CS543 Assignment 3

Part 1: Homography estimation

Describe your solution, including any interesting parameters or implementation choices for feature extraction, putative matching, RANSAC, etc.

My homography is

```
[[-7.05746371e-04  2.12185540e-04 -9.92645846e-01]
 [ 3.28873521e-04 -1.27621356e-03 -1.21029477e-01]
 [ 1.09202627e-06  5.16210486e-07 -1.95190322e-03]]
```

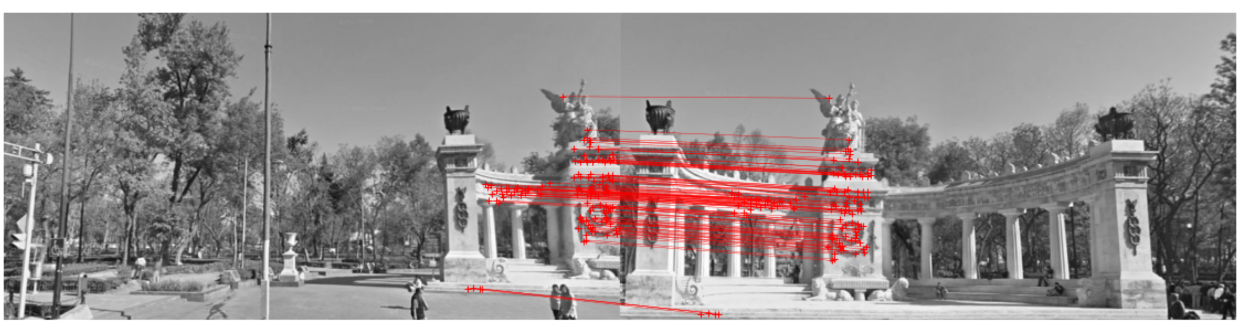
I used the sift algorithm from cv2.xfeatures2d package to calculate the keypoints and descriptors. I set the threshold to 0.005% so that about 700 pairs of keypoints are left. This threshold is based on the density and distribution of keypoints. I managed to visualize the keypoints, and my selected keypoints are well distributed around the monument. I then eliminated the keypoints calculated from different sigma but represent the same position in the pixel coordinates. I set the sample size to 10. The fitting algorithm tries to minimize the error of 10 pairs of input keypoints.

For the image pair provided, report the number of homography inliers and the average residual for the inliers.

Inliers 115

Average residual(mean 2-norm of difference) = 1.86

Also, display the locations of inlier matches in both images.



Display the final result of your stitching.

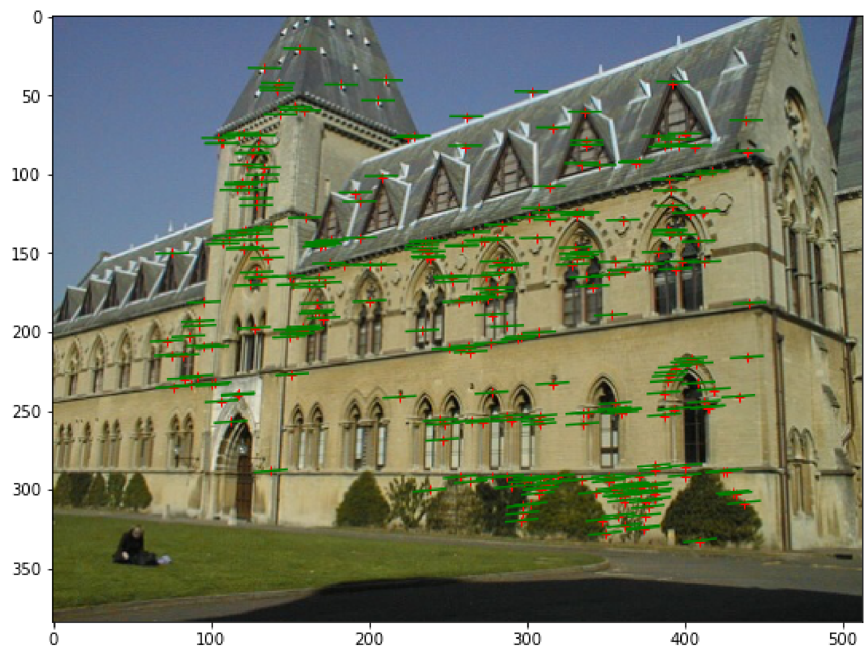
Part 2: Fundamental Matrix Estimation, Camera Calibration, Triangulation

For both image pairs, for both unnormalized and normalized fundamental matrix estimation, display your result (points and epipolar lines) and report your residual.

