CS 4391 Introduction to Computer Vision
Homework 4

Professor Yu Xiang

March 26, 2024

Download the homework4_programming.zip file from eLearning, Assignments, Homework 4. Finish the following programming problems and submit your scripts to eLearning. You can zip all the data and files for submission. Our TA will run your scripts to verify them.

Install the Python packages needed by

- pip install -r requirement.txt

Here are some useful resources:

- Python basics `https://pythonbasics.org/`
- Numpy `https://numpy.org/doc/stable/user/basics.html`
- OpenCV `https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html`

Problem 1

(3 points) Backprojection.

Implement the backproject() function in backproject.py. The function takes a depth image with height H and width W, and a camera intrinsics matrix as input, and outputs a point cloud with shape $H \times W \times 3$ generated from the depth image.

After your implementation, run the backproject.py in Python to verify it. Figure 1 shows an example of running the script. It shows the 3D points on the cracker box.

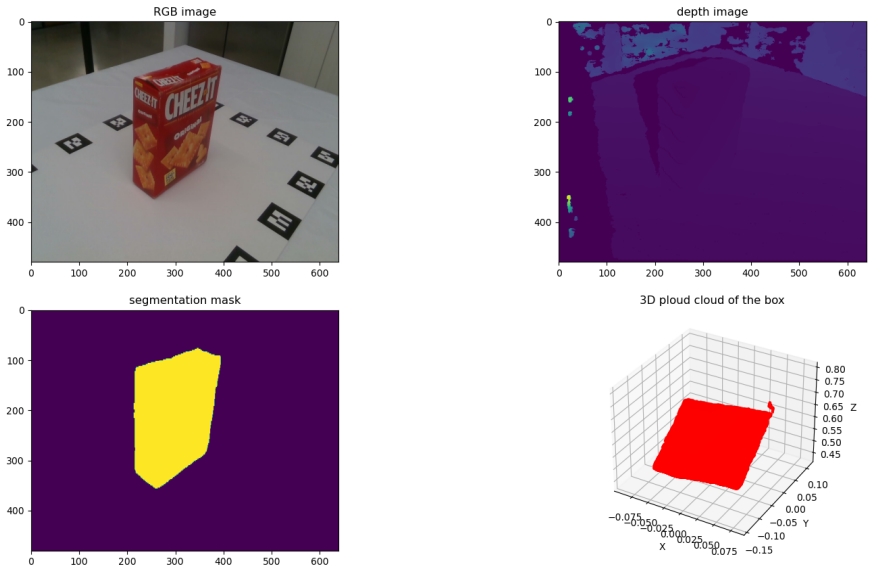

Figure 1: Backprojection.

Problem 2

(3 points) Correspondences.

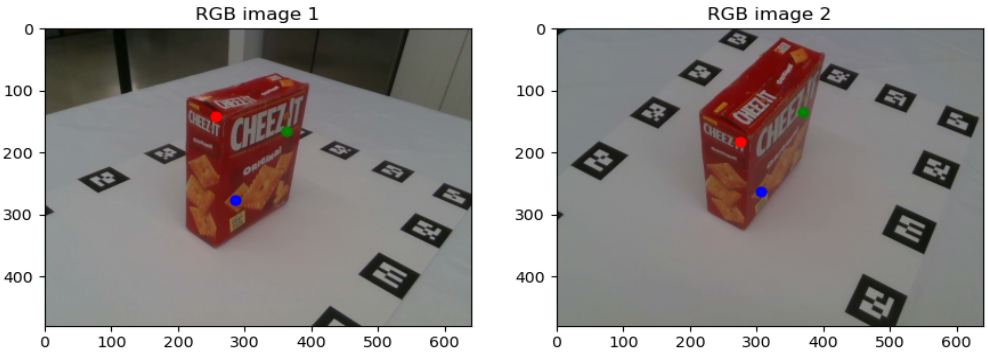

Figure 2: Correspondences

Implement the missing code in correspondences.py. This script finds the correspondences of pixels between two images using their camera parameters. You need to use the backproject() function in Problem 1.

