Project 3: Stereo from Disparity

Out: Tue Dec. 05, 2023
Due: Fri Dec. 15, 2023 (deadline: 11:55PM)
Revised version v2

Late submissions: Late submissions result in 10% deduction for each day. The assignment will no longer be accepted 3 days after the deadline.
Grading: 25% for implementations and 75% for experiments and written report including graphs, results, and critical discussion.

Office hours:

		Mon	Tue	Wed	Thur	Fri
Guido Gerig	gerig@nyu.edu				2-3pm ZOOM	
Pragnavi Ravuluri Sai	pr2370@nyu.edu	4-6pm				
Sai Rajeev Koppuravuri	rk4305@nyu.edu			4-5pm		

Please remember that we also use campuswire for communication on homeworks.

Please read the instructions carefully. Note: we will not be running code. Rather, we will check your code to make sure your implementation is your own, and it matches your results. Your grade is primarily based on your written report. This means going beyond just showing results, but also describing them. You should produce a standalone lab report, describing results in enough detail for someone else (outside of class) to follow. Please read every page in this assignment. **Please submit a single PDF/HTML with all code included as an appendix.**

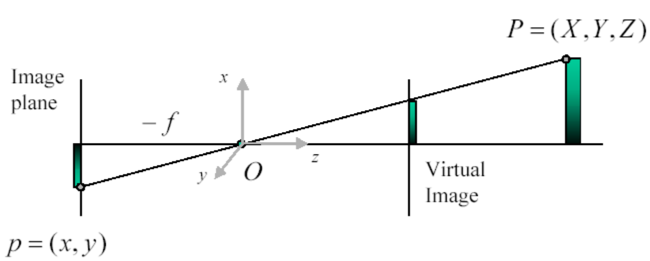
Important: Given the short time frame for this Project 4, you can either choose **Problem 1 (Triangulation)** or **Problem 2 (Dense depth map)** as your project. You can get a maximum grade of 100 with a successful solution of one of them, given convincing results and good critical discussion of your results.
Eventually providing solutions for both, Problem 1 and Problem 2, will earn bonus points.

Problem 1: 3D Object geometry from triangulation

Setup your own experiment with a stereo pair using your own camera or mobile phone, e.g. see the following example of a box where corresponding corners are easy to detect. We don't go through a complex calibration but estimate the major parameters via a simple pre-calibration:

1.1 Precalibration:

- Put an object of known size ***X*** in front of your camera, measure the distance ***Z*** of this object from your camera lens, best in [mm], and take a picture. Given our pinhole camera model, we get $\frac{x}{f} = \frac{X}{Z}$, so that the ratio of $\frac{x}{f}$ is known. On your picture, you measure the imaged object size **x** in pixel units. We also assume pixels to be squares so that pixel dimensions are the same in x and y, and you can do these measurements for a vertical or horizontal object. Given x of your test object in pixel units, you can get the focal length also in pixel units: $f(pixels) = x(pixels)\frac{X(mm)}{Z(mm)}$, where ***X***, ***Z*** and ***x*** are known.
- Assume that the intersection of the optical axis with the sensor is in the middle of your image, e.g.. for a sensor of 400ox3000 pixels, it would be at $(u_0,v_0)=(2000,1500)$. All measurements of pixel coordinates can then expressed relative to this center (u_0,v_0), i.e. $\boldsymbol{x}=(u_i-u_0,v_i-v_0)$ to get positive and negative coordinates.

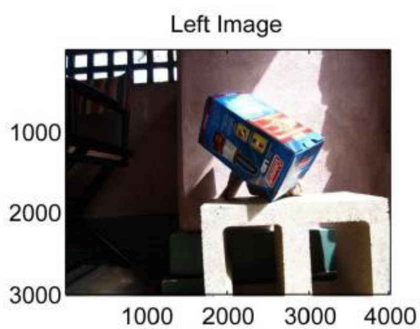


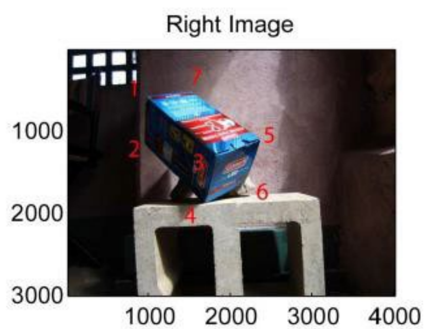
1.2 Calculate 3D coordinates with a set of manually selected landmarks.

