VE490 Undergraduate Research Report

Image Enhancement for Photoacoustic Microscopy and Computed Tomography

Wang Shi Bo 518370910109

Aug 8, 2021

Instructed by Prof. Sung-Liang Chen

1. Introduction

Photoacoustic imaging technology, a method uniquely combines the advantages of optical excitation and acoustic detection that optical excitation provides a rich contrast mechanism from either endogenous or exogenous chromophores, allowing PAI to perform biochemical, functional, and molecular imaging, and acoustic detection benefits from the low scattering of ultrasound in biological tissue, enabling PAI to generate high-resolution images in both the optical ballistic and diffusive regimes [1].

Computer Vision is one of the applications of Photoacoustic imaging technology, which enables computer to recognize some certain photos and analyze them. In our paper, we mainly designed a method with this kind of technology in order to find the center of four anchor bolts to ensure the master is not leaning.

The specification of anchor bolts is d = 68mm, where d is the diameter of the circle cross section. Four anchor bolts are placed at the four corners of a square with length of each side at 438-538mm. For the main post, it is a cylinder with diameter at a length between 1430 and 1600mm, and its height is unrelated to our experiment, so we do not concern. The sample graph of the whole post is shown in the following picture.

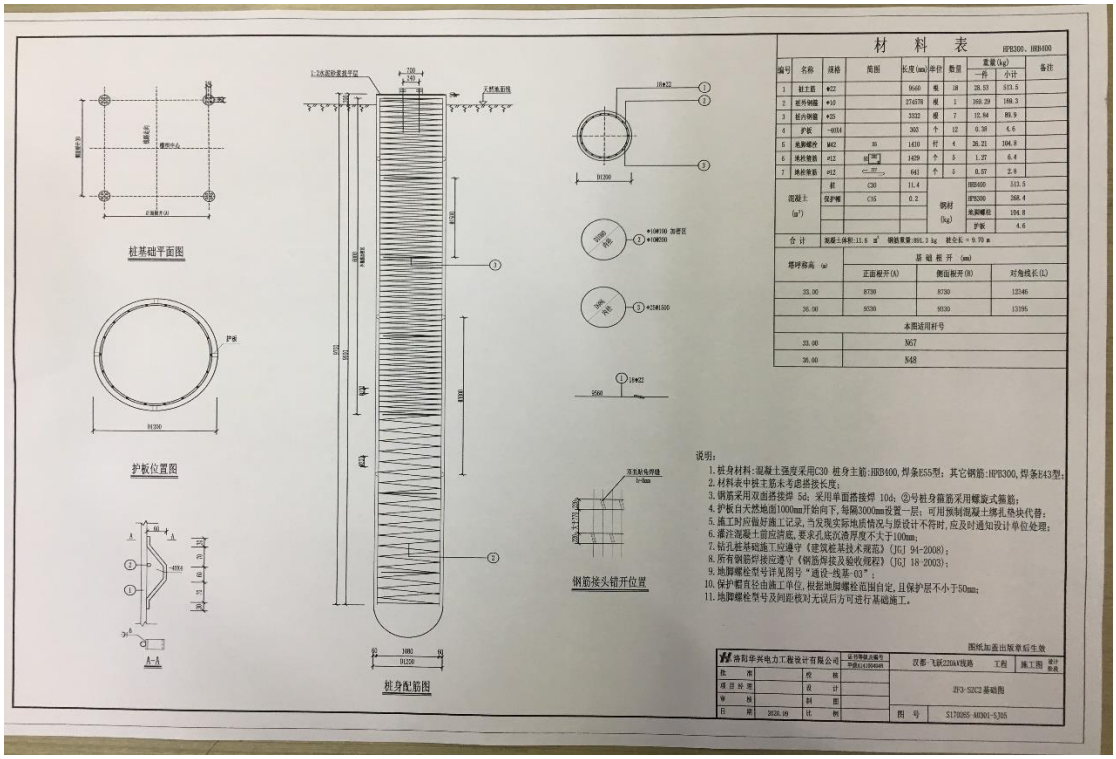
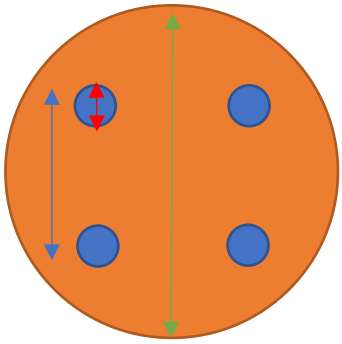


