Machine Vision Assignment 1

Name: Marcus Wong Yew Hon

ID: R00183595

Table of Contents

Task 1 2
- Subtask A 2
- Subtask B 2
- Subtask C 10
- Subtask D 16
- Subtask E 16
- Subtask F 20
- Subtask G 20

Task 2 25
- Subtask A 25
- Subtask B 25
- Subtask C 27

Task 1

All code implementations are in the **mv_assignment1.py** file, each question subtasks will be labelled with its respective implementation line with reference to the **mv_assignment1.py** file.

Subtask A

Load input image and convert to single channel grey value image: **lines 12-19**

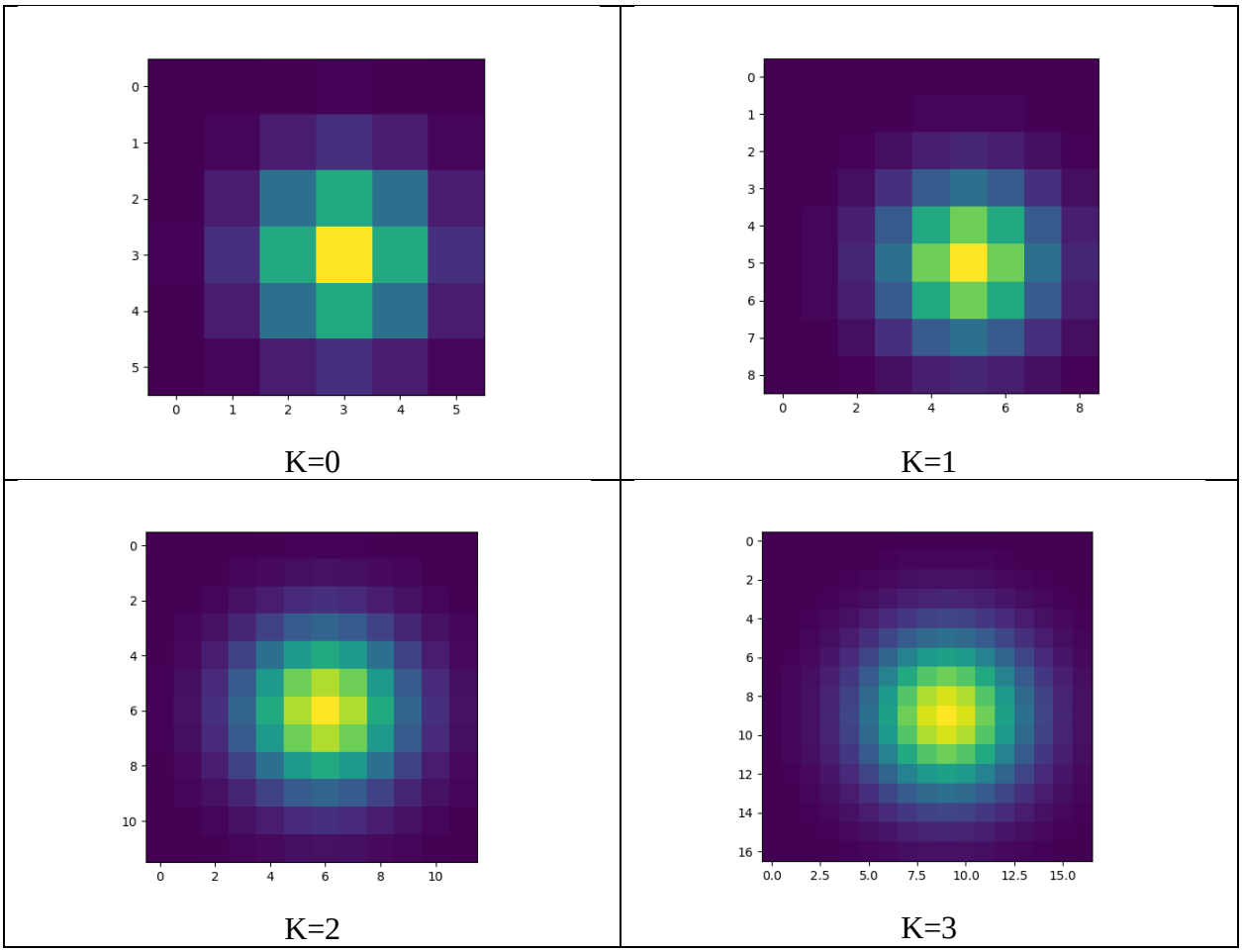
Convert image to data type float32: **line 20**

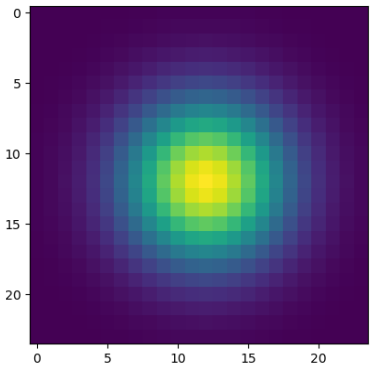
Determine the size of the image and double the image size: **lines 23-25**

Subtask B

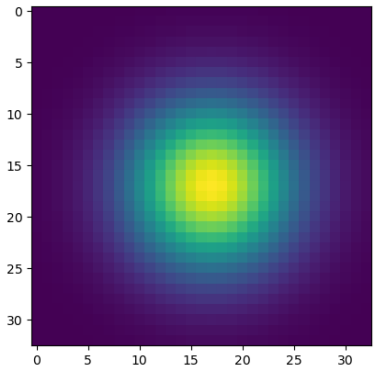
This subtask was implemented using a for-loop in **lines 31-50.**

Results for all 12 kernels (k = 0, …, 11) plotted as an image:

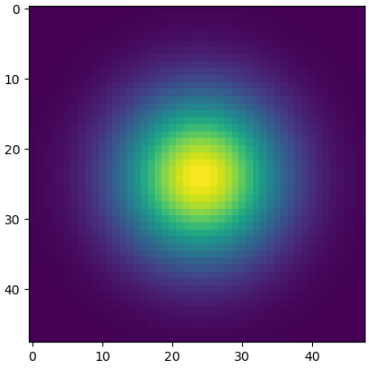




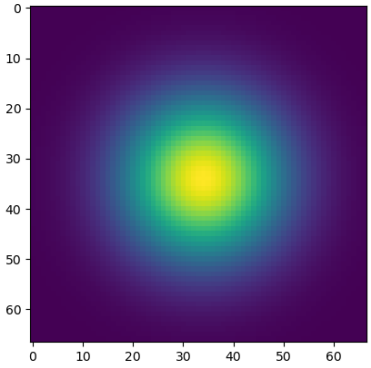
K=4



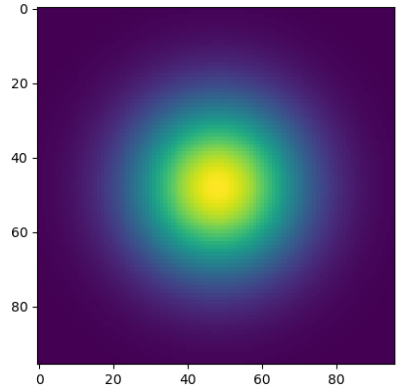
K=5



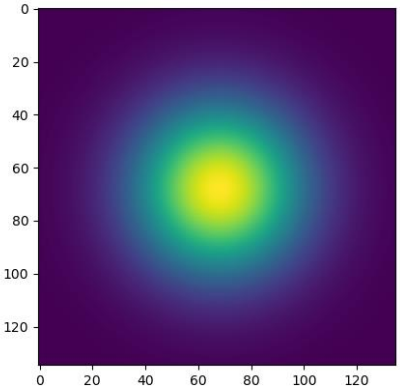
K=6



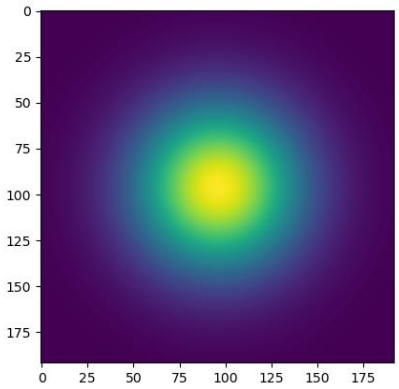
K=7



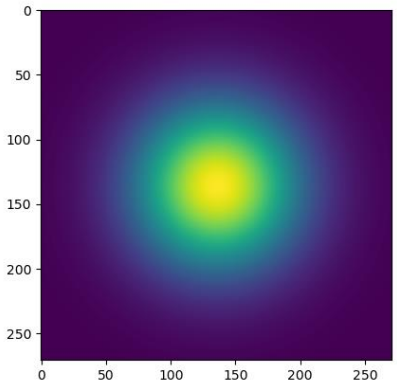
K=8



K=9



K=10



K=11

Results of all resulting scale space images after applying the 12 kernels above:

K=0

K=1

K=2

K=3

K=4

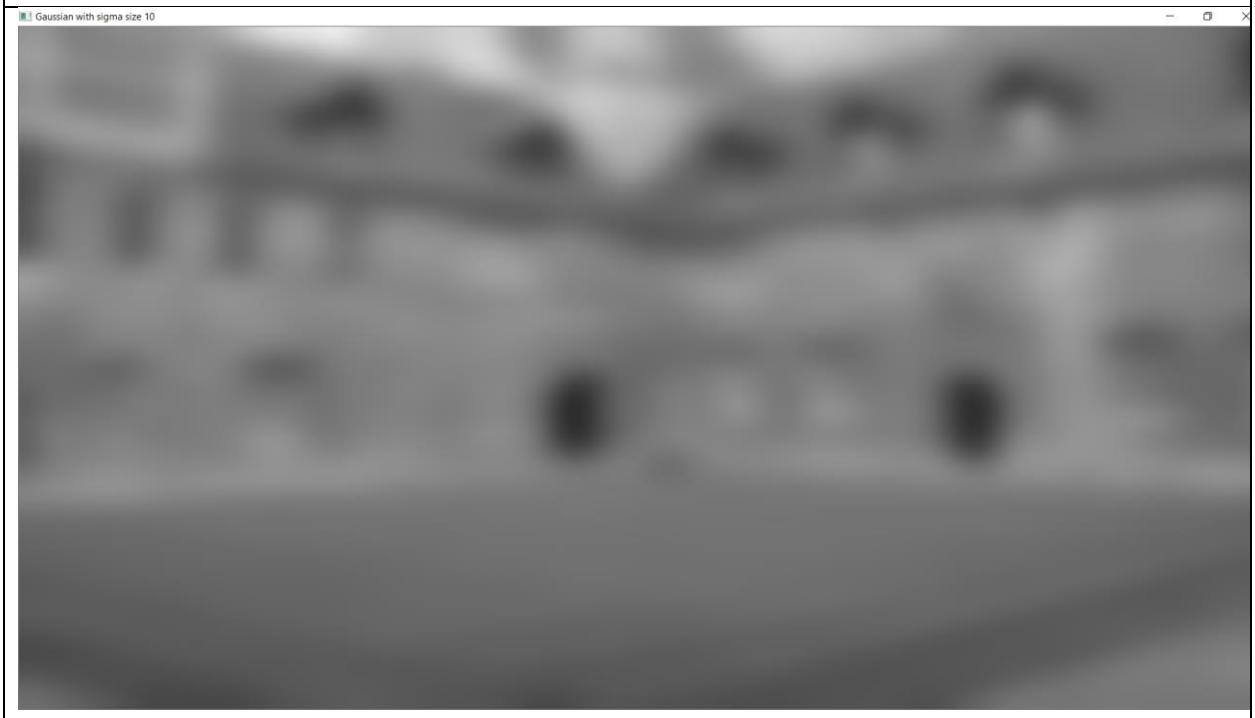
K=5

K=6

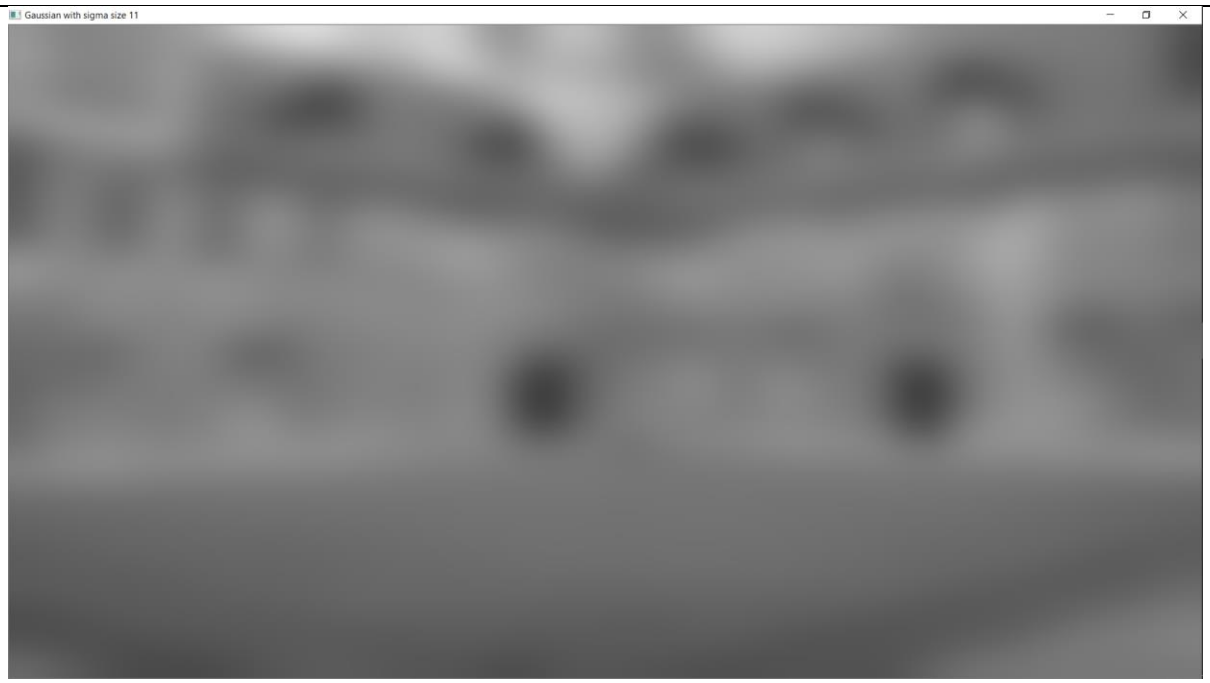
K=7

K=8

K=9



K=10



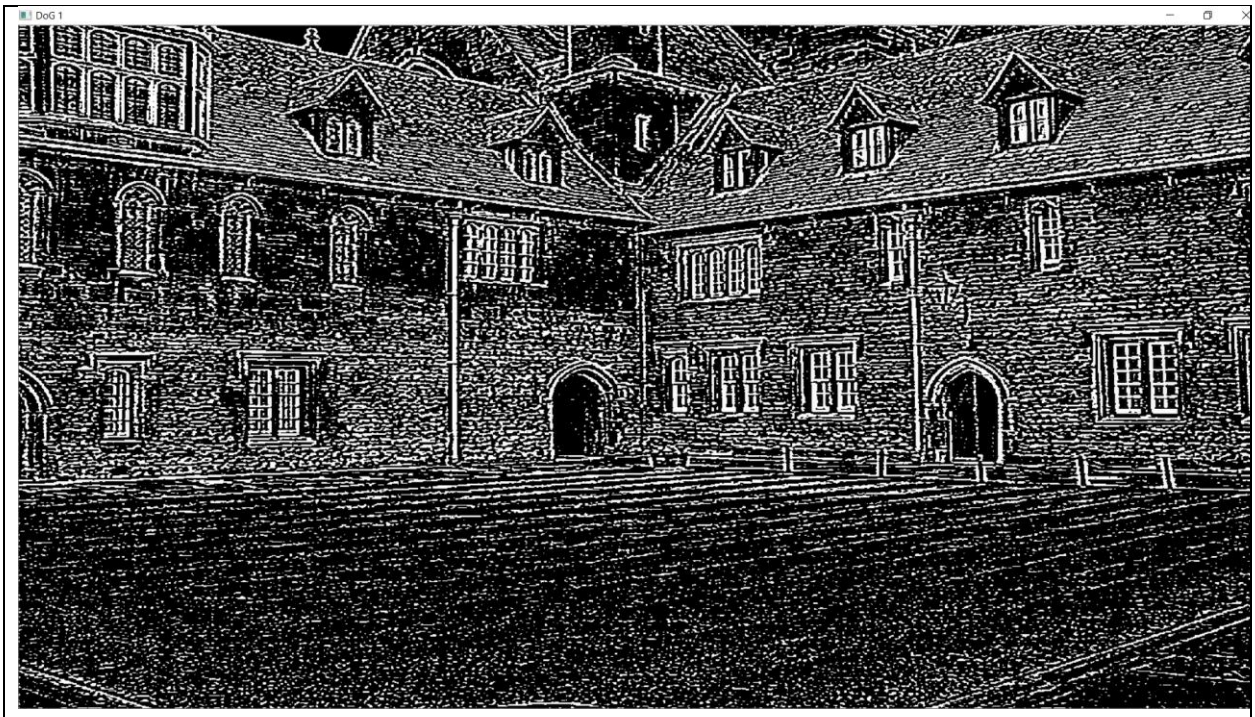
K=11

Subtask C

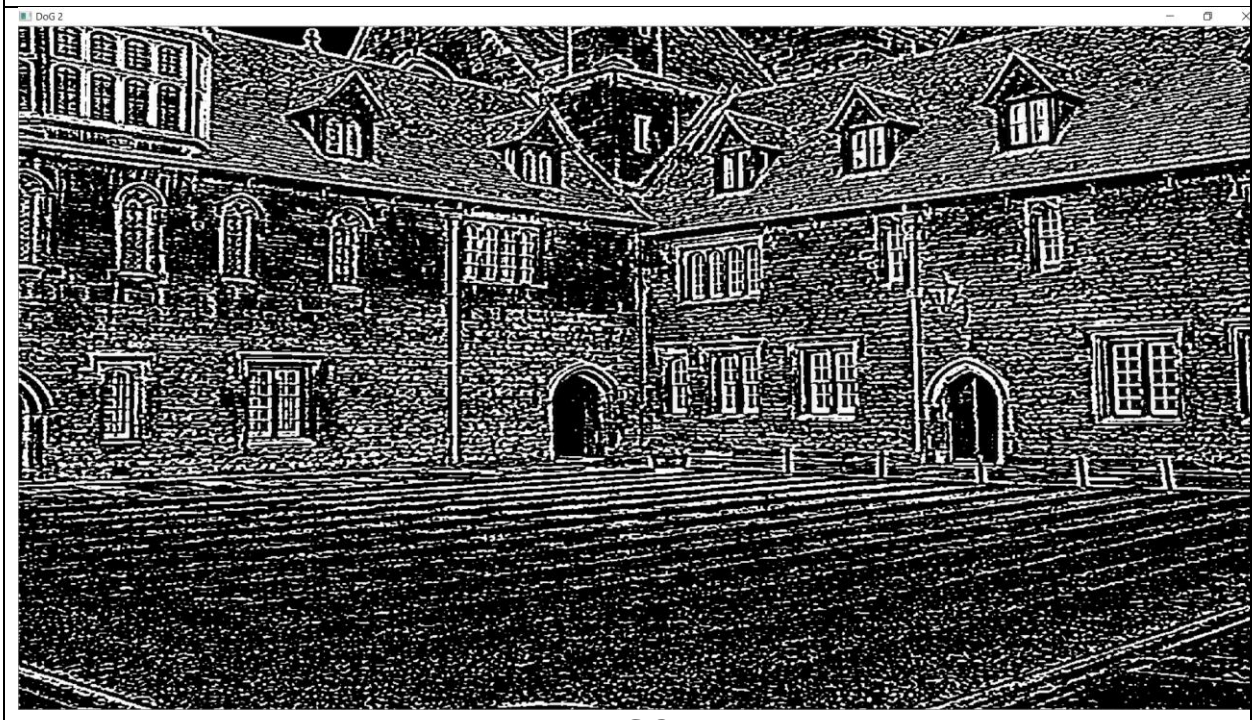
Calculate difference of Gaussian for all scales: **lines 68-83**

Resulting 11 DoG images are as below:

