**Report of Project 1**

**1, Algorithms:**

1, transfer the image into gray scale image by the formula:

Gray = 0.30 x Red + 0.59 \* Green + 0.11 \* Blue

2, transfer the gray image into normalized gradient magnitude image by convoluting the gray image with Sobel’s operator.

3, select a threshold to transform the image into binary edge map image. The threshold selected should retain most of the parallelograms in the image and make the edge of the image as thin as possible.

4, detected direct lines by using hough transform

(1), select suitable quantized parameters to quantize the parameter coordinate.

Here, we quantized the slope space into 360 accumulator cells, and quantize the p value space in to 2 x image\_width + 2 x image\_height accumulator cells.

(2), Assume that each cell in the parameter space is an accumulator. Initialize all cells to zero

(3), for each point (x, y) in the image space, increment by 1 each of the accumulators that satisfy the equation

(4), set a threshold to all the points in the parameter space, in each cell if the count is larger than the threshold, then it could be a direct line. Again the threshold here should retain as many as parallelograms as possible.

5, combine similar detected direct lines by keeping the line with the local maximum count and removing the other lines with less count. Similar lines can be defined as lines with similar slope, p value in the parameter space.

6, when the number of the detected direct lines is smaller enough. We select pairs of lines by certain conditions. Lines pair together should have similar slope, similar count in parameter space and a certain distance away.

7, select two pairs of lines and go over the four lines pixel by pixel, record the number of pixels that is foreground and background respectively. Compare the two numbers and set a threshold to determine whether the graph is parallelogram or not.

8, if the graph is parallelogram, then calculate it cross points and mark the graph on the input image.

9, repeat seven and eight until all pairs of lines were checked.

**2, Program Language selection and Instructions of program compiling and running**

**2.1, programming language selection**

I use Python 3.5.2 to compile my program.

**2.2, Instructions for program compiling and running**

Before the program was compiled, we have to install the module “Pillow”, one of the most popular image processing module, into our python environment. The introduction and the instruction of installing and using “Pillow” can be found on the website: <http://pillow.readthedocs.io/en/4.3.x/>.

For the purpose of easy to examine, I programed all my code into one file. After the “pillow” module was installed, we can just run the program by clicking run on the IDE program or use the command “python Parallelograms\_detection.py” in a command line window. Then the program will automatically print out the 4 corner points detected of a parallelogram and save normalized gradient magnitude image, edge map image and the final image with parallelograms detected superimposed on the original image.