Figure 1: design drawing



Red: 68mm green:1430-1600mm blue: 438-538mm

Figure 2 cross section sample

2. Method

For the whole program, we decide to use client-server structure. At the client-side, users should place checker on the post and use an app to take several photos of the post from different directions. Then, they should ensure the pictures to be clear and send them to the server. At the server-side, we first use MATLAB to analyze the pictures and calculate the homography so as to restore the real 3-D rectangular coordinate system from the given pictures. Next, we use computer vision technology to find the center of the post and justify whether it is close to the requested one. Finally, we get the result at server-side and send it back to the client-side, which can be shown to the users. Thus, our research is mainly divided into three parts, design app, calculate homography, and find and test the center. We have totally tree group members and I mainly take the responsibility of calculating the homography. In my part, I used two different methods to calculate the homogrphy and conducted experiments with each method to test the precision. In this section, I will introduce the two different methods.

2.1. Calculate by Corresponding Points

When we rotate a group of coordinates, we can get the original coordinates by

$$x' = x \cdot \cos\theta - y \cdot \sin\theta$$

$$y' = x \cdot \sin\theta + y \cdot \cos\theta$$

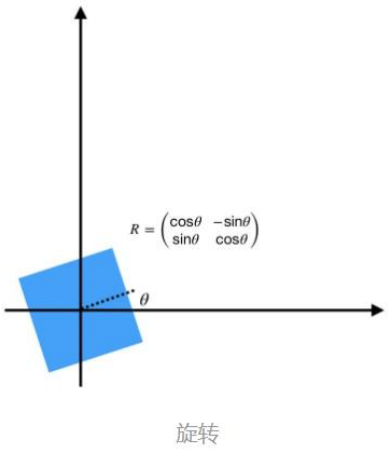


Figure3: rotation matrix

It is easy to find that we can write this rotation as a matrix multiplication

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = R \begin{pmatrix} x \\ y \end{pmatrix}$$

When we translate a group of coordinates, we can get the original ones by

$$x' = x + t_x$$

$$y' = y + t_y$$

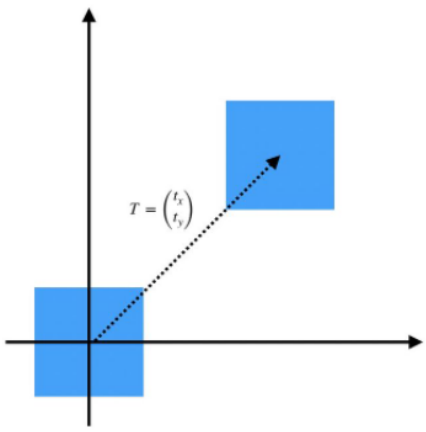


Figure 4: translation matrix

We can also write this translation into matrix form

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Now, we would like to add both matrix together, so we use a homogenous coordinate and get:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} I_{2\times 2} & T_{2\times 1} \\ 0^T & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Besides rotation and translation, we need to find the affine transformation of the coordinates, so we change I = [1 0;0 1] into A = [a_{11} a_{12};a_{21} a_{22}]

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{bmatrix} A_{2\times 2} & T_{2\times 1} \\ V^T & s \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = H_{3\times 3} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

