Assignment report

Introduction: Roberts: The Roberts operator detects edges by **approximating the image gradient** through diagonal pixel differences, employing two **2x2 kernels** $G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ and $G_y = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$ for the **first-order derivative**. These kernels are convolved with the image to form $G_x \otimes \mathrm{I}$ and $G_y \otimes \mathrm{I}$,representing the gradient components. The gradient magnitude is then calculated as $|G| = \sqrt{G_x^2 + G_y^2}$.

Sobel: The Sobel operator detects edges by computing **first-order spatial derivatives in both the x and y directions**, using two **3x3 kernels** $G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ and $G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$ After convolving these kernels with the image to obtain **horizontal** and **vertical** edge components, the gradient magnitude is calculated as $|G| = \sqrt{G_x^2 + G_y^2}$.

First-Order Gaussian: The First-Order Gaussian filter detects edges by employing the **first derivative of the Gaussian function** to calculate the image gradient. Gradient components G_x and G_y are extracted through convolution, and the overall edge strength is determined by the gradient magnitude$|G| = \sqrt{G_x^2 + G_y^2}$. $g'_\sigma(x) = \frac{\partial g_\sigma(x)}{\partial x} = -\frac{x}{\sigma^2}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{x^2}{2\sigma^2}\right)$

Laplacian: The Laplacian edge detection uses a single matrix $G_x = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ to **approximate image gradient changes**. This matrix, when convolved with the image, highlights areas of **intensity change**, effectively detecting edges by identifying points where values shift from **negative to positive**. $L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2}$

Laplacian of Gaussian (LoG): The LoG operator applies convolution of the **Gaussian kernel** with the image matrix. Subsequently, it convolves this outcome with the **Laplacian kernel**, employing the Laplacian edge detection method. This dual convolution process serves to both **smooth** the image and diminish noise while effectively detecting edges. $LoG(x, y) = -\frac{1}{\pi\sigma^4}[1 - \frac{x^2+y^2}{2\sigma^2}]e^{-\frac{x^2+y^2}{2\sigma^2}}$

Task1: Aim: implement and use Gaussian Laplace (LoG) filters for edge detection of Shakey.gif.

Method: I implemented a LoG filter with a kernel size of 7(mean ± 3 * sigma) and a sigma value of 1.0 for edge detection in images. While this approach was **somewhat effective** in reducing noise, it still introduced **some noise** and potentially **missed certain edges**. To address these issues, I created the ***create_gaussian_kernel0*** function to generate a **zero-order Gaussian kernel** of size 7*7 with a sigma value of 3. This kernel was used for **Gaussian smoothing** before applying the LoG filter, significantly **reducing noise** and smoothing the image. Subsequently, I employed the **Otsu** method for the automatic determination of the optimal threshold for the image. This method works by calculating the **grayscale histogram of the image** and identifying the **threshold** that **maximizes the variance between the foreground and background**, effectively **enhancing the contrast** between them. For the combined Gaussian smoothing and LoG method, the Otsu threshold was set at 0.56, compared to 0.59 when only using the LoG filter.

Results and Conclusion: The integration of zero-order Gaussian smoothing with the LoG filter notably **improves edge detection** in images. The preliminary Gaussian smoothing **effectively minimizes noise**, enhancing the LoG filter's input and resulting in more precise edge detection. While LoG inherently reduces some noise, its effectiveness is **limited in high-noise contexts**, making **additional Gaussian smoothing essential**. This combined approach proves especially effective in high-noise scenarios, underscoring the value of appropriate image preprocessing for **optimal edge detection**.

