REAL-TIME DIGITAL SYSTEMS DESIGN AND VERIFICATION WITH FPGAS
ECE 387 – LECTURE 8

PROF. DAVID ZARETSKY

DAVID.ZARETSKY@NORTHWESTERN.EDU

AGENDA

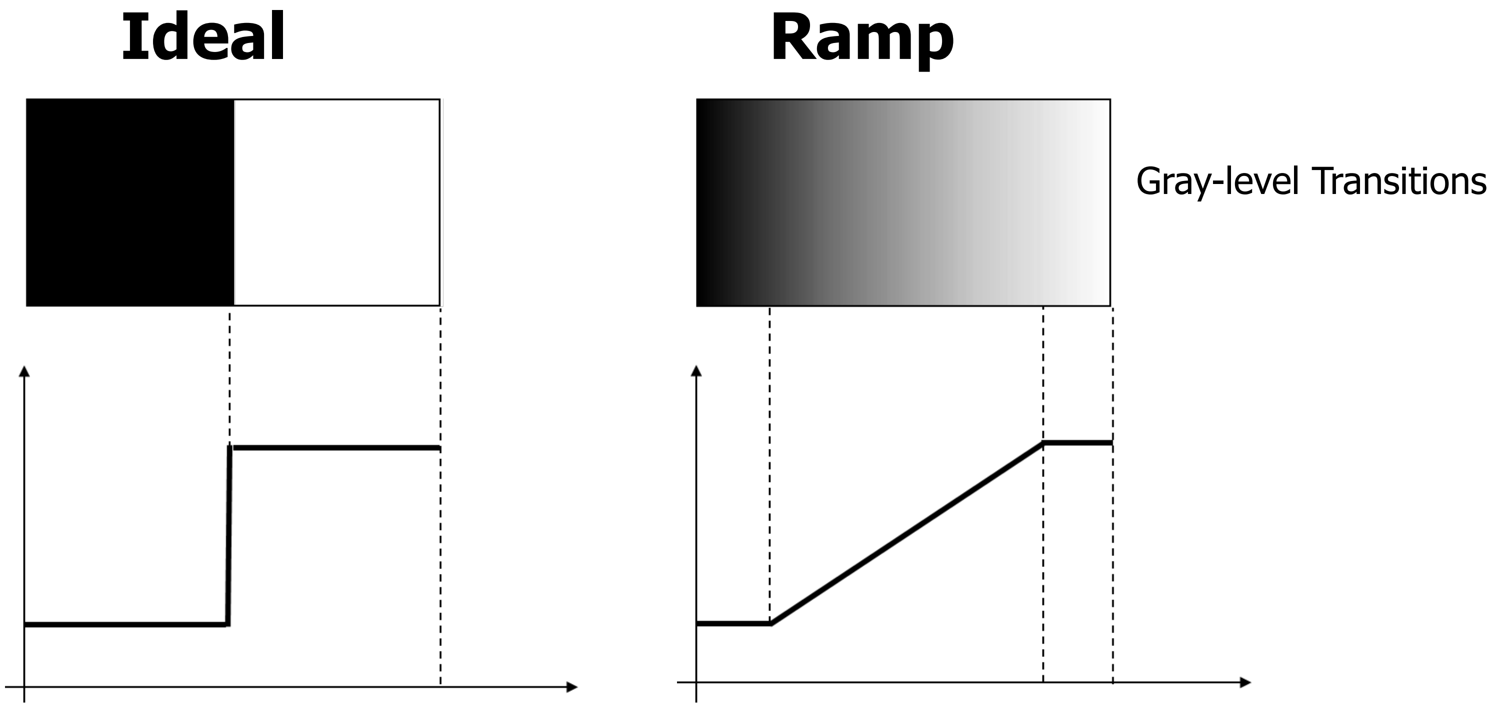
- Computer Vision
- HW4: Edge Detection

COMPUTER VISION

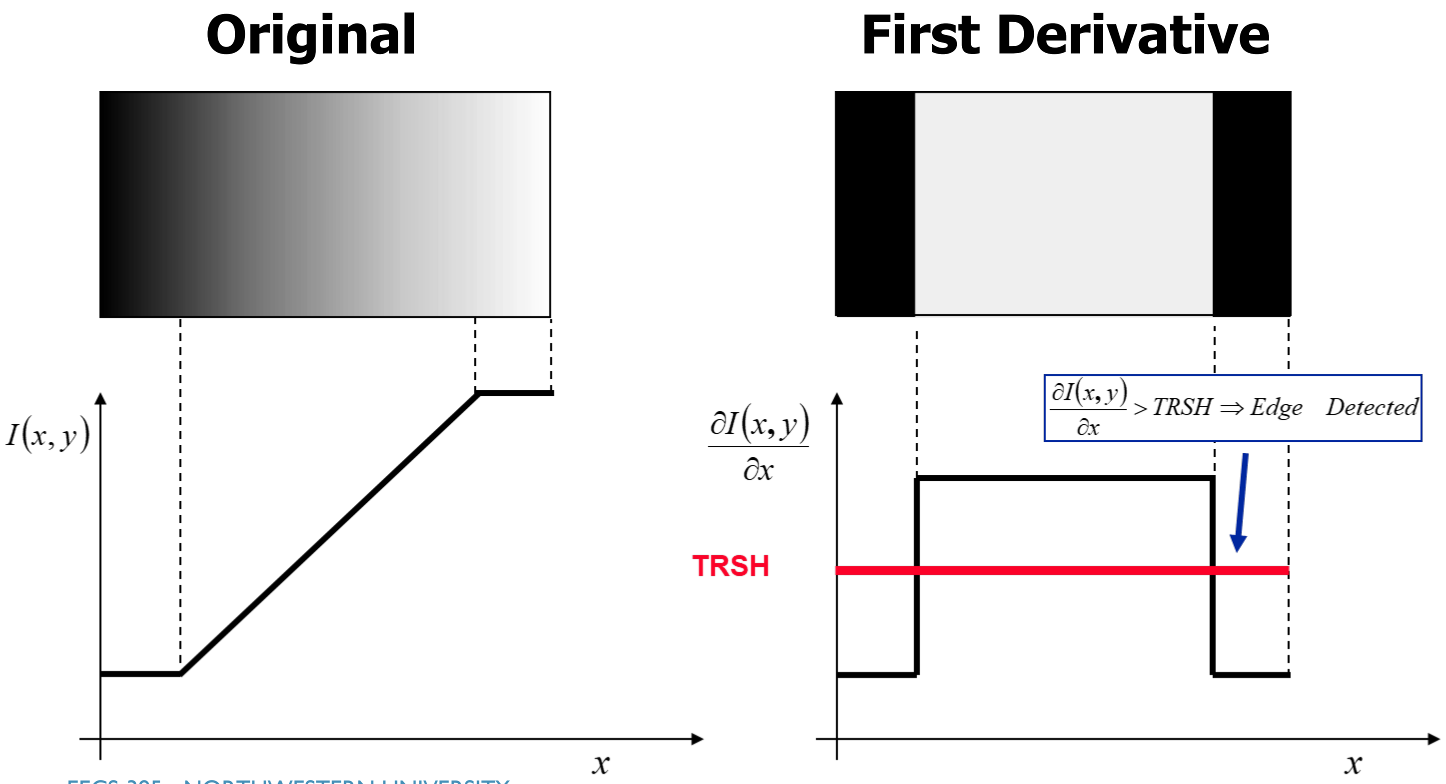
- Introduction to acceleration techniques for computer vision
- Implement a variation of the Canny edge detector, which is a widely-used edge-detection scheme in computer vision applications
- Applications:
  - Autonomous Vehicles
  - Machine Learning
  - Image Analysis
  - Object Detection
  - Motion Detection
  - Tracking
  - Security
  - Maps / Routing

EDGE DETECTION

- What is Edge Detection?
  - The ability to measure gray-level transitions in a meaningful way.