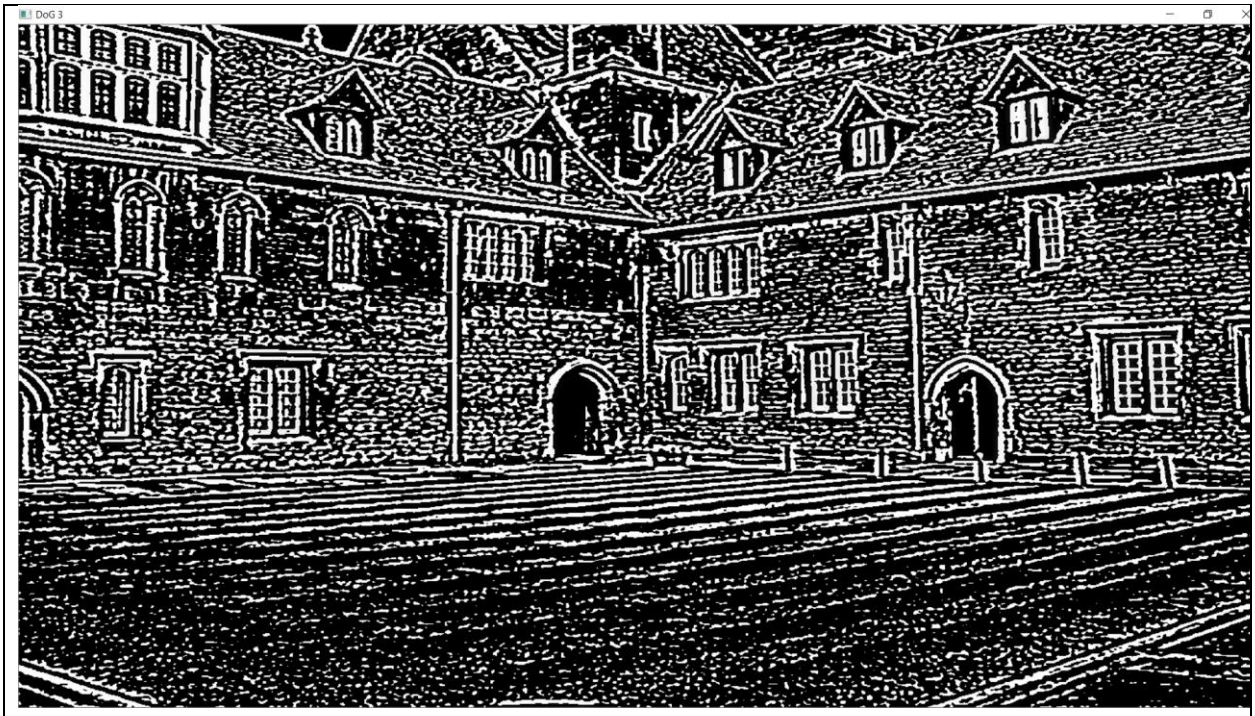
DoG 1



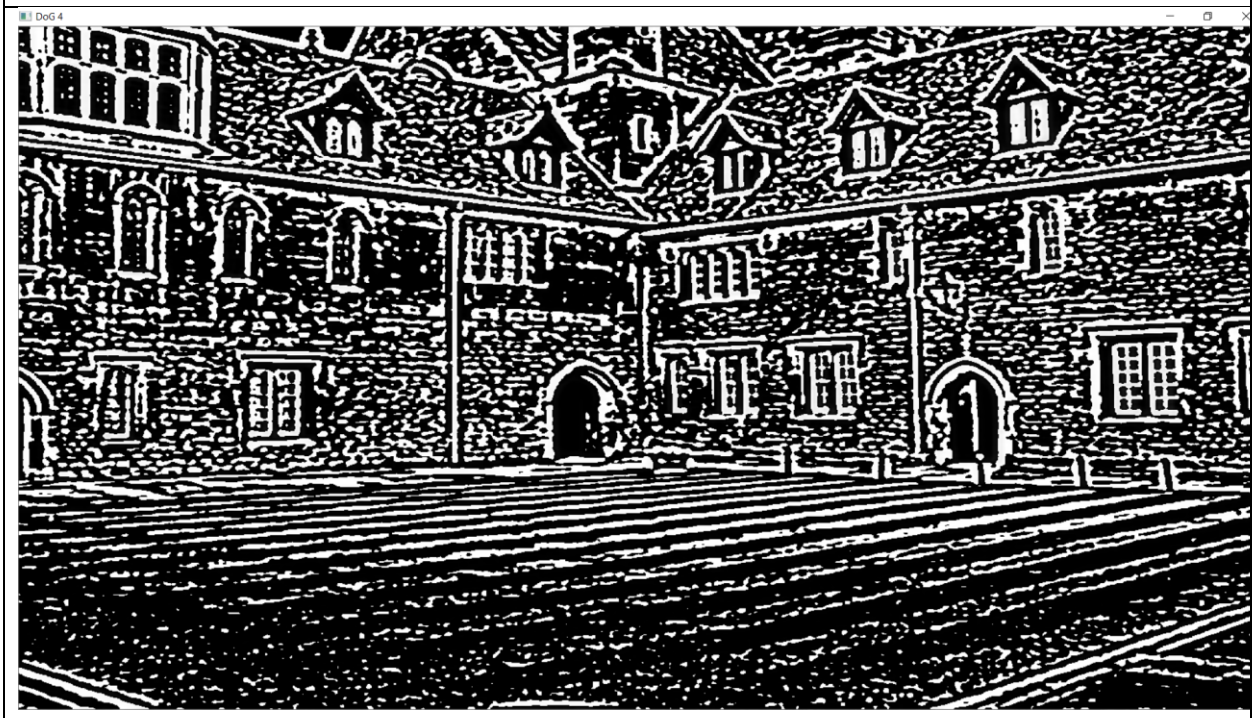
DoG 2



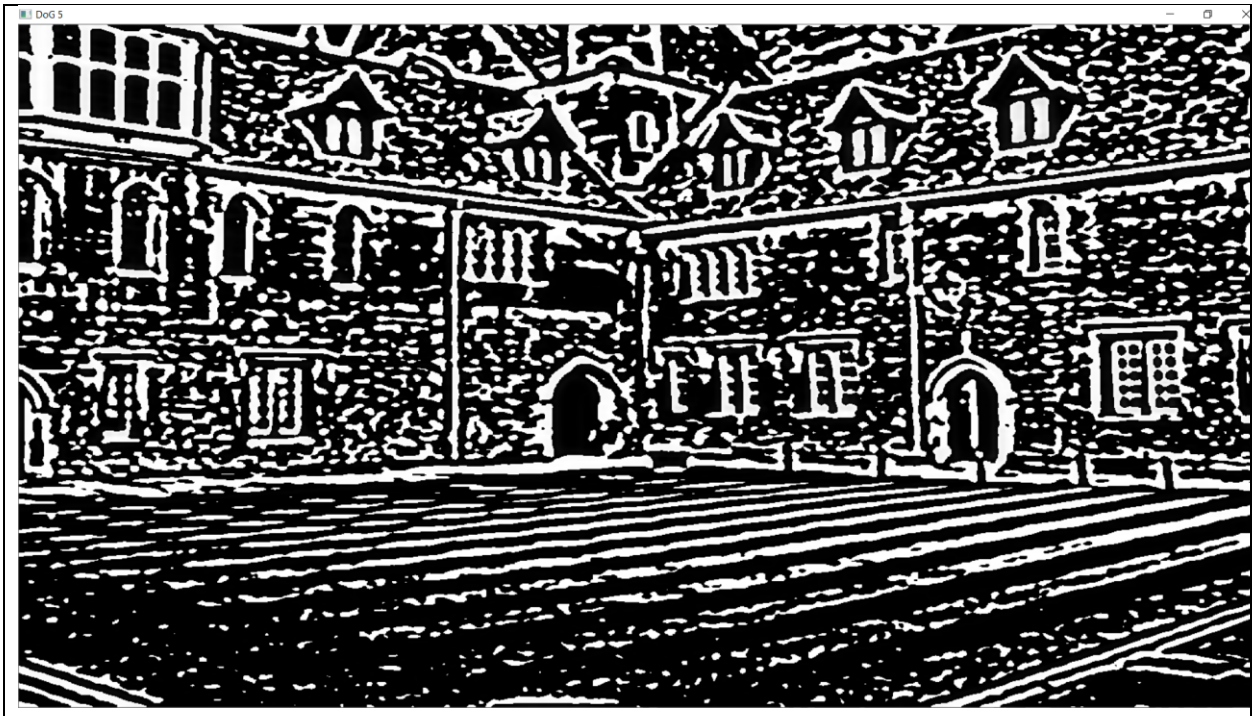
DoG 3



DoG 4



DoG 5

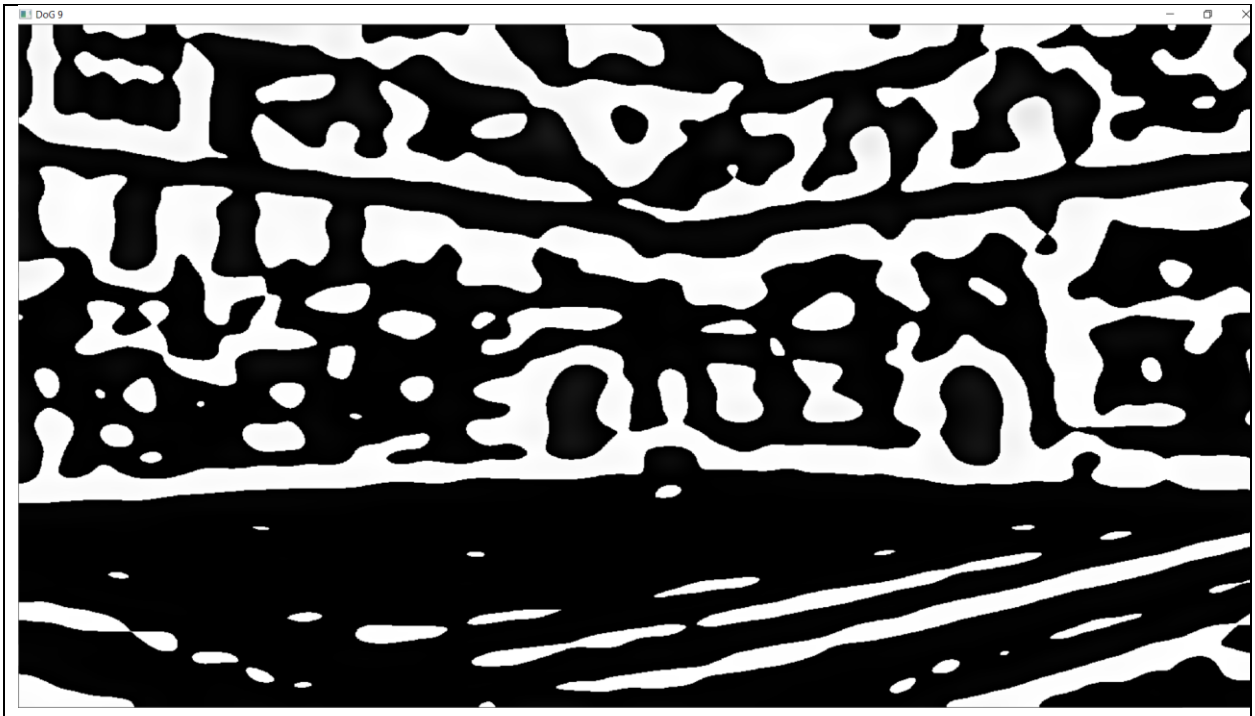


DoG 6

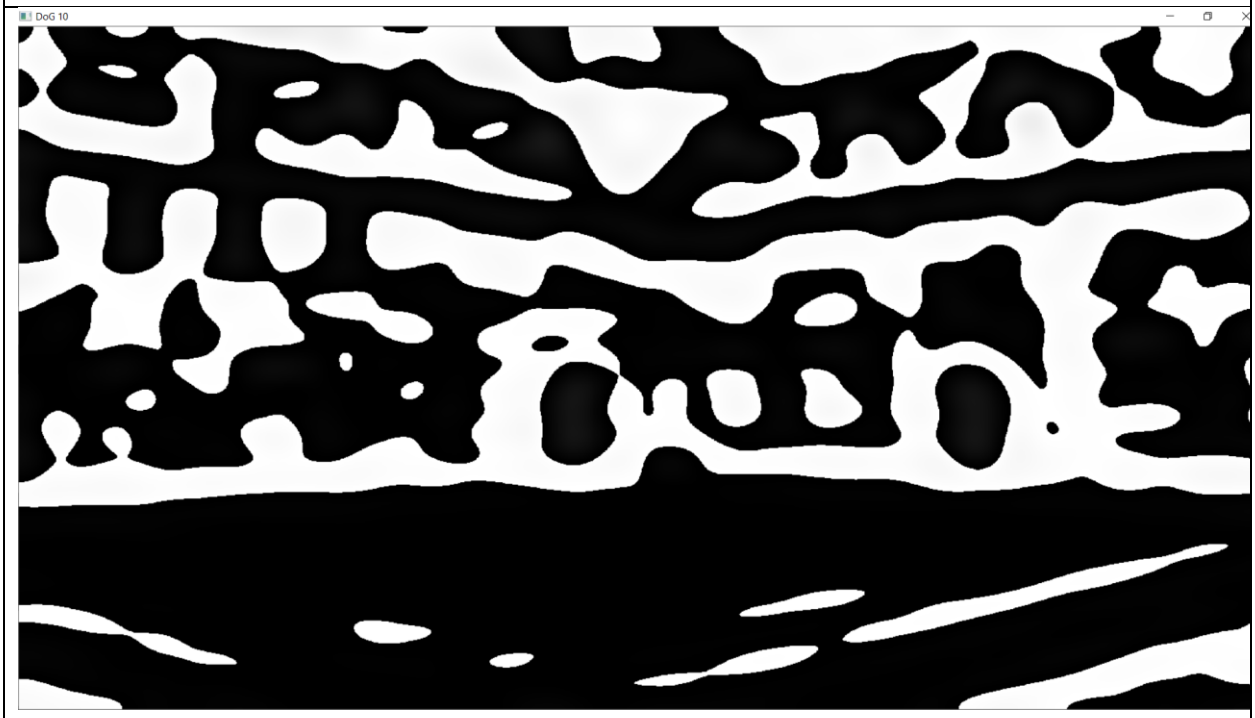
DoG 7

DoG 8

DoG 9



DoG 10



DoG 11

Subtask D

A loop to compare the stack of 3 images for 26 neighbour non-maximum suppression can be found in **lines 141-146.**