Figure 1. Only LoG

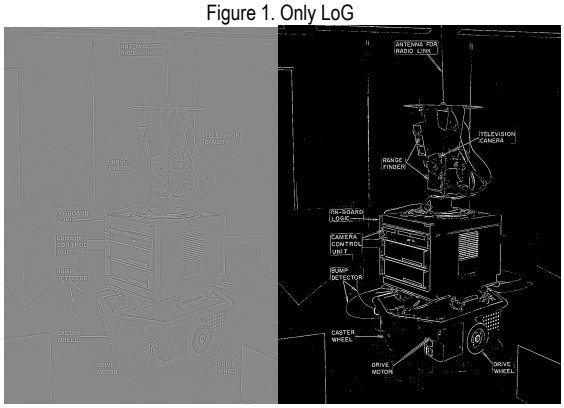
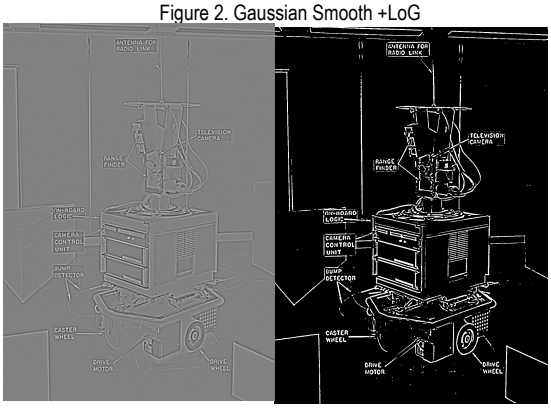


Figure 2. Gaussian Smooth +LoG



Task2:

Aim: Apply the five different **edge detection techniques mentioned in the introduction** to three sets of provided HEp-2 cell images and analyze and compare their effectiveness in identifying edges under various conditions.

Method: I implemented five distinct edge detection techniques: Roberts, Sobel, first-order Gaussian, Laplacian, and Laplacian of Gaussian (LoG). Initially, when using these techniques alone for edge detection, I found their noise reduction

capabilities limited, **with noticeable noise in both the edges and internal areas of the cells**. Therefore, I decided to apply **Gaussian smoothing** to the images before proceeding with further edge detection. This smoothing process significantly improved the performance of each edge detection technique in terms of edge brightness and noise reduction, particularly the Roberts method, which showed the most notable improvement (Figure 1). To further enhance edge highlighting, I then used the Otsu method on the original images to determine the **optimal thresholds**, which were **0.0906** for 9343 AM **(Medium intensity)**, **0.1341** for 10905 JL **(High intensity)**, and **0.0506** for 43590 AM **(Low intensity)**. Subsequently, I combined Gaussian smoothing with edge detection algorithms on these thresholded images and conducted **normalization** and **image inversion**. The results indicated that the Roberts, Sobel, and first-order Gaussian methods performed relatively well. For the Laplacian and LoG methods, as they are very sensitive in edge detection and to noise, I applied these techniques to the original image processed with Otsu's method and combined them with zero-crossing techniques (threshold = 0.75 times the absolute value of the image) (Figure 2). The reason for using this approach is because the output of LoG and Laplacian is the s**econd-order derivative value at each pixel point in the image**, typically used to **highlight areas of rapid brightness change in the image**. The zero-crossing technique helps to **locate the points in the image** where pixel values change from **positive to negative** (edges).

Results and Conclusion: In my exploration of edge detection in HEp-2 cell images, Gaussian smoothing significantly **improved** the results of all methods. For the Roberts method, the improvement is modest, but the edge sharpness is slightly less than for the other methods. The Sobel and first-order Gaussian methods **balance edge highlighting and noise suppression**. In the Laplacian and LoG methods, the **gray-scale changes** reflect the edge details, but these changes are enhanced by the introduction of Otsu thresholding and zero-crossing techniques. Since Otsu can easily detect edges in binarized images, these additions **significantly improve their performance**, especially on such **high-contrast cell images**. Specifically, for different images, based on **visual observation** and the **results of edge detection**, 9343 AM exhibited **medium** intensity variations, 10905 JL showed **high** intensity variations, and 43590 AM presented **low** intensity variations. This suggests that different images might require distinct methods of optimization in edge detection.