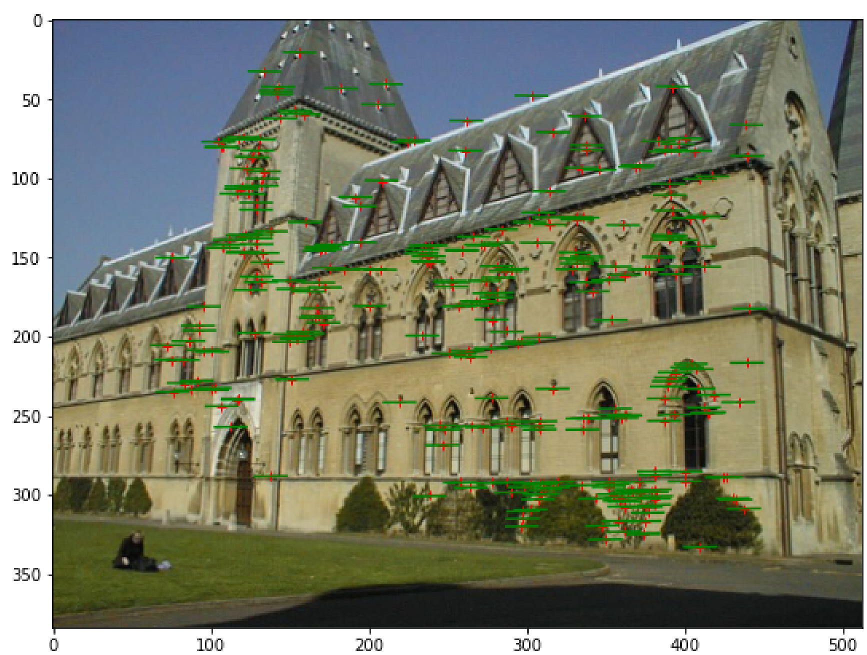
Normalized: `1.3187982852873092`

Unnormalized:`13.20106138737857`

Unnormalized



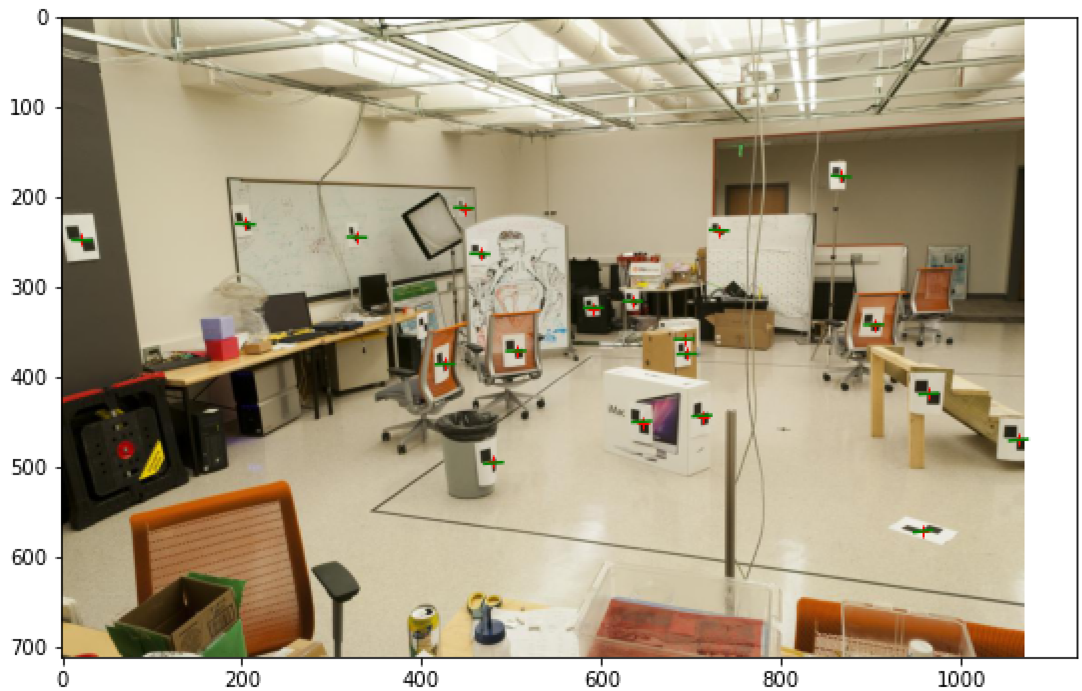
Normalized



```
Normalized: 0.9064654437128721
Unnormalized: 12.200768947343281
```