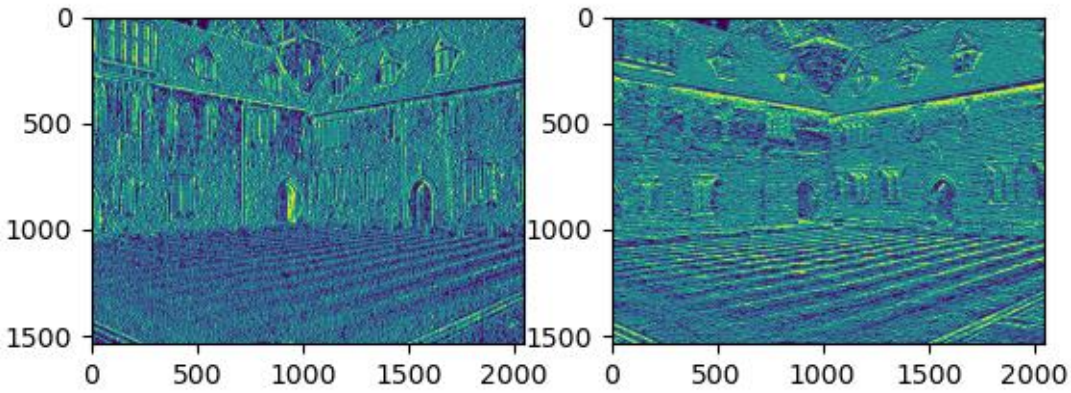
The implementation code for the non-maximum suppression itself is enveloped in a function and can be found in **lines 85-135.**

Subtask E

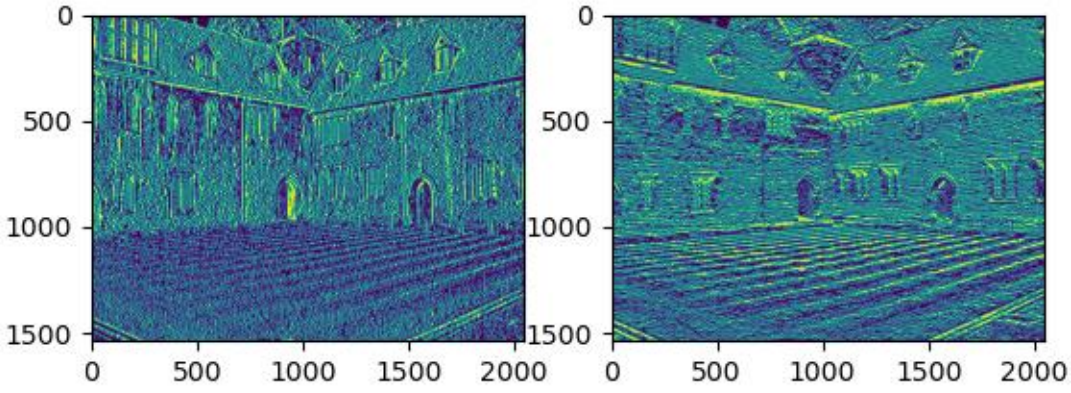
The dx and dy kernels are initialized in **lines 153-156.**

The kernels are then applied to all images from subtask B, which was implemented in **lines 160-175**

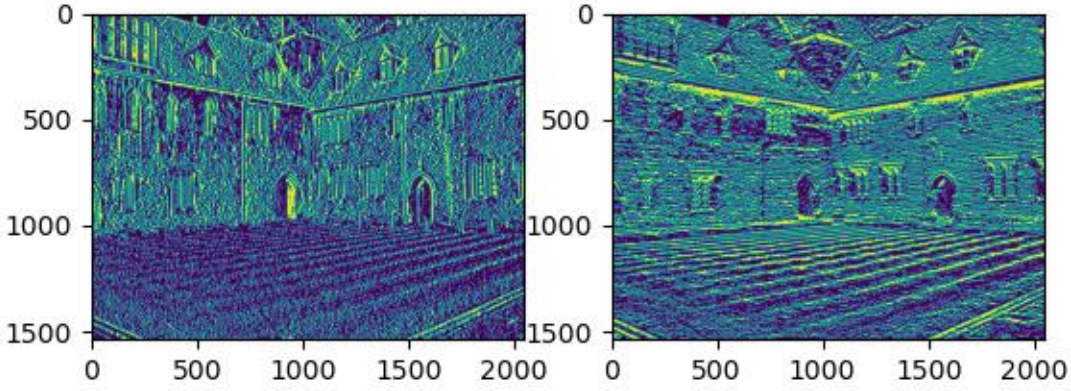
The resulting XY derivative images for all 12 images in subtask B are shown below, the left image shows the x-derivative and the right image shows the y-derivative.



