CGRA352 Assignment 3:

How to run the program:

To run the program, ensure that there is a folder named ‘res’ the same ‘level’ as the executable. Inside the res folder include a folder named ‘jellyBean’ which contains all the light field images. When run, the program will do core,completion and challenge at once. Move the sliders provided to change the focus, aperture, xPos and yPos. After running the program, it will automatically create 5 different images with different focuses and print out the 4D point r=7 ,c=10 ,t=384 ,s=768.

generateSTArray():

This function will, for a given section, generate a set of concatenated images of a given number of rows and columns and use a given radius for its aperture.
The function will set any value outside the radius of the current section it's looking at, as black. Implemented in the program are two shapes for the aperture. The first uses sum of squared differences (SSD) to use a circular aperture. The second is the cityBlock function which instead uses the sum of the absolute values of the points to return a square/diamond shaped aperture instead.

Refocus():

This function uses a given focus and aperture value to sample the given images accordingly and produce a refocused image that is accordingly blurred/focused. This is done by for each row and column, checking each point in an image, and determining if it is ‘captured’ by the camera, in this case if the calculated SSD is less than the aperture squared. By doing this, we sample points at different locations and combine the pixels together creating a blur effect. Note that in this implementation we have to count how many times a pixel is added together so that it can be accurately divided afterwards, else the pixel values will be over 255 and thus the resulting image will just be white.
Note that when the aperture is a small value, then the SSD must also be a small value mimicking how a camera would sample less light by having a smaller aperture where light can enter the camera from.

In order to find the pixels to sample, I used the equation on page 53 of the reference:
Sum(u*(1-1/focus)+s,v*(1-1/focus)+t)/N

Moving the camera position:

By changing the centre of the UV coordinates, we are able to change where the virtual camera is located at, resulting in being able to move the virtual camera vertically or horizontally.

References:

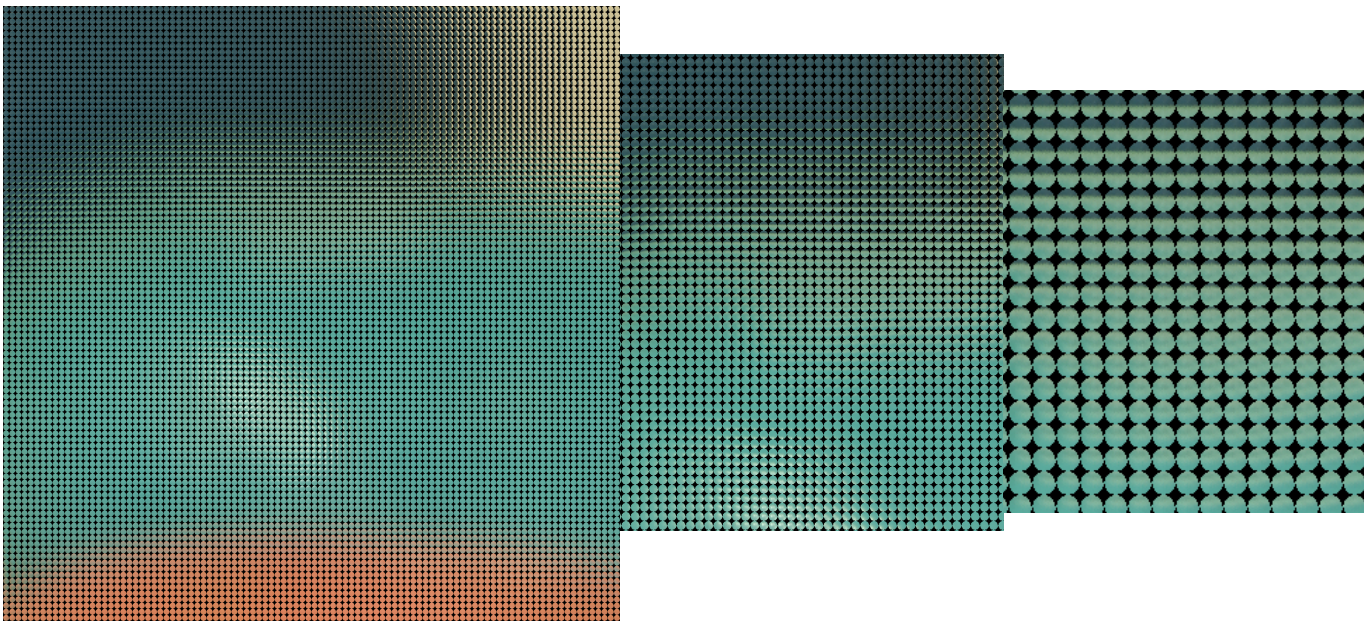
https://people.eecs.berkeley.edu/~ren/thesis/renng-thesis.pdf