Figure 1. Edge Detection Only

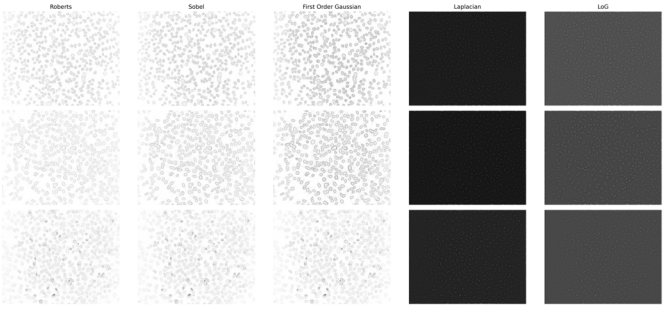
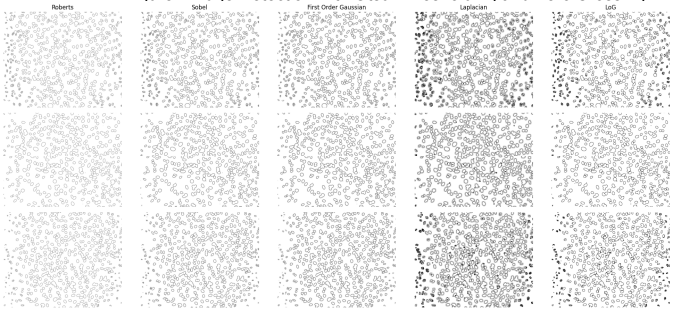


Figure2. Edge Detection with Otsu Thresholding and Zero Crossing



Task3: Aim: Apply a new edge detection algorithm to three image sets and evaluate whether it is better or worse compared to the techniques used in Task 2.

Method:I chose the **Canny edge detection** algorithm for its renowned ability to provide a clear and **robust** method for edge detection in images. Starting with Gaussian Filter Application, I utilized a Gaussian kernel defined by a sigma value of 1 and a kernel_size of 9. This setup was crucial in **smoothing** the image adequately, **reducing noise** without overly blurring essential edge details. During the Gradient Calculation phase, I computed the **gradient's magnitude and direction from the convolved images,** ensuring that the direction of the edges was consistent and within a **0 to 180-degree range**. This step was fundamental in establishing the orientation of potential edges in the image.In the **Non-Maximum Suppression** process, I compared the **magnitude of each pixel** with its **immediate neighbors along the gradient direction**. Pixels not having the highest magnitude in their local area were set to white (255), effectively thinning the edges. This approach was key in enhancing the clarity and distinction of true edges.The final step involved Thresholding and **Edge Tracking by Hysteresis**. Here, I selected **high and low thresholds** at 0.90 and 0.50, respectively. These thresholds were critical in differentiating between strong and weak edge pixels. Strong edge pixels were preserved, while weaker ones were suppressed **unless** they were **connected to strong edges**. Finally, the **inverted** image is output.

Results and Conclusion: The Canny edge detection algorithm demonstrated its strengths compared to the methods used

in Task 2, the Canny algorithm excelled in **effectively reducing noise** and **accurately detecting a wide range of edges**, a significant advantage in complex image analysis. This precision, combined with its ability to **minimize false edges**, marked a **key improvement over other methods**. While more computationally intensive than simpler techniques like Roberts or Sobel, and potentially more complex than Laplacian methods, Canny's meticulous multi-stage process typically **resulted in cleaner**, **more reliable edge detection**. Furthermore, the incorporation of **Otsu's thresholding** on the original images before applying Canny significantly improved the **clarity of edge detection**, producing very clear and distinct edge definitions. This makes Canny particularly valuable in scenarios where precision and clarity are paramount, despite the potential trade-off in processing time and complexity.The Canny algorithm is highly suitable for processing high-contrast green cellular images, **effectively reducing noise** and **ensuring precise and continuous edge detection** through **non-maximum suppression**. Its **high sensitivity** and **adaptability** make it an ideal tool for edge analysis in such images.

Figure 1. Only Canny

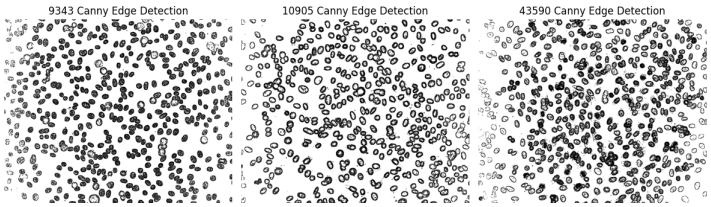
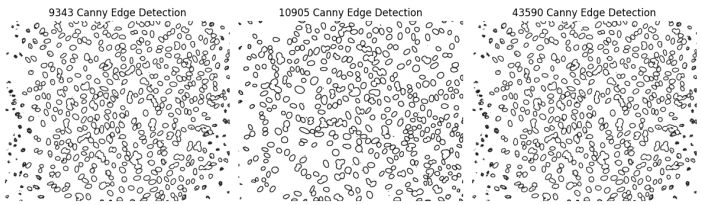


