IRL #4: Progress Review

Man-Ning Chen (Mandy)

Team G: EXCALIBR

Teammates: Yiqing Cai, Huan-Yang Chang, Siddharth Raina, Sambuddha Sarkar

Individual Progress

Overview

This week, our main objects were photometric and geometric calibration. Because Cece and I are both assigned to sensor calibration (CMOS) in the team and we collected data and did experiments together, our ILR report have many overlap. She helped me with FPN calibration last week, and after FPN finished, I move on to work on photometric with her. In addition, since most of the programs in Oculus are developed on Linux operating system. For convenience in the future, our team decides to move our working space from Windows/Mac to Linux. Meanwhile, I am now working on translating my code from Matlab to C++. This is for better integration with the whole system.

Photometric Calibration

For photometric calibration, we have tried two methods. The first one is from "Recovering High Dynamic Range Radiance Maps from Photographs "[1]. The second one is from " A Photometrically Calibrated Benchmark For Monocular Visual Odometry"[2]. The first one is shown as followed,

$$Z_{ij} = f(E_i \Delta t_j)$$ z: pixel value, E: irradiance, delta t: exposure time

Let $g = \ln f^{-1}$, irradiance E can be computed by the following equations:

$$g(Z_{ij}) = \ln E_i + \ln \Delta t_j \qquad w(z) = \begin{cases} z - Z_{min} & \text{for } z \leq \frac{1}{2}(Z_{min} + Z_{max}) \\ Z_{max} - z & \text{for } z > \frac{1}{2}(Z_{min} + Z_{max}) \end{cases}$$

$$\ln E_i = \frac{\sum_{j=1}^{P} w(Z_{ij})(g(Z_{ij}) - \ln \Delta t_j)}{\sum_{j=1}^{P} w(Z_{ij})}$$

Since this algorithm is based on solving Singular Value Decomposition of all image pixels with all exposure ranges, with over 12,000,000 pixels in each images and more than 1000 different exposures required, the matrix will be very huge. Doing SVD of the matrix requires a lot of memory. Therefore, we have to downsample the image. Nevertheless, we found that it is not easy to find a good scene which provides constant light source and enough intensity diversity. Although downsampling can resolve memory diversity, but the intensity diversity is still insufficient for solving the SVD. The rank deficiency problem prevented us from getting best results of inverse irradiance function g. To solve this problem, we tried the 2nd method. In the second method, I(x) represents observed pixel values; both G and V are only observable up to a scalar factor.

$I(\mathbf{x}) = G\big(tV(\mathbf{x})B(\mathbf{x})\big)$ let $B'(\mathbf{x}) := V(\mathbf{x})B(\mathbf{x})$ and $U := G^{-1}$

we get

$$E(U, B') = \sum_i \sum_{\mathbf{x} \in \Omega} \left(U\left(I_i(\mathbf{x})\right) - t_i B'(\mathbf{x}) \right)^2$$

E is the difference between the image corrected by inverse irradiance function and the true irradiance (Exposure time times irradiance map).
We should make E as small as possible.
The algorithm is shown as follows,

1. Set the input image as B' and times it with t
2. calculate U(k):

$$\Omega_k := \{i, \mathbf{x} | I_i(\mathbf{x}) = k\}$$

$$U(k)^* = \underset{U(k)}{\operatorname{argmin}} E(U, B') = \frac{\sum_{\Omega_k} t_i B'(\mathbf{x})}{|\Omega_k|}$$

3. Use U(k) to calculate B'(k)

$$B'(\mathbf{x})^* = \underset{B'(\mathbf{x})}{\operatorname{argmin}} E(U, B') = \frac{\sum_i t_i U\left(I_i(\mathbf{x})\right)}{\sum_i t_i^2}$$

