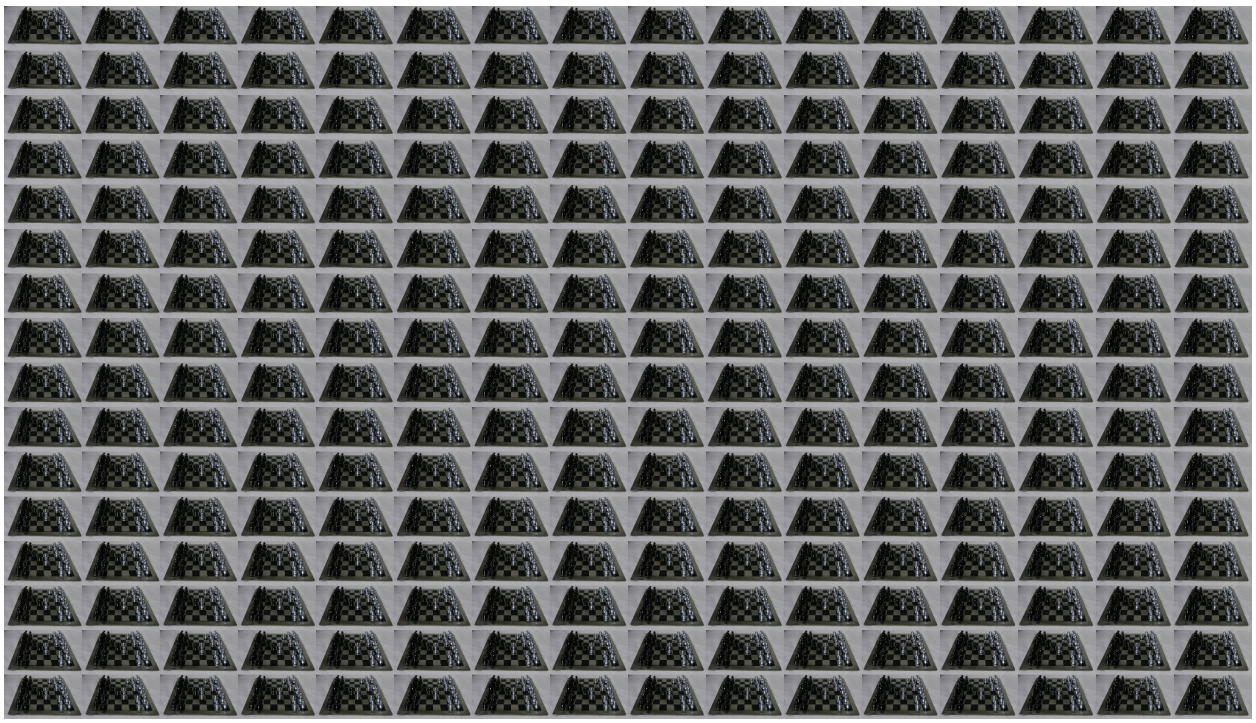
Homework Assignment 4
15-463 Computational Photography F22
Kavya Tummalapalli
Monday, November 7th, 2022

