Step 3: Command line

```
imgtool lightfield --focalplanez 6.6 --camerapos  -0.15 -0.2 -1.2 --inputfov 50 --outputfov 40 --filterwidth
0.29 --griddim -0.6 0.6 -0.6 0.6 --camsperdim 16 16 --outputdim 300 300 book_rf.exr resolved.exr
```

‘book_rf’ here refers to output file of step2 with ray differentials.

Step 5:

- What happens to the images you generate if your light field is under sampled (the number of camera per dimension is small)? Why?

  If the light field is under sampled, then after bilinear interpolation, the rendered image would appear aliasing and look unrealistic. The aliasing effect results in multiple copies of a single feature appear in the final image. In order words, duplicated pixels would take up the space for other original features in pixel points. This would cause missing information/ features in the image and make the rendered light field look aliasing and fake.

- What happens to your final images if you seek to fix the artifacts described in the previous question by using many cameras but with very low resolution (say a 128x128 grid of 64x64 resolution cameras)?

  I don’t think that would fix the artifact in the previous question. Cameras with 64x64 resolution do not contain much information/ features originally. Thus, even if we did the bilinear interpolation using 128 x 128 grid of camera, it would not be able to recollect most of the missing features either, and thus would not have much effect in reducing the aliasing effect.

- Difficulties I came across when implementing assignment3:

  Assignment 3 is by far the most challenging assignment for me in this course. (Haven’t take a look at assignment 4 yet, hope it won’t take so much time). It took me a long time to understanding the pbrt, light field system and implementing coordinates transform.

  - First, it took me a long time to figure out how pbrt calls the ‘generateRay ‘function. I am totally unaware of the fact that each time it would pass in a single pixel point. I kept looping over all the points which causes a lot of trouble.
  - Second, although the paper seems quite clear. I was stuck in finding the position of output image in the synthetic aperture system. I used to think the focal plane is the output plane which is ridiculous. And I also think about the why the light field would create a ‘focus’ on the final image. When we use the pinhole camera with zero lens, everything is in foucs. But in step3, we gather several pinhole cameras to create a virtual aperture, which is similar to use a pinpole with a large aperture and thus he depth of field has thus changed.
  - It is a lot easier for me once I got the main idea about the light field and how to generate a new image from the lightfield. The only trouble I had is how to do the coordinates transform. I am still confused about the whole RasterToCamera, CameraToRaster transform. So in my implementation of step3, I do not use any of those transforms and simply calculate the geometric relation based on the ray direction and fov. I used the

similar triangle principle to calculate the ratio between different segment that I want to use and obtained their coordinates. It is more straightforward to me, but I hope I could get more intuition and dive a little deeper about why and how we should use these transforms.