Final Project Report
Junqi Yang
221004901

1. Introduction

In this final project, I will implement the algorithm from the research paper "**Relative 3D Reconstruction Using Multiple Uncalibrated Images**". As mentioned in the paper, this algorithm can be easily implemented and will not require camera calibration during the 3D reconstruct. All it needs is five reference points to hold Euclidean shape and distance.

In this report, I will discuss 2 different approaches, which based on the same equation, to reconstruct 3D and their individual result. I also included one real image result and 2 simulated results to test the algorithm that I implemented.

And I will discuss the correctness, efficiency and possible outcome in the end.

2. Math approach

The logic behind the algorithm is very easy to understand. As we know that, by knowing the correct projection matrix of the camera. We can easily transfer between a 3D homogeneous coordinate and its projection, which is a 2D homogeneous coordinate by the equation:

$$\rho_{ij} m_{ij} = P_j M_i$$

In which, ρ is an unknown scaling factor, Mi is a 3D homogeneous coordinate, Pj is the camera matrix and m_{ij} is the 2D projection of Mi. In this paper we can assume ρ can be hidden and ignore during the computation, this can be also represented as:

$$u_{ij} = \frac{P_{11}x_i + P12y_i + P_{13}z_i + P_{14}t_i}{P_{31}x_i + P32y_i + P_{33}z_i + P_{34}t_i}$$
$$v_{ij} = \frac{P_{21}x_i + P22y_i + P_{23}z_i + P_{24}t_i}{P_{31}x_i + P32y_i + P_{33}z_i + P_{34}t_i}$$

This paper is using a idea can be explained like this: ideally, when we know a 2D point m, if we find a perfect correct camera matrix Px and its correct 3D homogeneous coordinate Mx. We can have the result of $m - P_x M_x$ equal to 0.

So when we only know the information of m, to find a correct P and its M in 3D, we need to find a pair of them which can minimize the result of the equation $m - P_x M_x$.

By doing so, we can recover the projective transformation information. So we need to have another 5 reference points to hold the Euclidean shape and distance information so that we can remove affine.

a. 5 Reference Points

We need 5 reference point to construct the euclidean coordinate system of the object. The 5 point can be auto select by a greedy algorithm with coplanarity test.

1. First, we randomly pick the first point M1 and second point M2
2. Second, we pick the third point M3, not in line with M1 and M2
3. Third, we pick the fourth point M4 that is not coplanar with M1 to M3, this can be done by using a coplanarity test. The coplanarity test is a theorem that

   "if neither a, b, c, nor d are the epic pole point of image 2 with respect to image 1, then it exists at least one diagonal intersection m such that m and its corresponding intersection m satisfy the epic polar constraint if and only if A, B, C, D are coplanar."

   This can guarantee that no any 4 points we select are coplanar.
4. Last, we pick the last point 5 that is not coplanar with any 3 points set that combined by M1 to M4

   For manual selection, we usually pick one point and original point(0,0,0) and three points go alone x, y, z three axes. And one point does not on the surface that combined by two axes.

   So in the experiment, we use to have 5 reference point following the following format:

$M_1 = \rho(0,0,0,1), M_2 = \rho(1,0,0,1), M_3 = \rho(0,1,0,1), M_4 = \rho(0,0,1,1), M_5 = \rho(1,1,1,1)$

ρ is a scalar. After finding out the 5 reference points, we can pick rest of the points that will be used in the 3D reconstruction. As a rule, all points we picked should have its project coordinate can be found in all the test image we have.

I will talk about 2 different methods which are very similar but I obtained different result from them in the end

b. Method 1

Method 1 the direct nonlinear reconstruction method this paper proposed in section 2. Again, we know that:

$$\rho_{ij} m_{ij} = P_j M_i$$

Transfer m into Euclidean representation:

$$u_{ij} = \frac{P_{j-row1} Mi}{P_{j-row3} Mi}, v_{ij} = \frac{P_{j-row2} Mi}{P_{j-row3} Mi}$$