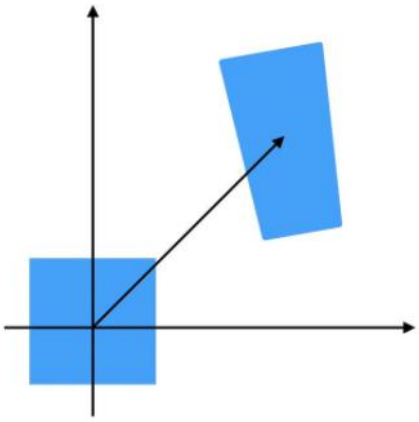


Figure 5: projection matrix

To calculate each element of the projection matrix, we denote them as h_{11} to h_{33}, and calculate them by the following two equations:

$$(h_{31}x_i + h_{32}y_i + h_{33}) \cdot x'_i = h_{11}x_i + h_{12}y_i + h_{13}$$

$$(h_{31}x_i + h_{32}y_i + h_{33}) \cdot y'_i = h_{21}x_i + h_{22}y_i + h_{23}$$

We can find that one pair of corresponding points can get two equations. Since it is

linear, we can let $h_{33} = 1$, then we need to calculate the rest 8 variables, which can be get from 8 equations. Therefore, we should find four pairs of corresponding points.
In order to get those points, we upload the pictures sent from client to MATLAB.
By using the command "a=imread('picture.jpg')", the picture we choose can be translated into array as a variable, then we use the command "imshow(a)" to show the picture in MATLAB. Next, we use the command "ginput" to get the point we click like the following picture

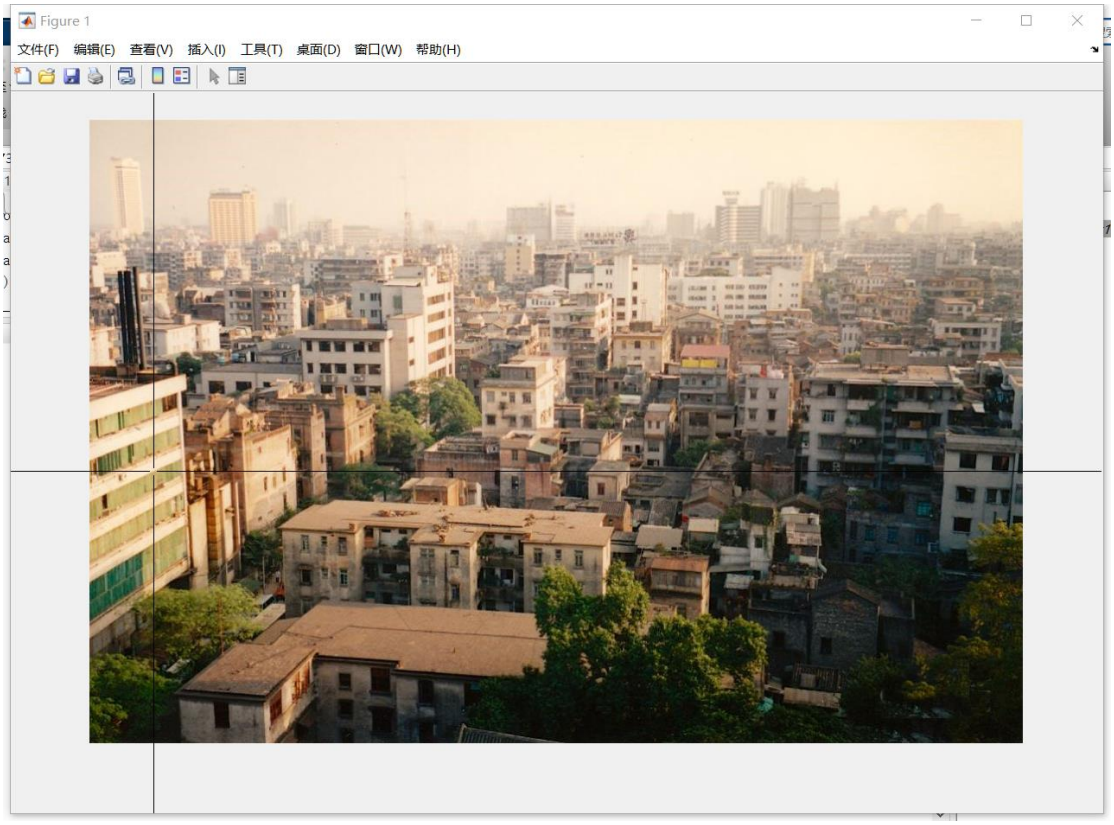


Figure 6: ginput command

We choose four points in the image which we already know their origin coordinate and calculate with the script show in Appendix. Finally, we can get the homogrphy H from the equations shown above.