4. Go back to 2. Iterate 10~50 times

Figure 1 compares Method 1 and Method 2 with different

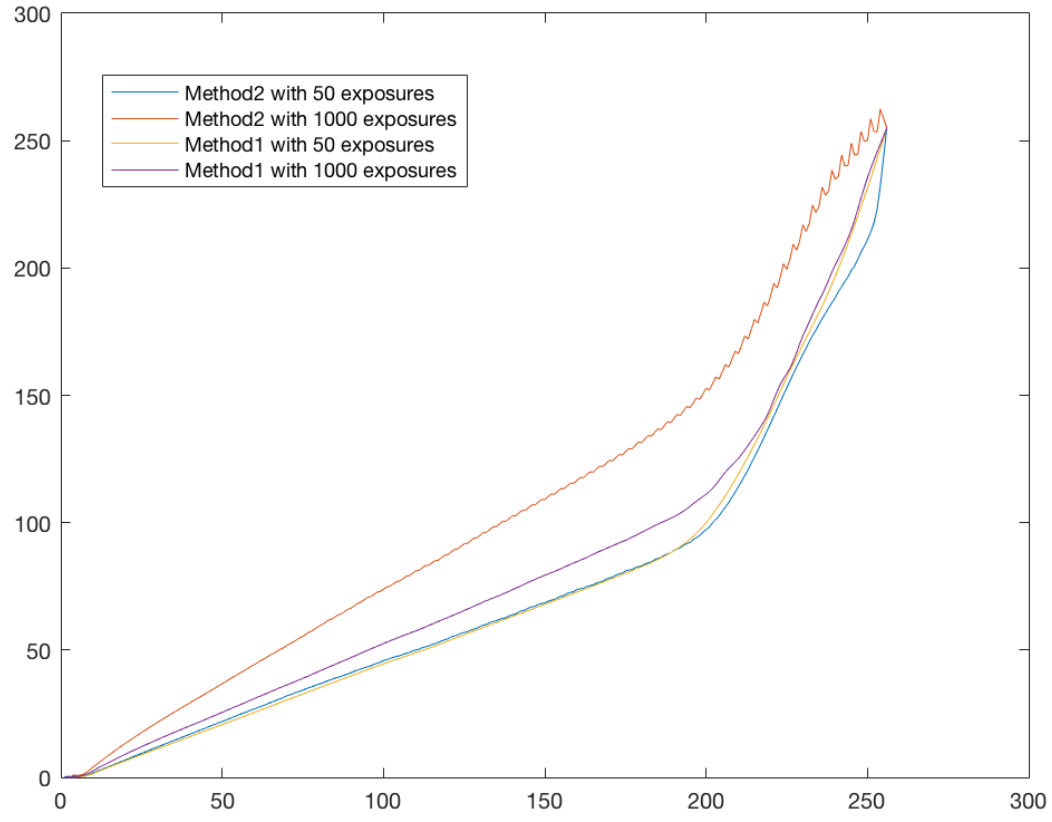
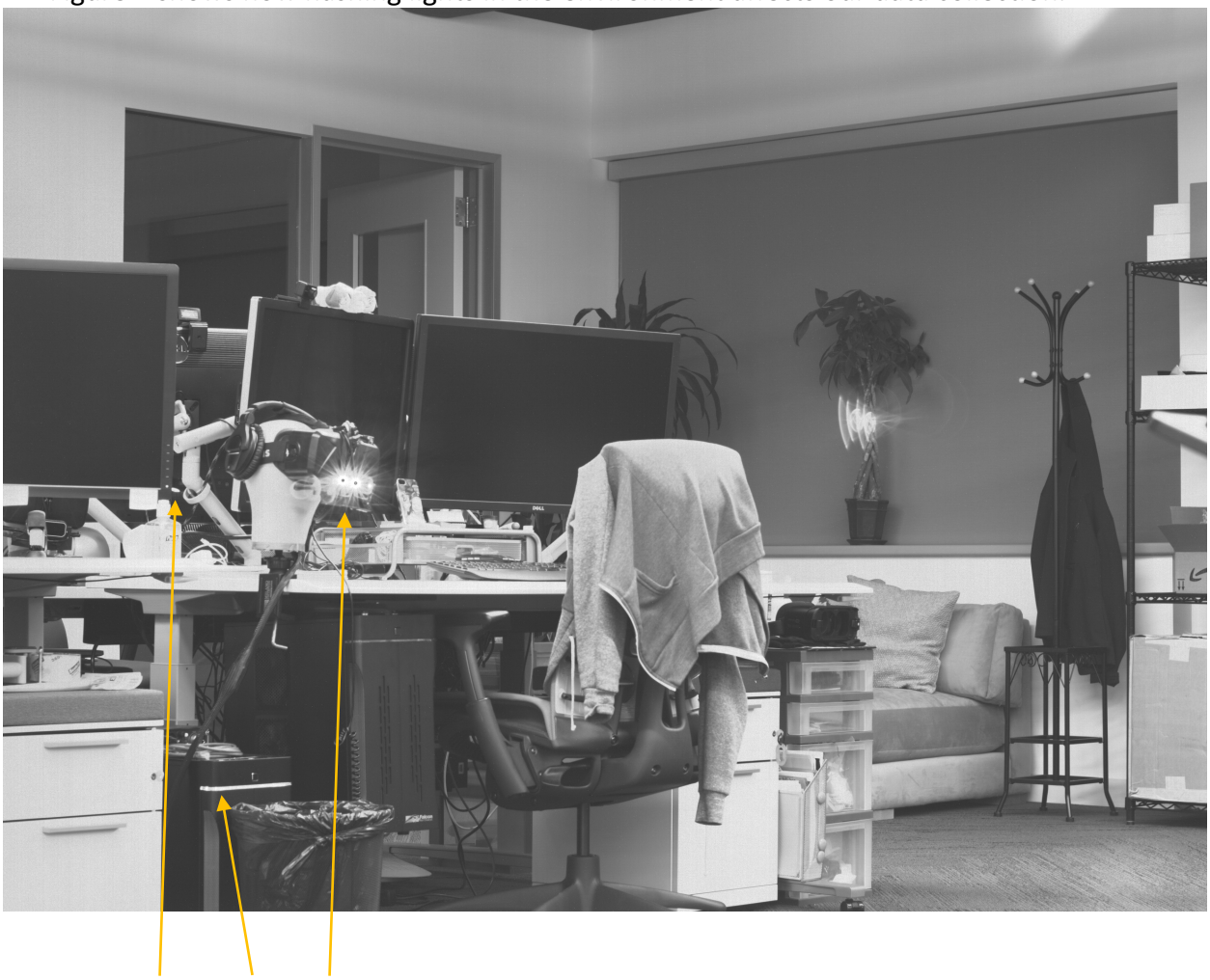