Unnormalized

Normalized



For the lab image pair, show your estimated 3x4 camera projection matrices.

```
[[-3.09963996e-03 -1.46204548e-04  4.48497465e-04  9.78930678e-01]
 [-3.07018252e-04 -6.37193664e-04  2.77356178e-03  2.04144405e-01]
 [-1.67933533e-06 -2.74767684e-06  6.83964827e-07  1.32882928e-03]]

[[ 6.93154686e-03 -4.01684470e-03 -1.32602928e-03 -8.26700554e-01]
 [ 1.54768732e-03  1.02452760e-03 -7.27440714e-03 -5.62523256e-01]
 [ 7.60946050e-06  3.70953989e-06 -1.90203244e-06 -3.38807712e-03]]
```

Report the residual between the projected and observed 2D points.

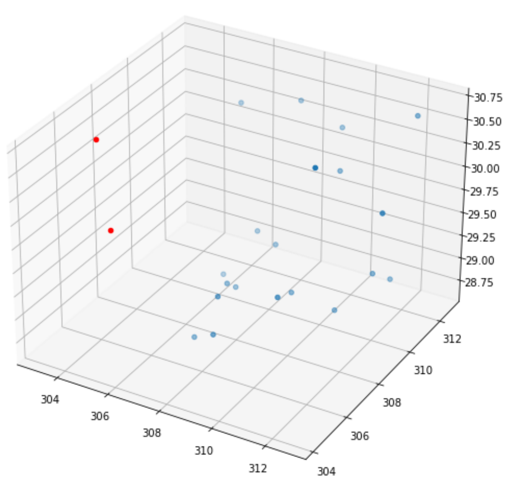
```
camera0: residual=  13.545832894770703
camera1: residual=  15.544953451653466
```

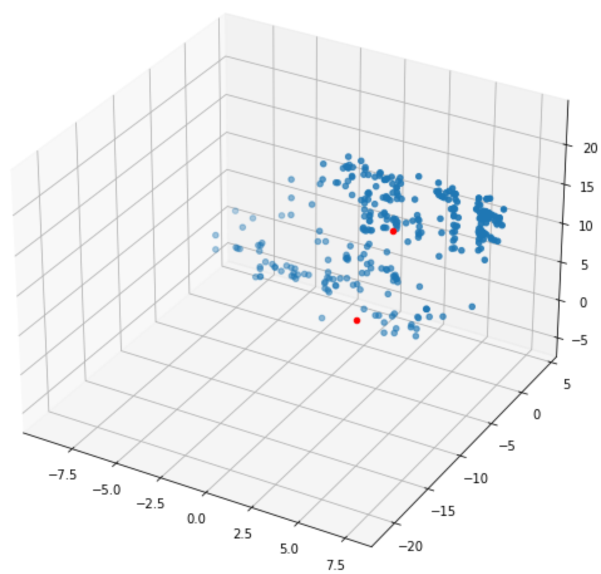
For both image pairs, visualize 3D camera centers and triangulated 3D points.

Camera0: `305.83276769 304.20103826  30.13699243`

`Camera1:303.10003925 307.18428016  30.42166874`

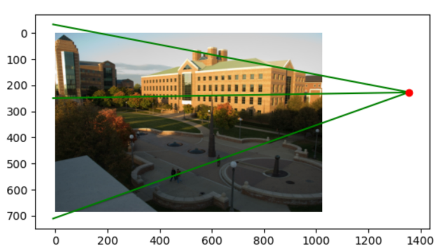
```
Library:
Camera0:7.28863053 -21.52118112  17.73503585
camera1:6.89405488 -15.39232716  23.41498687
```

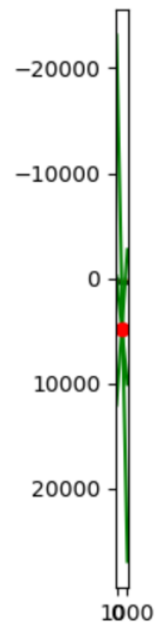


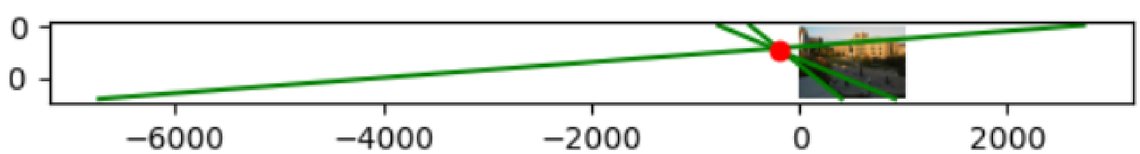


Part 3: Single-View Geometry

Plot the VPs and the lines used to estimate them on the image plane using the provided code.







