Robotics HW3[1]

R12921008 Che-Jung Chuang
R12922135 Cheng-Yen Yu
R11631045 Wei-Hsuan Cheng
B10901042 Wen Perng

Part A: Camera Calibration

1. Camera and Checkerboard Descriptions

The camera we used is the Intel® RealSense™ D435 depth camera, and the checkerboard used for calibration is of 7 × 9 squares with each square having the size of 22.5 [mm] × 22.5 [mm].

The D435 camera can provide both RGB image and depth image, while here only the RGB image is needed. Each photos are captured and saved using the software realsense-viewer, and the resolution and frame rate are set as 640 × 480 [pixels] and 30 [fps], respectively.

The checkerboard and square sizes, camera in use, and the resolution and frame rate set in the software are shown in the figures below.

It's interesting to point out that the square size of the checkerboard doesn't affect the calibration result if there's no need for knowing the real-world scale.

Figure 1: RealSense D435 Depth Camera

[1]Code provided in github: eps46656/Robotics-AssignmentIII, and wei-hsuan-cheng/assignments_robotics_2023-24.

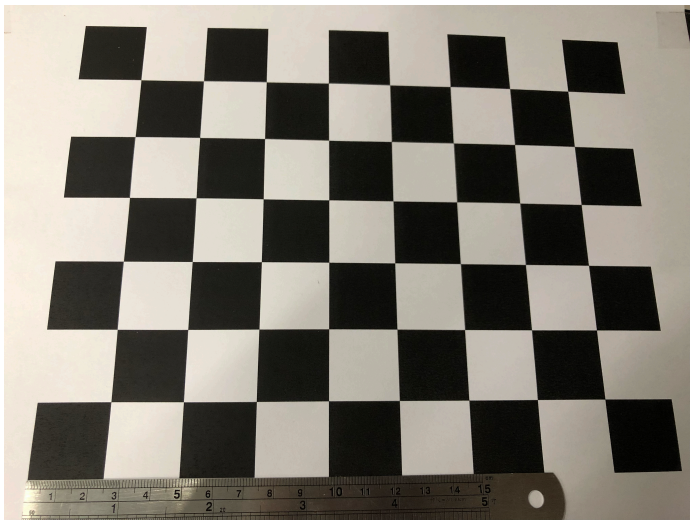


Figure 2: The Checkerboard and the Square Sizes

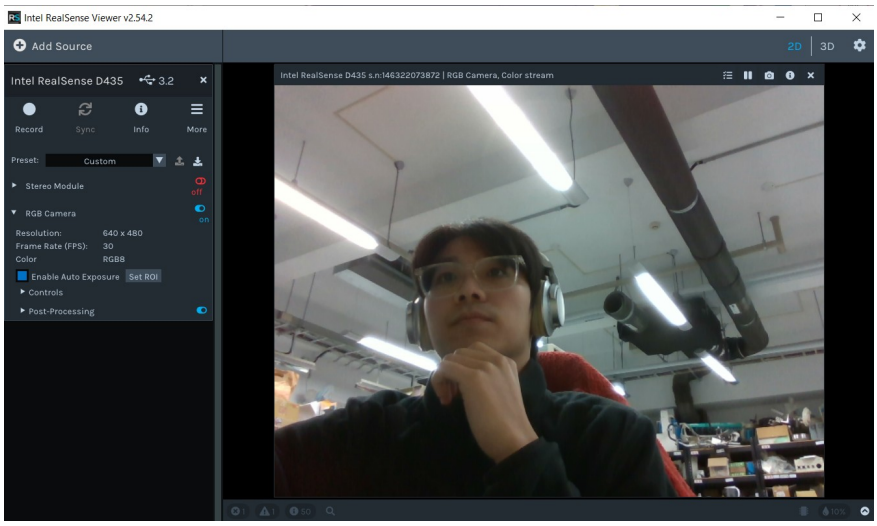


Figure 3: Resolution and Frame Rate set in RealSense Viewer

2. Intrinsic Parameters of Camera

The intrinsic parameters are obtained via the following procedures:

1. Using the function cv.findChessboardCorners() to obtain the corners of the chessboard.
2. Then cv.calibrateCamera() was used to find the intrinsic parameters of the camera given the corners of 2D image and their real world position.

