Lappeenranta-Lahti University of Technology LUT
School of Engineering Science
Computational Engineering and Technical Physics
Computer Vision and Pattern Recognition

Group E : Tuo Yang , Subhashree Rautray

IMAGING MEASUREMENTS

Practical Assignment

Examiners: Professor Arto Kaarna

Supervisors: Professor Arto Kaarna

CONTENTS

1 PROBLEM DESCRIPTION

The purpose of this image measurement task is to recognize various euro coins 2-euro, 1-euro, 50-cent, 20-cent, and 10-cent from an image using proper image processing methods. During the process, the intensity and geometric calibration has considered to eliminate some uneven illumination, space distortion after that we have taken object extraction methods to extract coin's shapes so that we can judge types of coins in the image. In general, the goal of the assignment is to implement a system that can estimate the number of various coins in the image.

2 IMAGE CALIBRATION

2.1 Intensity Calibration

To realize the intensity calibration, three types of images we have used here, they are bias, flat, and dark images (B, F, and D for short), from which we have calculated their mean images $\hat{\mathrm{B}}$, $\hat{\mathrm{F}}$, and $\hat{\mathrm{D}}$, in the end we used the formula below to get the intensity-calibrated image R when we used unmeasured image I as the input:

$$R = \frac{I - \widehat{B} - \widehat{D}}{\widehat{F}} \tag{1}$$

2.2 Geometric Calibration

The next step after the intensity calibration is to use matlab toolbox Camera Calibrator to do the geometric calibration for eliminating radial and tangential distortion in the intensity-calibrated image, and part of matlab realization codes are shown as Fig. 1 below:

```
%corner at (0,0)
squareSize = 12.5;      %in millimeters 23.25 cents(14/25)
worldPoints = generateCheckerboardPoints(boardSize, squareSize);

%Calibrate the camera
imageSize = [size(measurement,1),size(measurement,2)];
cameraParams = estimateCameraParameters(imagePoints,...
    worldPoints,'ImageSize',imageSize);

%Evaluate cabibration accuracy
figure;
showReprojectionErrors(cameraParams);
title('Reprojection Errors');

%because the lens introduced little distortion, use 'full'
%output view to illustrate that the image was undistored
[calibrated_image,newOrigin] = undistortImage(measurement,cameraParams,...
    'OutputView','full');
%display the image after geometric calibration
figure;
imshow(calibrated_image,'InitialMagnification',magnification);
title('Undistorted Image');
```

Figure 1. matlab realization codes for geometric calibration

As from the Fig. 1 , first we set the real size of checkboard square as 12.5 millimeters. Then we use the function generate- CheckboardPoints to let the system itself generate some standard checkerboard images . The function estimateCameraParameters will be making comparison between the standard checkerboard image and all measurement images to get some parameters stored inside the structural variable cameraParams and two variables among which are very important: RadialDistortion and Tangential Distortion, all elements inside these two variables will be coefficients which will be used for eliminating radial and tangential distortion. Then we have used the measurement image and cameraParams as inputs of the function undistortedImage to make the geometric calibration to the measurement image. During this process, ReprojectionErrors, as a estimation parameter of the camera accuracy will show the error between the reprojected pattern key points and the detected pattern key points of each measurement image respectively , the reprojection error shown in Fig. 2

