Huan-Yang Chang
Team G: Excalibr
Teammates:
Man-ning Chen, Yiqing Cai, Sambuddha Sarkar, Siddharth Raina

ILR05
11/22 2016

Individual Progress

1. Integrate the geometric calibration pipeline

Basing on the structure of calibration pipeline we designed, we combined the FPN and photometric calibration in the geometric pipeline(fig 1). That was we first read the raw image and using the FPN parameter to reduce the dark noise and then used the inverse response function to get the better image values. After we finished the image processed, we used that filtered image to detect corner and conduct the original camera calibration.

Fig 1. Combined FPN and Photometric calibration in geometric calibration pipeline

However, because we have the slowness issue, we thought that some of the calibration might not necessary for our calibration pipeline. We thought photometric possibly had not much influence on the geometric calibration because it was using the inverse response function to map the original pixel values to new one. For geometric calibration site, the corner detection was not influenced by that mapping. To prove that we make an experiment to show the difference of reprojection error between photometrically calibrated images and no-photometrically calibrated images(Table 1.).

# of images	Reprojection error with photometric calibration	Reprojection error without photometric calibration
88	0.335	0.333
176	0.331	0.329

Table 1. The reprojection error in geometric calibration with photometric calibration and without photometric

calibration

By that result and consult with our sponsor, we decided to remove the photometric calibration from geometric calibration pipeline(Fig 2.).

Fig 2. Calibration pipeline without photometric calibration

2. Time cost and the slowness analysis of the OpenCV calibration algorithm

For increase the speed of our geometric calibration, we first wanted to investigate the slowness part of the geometric calibration. In the OpenCV camera calibration function, we can approximately separate to two steps : corner detection and calibration parameters optimization. Hence, we conducted experiments to analyze the time cost in the OpenCV.

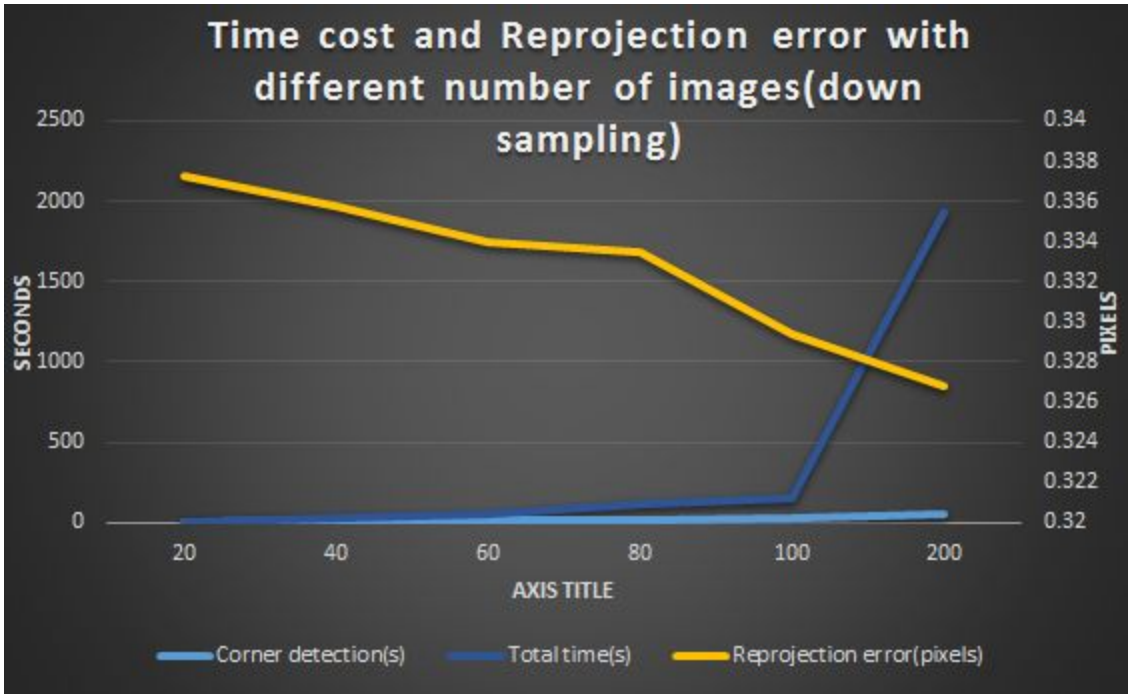

Fig 3. The time cost and Reprojection error in different number of images

By this experiments, we could found out that the most of the time was spending on calibration parameters optimization in a large amount of images (> 100 frames). By reading the document of the OpenCV, we found that the method OpenCV used was Levenberg-Marquardt optimization. For the same time, the Matlab camera Calibrator was using the similar algorithm to conduct the optimization, but it much faster.(400 images for about half hour in Matlab v.s. hours in OpenCV) Hence, we might need either to find a suitable algorithm or study the method used in Matlab to improve ours.

Challenges

1. Algorithm improvement

For the algorithm improvement, I had to read the technical document and try to understand the idea and the method they used and maybe try to implement it. It always took me a lot of time and made a little progress. It felt like there is no correct answer there, and you had to try and adjust the direction in the same time. Hence, I didn't have a very clear path for this and worried about that it would delay our whole project's progress.

.

Teamwork

Work was done in this week	Name
Aerotech robot arm trajectory design and programming environment implemented in MATLAB	Sam
Combine FPN and Photometric calibration in geometric pipeline	Peter / Cece/Mandy
Finished Sensor noise and Photometric Pipeline	Cece/Mandy
Conducting the experiment for slowness analysis	Peter/Sam
Using FFT to validate the result of FPN	Sid/Mandy

In this week, the team tried to integrate the individual works on geometric calibration together. For Aerotech robot arm, Sam was doing the Matlab programming to simplify the future usage with the robot arm. I, Cece and Mandy tried to make our individual program into a single function that could be reused in another program. And then we combined the filter part and the geometric calibration. Cece and Mandy also finished the pipeline for FPN and photometric calibration parameter creation pipeline. Sid and Mandy started to use FFT to validate the result of FPN.

Future Plans

1. Visualize the calibration result with reprojection error and it positions in field of view

   In earlier progress, we only using the RMS of the reprojection error, however, it is not informative for validation, because the single value can only tell us the approximate results. Hence, we want to visualize the position and its corresponding reprojection error. And this figure can help us to validate our calibration results.

2. Improve the slowness problem in geometry calibration

   I will try to understand the difference in the implement of estimation of camera calibration in the Matlab calibration and OpenCV camera calibration.