Specify the VP pixel coordinates.

Right:`1.35617939e+03, 2.27120362e+02`

Vertical:`5.49926702e+02  4.84513324e+03`

Left:`-1.97639747e+02  2.24906754e+02`

Plot the ground horizon line and specify its parameters in the form a * x + b * y + c = 0. Normalize the parameters so that: a^2 + b^2 = 1.

```
[ 1.42462252e-03 -9.99998985e-01  2.25188088e+02]
```

Using the fact that the vanishing directions are orthogonal, solve for the focal length and optical center (principal point) of the camera. Show all your work.

$$\underset{3}{v_i^T}\ \underbrace{K^{-T}K^{-1}}_{9}\ \underset{3}{v_j} = 0.$$

f, p_x, p_y

$K^{-T}K^{-1}$

$$= \begin{bmatrix} 1/f & 0 & 0 \\ 0 & 1/f & 0 \\ -p_x/f & -p_y/f & 1 \end{bmatrix} \begin{bmatrix} 1/f & 0 & -p_x/f \\ 0 & 1/f & -p_y/f \\ 0 & 0 & 1 \end{bmatrix}$$

$a = 1/f$

$b = -p_x/f$

$c = -p_y/f$

$$\left(\frac{p_x^2}{f^2} + \frac{p_y^2}{f^2} + 1\right) f^2$$

$$p_x^2 + p_y^2 + f^2 = \gamma$$

$$= \begin{bmatrix} a & 0 & 0 \\ 0 & a & 0 \\ b & c & 1 \end{bmatrix} \begin{bmatrix} a & 0 & b \\ 0 & a & c \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} a^2 & 0 & ab \\ 0 & a^2 & ac \\ ab & ac & b^2+c^2+1 \end{bmatrix}$$

$$-\frac{p_x/f}{1/f} = -p_x$$

$$= \begin{bmatrix} 1 & 0 & b/a \\ 0 & 1 & c/a \\ b/a & c/a & \frac{b^2+c^2+1}{a^2} \end{bmatrix} a^2$$

$$\underset{v_i}{(x\ y\ 1)} \begin{pmatrix} 1 & 0 & b/a \\ 0 & 1 & c/a \\ b/a & c/a & \frac{b^2+c^2+1}{a^2} \end{pmatrix} \overset{v_j}{\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix}} a^2 = 0$$

$b/a = \alpha = -p_x$

$c/a = \beta = -p_y$

$$\frac{b^2+c^2+1}{a^2} = \gamma = p_x^2 + p_y^2 + f^2$$

$$(x\ y\ 1) \begin{pmatrix} x' + b/a \\ y' + c/a \\ x'\,b/a + y'\,c/a + \frac{b^2+c^2+1}{a^2} \end{pmatrix} = 0$$

$$\left((x+x'), (y+y'), 1\right) \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = -x'x - yy'$$

$3+3.$

$\cancel{xx' + x\,b/a}$

$$(x\ y\ 1) \begin{pmatrix} x' + \alpha \\ y' + \beta \\ x'\alpha + y'\beta + \gamma \end{pmatrix} = 0$$

$$x'x + \alpha x + y'y + \beta y + x'\alpha + y'\beta + \gamma = 0$$

Focal length = `765.5233269011928`

`U = 556.3212597477082`

`V = 356.53979633140676`

Compute the rotation matrix for the camera.

```
[[ 0.71755684 -0.00140434 -0.69649854]

 [-0.11610284  0.98576538 -0.12160075]

 [ 0.68675491  0.16812091  0.70717964]]
```

Estimate the heights of (a) the CSL building, (b) the spike statue, and (c) the lamp posts assuming that the person nearest to the spike is 5ft 6in tall. In the report, show all the lines and measurements used to perform the calculation.

In meters

```
Estimating height of CSL building

28.519733474975 68

Estimating height of the spike statue

12.358220638234098

Estimating height of the lamp posts

4.896751934872732
```

How do the answers change if you assume the person is 6ft tall?

In meters

0
100
200
300
400
500
600
0
200
400
600
800
1000

0
100
200
300
400
500
600
0
200
400
600
800
1000

```
Estimating height of CSL building

31.77759584949543

Estimating height of the spike statue

13.526281612945644

Estimating height of the lamp posts

5.354648948956677
```

Extra Credit

Don't forget to include references, an explanation, and outputs to receive credit. Refer to the assignment for suggested outputs.