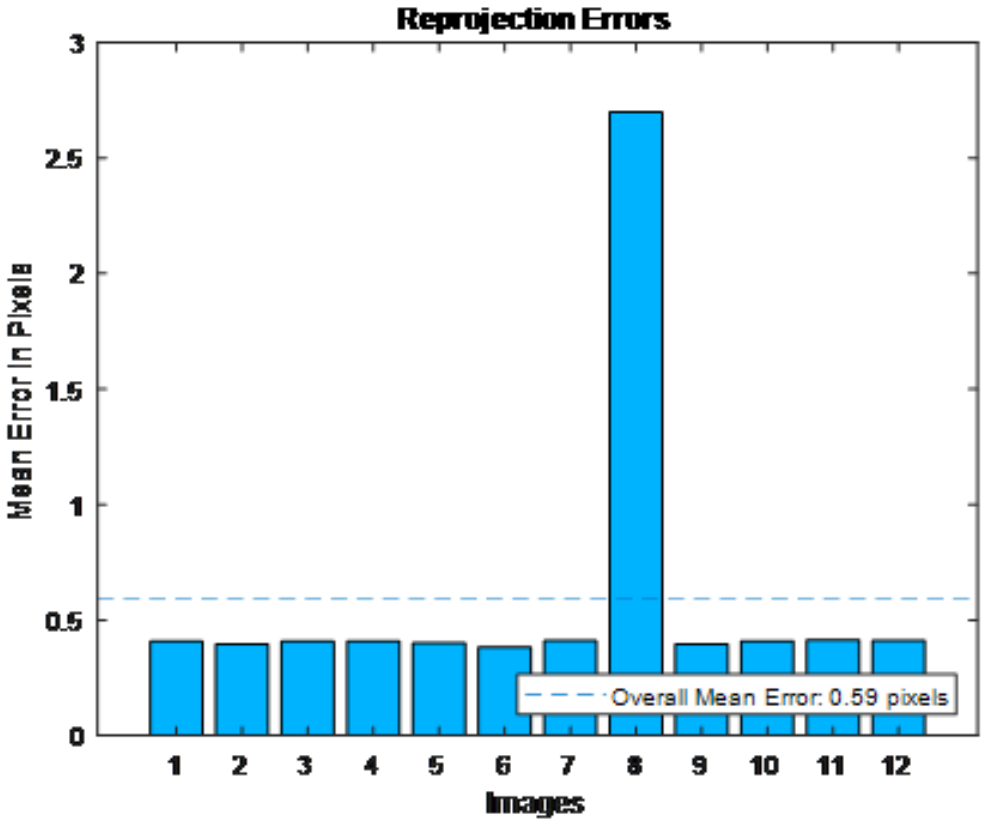

Figure 2. Reprojection errors

As we can see from the Fig. 2, the mean of all measurement images' reprojection errors is 0.59 pixels, and we got one outlier here, it's the 8th image, if we remove it from this image set, the overall mean error will become less.

3 MEASUREMENTS AND ANALYSIS

In fact, to judge the types of all coins inside the image, we need to find some good features to distinguish these coins with the background. The strategy we have taken is to convert it to the other color space, the color space we take is the HSV color space including Hue, Saturation, and Value three parameters, then our next step is to do the binarization to the image for getting some initial shapes of coins by selecting the optimal threshold value in the saturation component, and the optimal value we selected is the half of optimal value getting from the function graythresh(OSTU method) using saturation channel of this image as input, and the initial binarization results are shown as Fig. 3 below (taken

image is_DSC1776.JPG as the example):

3.1 Binarization process of an image

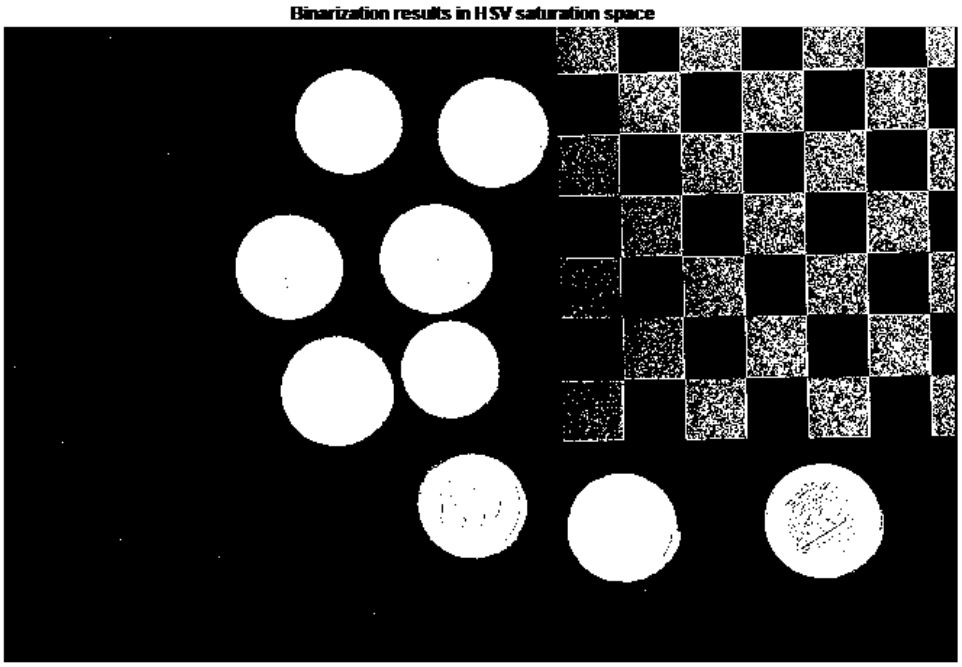

Figure 3. binarization results in HSV saturation space

As the Fig. 3 shown as below, after doing some initial binarization to the calibrated image, several noisy background and small white holes will show up, and we need to get rid of them. But there are two different situations needs to be dealt with by using two totally different algorithms (experience from several times' debugging).