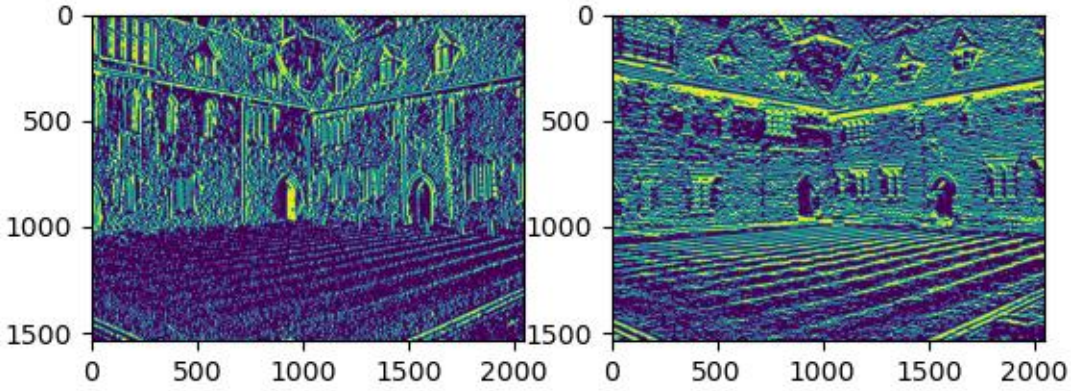
Derivatives for image 1



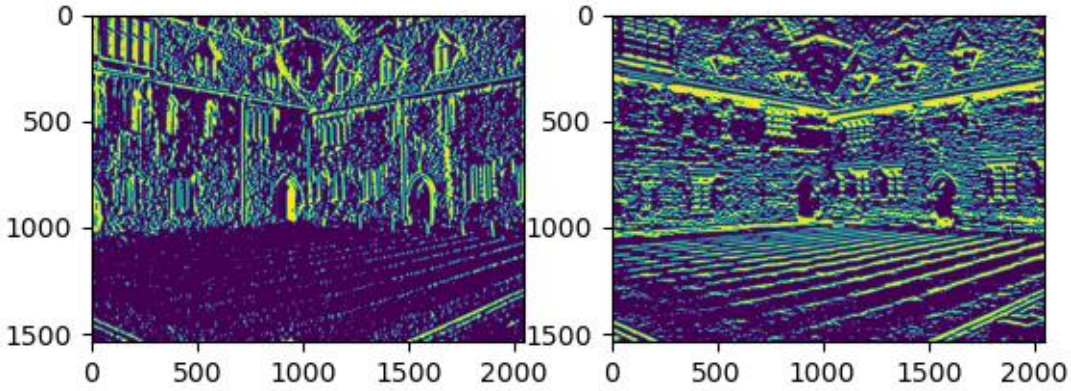
Derivatives for image 2



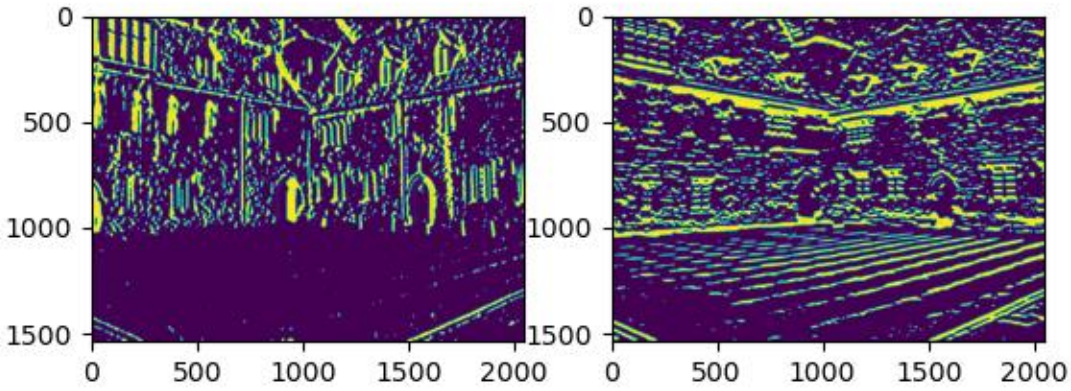
Derivatives for image 3



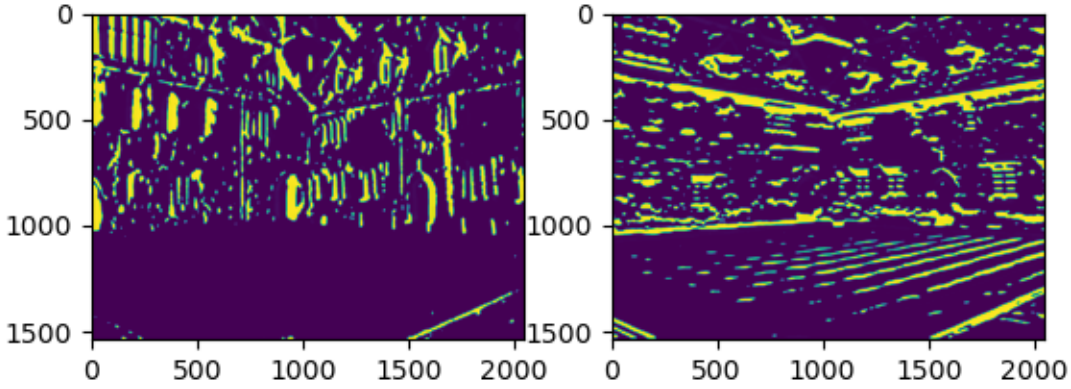
Derivatives for image 4



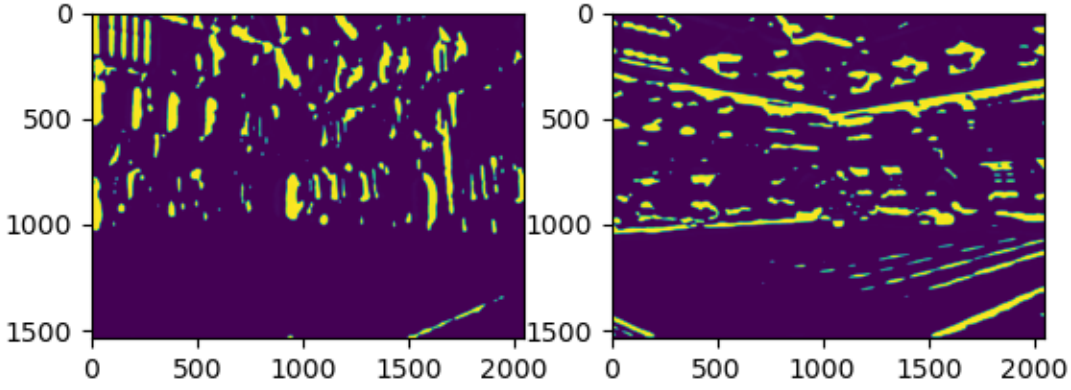
Derivatives for image 5



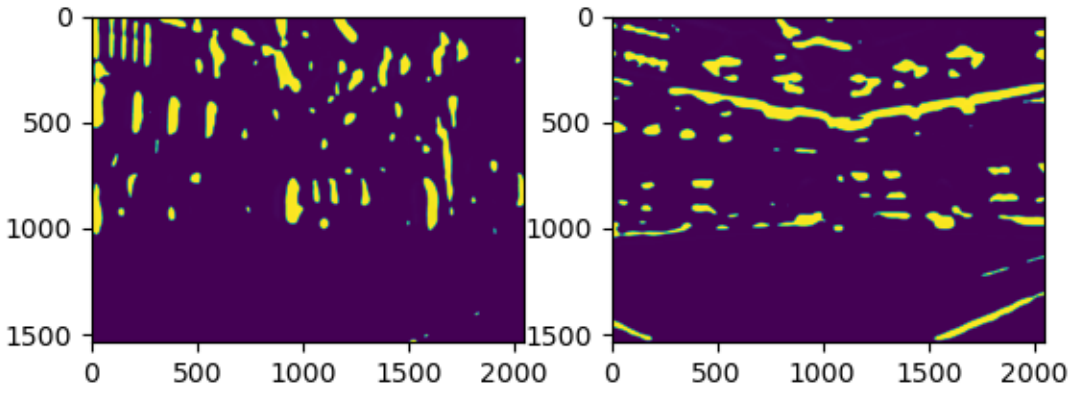
Derivatives for image 6



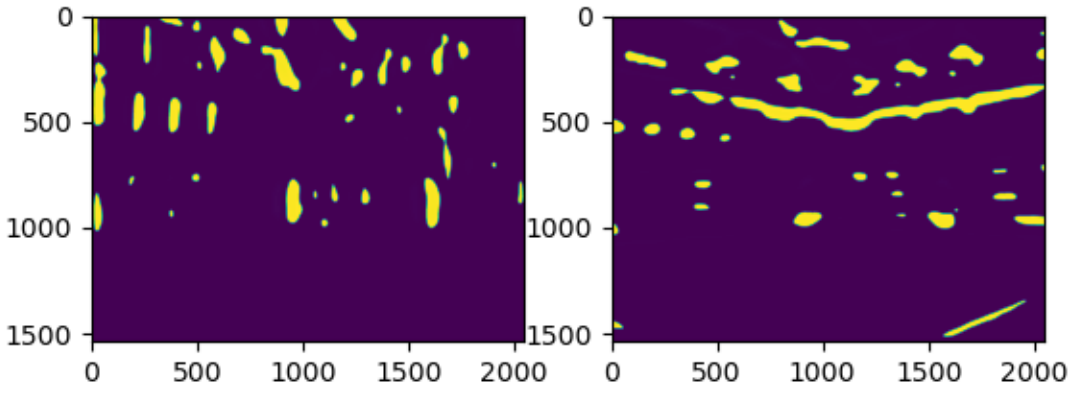
Derivatives for image 7



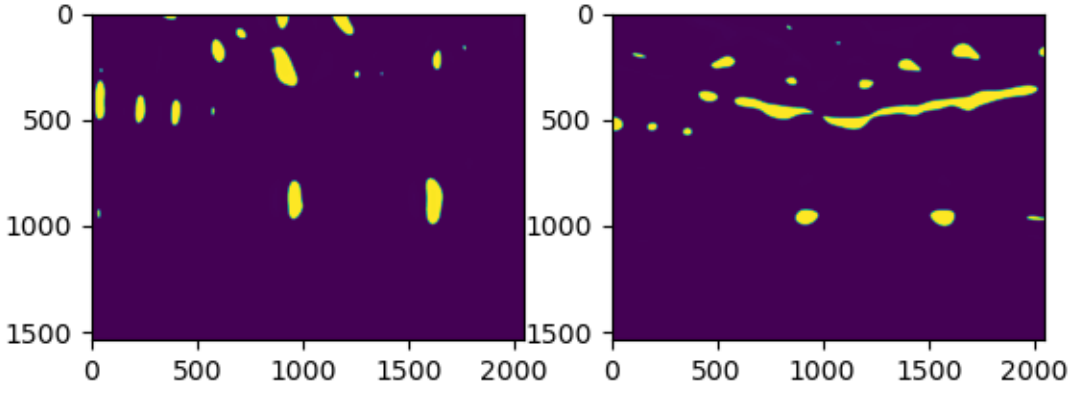
Derivatives for image 8



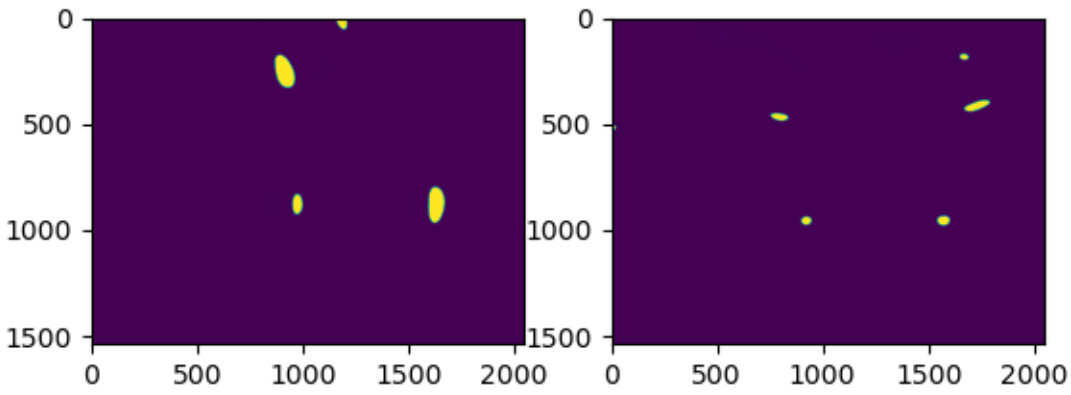
Derivatives for image 9



Derivatives for image 10



Derivatives for image 11



Derivatives for image 12

Subtask F

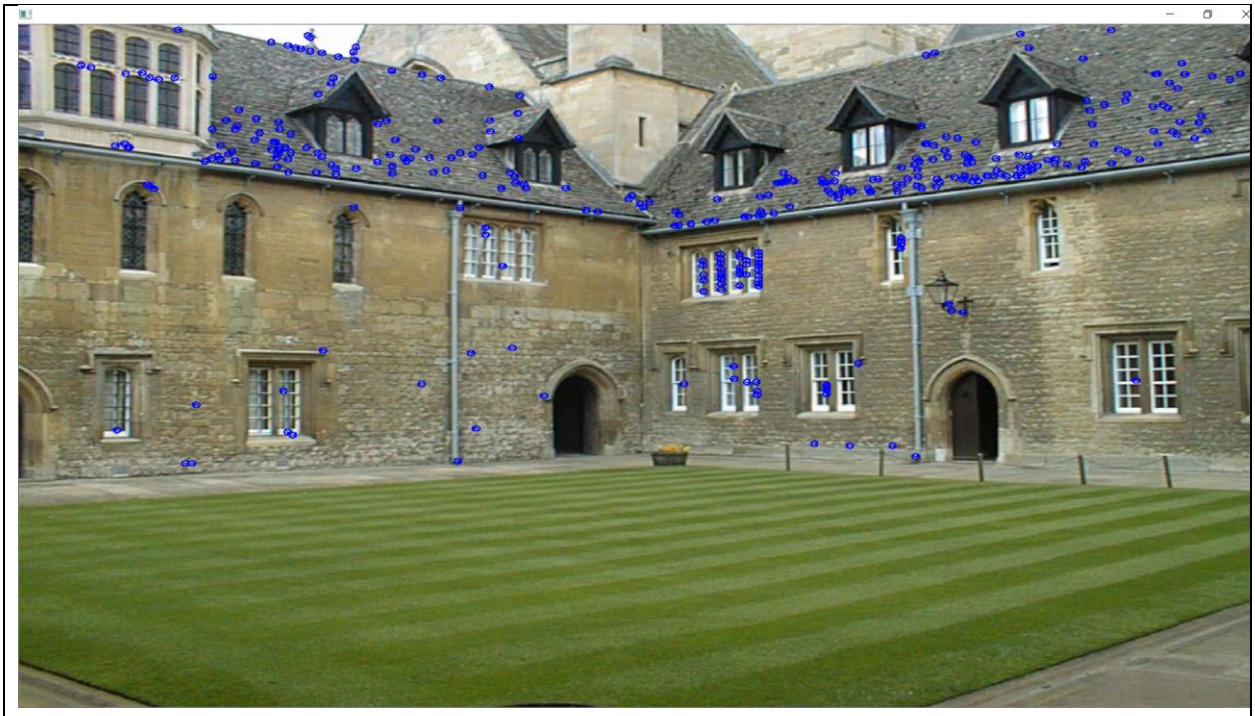
In this subtask, the gradient magnitude and gradient length was calculated for the entirety of each derivative belonging to its key-point image's sigma value obtained in subtask D **(lines 233-243)**. Once the gradient magnitude and gradient length are calculated, a 7x7 window was used to slide through each detected key-point coordinate obtained from subtask D. The weighting function was then applied to the 7x7 window upon the calculated magnitude image and an angle conversion was performed on the calculated gradient direction image through the 7x7 window, which converts degree angle values to encompass a range of up to 360 degrees **(lines 249-258)**.

The 7x7 window values of gradient magnitude and gradient length were then fed into the 36-bin histogram to obtain the mode gradient direction value. The mode direction value will then determine the final angle of the detected key point **(lines 261-265)**.

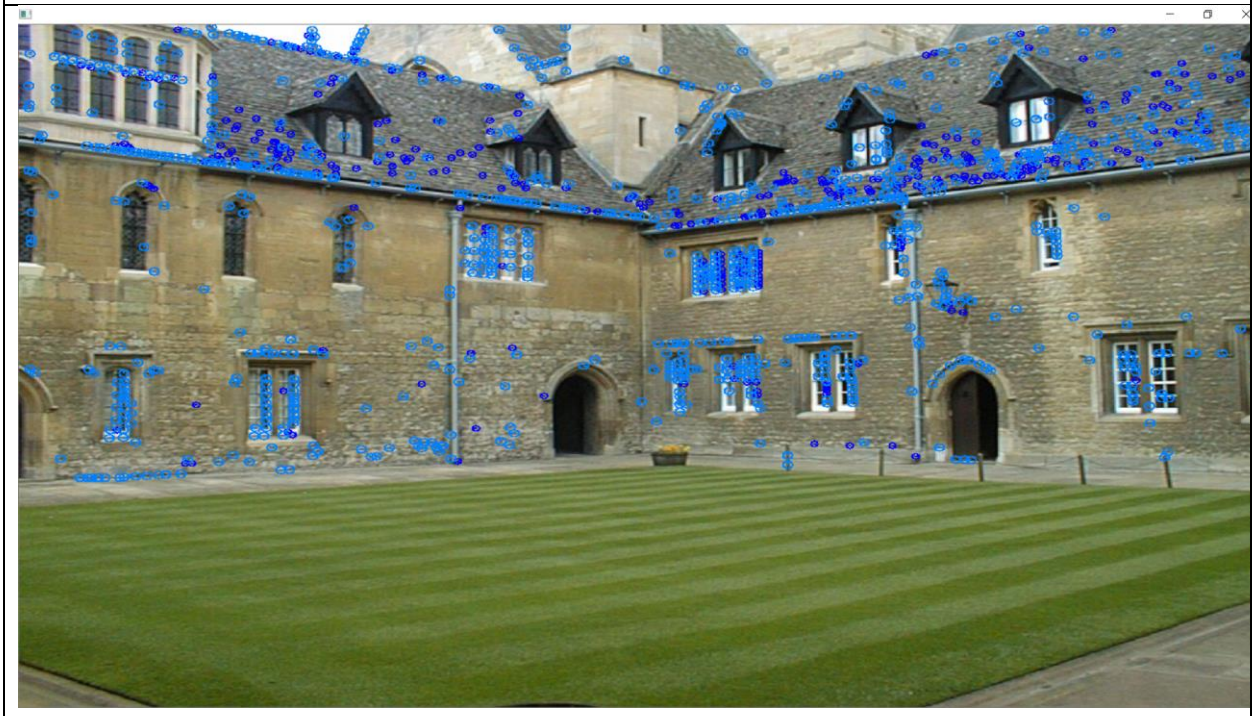
Functions for gradient magnitude calculation **(lines 183-185),** gradient direction calculation **(lines 187-190),** weighting function calculation **(lines 192-200)** and histogram calculation **(lines 202-217)** were created.