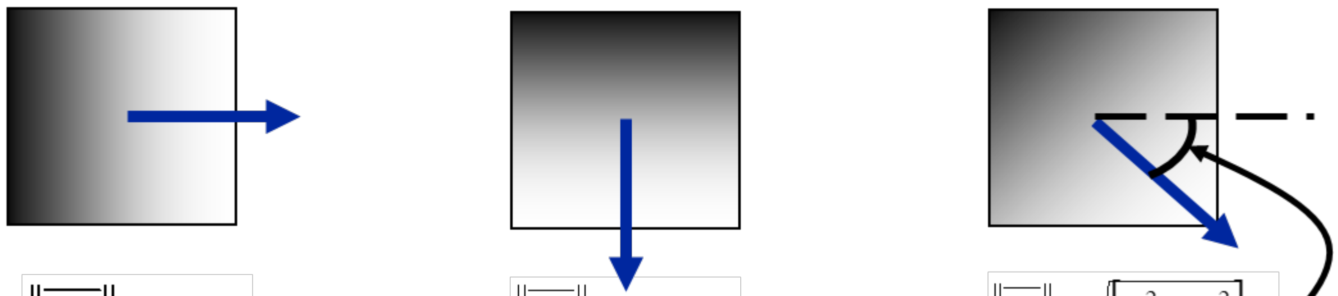


DETECTING THE EDGE



THE MEANING OF THE GRADIENT

- The direction of the strongest variation in intensity

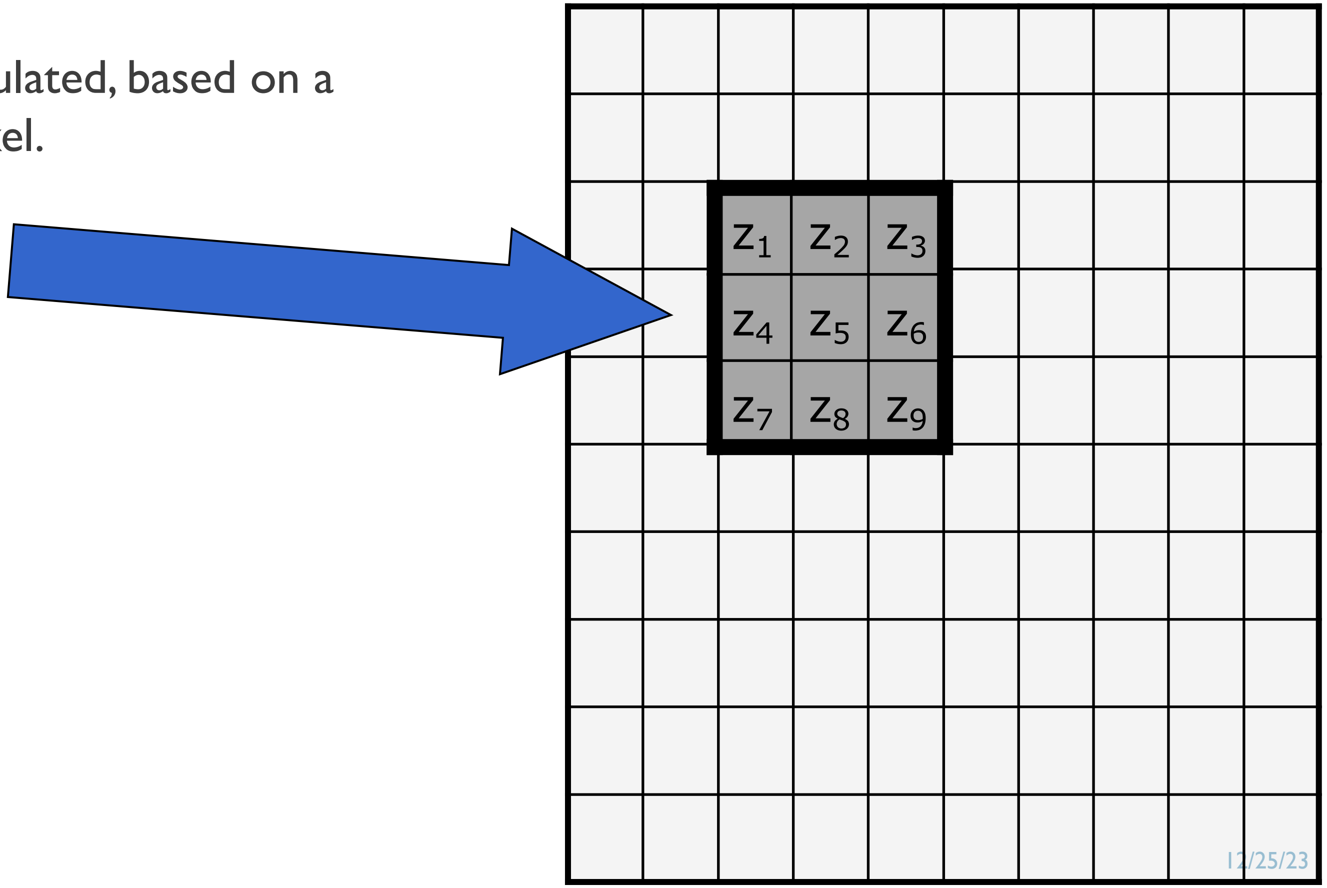


	Vertical	Horizontal	Generic
Edge Strength:	$\left\|\nabla I\right\| = G_x$	$\left\|\nabla I\right\| = G_y$	$\left\|\nabla I\right\| = \sqrt{\left[G_x^2 + G_y^2\right]}$
Edge Direction:	$\theta(x, y) = 0$	$\theta(x, y) = -\frac{\pi}{2}$	$\theta(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$

The direction of the edge at location ***(x,y)*** is perpendicular to the gradient vector at that point

CALCULATING THE GRADIENT

- For each pixel the gradient is calculated, based on a 3x3 neighborhood around this pixel.



THE SOBEL EDGE DETECTOR

-1	-2	-1
0	0	0
1	2	1

HORIZONTAL

$$G_x \approx (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

-1	0	1
-2	0	2
-1	0	1

VERTICAL

$$G_y \approx (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

THE CANNY METHOD

- The Canny edge-detection algorithm involves five stages which are applied to the input image in succession
- Key Difference: The image is convolved with a Gaussian filter before gradient evaluation