**3, Results**

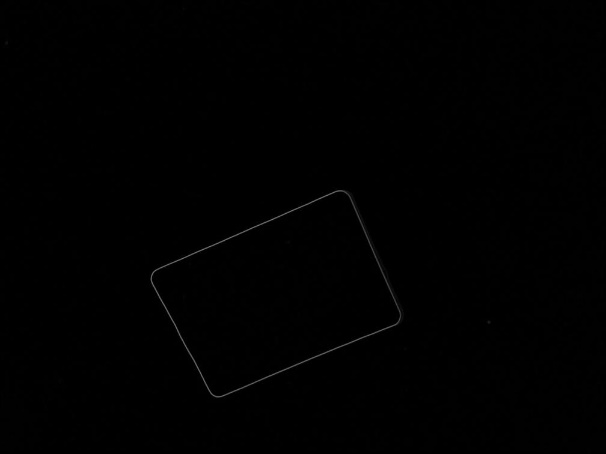
**3.1 For the first image:**

The threshold use is **50**

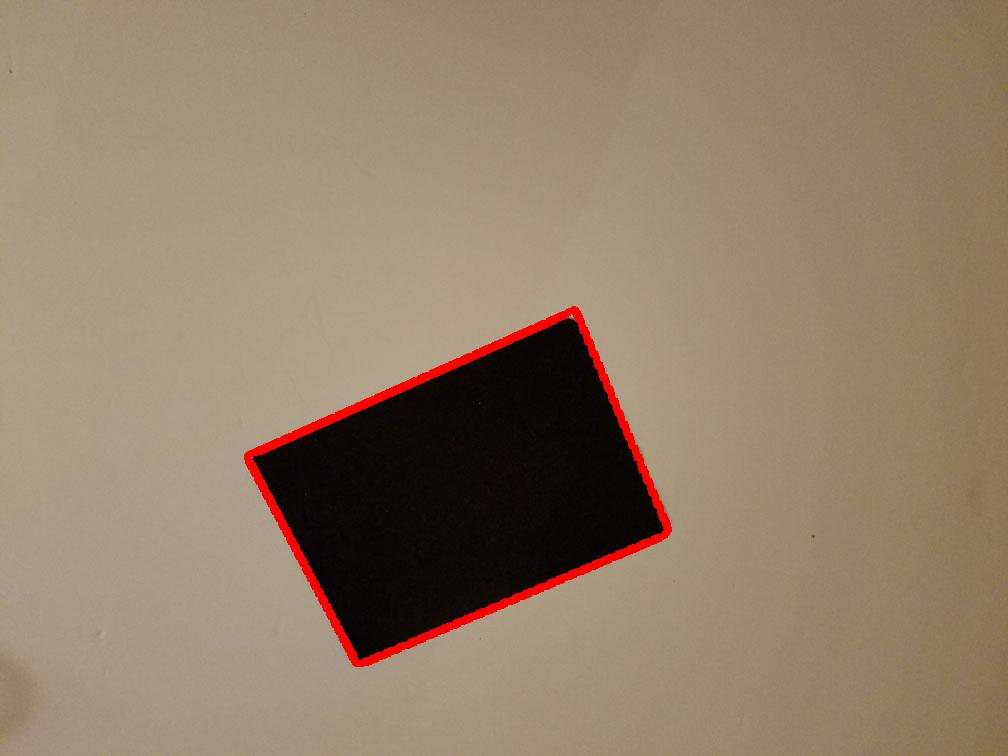
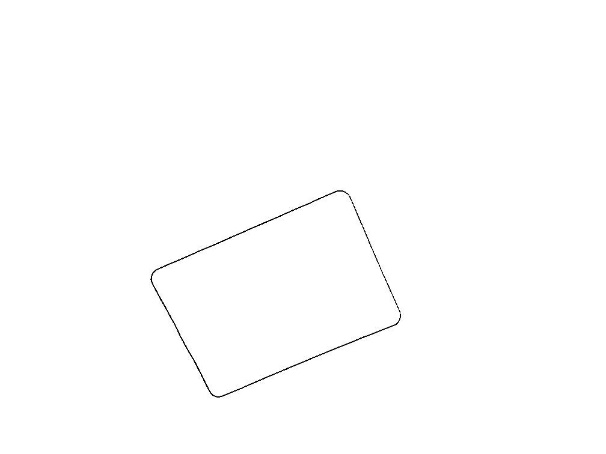
The four cross points of the parallelogram detected in the first image is

**[(246, 456), (357, 665), (574, 308), (669, 531)]**

The original image and the detected parallelogram is showed in below images.



1. The original image (b) Normalized gradient magnitude image



(c) Binary edge map image (d) Final output image

**3.2 For the second image**

The threshold used was **30**

Four parallelograms detected, two of them are actually the same parallelogram in the original image.