Part 1

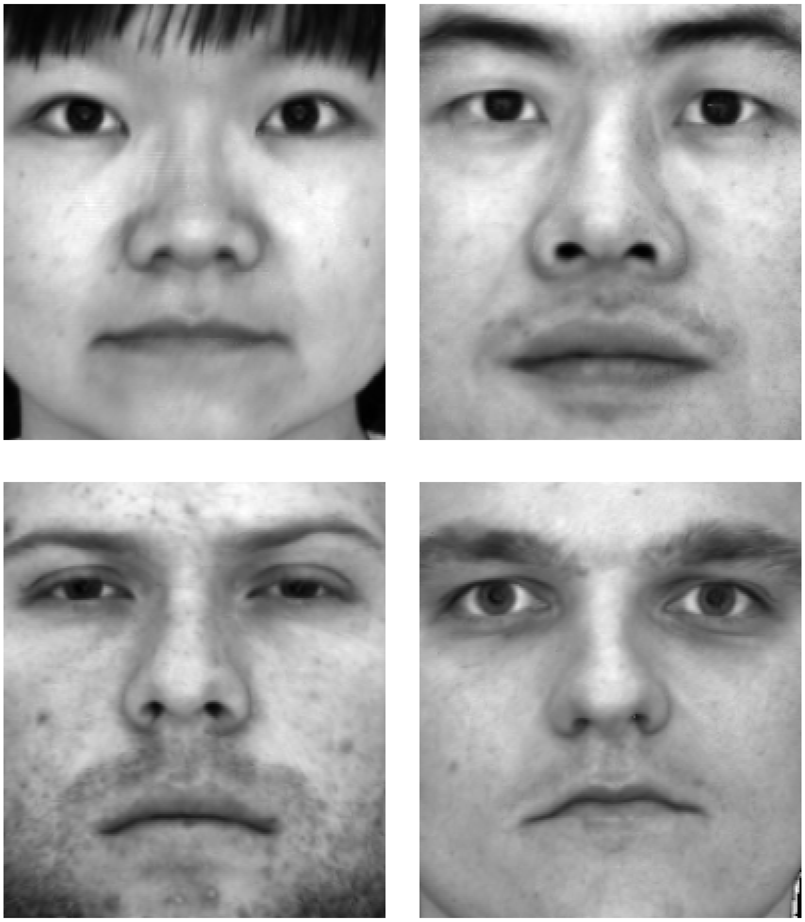
Part 2

Part 3

The person in red jacket and jeans is the tallest. If we assume the person under the statue is 1.82m, the man in red is 1.91m

Part-1 : Estimate the albedo and surface normals

1) Insert the albedo image of your test image here:



2) What implementation choices did you make? How did it affect the quality and speed of your solution?

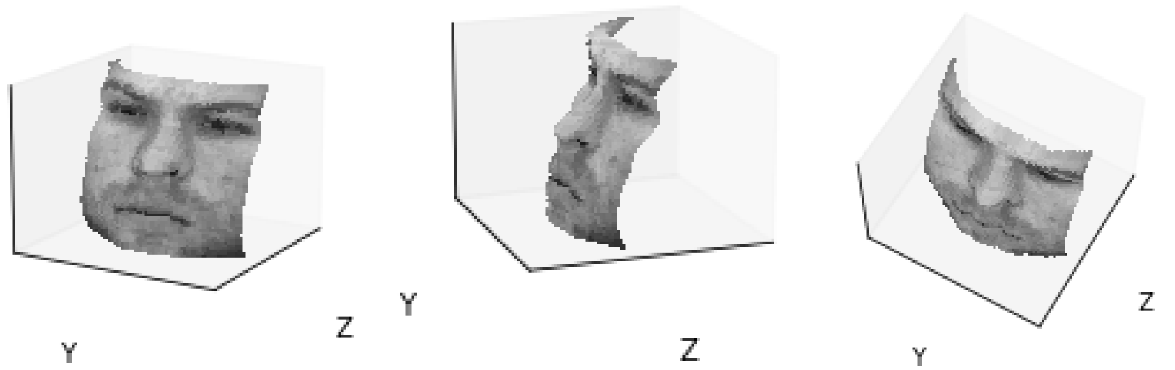
I implemented the algorithm that for each pixel, it solves the least square problem V*g = I, where V contains the directions of light sources, I contains the pixel values in each light source. An interesting part of my solution was that instead of solving SVD of the light direction matrix on each pixel, I solved the SVD in the beginning of the algorithm, and used the same decomposed parts through the loop. This greatly reduced the time complexity of my algorithm.