3.2 Noise removal process from image

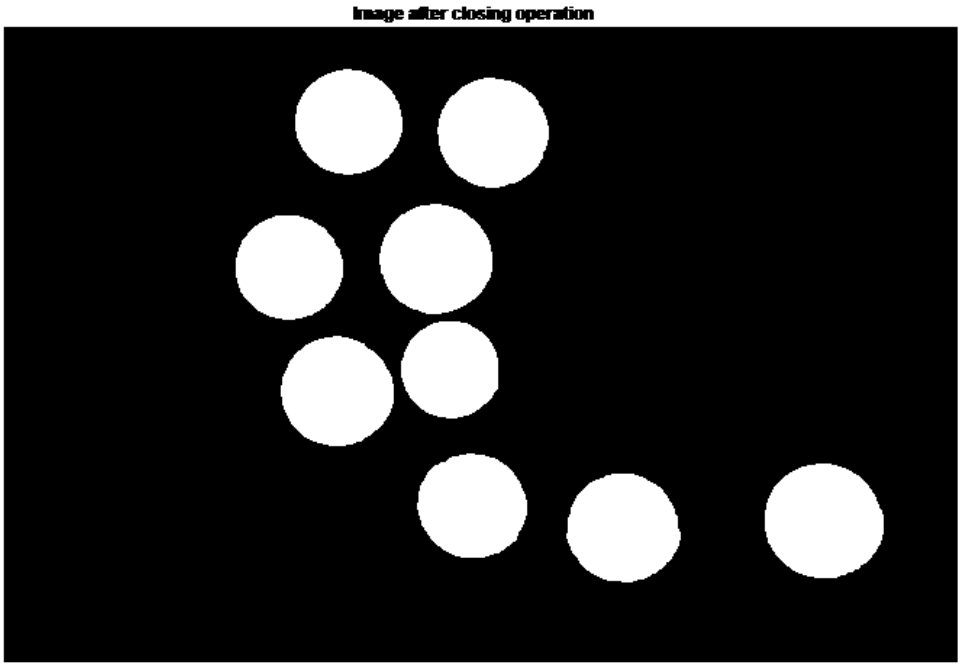

Figure 4. image after doing further processing of getting rid of the noise

Taken image _DSC1779.JPG as the example: Fig. 5 after doing further processing of getting rid of the noise