$$h(r) = -e^{-\frac{r^2}{2\sigma^2}}$$

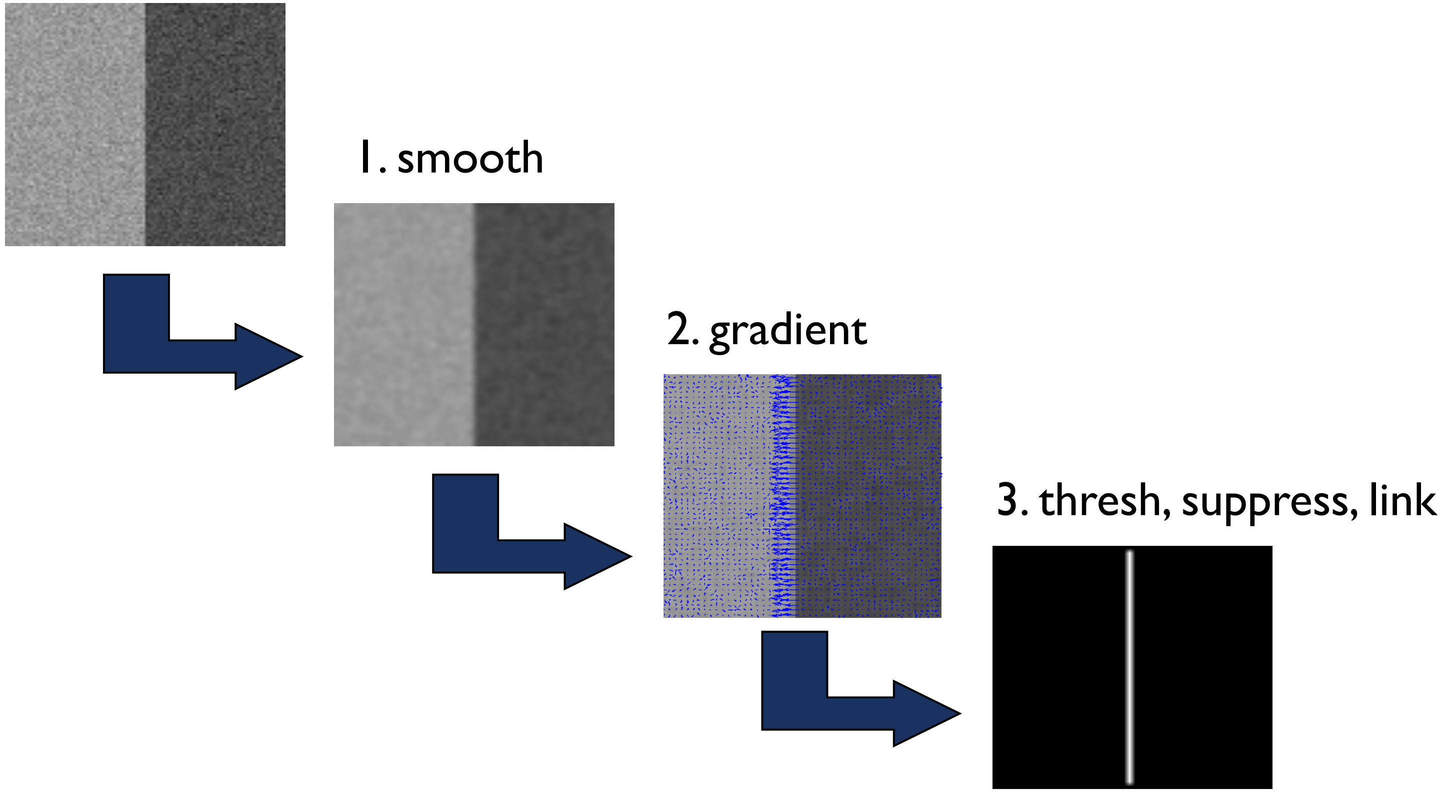
$$r = \sqrt{x^2 + y^2}$$

THE EDGE DETECTION ALGORITHM

- The gradient is calculated for each pixel in the picture.
- If the absolute value exceeds a threshold, the pixel belongs to an edge.
- The Canny method uses two thresholds, and enables the detection of two edge types: strong and weak edge.
  - If a pixel's magnitude in the gradient image, exceeds the high threshold, then the pixel corresponds to a strong edge.
  - Any pixel connected to a strong edge and having a magnitude greater than the low threshold corresponds to a weak edge.

EXAMPLE CANNY EDGE DETECTION

- Images are converted to grayscale



STAGE 1: GRAYSCALE CONVERSION

- This stage converts the input 24-bit bitmap color image (8 bits each for red, green, and blue) into an 8-bit grayscale image.
- The grayscale value at each pixel is calculated as the average of the three 8-bit color values of the original image.