3) What are some artifacts and/or limitations of your implementation, and what are possible reasons for them?
Though the albedo images look fine, there are artifacts. The albedo image should reflect the true color of the surface. So the dark areas on the side of the nose, under eyelids, and in the nose should not appear on the albedo image. This is due to the presence of shadows when the pictures are taken. For example, the shadows on the side of the nose will always appear in the presence of light. Also, there must be artifacts at edges like eyelids. At edges, surface normals change abruptly. A small change in the input can lead to a huge change in the result.

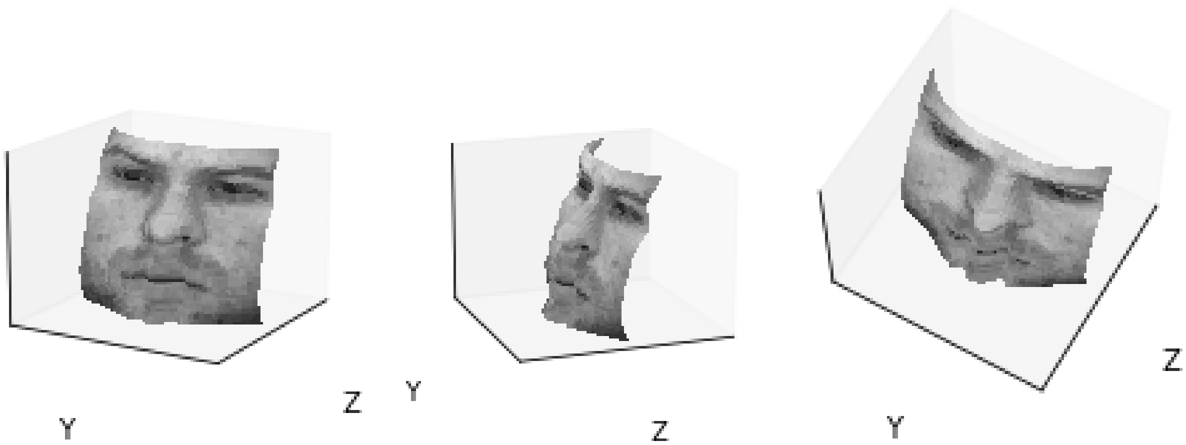
4) Display the surface normal estimation images below:

Part-2 : Compute Height Map

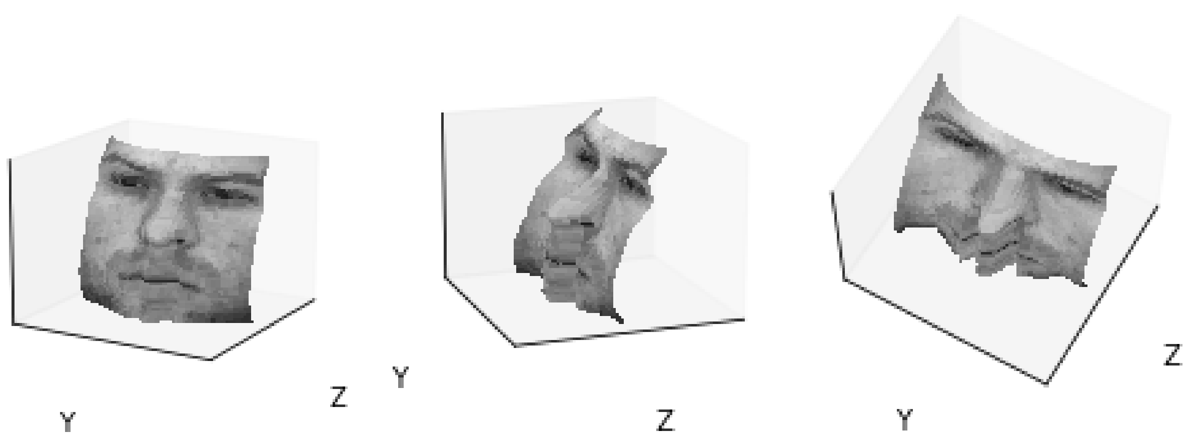
5) For every subject, display the surface height map by integration. Select one subject, list height map images computed using different integration method and from different views; for other subjects, only from different views, using the method that you think performs best. When inserting results images into your report, you should resize/compress them appropriately to keep the file size manageable -- but make sure that the correctness and quality of your output can be clearly and easily judged.