Printing a 4D Point

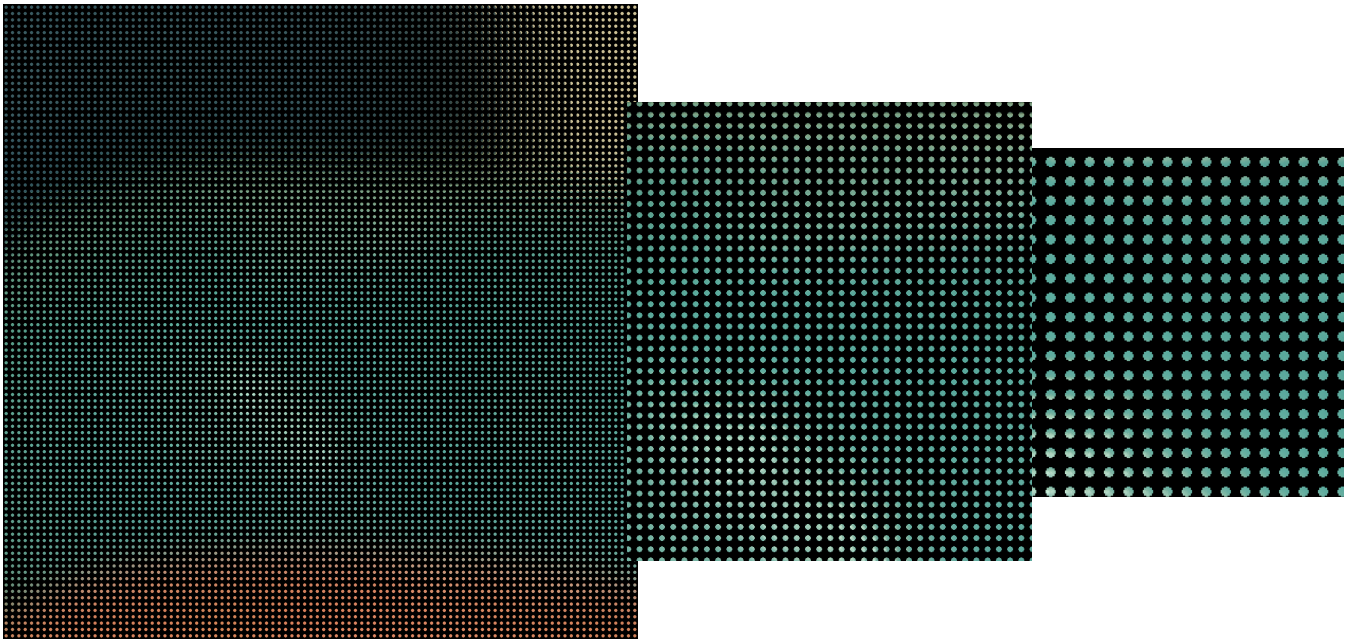
```
Loading light field ...
Finished loading light field

4D Point for r = 7 c = 10 t = 384 s = 768
[99, 135, 219]
```

ST Output:

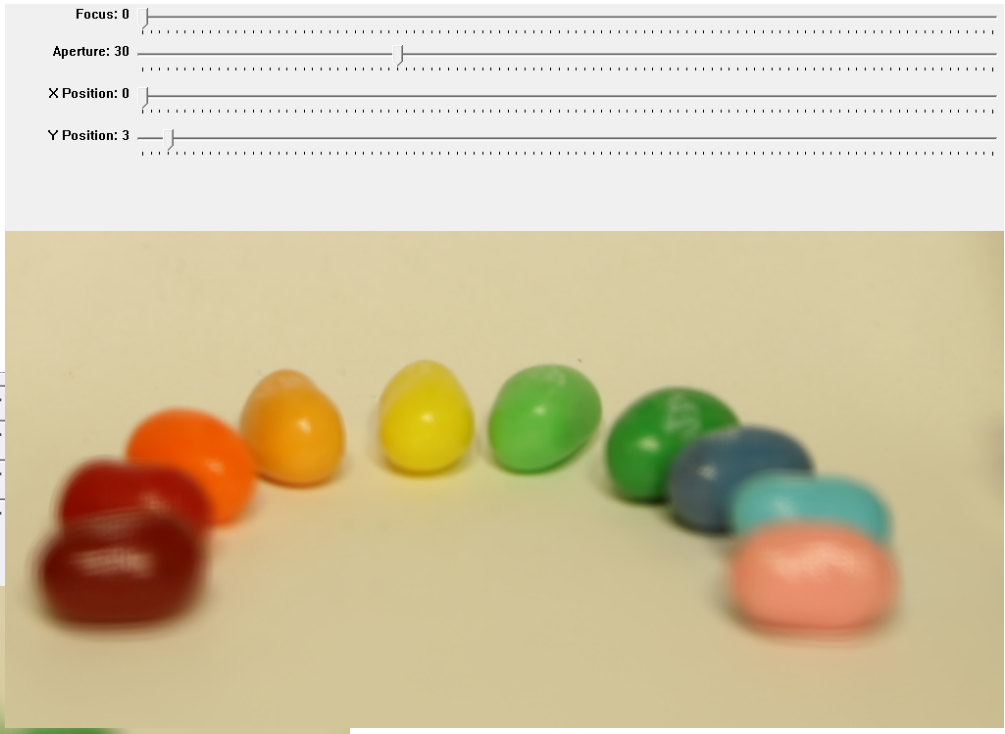
Aperture = 75

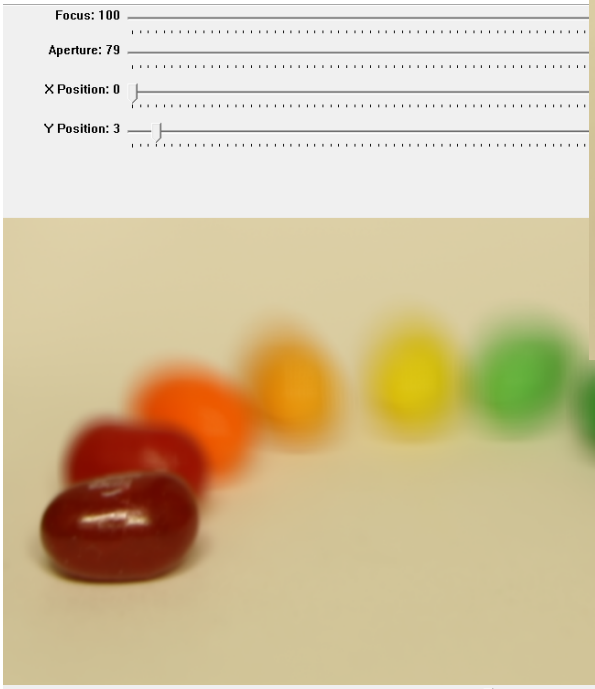


Aperture = 40



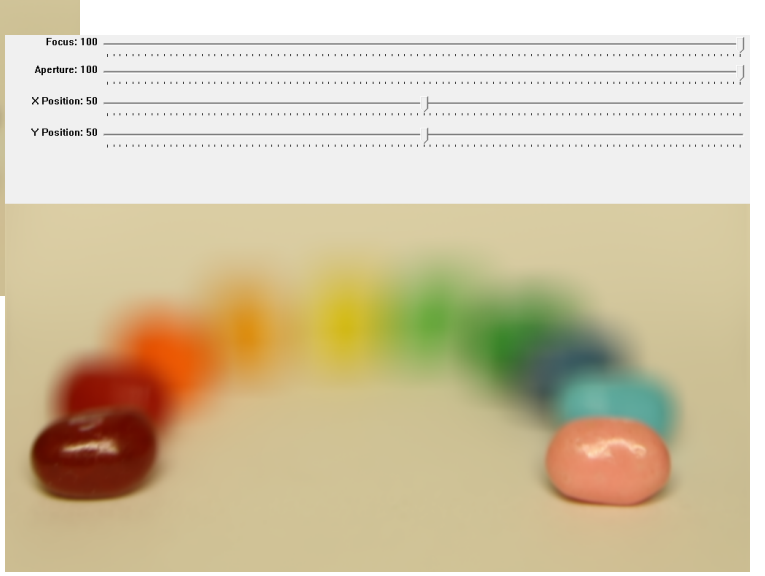
Refocused:


Focus: 0
Aperture: 30
X Position: 0
Y Position: 3


Focus: 100
Aperture: 79
X Position: 0
Y Position: 3


Focus: 28
Aperture: 47
X Position: 50
Y Position: 50


Focus: 47
Aperture: 50
X Position: 50
Y Position: 50


Focus: 100
Aperture: 100
X Position: 50
Y Position: 50