So we need to find P_{ij} and $M_i(x_i, y_i, z_i, t_i)$. And to normalize the data we will need to add third constraint that $x_i^2 + y_i^2 + z_i^2 + t_i^2 = 1$. So this problem is formulated into minimizing:

$$F(x_i, y_i, z_i, t_i, p_{11}, ...p_{34}) = \sum (\frac{f(u, v, x_i, y_i, z_i, t_i, p_{11}, ...p_{34})}{\sigma})^2$$

$$F(x_i, y_i, z_i, t_i, p_{11}^{(j)}, ...p_{34}^{(j)}) = \sum_{k=1}^{2\times m\times n+n} (\frac{f_k(u_{ij}, v_{ij}; x_i, y_i, z_i, t_i, p_{11}^{(j)}, ...p_{34}^{(j)})}{\sigma_k})^2$$

Where we have three different f(;):

1. $$u_{ij} - \frac{p_{11}^{(j)}x_i + p_{12}^{(j)}y_i + p_{13}^{(j)}z_i + p_{14}^{(j)}t_i}{p_{31}^{(j)}x_i + p_{32}^{(j)}y_i + p_{33}^{(j)}z_i + p_{34}^{(j)}t_i}$$ or
$$u_{ij}(p_{31}^{(j)}x_i + p_{32}^{(j)}y_i + p_{33}^{(j)}z_i + p_{34}^{(j)}t_i) - (p_{11}^{(j)}x_i + p_{12}^{(j)}y_i + p_{13}^{(j)}z_i + p_{14}^{(j)}t_i)$$
2. $$v_{ij} - \frac{p_{21}^{(j)}x_i + p_{22}^{(j)}y_i + p_{23}^{(j)}z_i + p_{24}^{(j)}t_i}{p_{31}^{(j)}x_i + p_{32}^{(j)}y_i + p_{33}^{(j)}z_i + p_{34}^{(j)}t_i}$$ or
$$v_{ij}(p_{31}^{(j)}x_i + p_{32}^{(j)}y_i + p_{33}^{(j)}z_i + p_{34}^{(j)}t_i) - (p_{21}^{(j)}x_i + p_{22}^{(j)}y_i + p_{23}^{(j)}z_i + p_{24}^{(j)}t_i).$$
3. $x_i^2 + y_i^2 + z_i^2 + t_i^2 - 1$

The solution can be found using Levenberg-Marquardt algorithm by using nonlinear least square routine. The result we obtained a set of 3D coordinate that only projective properties are preserved. So aligned points still aligned but not in the right shape. So we need to find a homography H that can remove affine.

We already have 5 reference point here. So for this five reference point, we want to find this Homography that will project the current coordinate into their measured coordinate.

$$a_i = Ae_{measured}$$

To solve this I formulated the problem into find a A that will minimize:

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} - A[e_1, e_2, e_3, e_4, e_5]^T$$

Then for every points, we apply

$$M_x = A^{-1}m_x$$

c. Method 2

The method 2 is a simplified version of method 1, I found this more reliable compare to method 1. As the author said in his paper, if we assign the reference points to their Euclidean coordinates at the beginning of the minimization process, we can directly obtain coordinated that has its euclidean shape.

So in this process, we need to assign the measured coordinate to reference points before minimization instead of normalizing them into the scale of 1. And we should not normalize any of other points.

So in method two, we need to minimize:

$$F(x_i, y_i, z_i, t_i, p_{11}^{(j)}, ...p_{34}^{(j)}) = \sum_{k=1}^{2\times m\times n+n} (\frac{f_k(u_{ij}, v_{ij}; x_i, y_i, z_i, t_i, p_{11}^{(j)}, ...p_{34}^{(j)})}{\sigma_k})^2$$

Where we have two different f(;):

1. $u_{ij}(p_{31}^{(j)}x_i + p_{32}^{(j)}y_i + p_{33}^{(j)}z_i + p_{34}^{(j)}t_i) - (p_{11}^{(j)}x_i + p_{12}^{(j)}y_i + p_{13}^{(j)}z_i + p_{14}^{(j)}t_i)$
2. $v_{ij}(p_{31}^{(j)}x_i + p_{32}^{(j)}y_i + p_{33}^{(j)}z_i + p_{34}^{(j)}t_i) - (p_{21}^{(j)}x_i + p_{22}^{(j)}y_i + p_{23}^{(j)}z_i + p_{24}^{(j)}t_i).$

We can directly obtain what we want after minimization process.