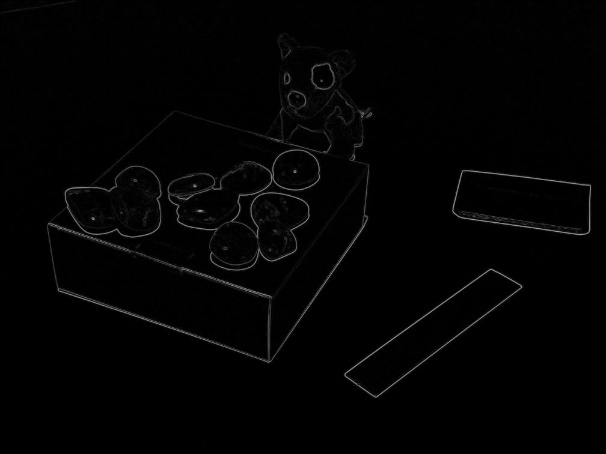
The cross points of the three parallelogram are:

Lower left: **[(95, 478), (77, 364), (444, 602), (447, 496)]**

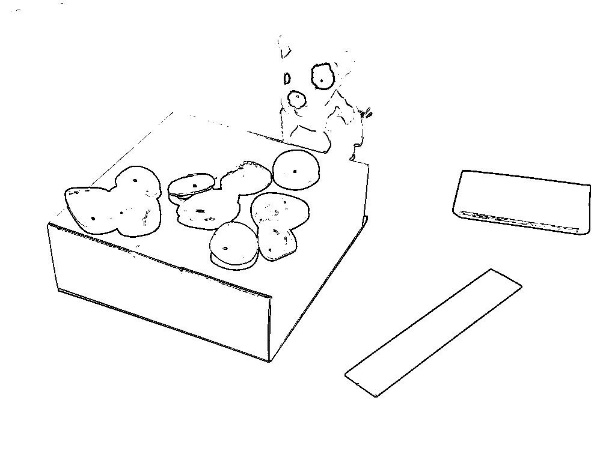
Lower right:  **[(869, 476), (818, 445), (624, 660), (570, 624)]**

Upper right: **[(981, 308), (980, 392), (770, 282), (749, 361)]**

The original image and the detected parallelogram is showed in below images.



1. The original image (b) Normalized gradient magnitude image



(c) Binary edge map image (d) Final output image

**3, For the third image**

The threshold used was **15**

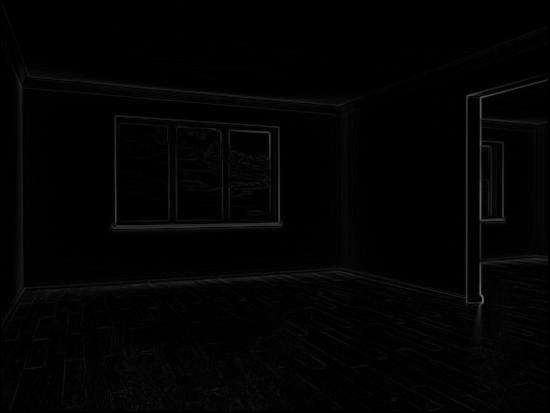
Three parallelograms detected

The outer parallelogram: **[(279, 223), (278, 126), (116, 226), (115, 115)]**

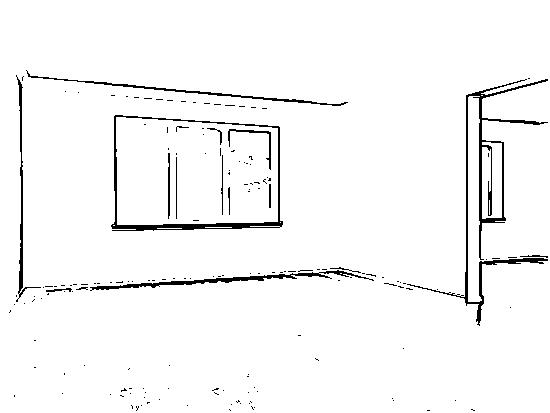
The right bigger parallelogram: **[(279, 223), (278, 126), (176, 224), (176, 119)]**

The left smaller parallelogram: **[(116, 226), (115, 115), (176, 224), (176, 119)]**

The original image and the detected parallelogram is showed in below images.



1. The original image (b) Normalized gradient magnitude image



(c) Binary edge map image (d) Final output image