STAGE 2: GAUSSIAN SMOOTHING

- Gaussian filter is used to smooth out the image, by modifying noisy pixels (pixels that are unlike their neighboring pixels)
- The effect of this operation is that each pixel gets assigned the weighted average value of the 5 x 5 grid of pixels surrounding each pixel.

$$B = 1/159 \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * A$$

STAGE 3: SOBEL OPERATOR

- Sobel calculates each pixel from the overall intensity gradient
- Gradient is calculated in horizontal (Cx) and vertical (Cy) directions across the pixel, using the matrices to the right.
- The magnitudes of the two gradients are added to calculate the overall gradient intensity value for each pixel,
- In the resulting image, the edges of the original image are highlighted as brighter pixels. Non-edges, which are areas with low intensity gradients, appear as darker pixels.

$$C_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * B$$

$$C_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * B$$

$$C = 0.5|C_x| + 0.5|C_y|$$

STAGE 4: NON-MAXIMUM SUPPRESSION

- This stage aims to thin the thick and/or blurry edges that may have resulted from the sobel operator stage.
- Thick edges are problematic as many applications of edge detection benefit from the edges being as thin as possible
- Non-maximum suppression stage thins the edges by removing the weaker (non-maximum) pixels of each edge, and keeping only the maxima.

0	41	134	45	0
0	43	135	46	0
0	35	136	41	0
0	41	132	35	0
0	44	131	41	0

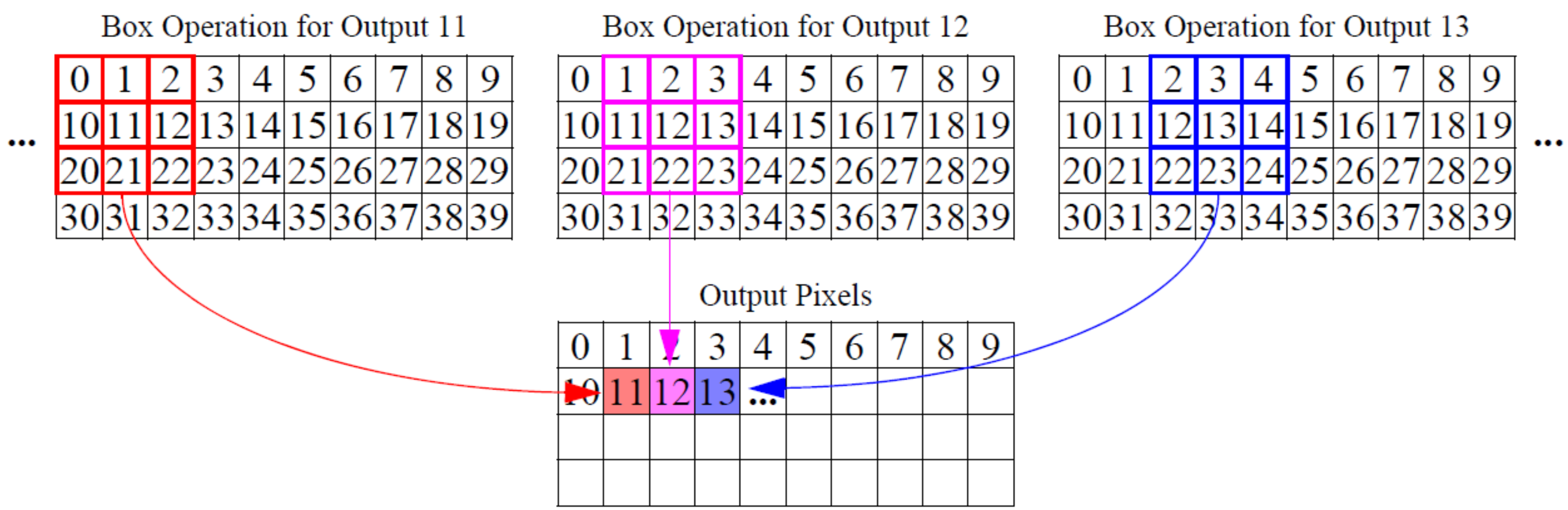
0	0	134	0	0
0	0	135	0	0
0	0	136	0	0
0	0	132	0	0
0	0	131	0	0

STAGE 5: HYSTERESIS

- The goal of the hysteresis stage is to remove pixels that do not belong to an edge and weak edges altogether.
- Hysteresis preserves each pixel if:
  - the pixel exceeds a high threshold, or
  - the pixel exceeds a low threshold value and there exists at least one adjacent pixel (horizontally, vertically, or diagonally) that exceeds the high threshold.
- If neither criteria are met, the pixel is removed by turning it black.