d. Initial guess

For the minimization process, the levenberg-marquardt algorithm will need a initial guess. For projection matrix, we give:

$$P = \begin{bmatrix} 0.1 & 0.1 & 0.1 & 20 \\ 0.1 & 0.1 & 0.1 & 20 \\ 0.1 & 0.1 & 0.1 & 1 \end{bmatrix}$$

For method 1, we give reference point:

$M_1 = \rho(0,0,0,1), M_2 = \rho(1,0,0,1), M_3 = \rho(0,1,0,1), M_4 = \rho(0,0,1,1), M_5 = \rho(1,1,1,1)$

For method 2, we give reference point ground truth. For other point, the author suggest to use: (0.5, 0.5, 0.5, 1). However for method2, I will use the mean value from reference points.

For initial guess for the homograph, I give

$$A = \begin{bmatrix} 0.5 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.5 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.5 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.5 \end{bmatrix}$$

3. Results

I implemented this paper in both methods, and method 2. I used two real images, which both methods provided me a bad looking result, a computer simulated images sequence, use which, method 2 provide me semi-good looking result and a set of simulated 3D coordinates, which method 2 provide me a nearly perfect result.

a. Real Image result

As I mentioned in the section introduction, I got a terrible result which the coordinate sets do not make any sense to me in 3D illustration. I used two picture from project 1 and picture of a cube box to do the testing:

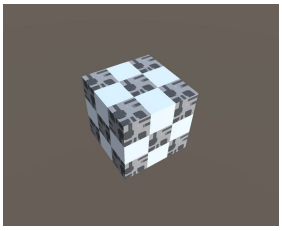
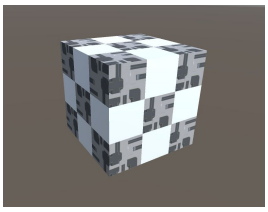
Use image 1 as the reference I set up the five reference points as with all other points that I used in project 5 to reconstruct 3D objects and left the the result I obtained, as blue point is estimate coordinate and green point are ground truth for reference points.

All the point stack together in one point.

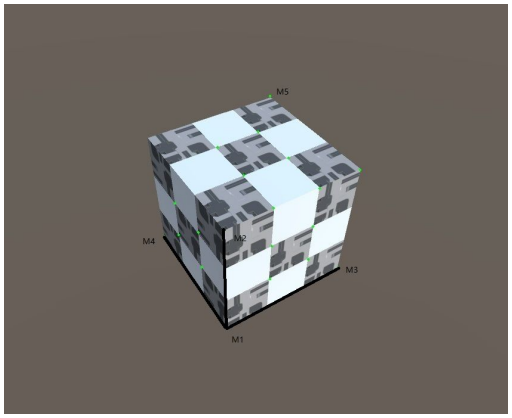
b. Simulated image result

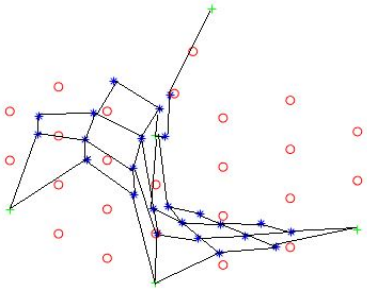
Since the error could be caused by measurement error, reference point selections, instead of taking real photos, I used Unity game engine with a hololens emulator to take photos of virtual object that the measurement can be obtained from the computer and the ground truth can be highly accurate.

I obtained a sequence of images and select points from the image. Here is a set of sample image I used for the experiment.

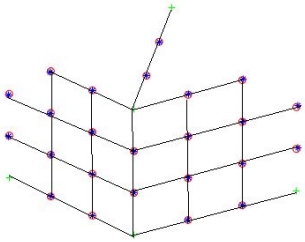
The following picture is my 5 reference point selection and the best result I obtained,after running days of tries.





c. Simulated point result

The author's paper also included a set of the result obtained by simulated data. I auto-created two camera matrix and project ground truth into 2D to obtain 2 set of 2D coordinated. I still use the ground truth from the section b, the square cube. And whatever projection matrix we used, we always obtained a best result.



