, which usually set in the interal of 0 to 1. Below we test 3 different μ.

(a) μ=0.9

(b) μ=0.4

(c) μ=0.1

Figure 7: The effect of different μ values on pictures

As the μ decreases, the main part of the picture becomes more prominent. When the contours are in good shape, there is no need for too large μ, otherwise the image noise will increase.

7.2 MorphACWE and MorphGAC algorithm

Morphological Active Contours without Edges (MorphACWE) and Morphological Geodesic Active Contours (MorphGAC) are improvement from the traditional Active Contours algorithm. Their main idea is to convert PDE solving into morphological operations, which improves the speed of the algorithm and numerically more stable.

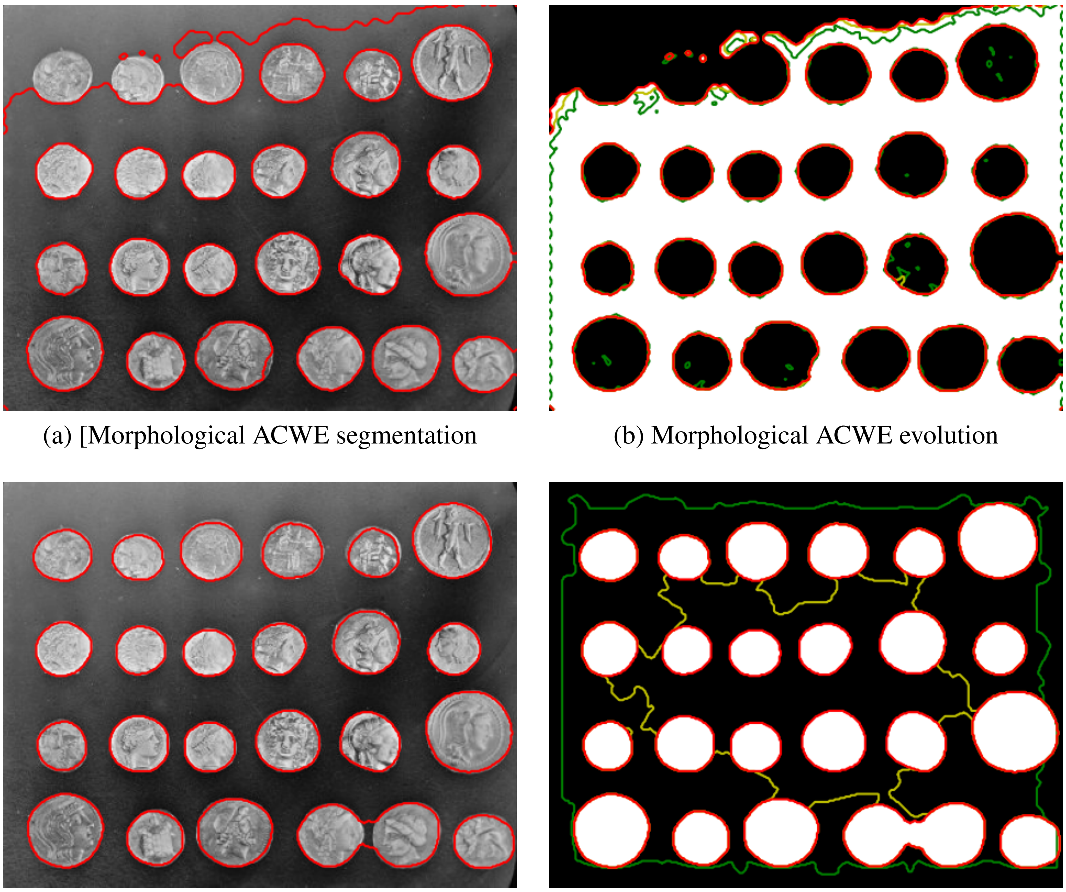
MorphACWE has good effects when pixel values of the inner and outer regions have different average. While MorphGAC is suitable for images with visible contours, even when these contours might be noisy or partially unclear. However, MorphACWE needs very strict pre-processing highlight the contours, which is the basic guarantee of image quality.

For MorphACWE, we mainly use the following code to do segmentation

```
morphological_chan_vese(image, 500, init_level_set=init_ls, smoothing=2,
    iter_callback=callback)
```

For MorphGAC, we fist use inverse Gaussian gradient to preprocess the image, then use following code to do segmentation

```
morphological_geodesic_active_contour(gimage, 500, init_ls, smoothing=1,
    balloon=-1, threshold=0.69, iter_callback=callback)
```



(a) [Morphological ACWE segmentation (b) Morphological ACWE evolution

(c) Morphological GAC segmentation (d) Morphological GAC evolution

Figure 8: coins: MorphACWE and MorphGAC

Because the coins picture has relatively clear contours. We can see that when MorphACWE algorithm processes coins, there are still some areas in the upper left corner that are not correctly segmented. However, after proper preprocessing the picture, the MorphGAC algorithm can achieve better results.