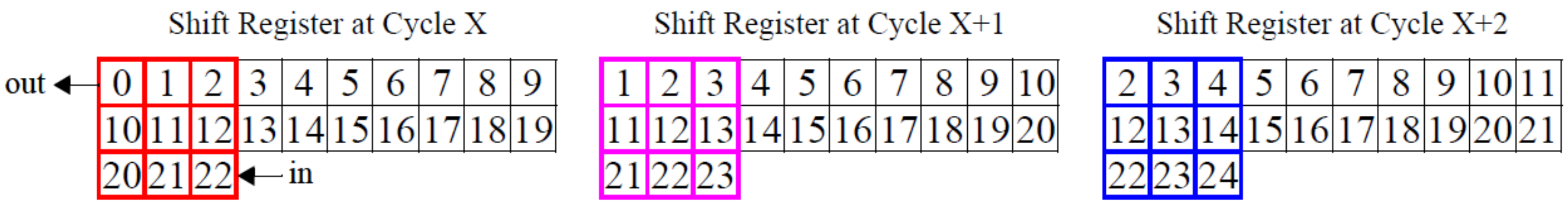
SOBEL ALGORITHM

- Note that the Gaussian smoothing, Sobel operator, non-max suppression, and hysteresis stages of the detector are all types of box operations, as they work on a box (or frame) of pixels to determine each output pixel.



BOX OPERATION

- To take advantage of the overlap in successive frames of box operations, we will use the FPGA's local memory resources to construct a 3x3 shift register design
- Once a sufficient number of pixels have been loaded, the shift register provides a new 3x3 frame for the box operation at every cycle
- Simply shift in a new pixel and shift out the oldest pixel which is no longer be used.

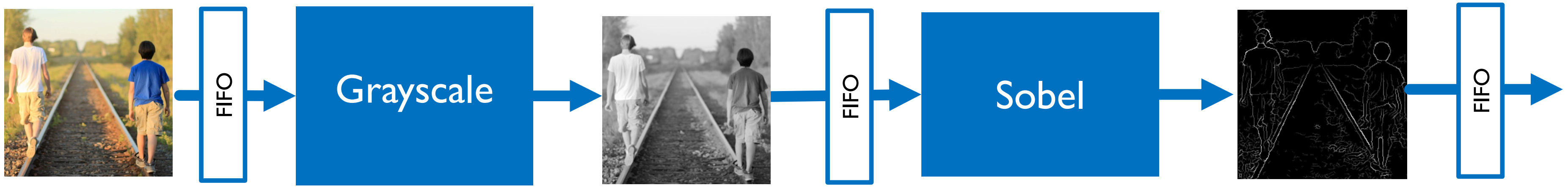


QUESTIONS FOR THOUGHT

- How many shift registers will you require for your canny edge detector?
- How large should each of these shift registers be, given that your circuit works on images that are 720 pixels wide?
- How many pixels must be loaded into each shift register before the corresponding box operation can start?
- Using such a shift register design requires you to zero pad the input image before shifting in its pixels, to properly operate on the boundaries of the image.
- Why is zero padding necessary, and what is the necessary padding size for a given box size?

PROGRAMMING ASSIGNMENT #3: SOBEL EDGE DETECTION

- Streaming Sobel
  - Implement a variation of the Canny edge detector
  - Only implement grayscale & sobel operations.
  - Implement the UVM model for verification
  - Read BMP image in from FIFO
  - Determine how to stream data for the sobel convolutional filter operation
- Simulate & Synthesize
  - Use given C program to generate image data
  - Simulate to get cycle count and verify correctness
  - Synthesize to get resource utilization
  - Compare results with other implementations



NEXT…

- HW #4: Edge Detection