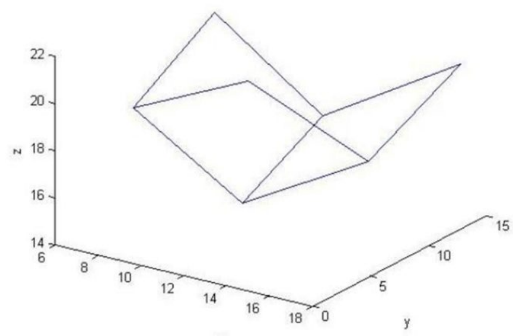
Now, we will make use of the basic perspective projection equations: $Z = \frac{fB}{d}$, $X = Z\frac{x}{f}$, and $Y = Z\frac{y}{f}$, in order to calculate world coordinates of each point detected via corresponding points in stereo pairs:

- Choose a 3D object of your choice. Given the discussion in our course, get relatively close to the object so that the disparity gets much larger and your depth perception is more pronounced.
- Move your camera along the horizontal x-axis to take a left and right picture. Best will be to move the camera along a thick book, wall or other supporting back.
- You need to measure the baseline shift **B** of the camera position between the left and right views.

- Click at a set of corresponding left and right points of your favorite object and record the $\boldsymbol{x}$=(x,y) locations, e.g. the example images of the box shows n=7 visible corners. Calculate these $\boldsymbol{x}$ coordinates as relative to the sensor center, $\boldsymbol{x}=(u_i-u_0, v_i-v_0)$.
- Calculate x-direction disparities d for all n left/right point pairs.
- Given your estimate of the focal length in pixel units, you can calculate world coordinate triplets (X,Y,Z) for each object corner point.
- Use a 3D polygon display tool (I will ask the TA's to provide python options) to display your 3D object as a wireframe structure by connecting the edges. You may encounter some limited precision, but own experiments by G. Gerig with a mobile phone show promising results given the strong simplifications.







Problem 2: Dense depth maps from dense disparity.

2.1 Camera precalibration: Follow guidelines as in 1.1 to get a rough calibration between disparity ***d*** in pixels and depth ***Z*** *in mm*.