Subtask G

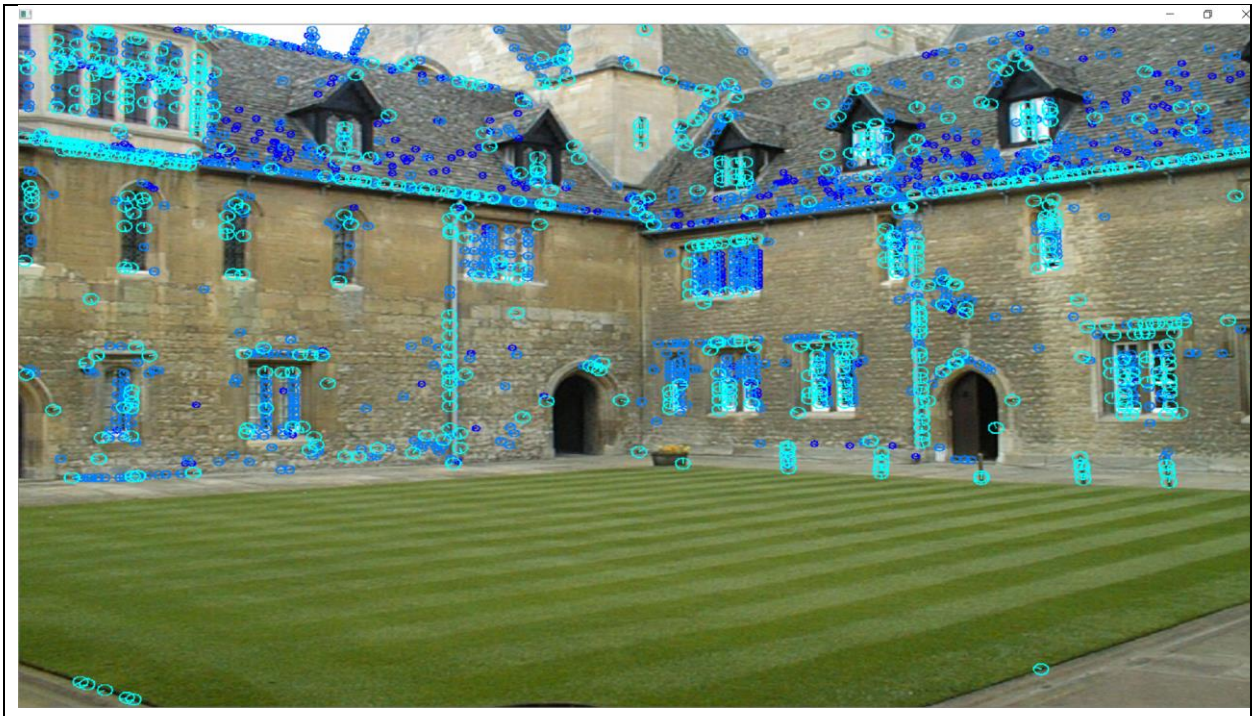
Circles are drawn around each key point, along with lines within the radius of the drawn circle to indicate key point direction **(lines 268-271)**. Each of the 9 layers of image key points drawn were given different colours, each colour representing different sigma key points. Each layer of key points drawn are displayed **(lines 281-285).** Each of the 9 layers of key points with its angle direction will be displayed below:



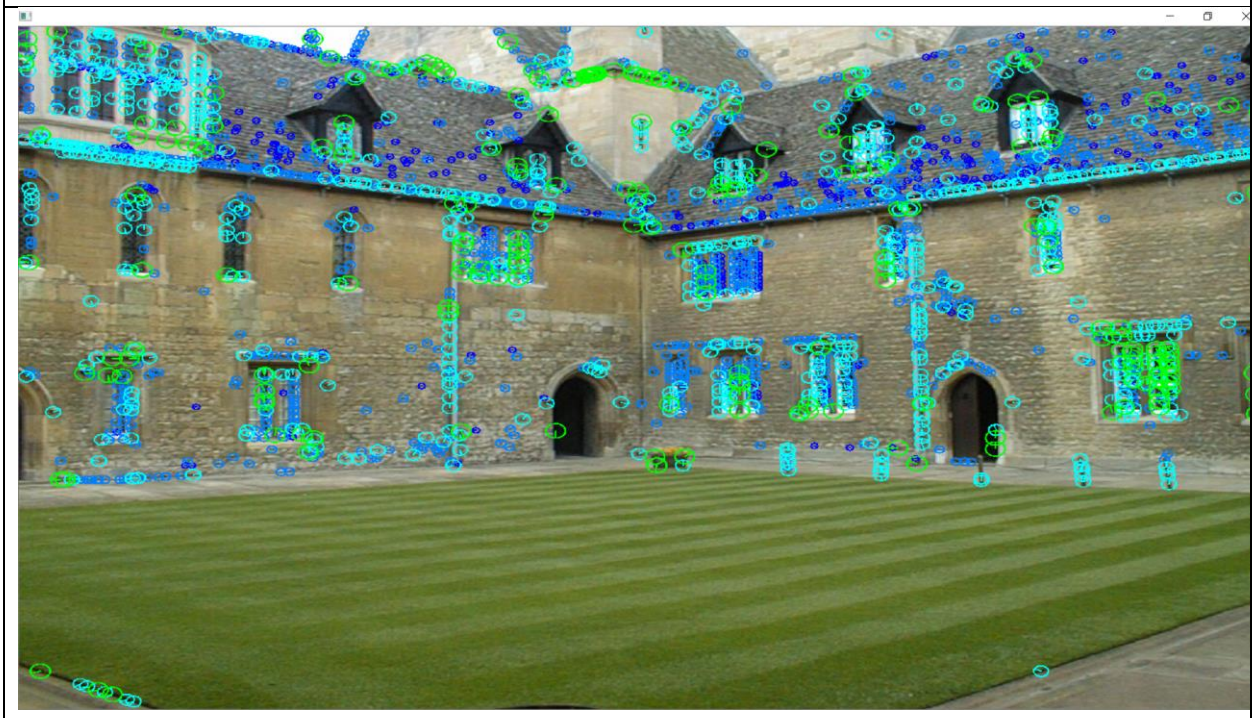
Layer 1



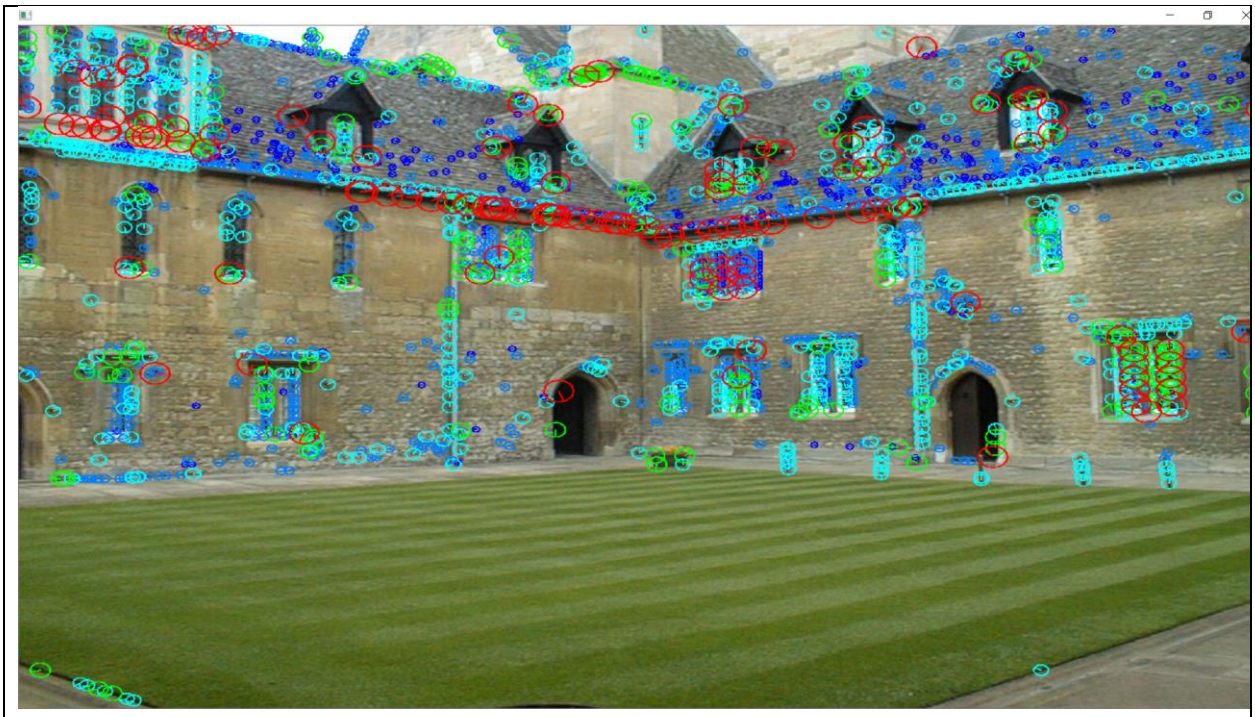
Layer 2



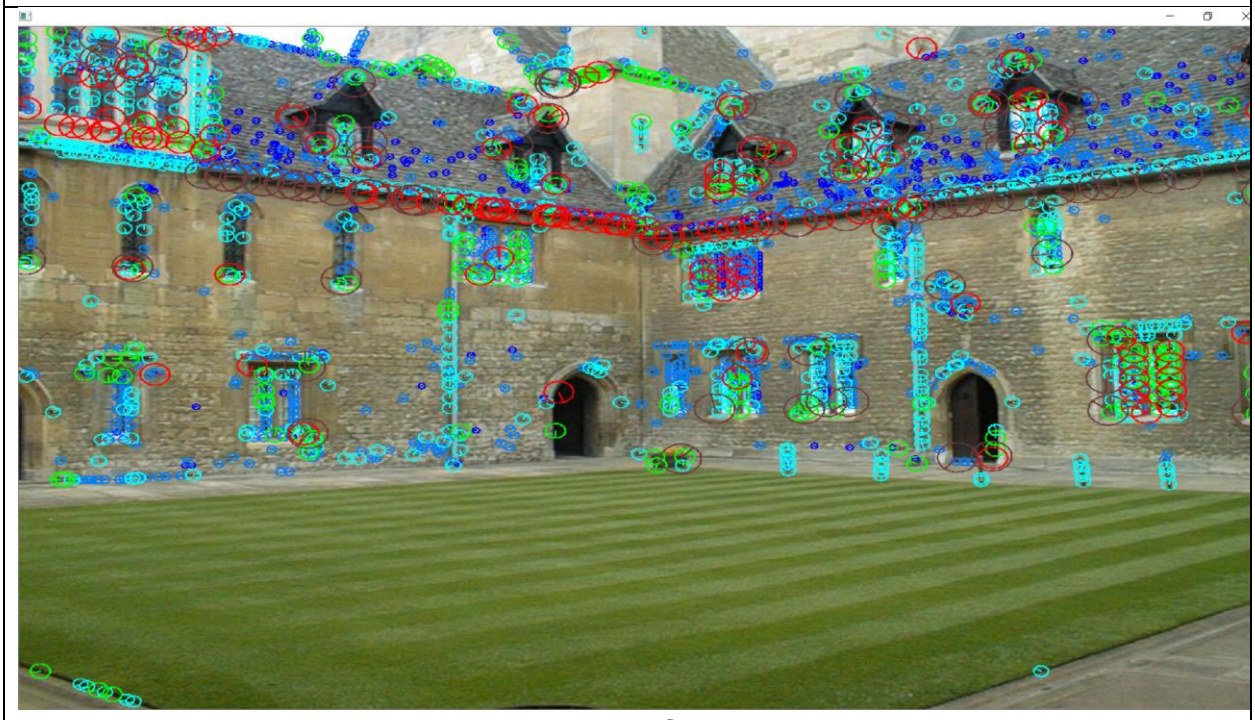
Layer 3



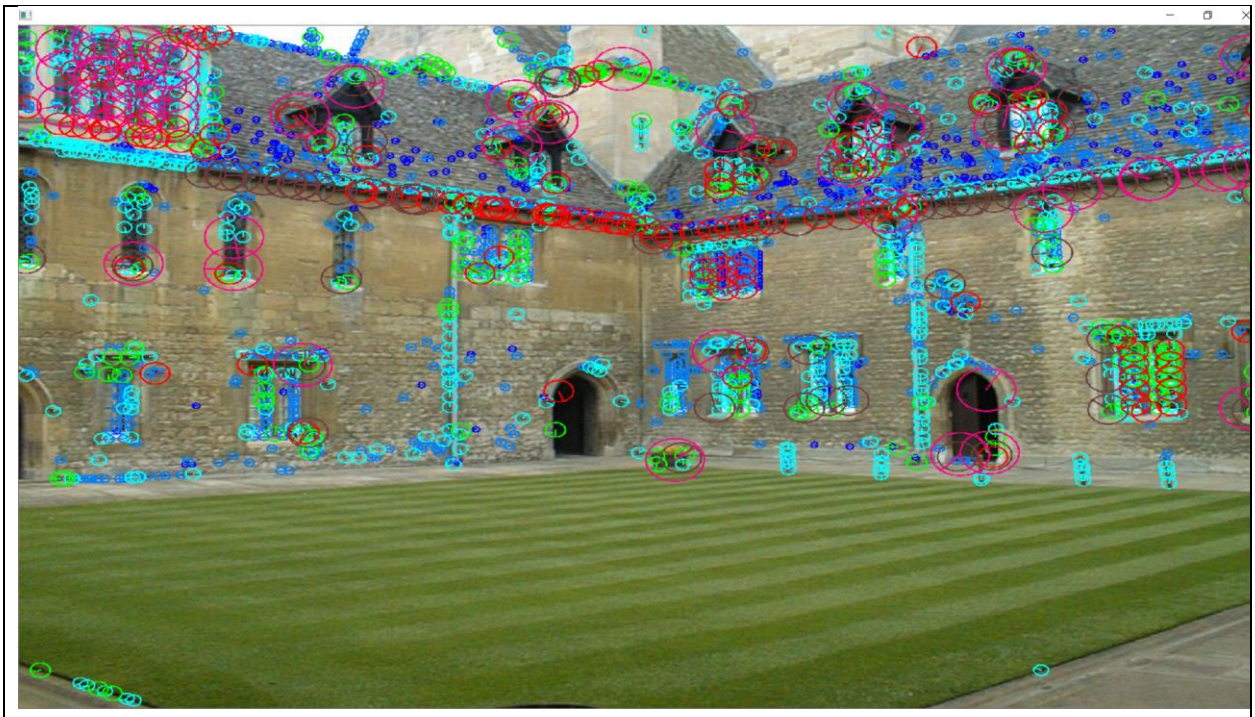
Layer 4



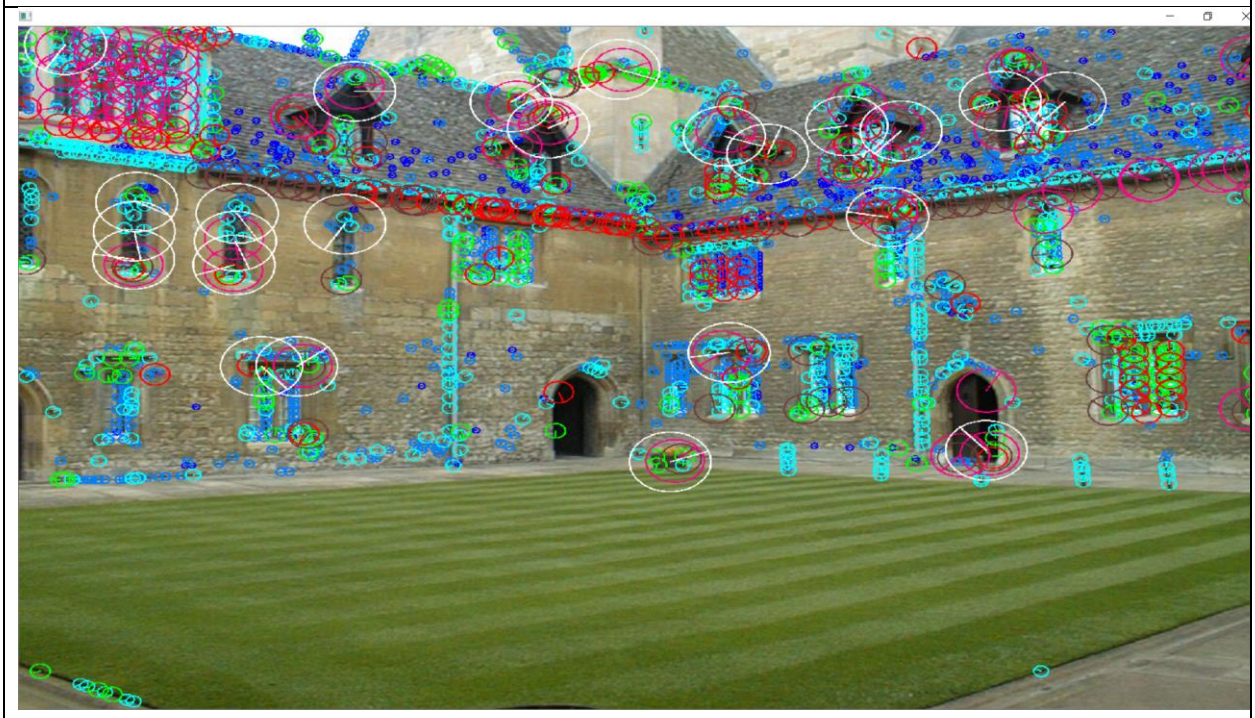
Layer 5



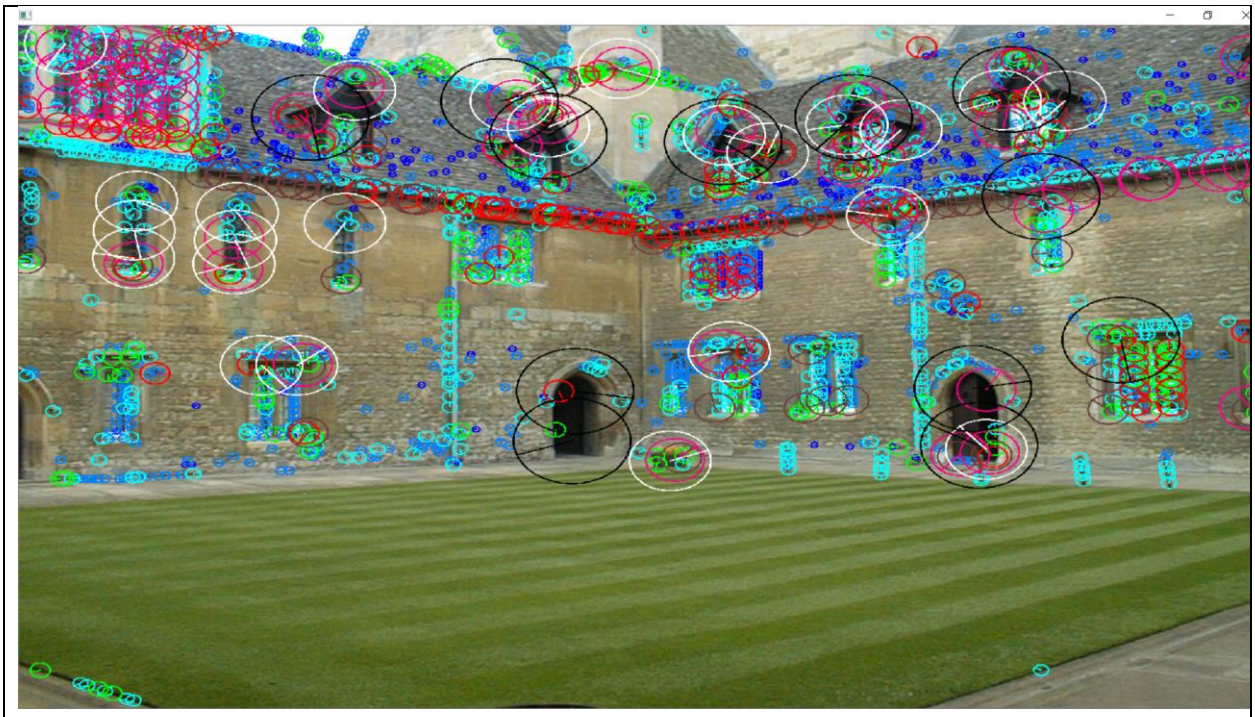
Layer 6



Layer 7



Layer 8



Layer 9 (Final Result)

Task 2

All code implementations are in the **mv_assignment1.py** file, each question subtasks will be labelled with its respective implementation line with reference to the **mv_assignment1.py** file.