Figure 2. Canny with Otsu Thresholding



Task4: Aim: Evaluate an edge detector's accuracy using ROC analysis by comparing it with a 'Ground Truth' image to determine its sensitivity and specificity.

Method: To evaluate the edge detection accuracy of the algorithms in Tasks 2 and 3 on real images. I improved on the use of **Otsu's** method in Canny method to achieve optimal binarization. At the same time, I integrated **zero-crossing** with **Otsu thresholding**, along with Laplacian and LoG methods, to precisely define edges. The other operators are binarized using the **threshold at which the highest AUC value** is found using the **probabilistic** input, and all are **normalized** and **inverted.** To evaluate the effectiveness of these techniques, I focused on calculating **sensitivity** and **specificity** through **ROC analysis**, evaluating the **proportion of true positive rate (actual edges correctly identified)** and **negative rate (actual non-edges correctly identified)**, respectively, and calculating **F1 score** to **balance** evaluation precision and recall.

Results and Conclusion: In analyzing edge detection in high-contrast cellular images, I observed that the Sobel operator performed **robustly** across multiple tasks, slightly outperforming first-order Gaussian and Roberts operators. This may be because Sobel is less sensitive to changes in **isolated high-intensity points**, which helps to **reduce noise interference** when detecting continuous edges, thereby **improving the accuracy of edge existence judgment** in some cases. Notably, the sensitivity of the Roberts operator, especially in noisy images, significantly improved when combined with Gaussian smoothing, evidenced by about a **3% increase** in the 43590 AM. This is because Roberts detects edges by calculating the diagonal difference of the image, which is quite sensitive to noise, and Gaussian smoothing is very effective in suppressing noise. Additionally, the Laplacian, LoG, and Canny operators, integrated with Otsu thresholding and zero-crossing techniques, demonstrated the highest AUC values in all test images. Particularly, the Laplacian operator, with an average Sensitivity of **97%**, Specificity of **95%**, and an F1 score of **98%**, excelled due to its **combination of Ostu thresholding and zero-crossing**. Ostu's method helps differentiate between edges and non-edges by automatically determining the **optimal binarization threshold**, while the zero-crossing(positive to negative) technique identifies edges in the **second-order derivative**(brightness change) of the Laplacian and LoG operator. Overall, these edge detection algorithms performed **best** and most evenly on the 10905 JL image (high intensity), also showed **good** results on the 9343 AM image (medium intensity), but were **weakest** overall on the 43590 AM image (low intensity). However, the application of Otsu and zero-crossing can **significantly improve the results**. In general, for these three high-contrast cellular images, the Laplacian, LoG, and Canny operators, incorporating Otsu thresholding and zero-crossing, proved to be the best choices, thanks to their exceptional **accuracy** and **reliability.**

Figure 1: ROC 9343 AM

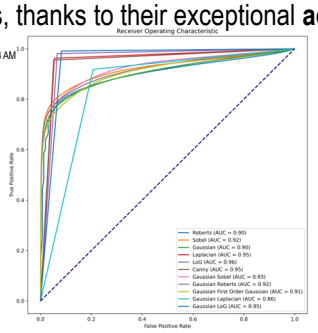


Figure 2: ROC 10905 JL

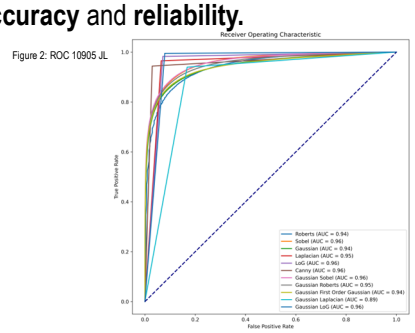


Figure 3: ROC 43590 AM