   Noted that the function cv.findChessboardCorners() takes the inputs of the 3D point in real world space, the 2D points in image plane, and the total number of pixels in each frame. The 3D and 2D points are stored in the variables objpoints and imgpoints in our code, respectively. The pixels number (640×480), can either be specified manually or be extracted by .shape[:: -1].

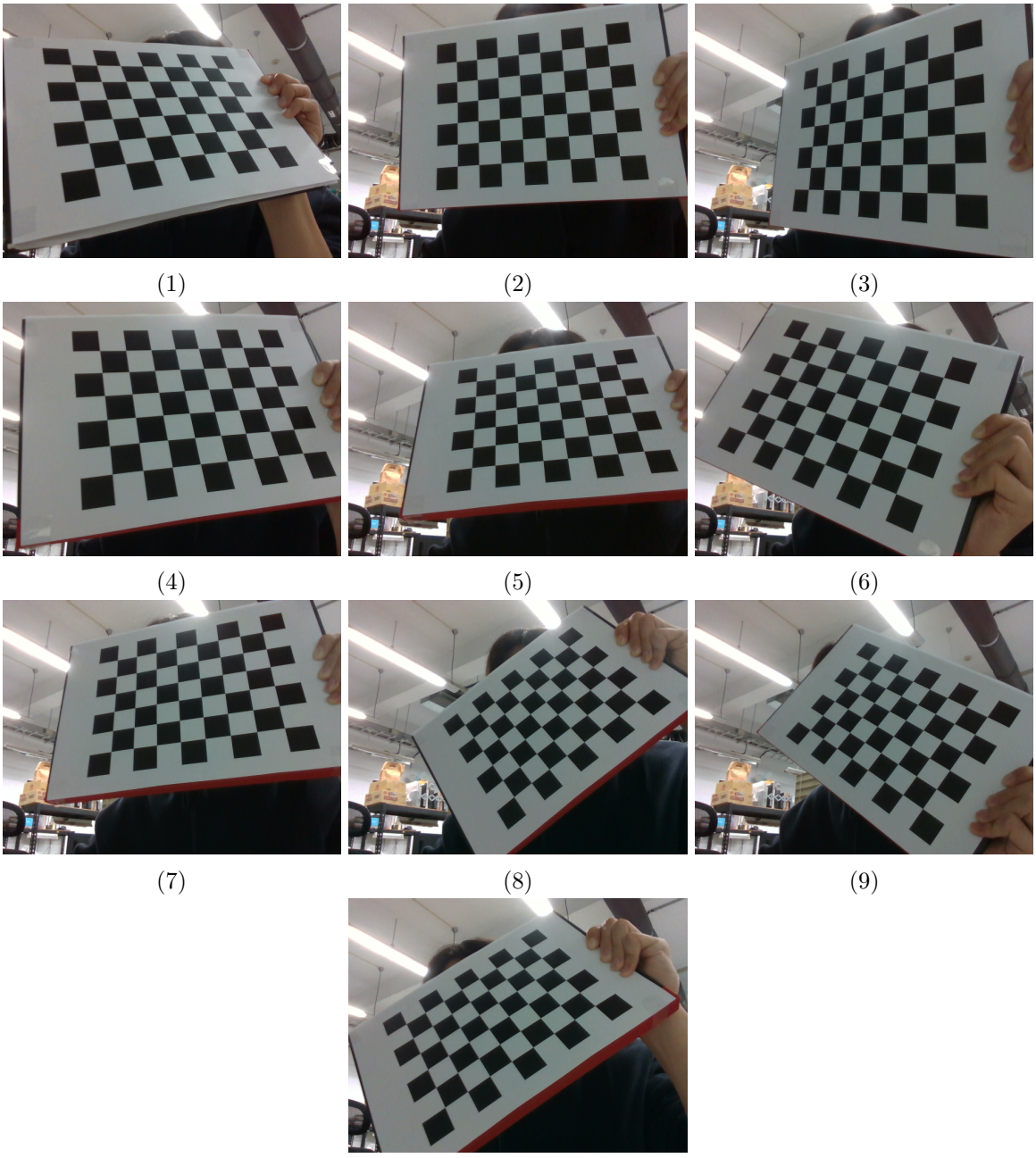
The resulting intrinsic parameters are:

$$\text{camera intrinsic matrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 585.83575158 & 0 & 329.7011727 \\ 0 & 585.44881547 & 268.63129684 \\ 0 & 0 & 1 \end{bmatrix}.$$

The intrinsic parameters can be understood as follows: f_x and f_y represents the focal length in the x and y direction respectively (think of it as the directional focal length of the lens the camera has); (c_x, c_y) is the distortion center of the camera, i.e. this point doesn't change position after distortion. All parameters are expressed in units of pixels.

3. Data

Original Photos



(1) (2) (3)

(4) (5) (6)

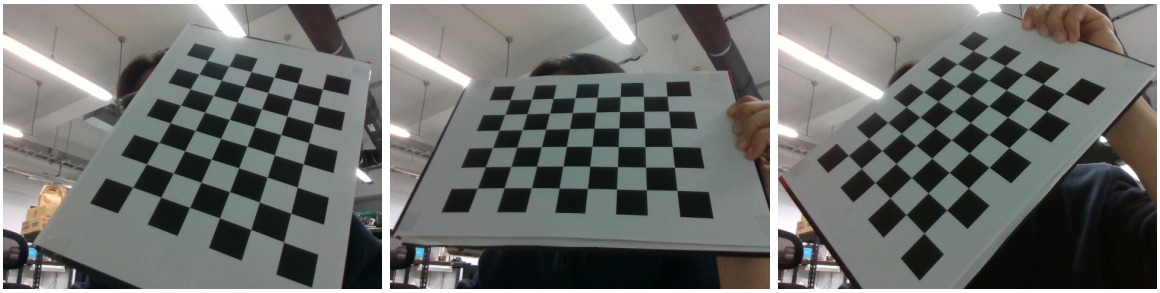
(7) (8) (9)

(10)

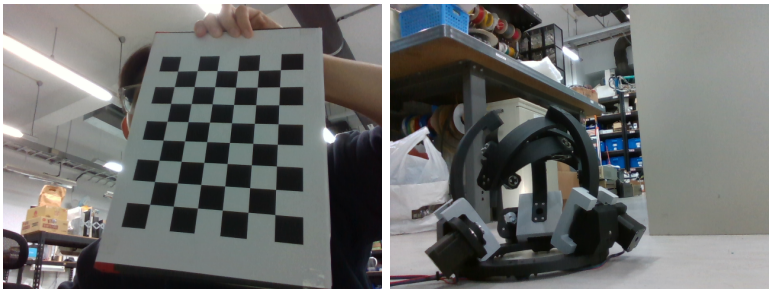
Figure 4: Original Photos (1∼10)

(11) (12) (13)

(14) (15) (16)