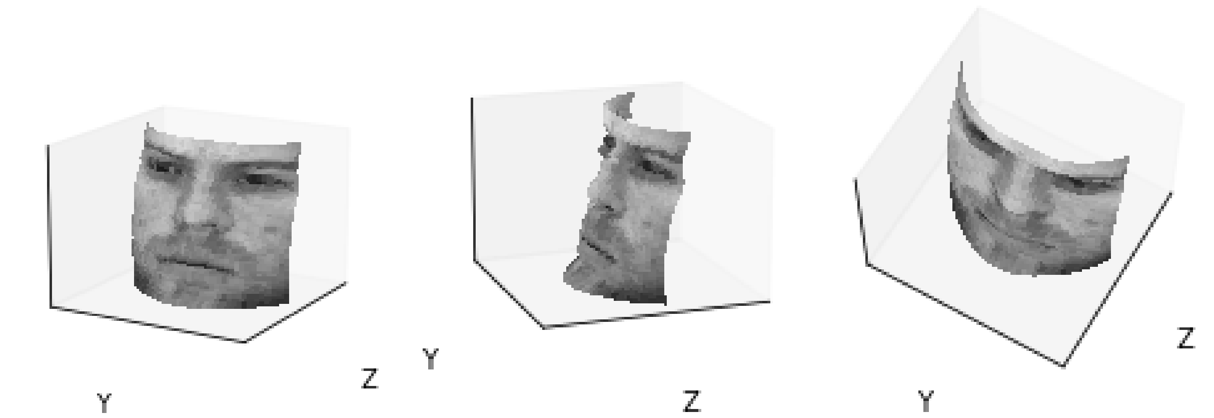
Random



Average



Row



Column

6) Which integration method produces the best result and why?
The random method produces the best results. In my implementation, each random path has a transit point. Instead of integrating directly to the target pixel, the random path first integrates to the transit point, takes a turn, then integrates to the target pixel. The transit point is a randomly selected point in the rectangle formed between the origin point and the target point. Additionally, each transit-path is integrated twice, using row-first and column-first integration. For each target pixel, a distance-dependent number of paths are integrated and averaged. The transits could reduce the effect that the path happens to be on top of a local extrema. Since our finite summation is an Euler's method approximation of the integration, it produces more accurate results along smooth paths. Because the random method takes more sample, and is more robust against local extrema, it produces the best result.

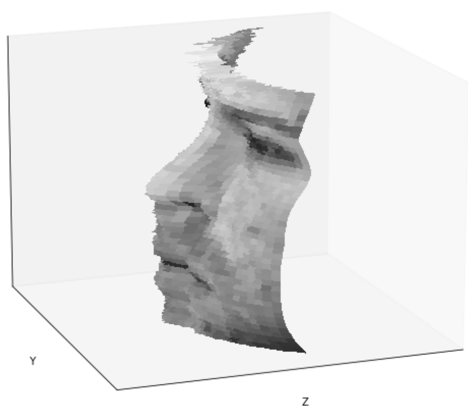
7) Compare the average execution time (only on your selected subject, "average" here means you should repeat the execution for several times to reduce random error) with each integration method, and analyze the cause of what you've observed:

Integration method	Execution time
random	1m11s
average	2s
row	2s
column	2s

The random function takes the longest time. Because it uses an explicit loop. It takes longer than vector operations. Also, at each pixel, the random function evaluates a number of random paths. The total time complexity is more than O(n^2)

Part-3 : Violation of the assumptions

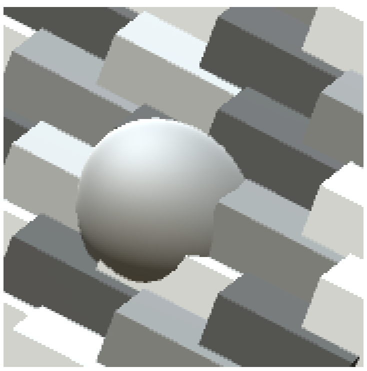
8) Discuss how the Yale Face data violate the assumptions of the shape-from-shading method covered in the slides.
   The face has specular reflections, it is not lambertian. It cast shadows on itself. It has edges where normals are undefined.
9) Choose one subject and attempt to select a subset of all viewpoints that better match the assumptions of the method. Show your results for that subset.

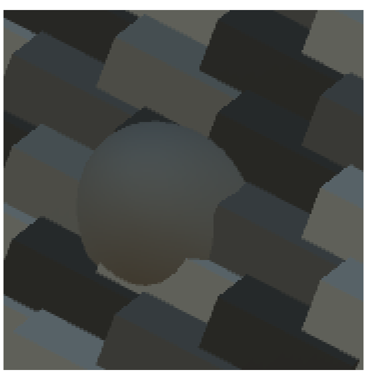


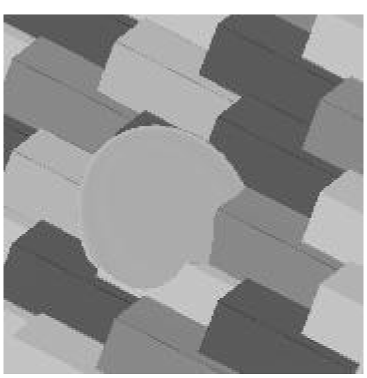
10) Discuss whether you were able to get any improvement over a reconstruction computed from all the viewpoints.
    I selected a subset of images where the face is properly lit. Most of the light hit the face vertically. So the effects from shadows are minimized. The result is better. The face stands vertical, and the mouth does not protrude out as much as the result that uses all the images. Also, the normal maps created from the subset contain more detail. Because many images in the full dataset are completely dark, the details are averaged out by them.