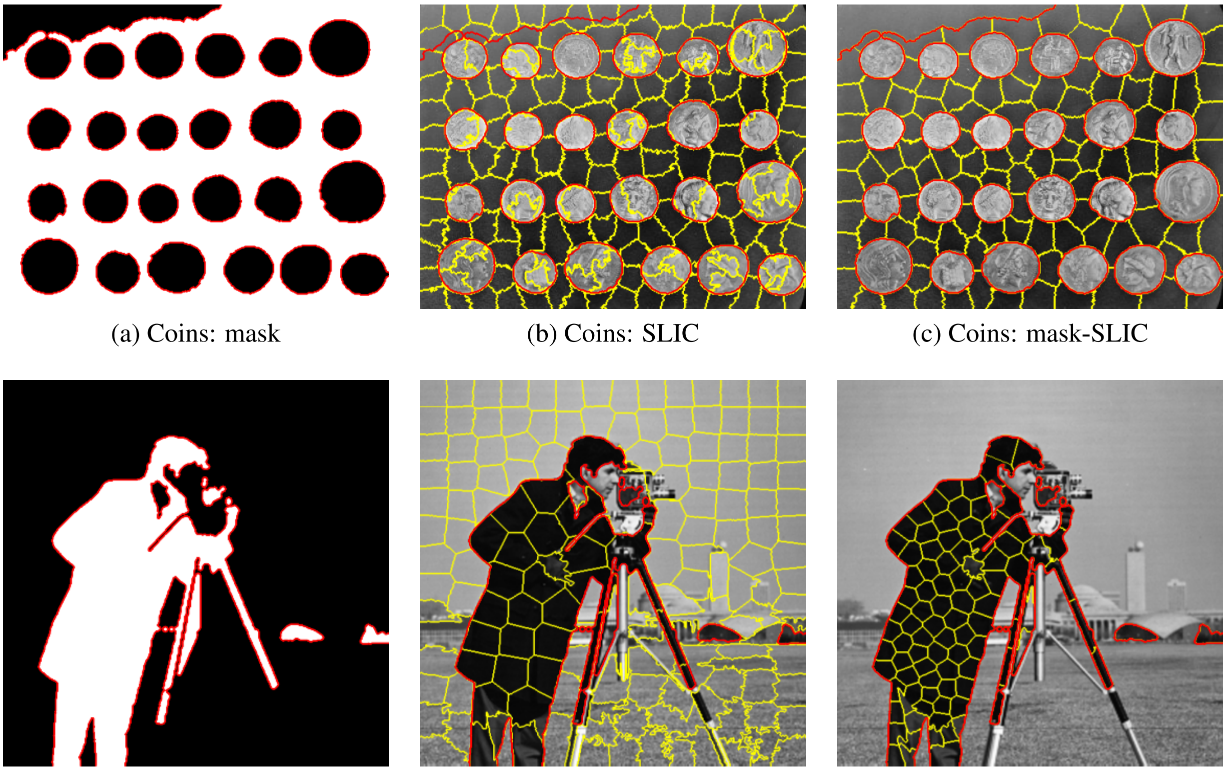
7.3 Simple linear iterative clustering algorithm

Simple linear iterative cluster algorithm (SLIC) algorithm is to form adjacent pixels with similar texture, color, brightness and other characteristics into irregular pixel blocks with certain visual significance. It uses the similarity of features between pixels to group pixels, and replaces a large number of pixels with a small number of superpixels to express image features.

The maskSLIC method is an extension of the SLIC method for the generation of superpixels in a region of interest, which can overcome border problems that affects SLIC method, particularely in case of irregular mask.

We mainly use the following code to generate mask and mask SLIC pictures

```
mask = morphology.remove_small_holes(morphology.remove_small_objects(lum
    < 0.25,100),500)
slic = segmentation.slic(img, n_segments=200, start_label=0)
m_slic = segmentation.slic(img, n_segments=100, mask=mask, start_label=0)
```



(a) Coins: mask (b) Coins: SLIC (c) Coins: mask-SLIC

(d) Camera: mask (e) Cmaera: SLIC (f) Camera: mask-SLIC

Figure 9: SLIC and mask-SLIC algorithm

For SLIC algorithm, we choose 200 segments, for mask-SLIC, we choose 100 segements. We can see that mask separates the main part of the picture we need. mask-SLIC limits the segmentation to the inside of the mask. For details, such as camera hands and pants, mask-SLIC achieves more precise and accurate segmentation.