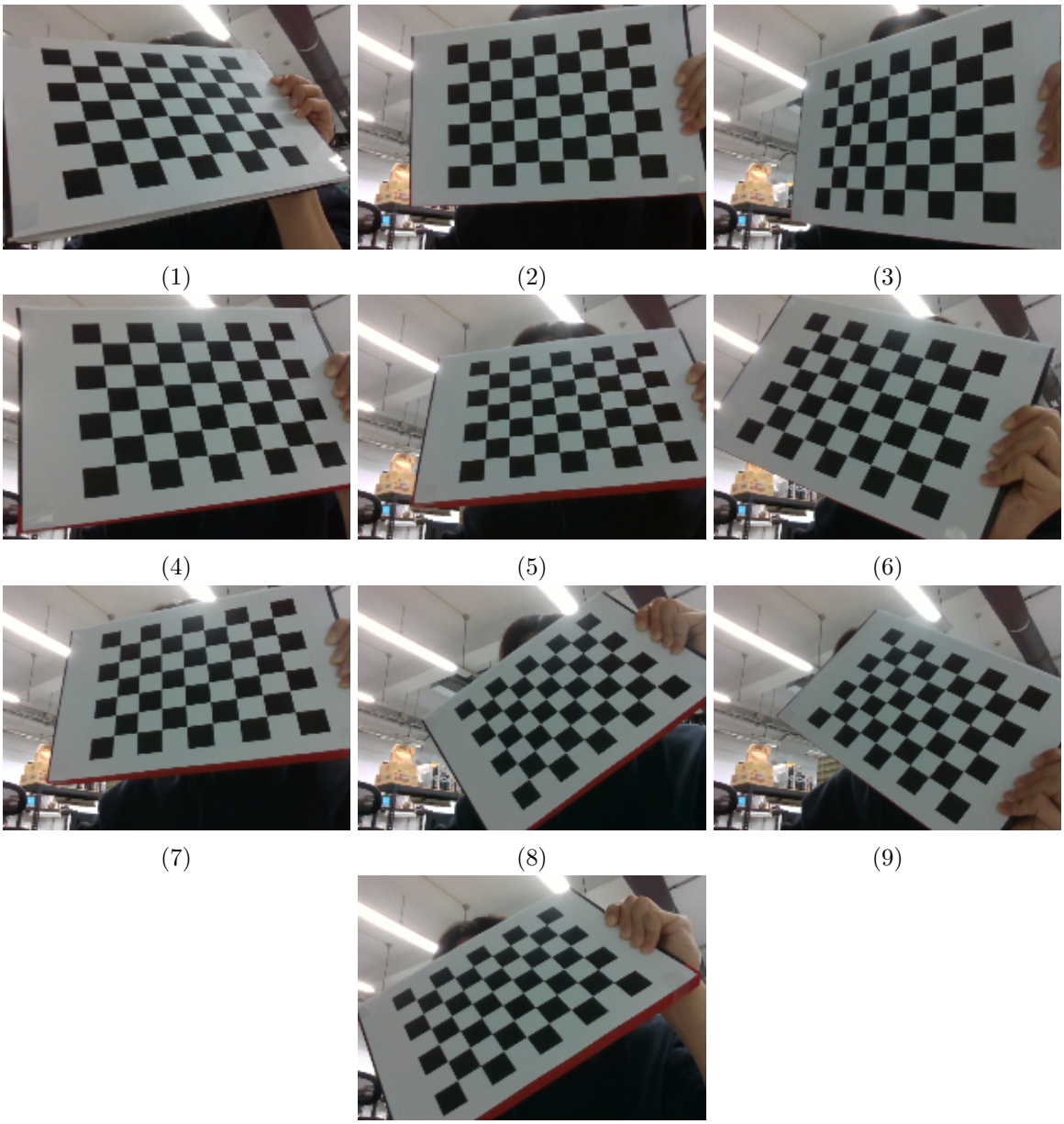
(17) (18) (19)



(20) (21)

Figure 5: Original Photos (11∼21)