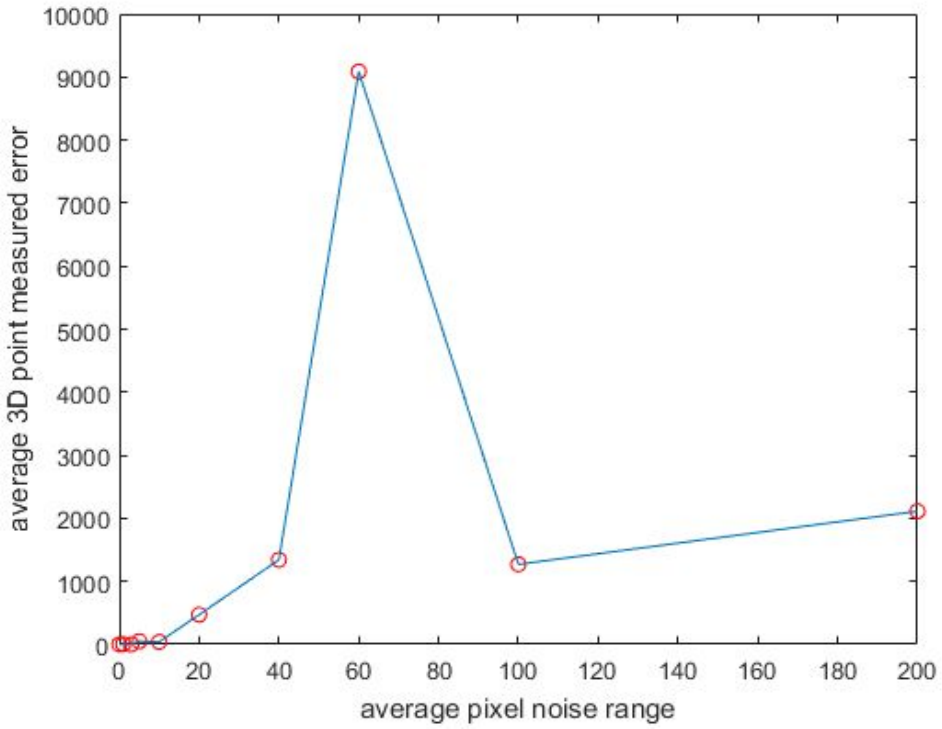
with a average error of 0.528e^-14

d. Error analysis

In order to analyze the cause of terrible output from the previous section, I ran several test by adding noise into the simulated 2D points. The noise is calculated by:

$$noise(n) = n * random(0,1) - \frac{n}{2}$$

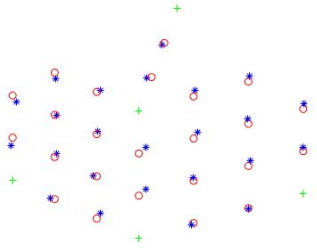
So if we have a noise of 10, we will have the coordinate have a error in the range of (-5,+5). I increase the noise from 1 to 200 in order to test the data. The **first** experiment I did is adding noise into generated 2D coordinate, to see if wrong points selected from image can cause inaccuracy. The average error is obtained by average each points' error between estimate points and ground truth in 10 runs of experiment. The ground truth is in 100 scale.



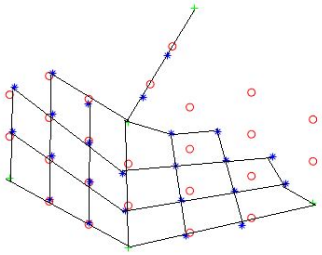
Average point errors from n = 0 to 10

n	Error
0	0.289595921769634*e^-14
1	4.9902315
3	0.418391139
5	50.881583916
10	40.366963268081541

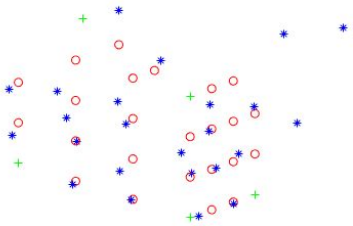
As we can see from n = 5, which means the generated 2D coordinate is increased/deceased <2.5 pixel, the project error has increased into 50. Next page is the selected result image produced by adding different noise size.



