Final Report

Nickolas Arustamyan

December 2020

Abstract

In this experiment, we focused on convolutional processes and their actions on images. Our first calculations were done using predefined matrices in conjunction with regular generic images. We also considered the time in which it takes to conduct different convolution methods and then took a look at animiations and how to work with those.

Convolution by Kernel

Beginning with a typical colored image, we proceeded to remove the color and convert the image to grey scale. We then defined a kernel matrix specifically designed with a purpose in mind. Our first matrix, a, was pictured below:

$$a = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

This matrix specifically derived a stronger pixel grey scale value based on the values above it, and a weaker value from the pixels directly to the bottom left, bottom right, and bottom middle of it. This modified the rectangular outline pictured in the image, removing the vertical lines, and accentuating the top of the letters as opposed to the bottom of the letters/figures pictured. The next matrix, b, taken was the transpose:

$$b = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

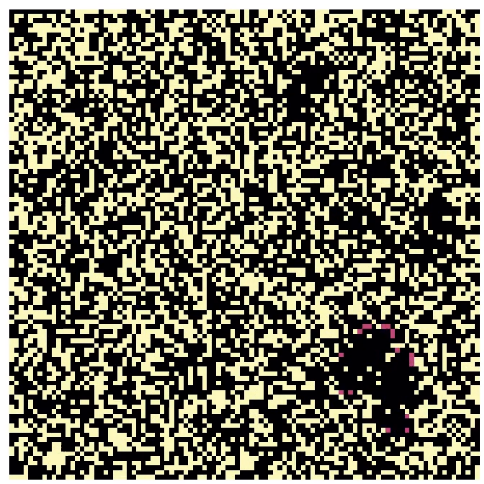
As opposed to the first matrix, we have that the horizontal features were generally removed, instead favoring the left hand side of given features of the image, and maintaining the vertical lines of the structure.
Our final matrix used for convolution was a randomized matrix ranging from 0 to 25 for each element. This convolution resulted in a more crude image that was simply a fuzzier version of our beginning image. Furthermore, we used randomness in convolution the image of the road, which could yield interesting results, since specific features are noticeable.

Timing

While having a good convolution method is important, the practical application of these methods requires speed and understanding the time it takes to convolute a figure is very important. In the lab, we observed the difference in time between different methods. The simple method with a loop took over 13 seconds for a fairly small image. The optimized method with a loop took just under a tenth of a second at $0.09975s$. The optimzed version without a for loop and processed on the gpu took just over a hundreth of a second at $0.0186s$. Rerunning the convolution with 960 filters instead of 96, we get a processing time of $0.009s$! This can help make an educated guess of how long it would take other projects and images.

Animations

The next major portion of the lab was the animation of iterations of automata. These animations included the Game of Life, Surface Tension model, Forest Fire model, Nonlinear Waves, Wireworld Wire and Oscillator, Fitzhugh-Nagumo Reaction Diffusion and Gray Scott Reaction Diffusion. In particular, I was drawn to the Gray Scott Reaction Diffusion and the numerous output animations that accompanied it, since the possibilities were intriguing. After experimenting with the various given varieties and images, I realized that when given the bacteria values, there was a possibility that the initial image could contain different amounts of 'bacteria', which vastly changed the final form of the image without altering the same processes that created it in the first place.



While this is obviously due to the noise and disturbance of the initial square, it is fascinating the beauty of the function itself. Furthermore, an equally interesting phenomenon is the forest fire simulation, which is interesting in its precise capturing of the essence of partial differential equations, since the probability of a forest fire increases based upon the increasing presence of forest. Therefore, upon every simulation, the probability for a unique and surprising scenario is high.

Conclusions

In conclusion, this lab examined in detail the convolution of images using an image and a defined image kernel in order to accentuate certain features imaged. Furthermore, the given animations and simulations utilized similar concepts in their methods of progression, utilizing methods modifying pre-existing values in order to create more processed images or features.