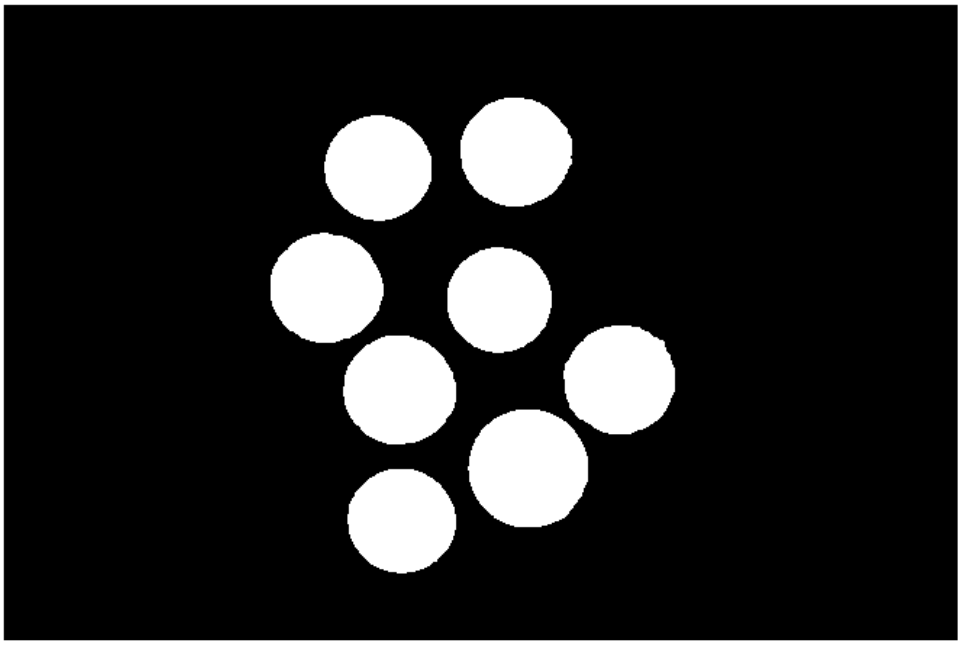

Figure 5. image after doing further processing of getting rid of the noise

For these two situations, four types of image processing methods used for binary images : opening and closing operation (used the matlab function imdilate and imerode), getting rid of small white area closing to the border(used the matlab function imclearborder and imfill), and bwareaopen operation(getting rid of white area less than specified pixel area).

4 RESULT

To get the number of each type's coin in the image, we need to do further processing to the optimized binarization results, using the function regionprops to measure the size of each coin's area in the image, and draw approximate circles to each coin, visualizing results we get Fig. 6 (Taken `_DSC1779.JPG` as the example):

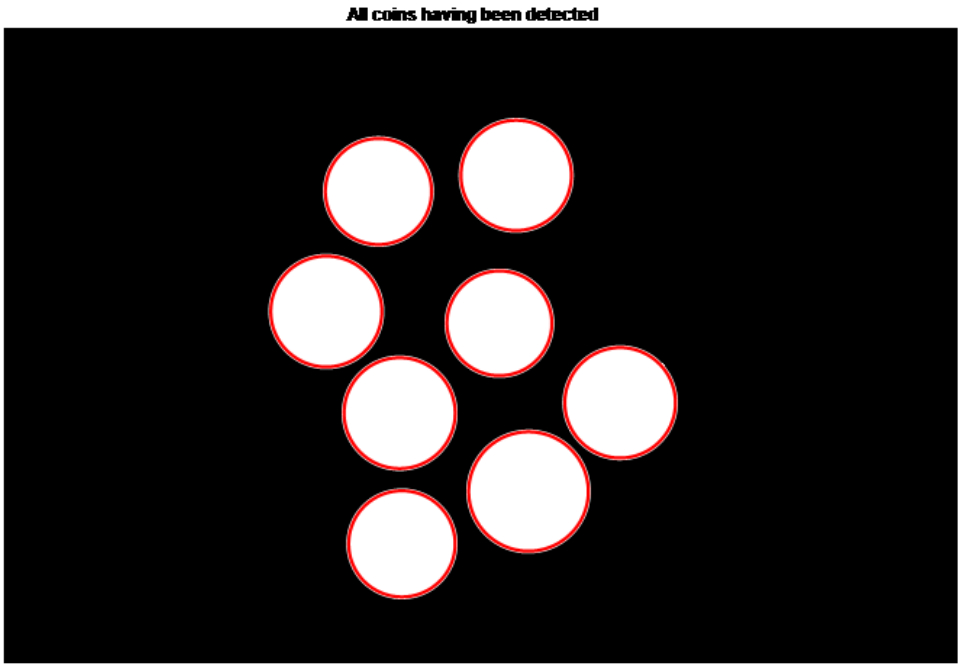

Figure 6. Approximate circles to all coins

And for each type of coin, according to our several times' experiments, we got approximate ranges of pixel for given types of coins.
2 euro: 200000 to 220000 pixels
50 cents:180000 to 200000 pixels
1 euro:170000 to 180000 pixels
20 cents:150000 to 170000 pixels
5 cents:140000 to 150000 pixels
10 euro:120000 to 130000 pixels
Using rules mentioned below, we can get the number of different types of coins from the image the output shown in the Fig. 7 for image `_DSC1779.JPG`:

```
Statistical results(_DSC1779.JPG):
the number of 2 euro:0
the number of 1 euro:0
the number of 50 cent:1
the number of 20 cent:4
the number of 10 cent:0
the number of 5 cent:3
```

Figure 7. Result of image `_DSC1779.JPG`

5 REFRENCE

https://www.mathworks.com/help/vision/ug/single-camera-calibrator-app.html
https://www.mathworks.com/help/vision/ref/cameraparameters.html
https://nl.mathworks.com/help/vision/ug/measuring-planar-objects-with-a-calibrated-camera.html