2.2. Camera Calibrator

Camera Calibrator is a tool in MATLAB toolbox, which can calculate homography by just uploading 10-15 pictures of checker taken in different directions.
We type "cameraCalibrator" in the command window fo MATLAB to open the tool.

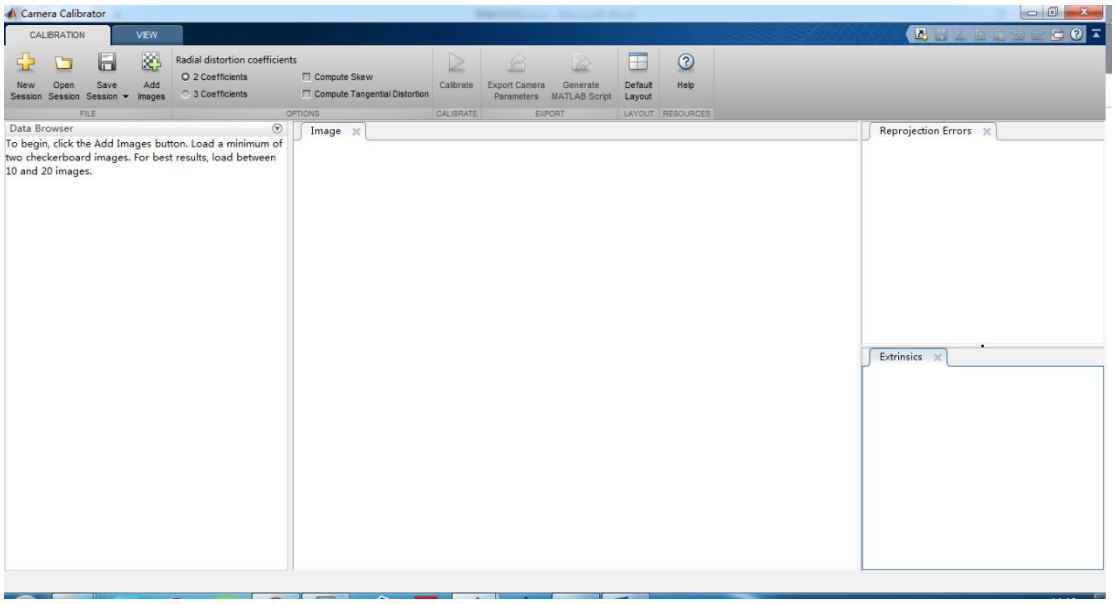


Figure 7: camera calibrator

Then we click add image to add the pictures and choose the tangential distortion mode.

$$x_{corrected} = x + [2p_1xy + p_2(r^2 + 2x^2)]$$
$$y_{corrected} = y + [p_1(r^2 + 2y^2) + 2p_2xy]$$

Simulate the tool, and we finally get the result like the following figure:

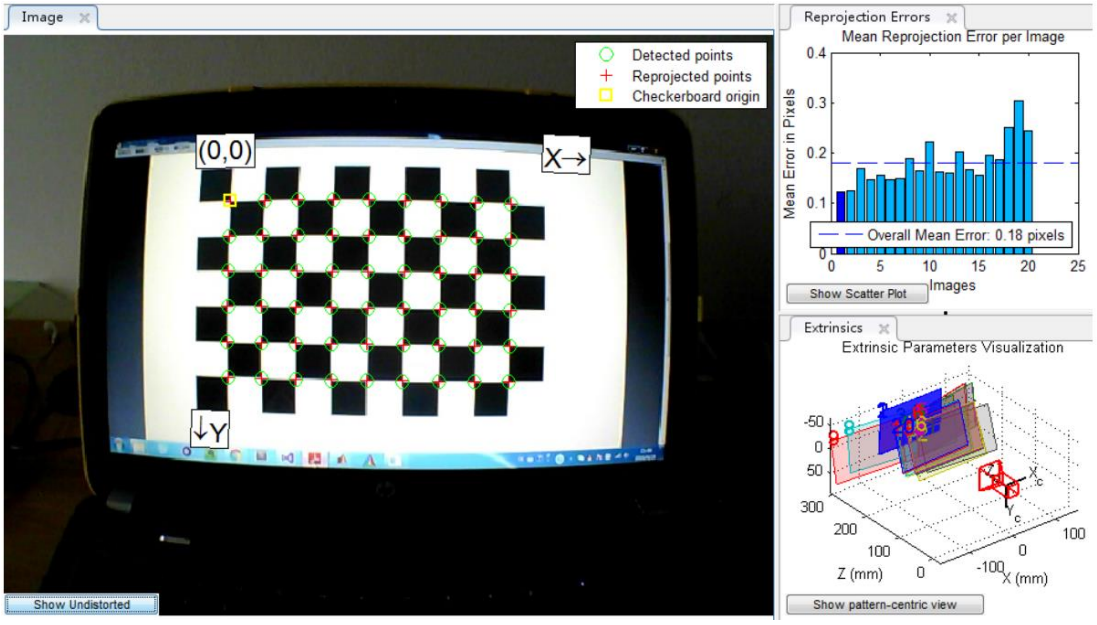


Figure 8: result of tool

We can find the homography from variable section of MATLAB. In the next part, I will show the result and analyze of both methods.

3. Result and Analyze

We use a checker like the following figure:

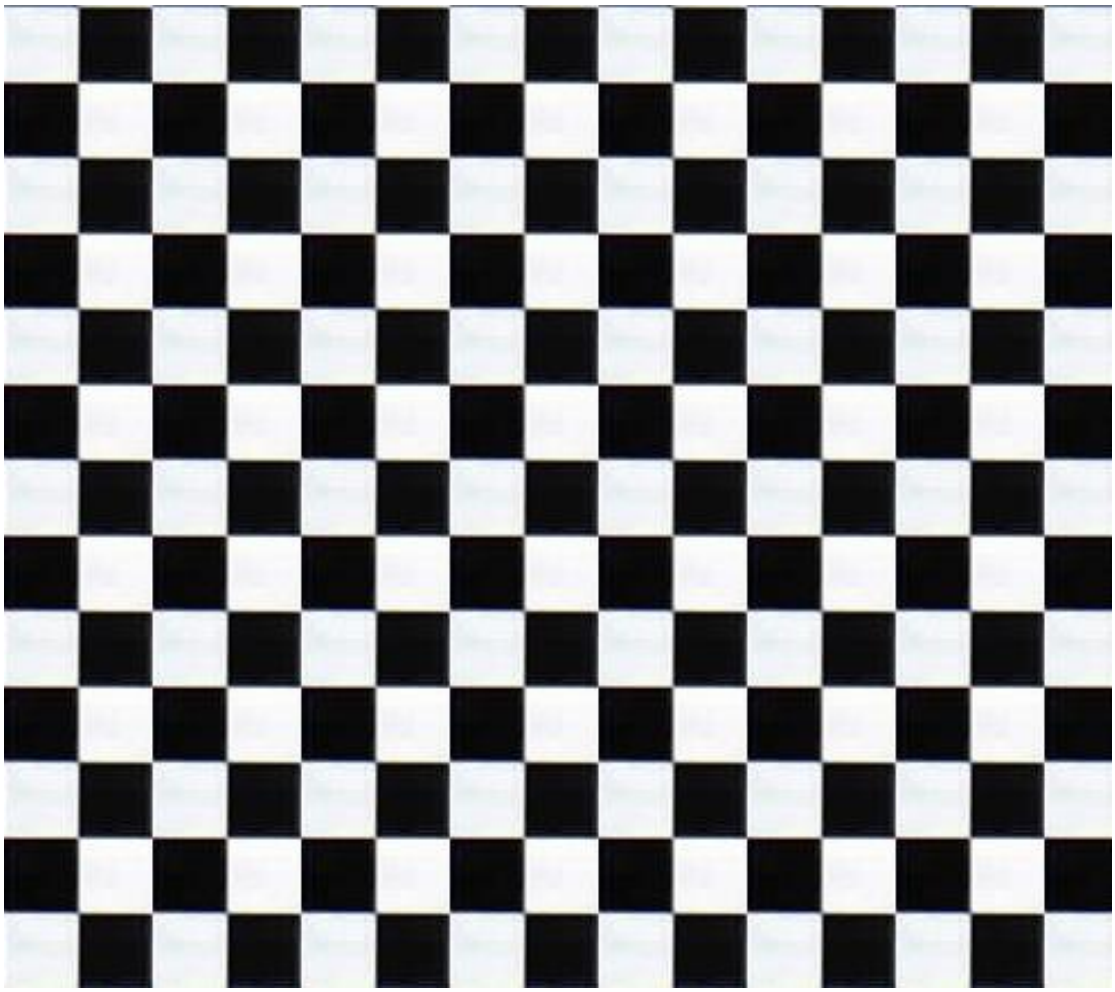


Figure 9: checker

The slide length of each square is measured to be 15mm.

3.1. Calculate by Corresponding points

We use this method to calculate the length of the ID card to test the precision of the homography.

Figure 10: test sample

From the result, we find that the original length of the ID card is 82mm, however, the calculated length is 110.93 mm.

3.2. Camera Calibrator

This part needs further test.

3.3. Analysis

For the first experiment, the error is noticeably large. We conclude two reasons for this error:

1) We take the points by clicking the figure, this procedure may lead to large error since we cannot get exactly the point we need.
2) The picture is distorted during taking the photo, and this distortion is not taking into consideration when we calculate the homography.

After we complete the second test, we can continue the analysis of that part.

4. Conclusion

From the test, we find that using camera calibrator can get more accurate homograpy since it takes camera distortion into consideration besides calculating the projection. Thus, we decide to use the second method in our whole program. From the test, we find some request of the photo we take. Firstly, it should be clear enough that computer can recognize the whole post and the checker. Secondly, it is important to change the direction which we take the photo because the internal algorithm needs those pictures to calculate the distortion of camera. Thirdly, we should write the whole procedure as a script to realize the automation of the program.

Besides completing the second test, we still have some other objectives. Firstly, there's still some problems about sending data between client and server. This part is quite important in a C-S structure, so we need further experiment to choose a stable and quick protocol. Secondly, it comes to the feel of users, we should simplify the operation of the app so that people can use it easily.

5. Appendix

This part will show the code of method 1.

```
K=[x1 y1 1 0 0 0 -xx1*x1 -xx1*y1 -xx1;0 0 0 x1 y1 1 -yy1*x1 -yy1*y1 -yy1;
   x2 y2 1 0 0 0 -xx2*x2 -xx2*y2 -xx2;0 0 0 x2 y2 1 -yy2*x2 -yy2*y2 -yy2;
   x3 y3 1 0 0 0 -xx3*x3 -xx3*y3 -xx3;0 0 0 x3 y3 1 -yy3*x3 -yy3*y3 -yy3;
   x4 y4 1 0 0 0 -xx4*x4 -xx4*y4 -xx4;0 0 0 x4 y4 1 -yy4*x4 -yy4*y4 -yy4];

H=null(K,'r');

size_of_H=size(H_initial);
width=size_of_H(2);
for i=1:width
    if H_initial(9,i)~=0
        break;
    end
end

H_temp=H_initial(:,i).';
H=[H_temp(1,1:3);H_temp(1,4:6);H_temp(1,7:9)];
```

H is the final result of the homography, which can help us get original coordinate by just multiply it to the points get from uploaded figure.

6. Reference

[1] Lin L , Wang L V . Photoacoustic Imaging[M]. 2021.