2.2 Calculation of dense depth maps based on normalized cross correlation (NCC)

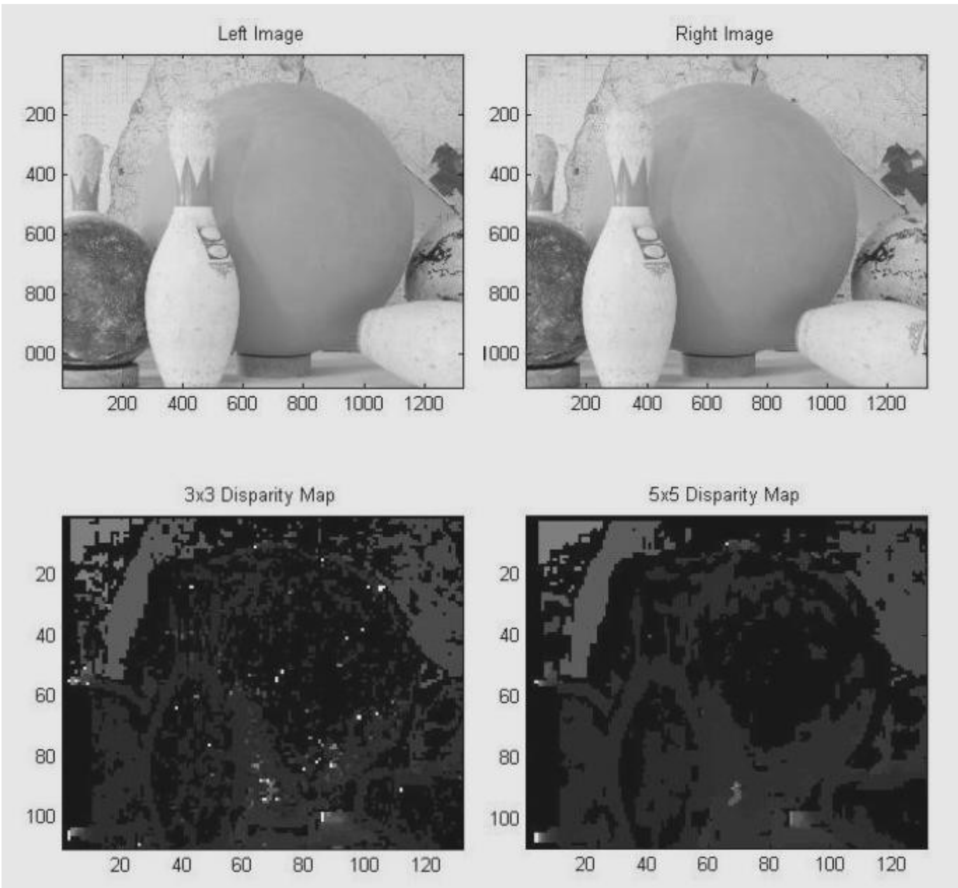
1. Use existing pairs that also come with ground truth depth maps from: https://vision.middlebury.edu/stereo/data/ or create your own set of parallel stereo images* with strong structures. Again here, please note that disparity gets much better if the object is relatively close to the camera. Given the large number of correlation operations, start with relatively small image sizes to test and debug.
2. *Take extra care with getting your own images if you wish, since even small shifts during taking the images may corrupt your results if we use small 3x3 or 5x5 filters. To be on the save side, you may choose to start with an existing set, and small subframe for testing/debugging.
3. Implement NCC along as discussed in the course slides based on template matching code which you already have implemented in Project 2. The difference will be that here, we normalize both, the left camera template window and the moving right camera template. Provide a size parameter so that you

can choose different window sizes of 3x3, 5x5 or else. As discussed, borders do not need to be processed.

4. Line by line along the vertical y axis, for each x-location in the left image, move the window along the x' axis in the right image. Find the best-match peak location for each run along right image pixel lines. The resulting pairs of x(left) and x'peak(right) locations are used to calculate the disparity d associated with each x(left pixel). These disparities can be stored in an additional disparity image array of the same size as the left and right images.
5. Given our precalibration, use these disparities (in pixel units) to estimate depth Z (in mm) for each pixel in the disparity image. Unlike Problem 1, we do not calculate (X,Y,Z) coordinates for each pixel as this would result in a dense point cloud. For simplification, we just calculate a depth map for each value in the disparity map and get a depth image the same size of the others.
6. The depth image can be displayed as a gray-level or pseudo-color image as you like. As you see in the following example image, use some contrast stretching to enhance depth perception, and note that the (x,y) pixels and depth Z have very different units. As discussed in the course. Z is nonlinearly related to disparity d, so that major depth differences can be seen in the foreground.
7. Test a 3x3 and 5x5 NCC run and compare.
8. As an optional choice you may even use a mesh display tool to get a 3D mesh of the (x,y,Z) depth map so that you could arbitrarily rotate this mesh.

Don't be disappointed if you get many failure pixels ore holes. First, not all pixels are to be seen by both camera views, and second, our NCC algorithm and peak detection may not always pick the “right” locations.

Display the left and right stereo pair and your resulting depth map for 3x3 and 5x5 NCC filtering, and write a short discussion paragraph with some critical assessment of your experiments.


Left Image
Right Image
3x3 Disparity Map
5x5 Disparity Map