Figure 1

Challenges

1. It is difficult to get images of a scene with constant light source.

Figure 2 shows how flashing lights in the environment affects our data collection.



Flashing lights

Many factors in the working environment such as reflections or flashing lights on computers are severely affects our data collection. Since photometric calibration requires regular scene rather than just a monotonous colors or brightness (like FPN). Good environment is needed.

2. Operating system changing is quite challenging. We have to get familiar with different operations on the computer and a more difficult but professional coding environment.

Teamwork

Peter:

1. Implement geometric calibration with OpenCV
2. Build up evaluation methods for the geometric calibration

Cece:

1. Conduct paper research
2. Implement photometric calibration algorithm in Matlab and in C++

Sam:

1 Generate AEROTECH robot trajectory
2 Do velocity profiling & troubleshooting

Sid:

1 Correct raw images with parameters derived from FPN and photometric calibration for geometric calibration.

Future Plan

1. Calibration Pipeline integration:
   Integrate FPN algorithm with photometric calibration
2. FPN and photometric calibration data collection
   To expedite data collection process and refine data quality, automate picture acquisition process and connect the process to FPN calibration.

References:

[1] Debevec, Paul E., and Jitendra Malik. "Recovering high dynamic range radiance maps from photographs." *ACM SIGGRAPH 2008 classes*. ACM, 2008.
[2] Engel, Jakob, Vladyslav Usenko, and Daniel Cremers. "A photometrically calibrated benchmark for monocular visual odometry." *arXiv preprint arXiv:1607.02555* (2016).