EE 562 Project
Fall 2024

Sobel edge detection algorithm calculates derivatives of an NxN input image with 3x3 K_x and 3x3 K_y kernels as shown below.

$$K_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} \text{ and } K_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$Sobel_X(i,j) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} K_x(k+1, l+1) * I(i+k, j+l) \qquad Sobel_Y(i,j) = \sum_{k=-1}^{1} \sum_{l=-1}^{1} K_y(k+1, l+1) * I(i+k, j+l)$$

for i = 0:N-1, j = 0:N-1

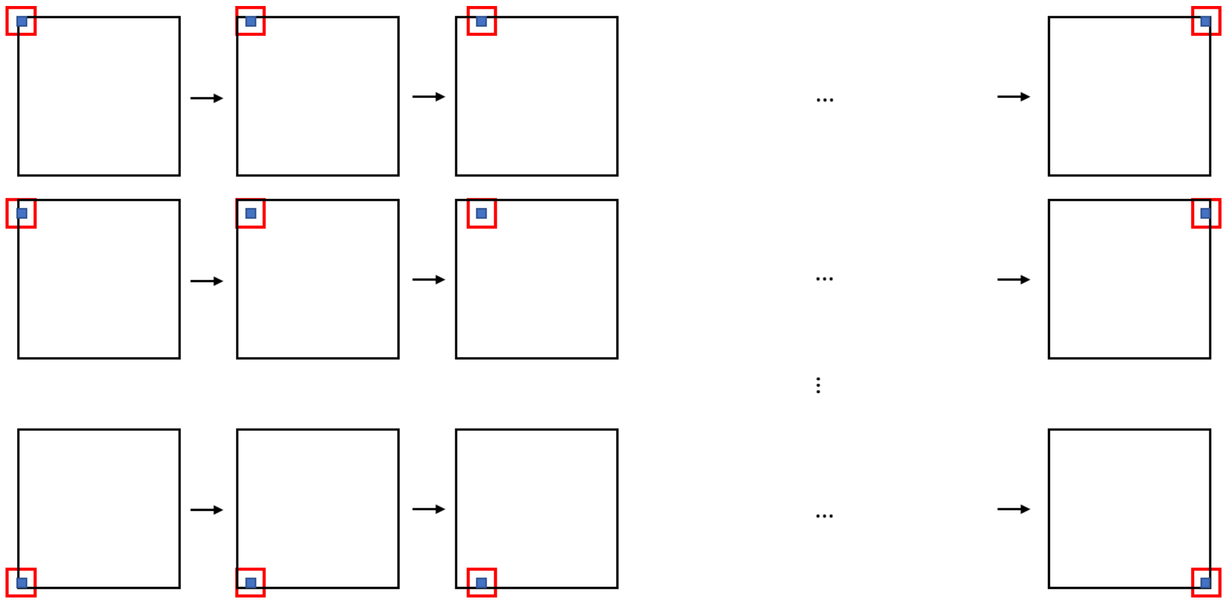
Then, the magnitude is calculated as shown below.

$$Sobel_{OUT}(i,j) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |\, Sobel_X(i,j)\, | + |\, Sobel_Y(i,j)\, |$$

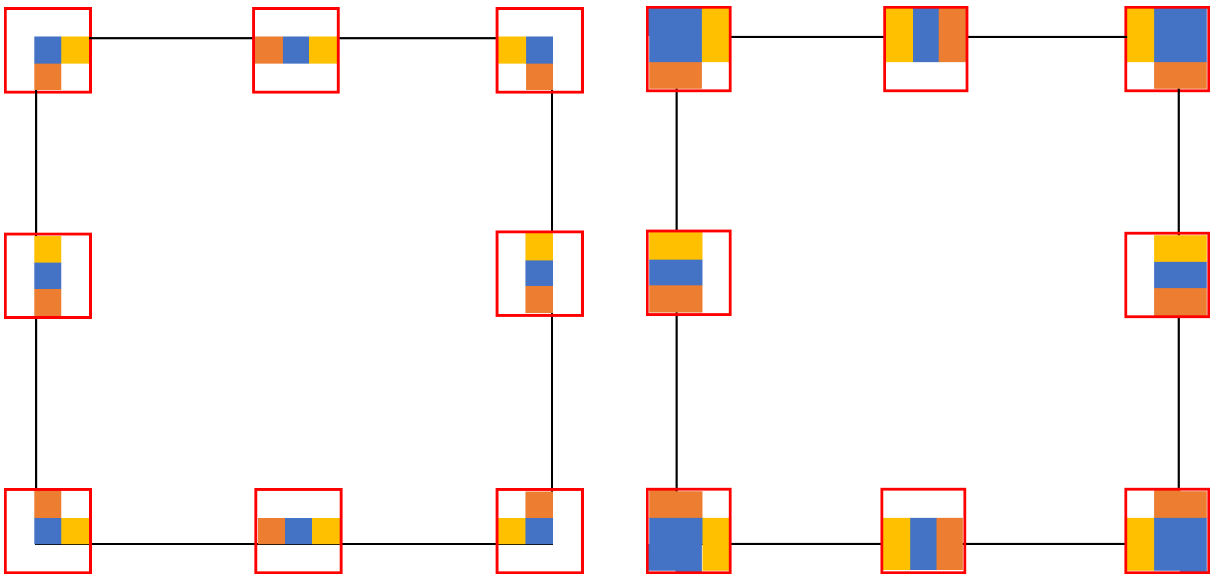
Finally, thresholding is applied in order to eliminate the noise on the edges.

```
Thresholding
if Sobel_OUT(i,j) ≥ 150
    Output_Image (i,j) = 0
else
    Output_Image (i,j) = 255
```

Sobel edge detection algorithm is applied to 3x3 windows for each pixel in the input image as shown below.

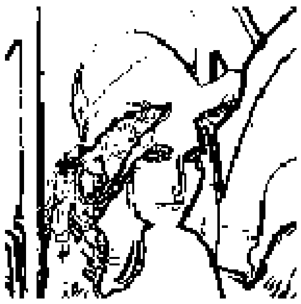


Since the pixels in borders of the input image do not have neighboring pixels, the closest pixels are duplicated for applying Sobel edge detection algorithm to these pixels as shown below.



The following figure shows an example grayscale (each pixel is 8 bits) input image and the edge detected output image.

Input Image


Output Image

In this project, you should design a digital hardware implementing Sobel edge detection algorithm, and implement it using Verilog RTL. You should also write a Verilog Testbench and verify your Verilog RTL code by simulating them using Xilinx Vivado.

You should synthesize and implement your Verilog RTL code to a Xilinx FPGA using Xilinx Vivado. There should not be any errors or latch warnings.

You should then write a short report explaining the status of your Verilog RTL code, testbench, verification, synthesis, and implementation. Submit all your Verilog files and report as a zip file to LMS.