One of the applications of programmable logic circuits is implementing image processing algorithms. In this project, we will familiarize ourselves with the basics of edge detection in image processing. An FPGA will be used as a high-speed processor to receive a grayscale image, detect the edges within the image, and output the results. A grayscale image is defined as a two-dimensional signal with two components (x, y) representing the continuous spatial axes. The value of the image at each point represents its brightness, denoted as f(x, y), which is defined over the continuous spatial components x and y.

This algorithm uses two kernels to perform derivative operations in two directions. To understand this better, imagine you have an array of values and want to calculate the derivative at each point. The derivative is calculated by subtracting the value of the previous cell from the value of the next cell. The kernel for this operation is as follows:

$$[+1 \quad 0 \quad -1]$$

We convolve this kernel with the array data to perform the derivative operation. The next step is to reduce noise, which can be done by applying Gaussian averaging at each point. The kernel for this operation is as follows:

$$\begin{bmatrix} +1 \\ +2 \\ +1 \end{bmatrix}$$

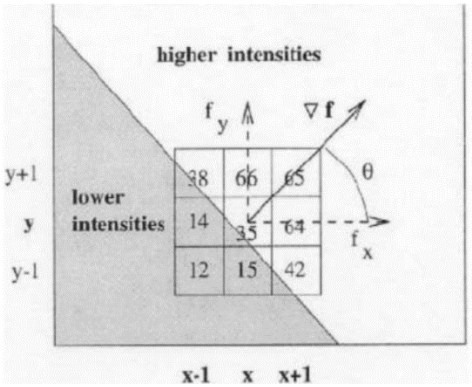
Now, by multiplying this average kernel by the derivative kernel, the derivative kernel is made in the x direction:

$$\begin{bmatrix} +1 \\ +2 \\ +1 \end{bmatrix} \times [+1 \quad 0 \quad -1] = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

And the kernel in the y direction will be as follows:

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

Now, the main gradient can be calculated by summing the two gradients. Now, in order to complete the averaging, the obtained size must be divided by 4. For example, suppose the following numbers correspond to a part of a photo.



$$Gx = (38 - 65) + 2 * (14 - 64) + (12 - 45)$$

$$Gy = (12 - 38) + 2 * (15 - 66) + (42 - 65)$$

Given that the resulting matrix values may become negative or exceed 255, we need to normalize the range of numbers to fit within 0 to 255. Based on the resulting matrix, we threshold the values using a number T, provided as input to the FPGA. For instance, if the resulting value is greater than T, the pixel value is set to 1; otherwise, the pixel value is set to 0. In this project, parallel implementation (using 4, 9, or 16 cores) is achieved through HDL implementation, and we use HLS tools to convert the RGB image to a grayscale image.