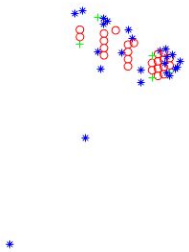
N = 1 The Levenberg-Marquardt algorithm took 15 iteration



N = 3 The Levenberg-Marquardt algorithm took 24 iteration



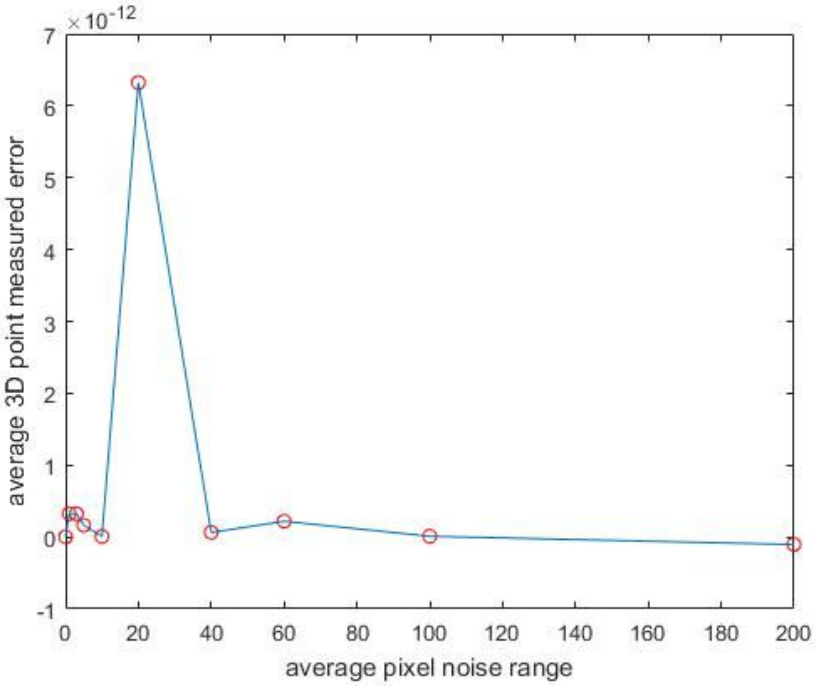
N = 4 The Levenberg-Marquardt algorithm took 40 iteration



N = 10 The Levenberg-Marquardt algorithm took over 5000 iteration

N = 30 The Levenberg-Marquardt algorithm took over 16200 iteration

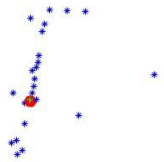
The **second** experiment I did is adding noise into the 5 reference point measurement when using those measurement into the minimization. The below is the result we obtained from the experiment.



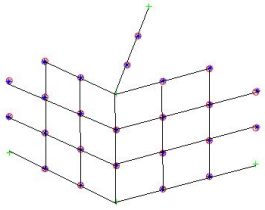
The noise add into the measurement does not effect so much on the final result.

The **third** experiment is using different initial guess from the beginning of the minimization process.

For different initial P, the result does not have many critical effect.The second target is the initial guess for the points other than the reference points. The first picture here is the result picture when using (0.5,0.5,0.5,1) as mentioned in the paper for the un-normalization method(method 2):



The second picture is the result picture when using the mean of the 5 reference point for method 2:



In conclusion, the initial points do affect the final result.

e. Discussion

From the experiment 1, we know the algorithm is very noise sensitive. The result output is highly depends on the correct selection of the 2D coordinate on the image. The more noise it has, the more computation circle it takes for matlab solver to perform the minimization.

The error could also be caused by the Levenberg-Marquardt algorithm, which could only find possible local minimum, so it missed the global minimum. It is also possible that the solver overdetermined the minimization result in the long iterations.

From the experiment 2, we know noise in the reference points before minimization process will not affect the result, however if the 4 points co-planned the result will be terrible displayed.

From the experiment 3, we know that select the initial guess is also important for un-normalized method. Using mean of the 5 reference points can be a good choice for the experiment. The different initial guess for P will not highly affect the result, but however, if the initial guess get closer to the real P, the higher result we could obtain.

4. Conclusion

During this final project, we learned how to construct 3D coordinate without doing camera calibration, however, the algorithm appears to be noise sensitive. There has been many paper since this paper published, the algorithm has been improved. A future study is necessary after this final project, to improve the qualify.

The result of the real image test appears to be poor quality, however the simulated test results are excellent. To improve the quality, a better method of points selection should be implemented.