Calibrated Photos



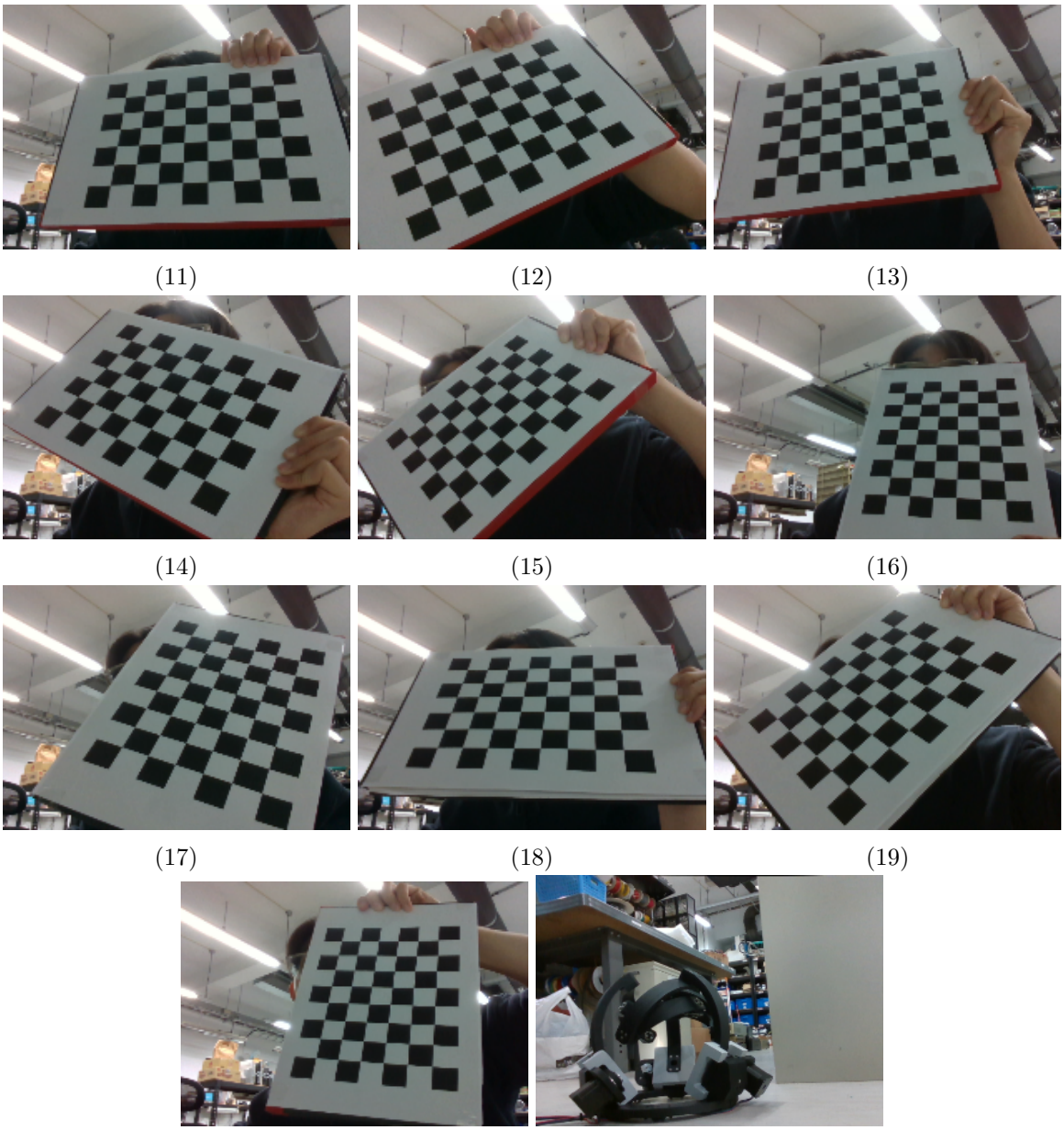
(1) (2) (3)

(4) (5) (6)

(7) (8) (9)

(10)

Figure 6: Calibrated Photos (1∼10)



(11) (12) (13)

(14) (15) (16)

(17) (18) (19)

(20) (21)

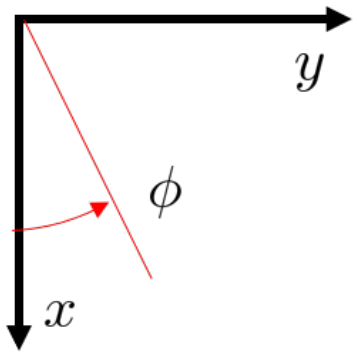
Figure 7: Calibrated Photos (11∼21)

Description of Differences

A mild difference in all the calibrated photos is that, when compared with the original ones, they are squished in the vertical direction and dilated in the horizontal direction.

Part B: Object Detection

The coordinate system in the slides is used (seen in the figure below).