Subtask A

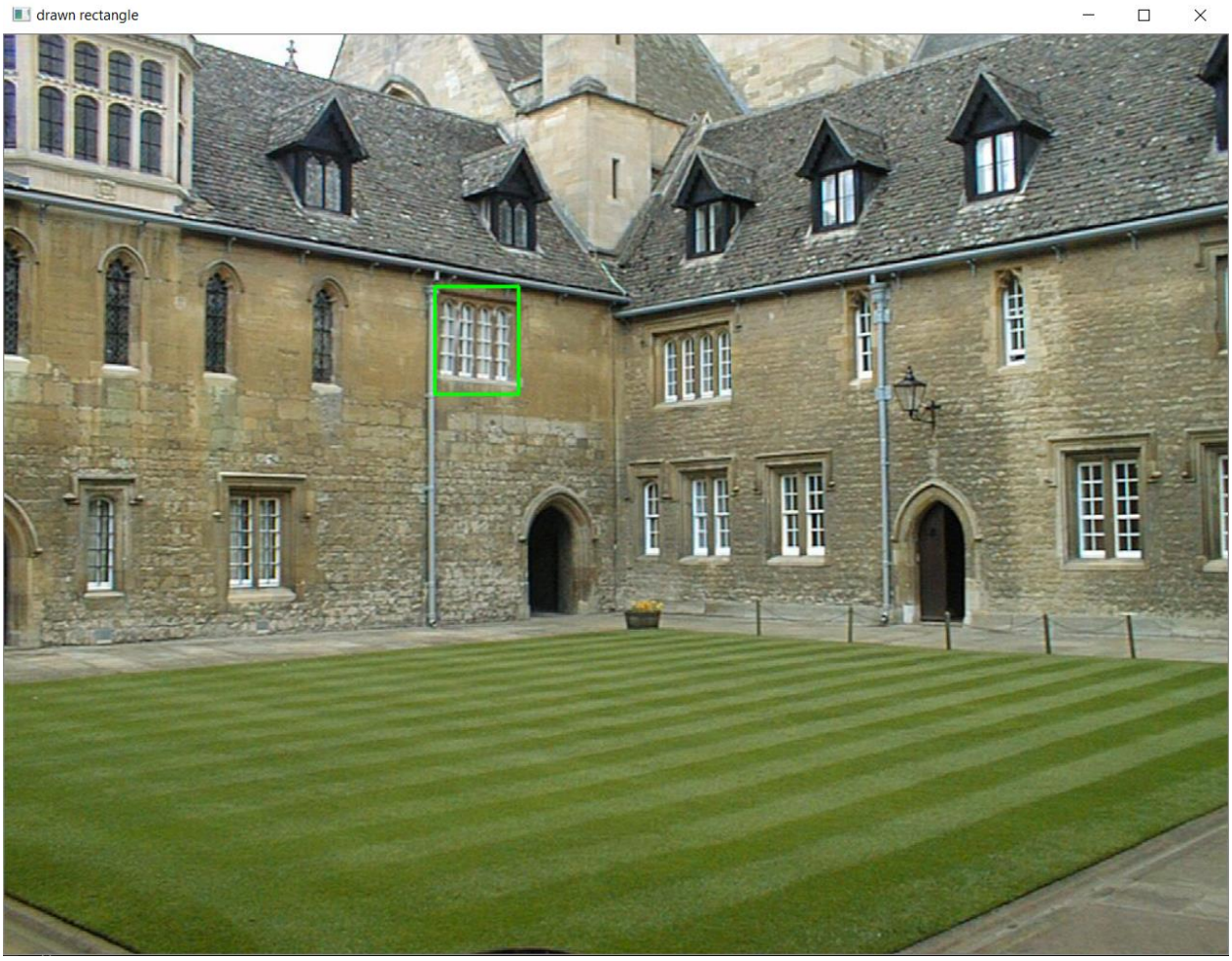
Load both input images and convert them to single channel grey value images: **lines 294-305**

Convert image to data type float32: **line 306**

Subtask B

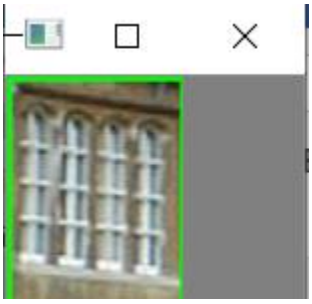
Draw a rectangle over the given window coordinate ((360,210),(430,30)): **lines 316-319**

Resulting image with rectangle drawn in input image 1 **(lines 322-324)**:



Cut out image patch containing only window and displaying it: **lines 327-333**

Resulting image with only the window cut out:



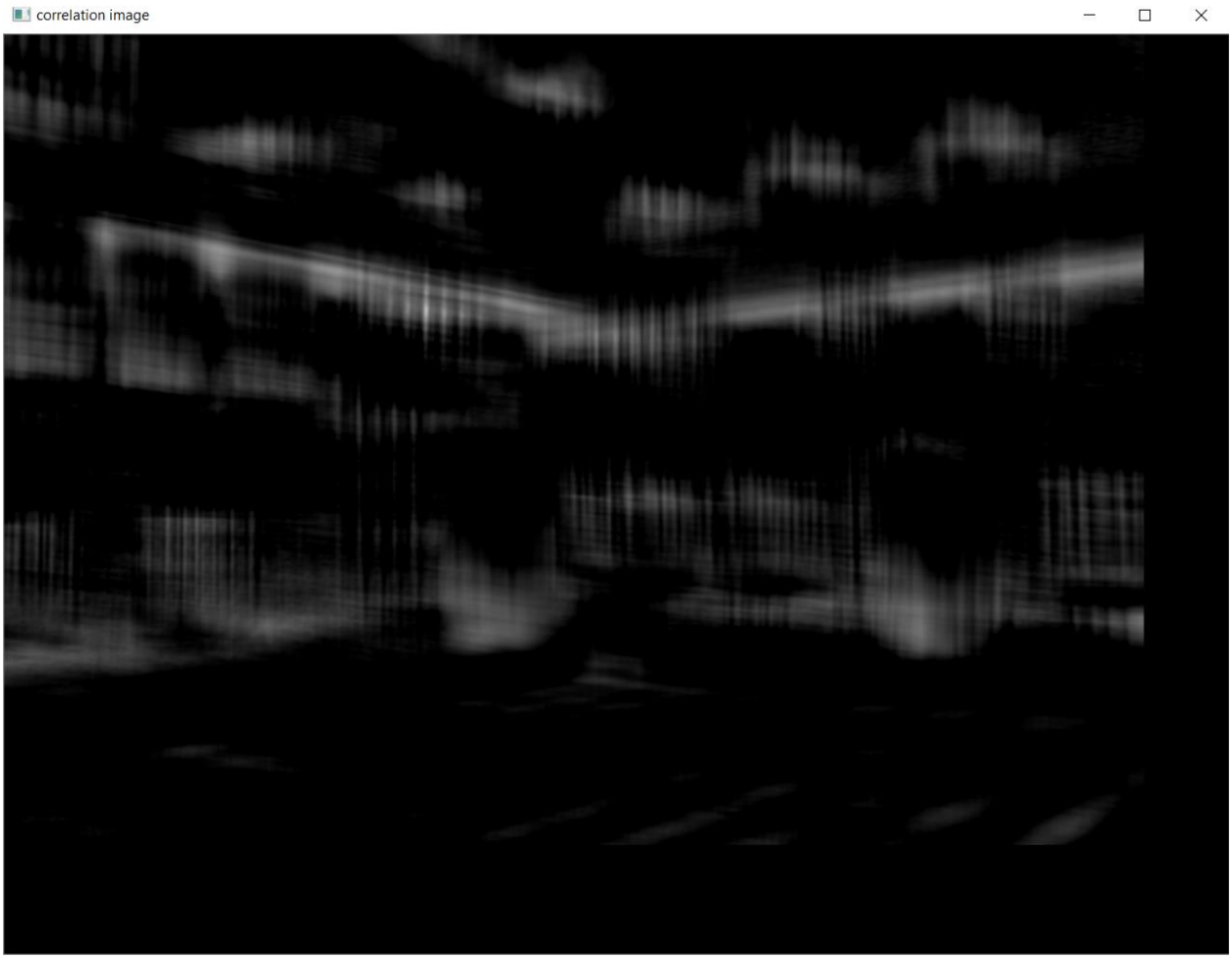
Subtask C

Calculate mean and standard deviation of window cut-out from subtask B: **lines 340, 341**

Looped through each position of image 2 with window size of the cut-out image dimension and calculate the cross correlation between the two patches: **lines 347-364**

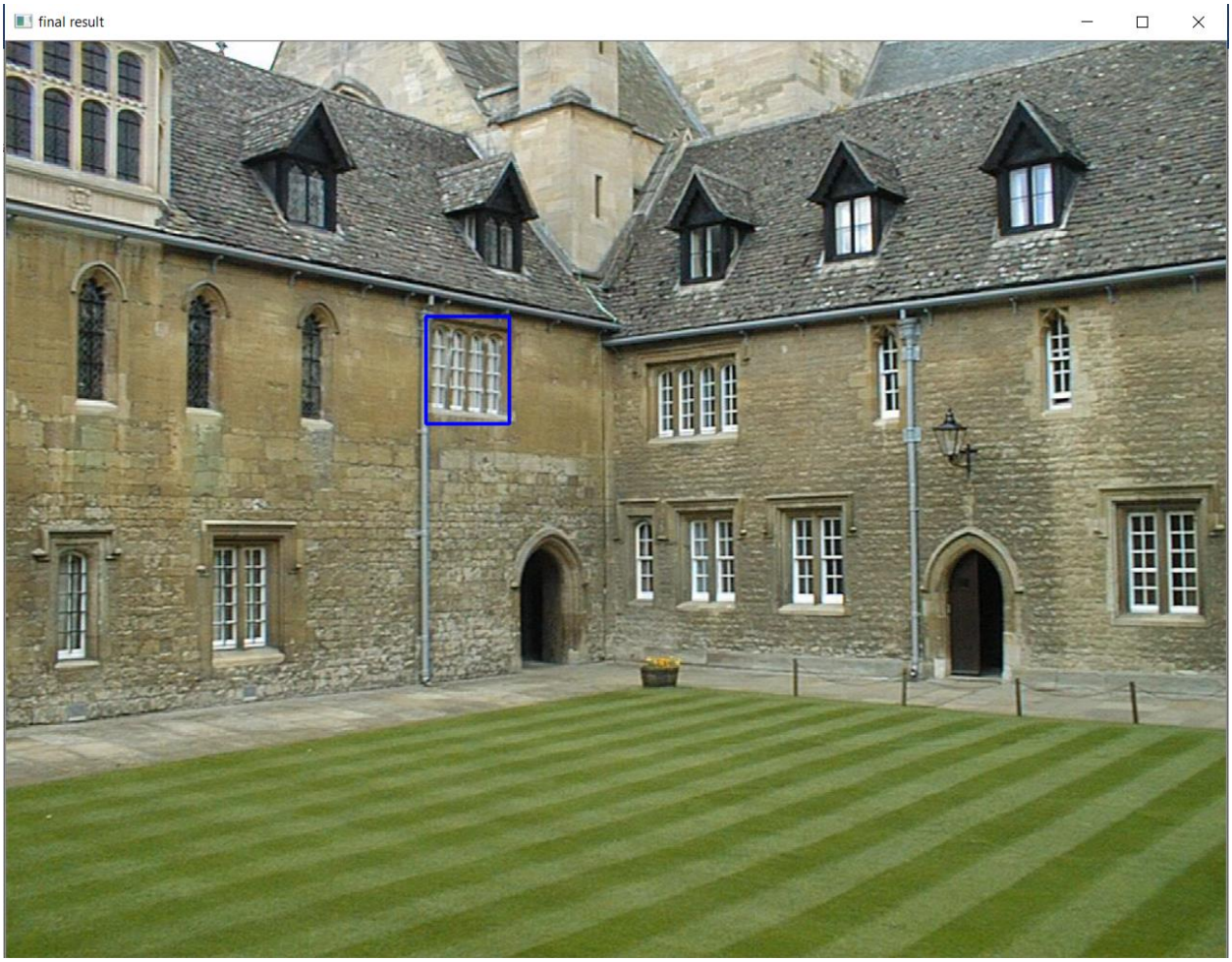
Create and display a cross correlation image for all potential positions in the second image: **lines 367-375**

Resulting cross-correlation image:



Obtain coordinates of the max cross correlation value and draw a rectangle around this position in the second input image: **lines 378-387**

Final image matching result **(lines 390-396):**


final result