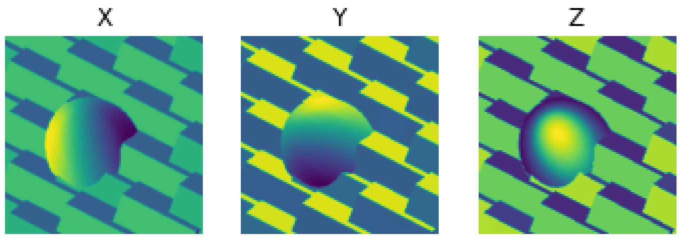
Part-4 : Bonus

I used Unity3D to generate a complicated surface including cubes and a sphere. The surfaces' smoothness are set to 0, so that there are no specular reflections. I turned off shadow casting. The camera is also orthographic. Therefore, the setup satisfies every assumption of the surface reconstruction. I used it to test the best performance of the model under ideal conditions.

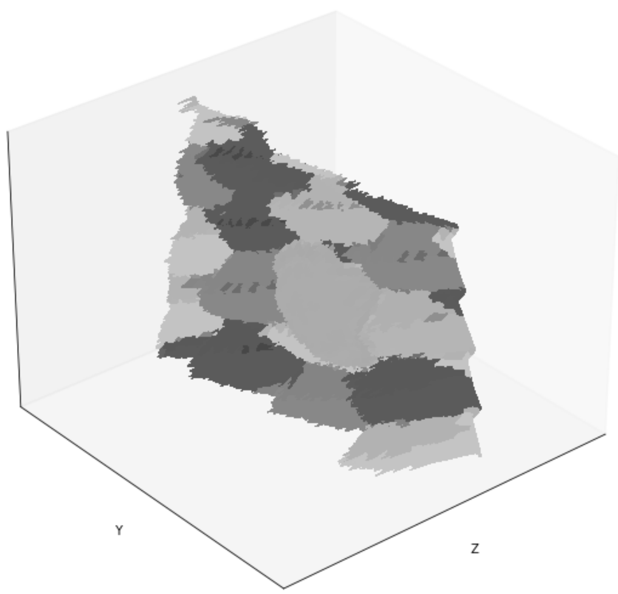

The 3D model consisting cubes and a sphere of different color

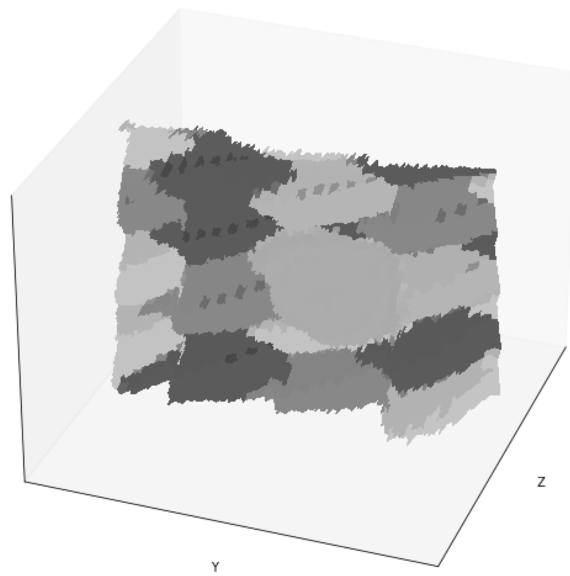

The screenshot taken in Unity3D with lights turned off. I used as a reference of ambient image


The recovered albedo map. The colors of each part are right relatively. But it is brighter than the original albedo image.



The recovered normal maps. It looks very good. The parallel planes on the cubes, and the gradients on the sphere can be clearly seen.





The reconstructed 3d surface. Except for the middle sphere region, it looks very bad. Because the surface is composed of many parts, and the edges are non-differentiable, the integral would not work in theory. Therefore, this is the limitation of the algorithm. Under ideal conditions, surface normals and albedo maps can be correctly calculated, surface reconstruction only works if the surface is differentiable.