After your implementation, run the correspondences.py in Python to verify it. Figure 2 shows an example of running the script. The 2D points with the same color correspond to each other.

Problem 3

(4 points) Epipolar Geometry.

Implement the compute_fundamental_matrix() function in epipolar.py. This script first samples a set of pixels on image 1, and then uses the depth and camera poses to find the correspondences of these pixels on image 2. You need to use your backproject function and the method for finding correspondences in problem 1 and problem 2 in this script.

After generating the correspondences, implement the 8-point algorithm to compute the fundamental matrix between these two images. Then the code will sample 3 pixels on image 1 and draw their epipolar lines on image 2 using the computed fundamental matrix.

After your implementation, run the epipolar.py in Python to verify it. Figure 3 shows an example of running the script.

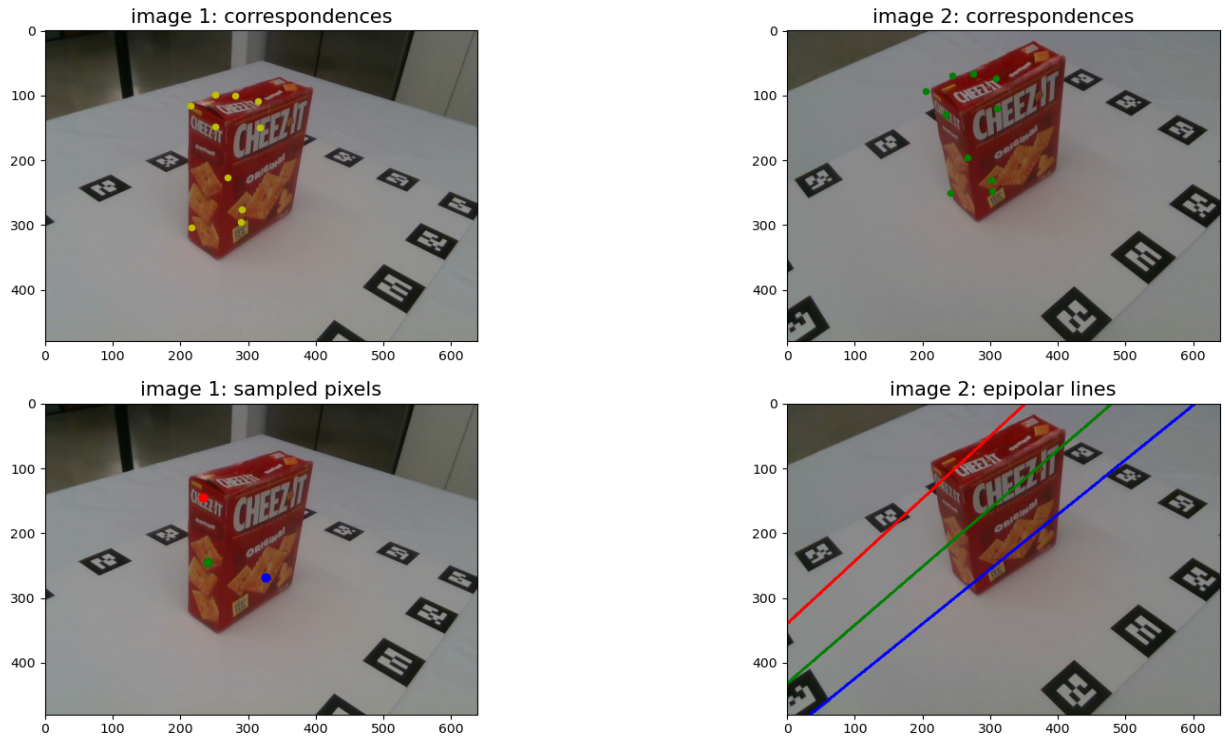

Figure 3: Epipolar lines from fundamental matrix.