**Classwork 5**

**Assignment 1:**

**Histogram Equalization**

**Implement** [**Histogram Equalization**](https://en.wikipedia.org/wiki/Histogram_equalization)**.**

**Define your procedure in the "myHistEq.py" script and all the logic functions in "myHistEqLogic.py" script.**

**The Process of Histogram Equalization algorithm steps:**

1. **Calculate the PDF of tones within the image.**
2. **Calculate the corresponding CDF .**
3. **Use the CDF as transformation function to remap the image tones.**

**Probability Density Function (PDF) calculation:**

**Step one: Calculate Image Histogram of tones.**

**Step two: Normalize the Histogram to turn it in to PDF by dividing each value by the sum of all values in the Histogram.**

**Cumulative Probability Function (CDF) calculation:**

**Remap tones of the image to Equalize the Histogram:**

**Using Matplotlib, compare the original (bad quality image) to the Equalized one.**

**Assignment 2:**

**Canny Edge Detector**

**Implement** [**Canny edge detector**](https://en.wikipedia.org/wiki/Canny_edge_detector)**.**

**Define your procedure in the "myCanny.py" script and all the logic functions in "myCannyLogic.py" script.**

**The Process of Canny edge detection algorithm steps:**

1. **Apply Gaussian filter to smooth the image in order to reduce the noise.**
2. **Find the intensity gradients (Sobel) of the image and its phase matrix.**
3. **Apply non-maximum suppression to get rid of spurious response to edge detection.**
4. **Apply double threshold to determine potential edges.**
5. **Track edge by Hysteresis: suppress all the weak edges which are not the continuation of a strong edge.**

**Gaussian Filter Mask:**

**Gradient and Phase:**

**Non-Maximum Suppression:**

**Round Phase to the nearest point from the list :**

**Since the and radian are part of the same line, replace the values of with .**

* **if the rounded gradient angle is (i.e. the edge is in the north–south direction) the point will be considered to be on the edge if its gradient magnitude is greater than the magnitudes at pixels in the east and west directions.**
* **if the rounded gradient angle is (i.e. the edge is in the east–west direction) the point will be considered to be on the edge if its gradient magnitude is greater than the magnitudes at pixels in the north and south directions.**
* **if the rounded gradient angle is (i.e. the edge is in the northeast–southwest direction) the point will be considered to be on the edge if its gradient magnitude is greater than the magnitudes at pixels in the north west and south east directions.**
* **if the rounded gradient angle is (i.e. the edge is in the north west–south east direction) the point will be considered to be on the edge if its gradient magnitude is greater than the magnitudes at pixels in the north east and south west directions.**

**Double Threshold:**

**Define High-Threshold and the Low-Threshold.**

**NOTE: Threshold is applied after normalizing the Intensity Gradient.**

**Remove any edges below Low-Threshold.**

**Define edges as weak if they are between Low-Threshold and High-Threshold.**

**Define edges as strong if they are above High-Threshold**

**Hysteresis edge tracing:**

**Remove all weak edges which are not connected to any of the strong ones.**

**Using Matplotlib, compare the Sobel and Laplacian edge detectors to the Canny edge detector.**