1. Lightfield rendering, depth from focus, and confocal stereo

Sub-aperture views



mosaic of sub-aperture views

Refocusing and focal-stack simulation

d = 0.4

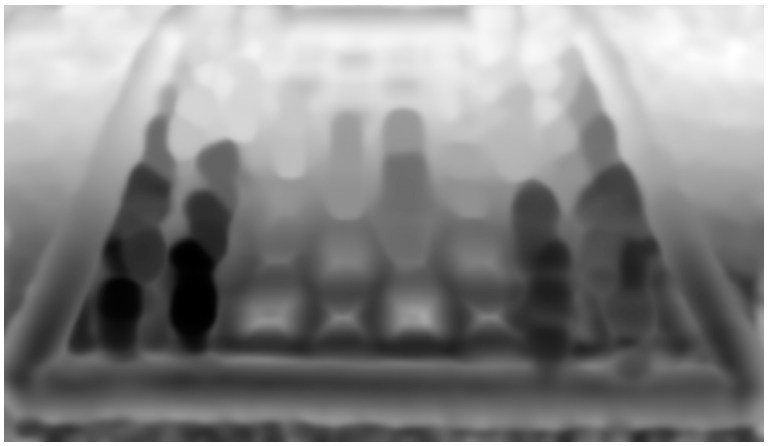
d = 0.0

d = -0.6

d = -1.0

d = -1.4

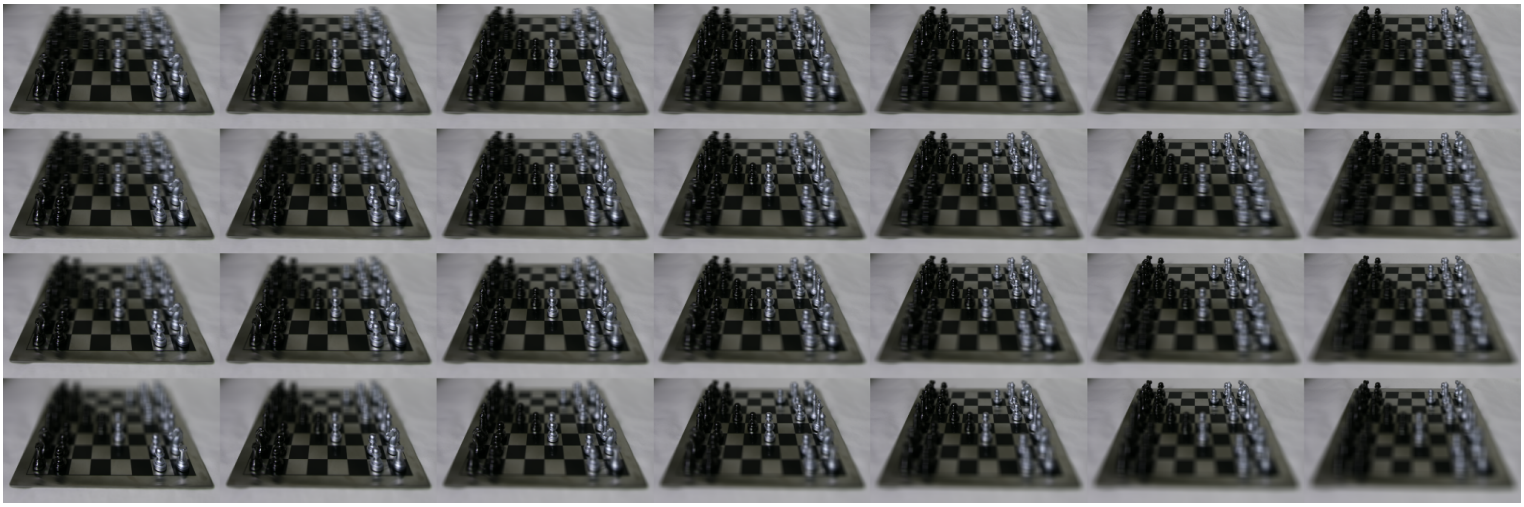
All-in-focus image and depth from focus

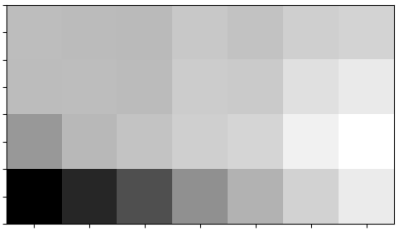


with $\sigma_1 = 2$ and $\sigma_2 = 5$ and
depth values: -1.4, -1.1, -0.8, -0.5, -0.2, 0.1, 0.4

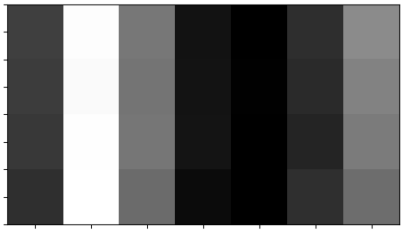
The parts of the depth map that have their depth incorrectly estimated are some of the parts of the chessboard and the overall front of the image. The all-in-focus image is similarly affected at these parts because the image is overall slightly blurry as we want to process entire neighborhoods of depths to compute sharpness.

Focal-aperture stack and confocal stereo

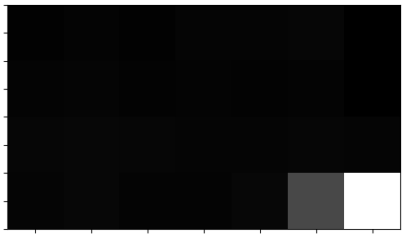




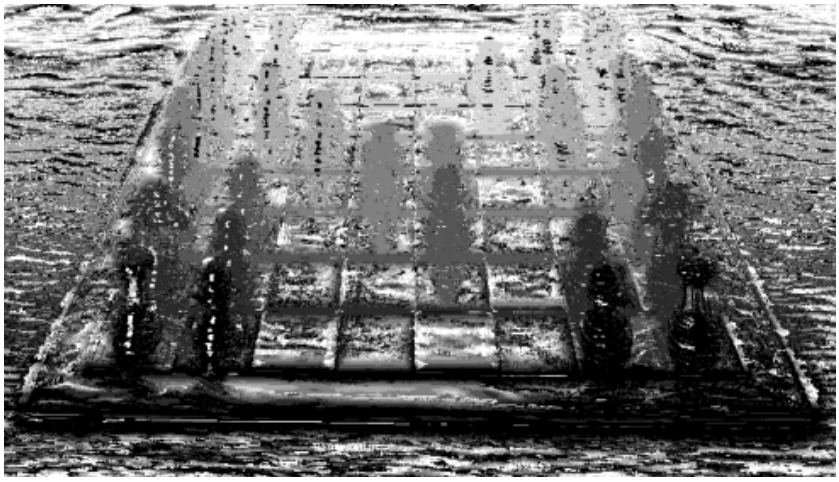
pixel (30,30)



pixel (100,241)



pixel (250,250)



with aperture values: 2, 4, 8, 16 and
depth values: -1.4, -1.1, -0.8, -0.5, -0.2, 0.1, 0.4

The depth map I recover here is more accurate and detailed compared to the one from the depth-of-focus procedure I implemented earlier as since we are computing depth by pixel, we can actually see the shapes of the chess pieces as the pieces are a different depth than their immediate background.

3. Capture and refocus your own lightfield

Capturing an unstructured lightfield

Video is attached to homework submission.

Refocusing an unstructured lightfield

The way I implemented the cross-correlation and shifting of the frames was that I took the first frame and found the box I wanted to use for my template g. This smaller image was now my template image that I wanted to look for in my other frames. I created a window of a size large enough that all the frames could find the template match in it, and this was my search window for all frames so that not all the pixels would have to be searched. Once I had this, I went through each frame and computed the normalized cross-correlation to find the shift of the template match in each frame, following the equation listed in the writeup. After this, I could shift each frame by the desired amount to have the template match match the first frame's position, and then I get the output of the template match (and any other objects at the same depth) being in focus while the rest of the image is blurred.