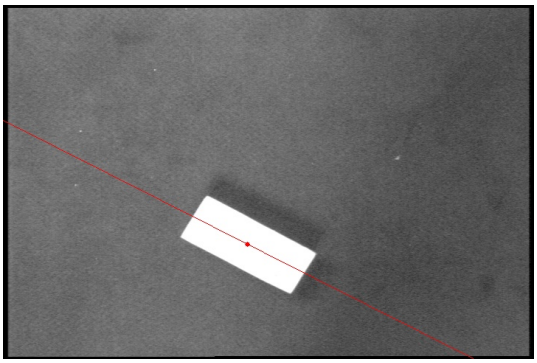
Firstly, the FindObjs function performs object detection through the following steps:

1. Convert the image to grayscale (if not already).
2. Blur the image.
3. Perform segmentation as described in the slides.

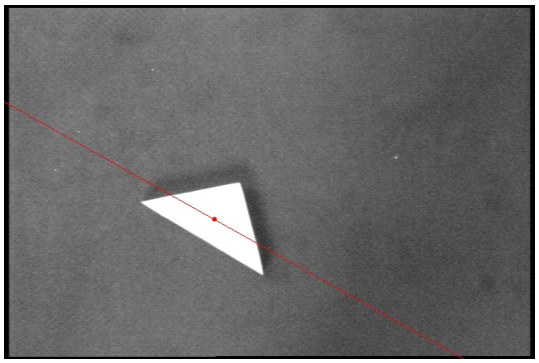
Secondly, the FindCL function finds the centroid (x_c, y_c) and principle angle ϕ for each object using the formulae in the slides.

Finally, for each object, the centroid and principle line are plotted, x_c, y_c, and ϕ are displayed through standard output.

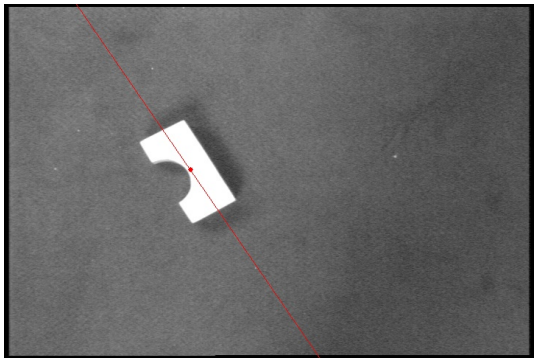
The results are shown as below in the next page:



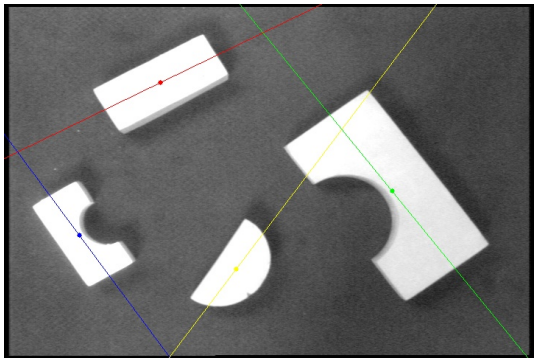
(1) er7-1.jpg
$(x_c, y_c, \phi) = (324.72, 331.84, 1.10)$



(2) er7-2.jpg
$(x_c, y_c, \phi) = (290.96, 285.19, 1.06)$



(3) er7-3.jpg
$(x_c, y_c, \phi) = (224.33, 253.14, 0.60)$



(4) er7-4.jpg
(red) $(x_c, y_c, \phi) = (106.22, 212.45, -1.12)$
(yellow) $(x_c, y_c, \phi) = (253.31, 527.49, 0.69)$
(green) $(x_c, y_c, \phi) = (313.60, 102.17, 0.64)$
(blue) $(x_c, y_c, \phi) = (359.15, 315.51, -0.65)$

Figure 8: Centroid and Principle Line of Objects

Division of Work

Che-Jung Chuang:	Report on part B
Cheng-Yen Yu:	Major contribution on coding of both part A and B
Wei-Hsuan Cheng:	Coding of both part A and B, preparing photos of part A, and report summarizing
Wen Perng:	Report on part A