# Assignment #1

## Thursday, 13th of November, 2025

---

The main aim of this assignment is to perform face detection by using the integral image. This is to be done by detecting eye area in faces through convolving a kernel that is designed to detect that area. The assignment is structured as follows:

1. Calculate the integral image and calculate the local sum
2. Detect the eye area

**Integral Image and Local Sum Calculation**

In this part, you are asked to implement two functions as follows:

1. **CalculateIntegral**
   - Input: 2D array representing the image (feel free to use a predefined function to transform an image into an array).
   - Output: 2D array representing the integral image.
   - Description: Implements the integral image technique as discussed in class. Feel free to implement it over two steps ($s$ & $ii$), or in one step, returning $ii$ as a result.
2. **CalculateLocalSum**
   - Input: An integral image, and two pairs of coordinates ($p_0 = (x_0,_0)$, $p_1 = (x_1,y_1)$).
   - Output: The local sum for the rectangular area defined by the pair of points (as $p_0$ being the upper left corner, and $p_1$ being the lower right corner of this rectangular area).
   - Description: Implement the local sum calculation using the integral image as discussed in class. This function **should not** contain any loops. Hint: test this function against normal local sum calculation to validate your implementation.

# Assignment #1

## Thursday, 13th of November, 2025

---

**Detect Eye Area**

In this part, you are asked to implement two functions as follows:

1. **DetectEye**
    - Input: the integral image and the kernel width.
    - Output: Coordinates (i, j) which represent the position of the maximum score achieved after convoluting the kernel.
    - Description: This method is responsible for calculating the local sums for the kernel provided in order to detect the eyes in the face image, where it gives the maximum score in the area where the eyes are present through convolving that kernel. **To convolve the kernel, the width of the kernel is provided as an input parameter and the height of the kernel is calculate based on the width where the parameter *m* is calculated as *m* = 0.15**n*, and *n* defines the kernel width.** Next, calculate the maximum score while convolving the kernel by calculating the different local sums of the different areas in the kernel and then save the position of the maximum score.

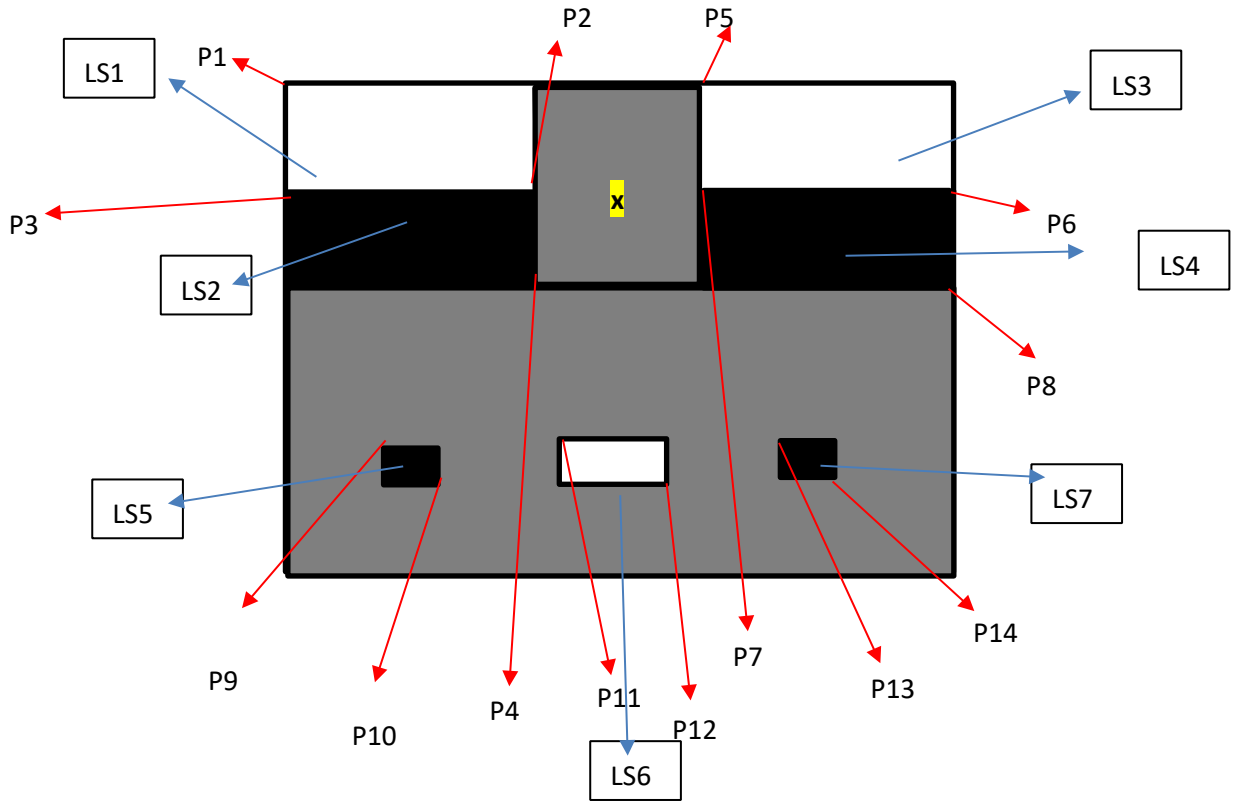
    - The kernel is shown below including the points representing each area (P1 … P14) in order to calculate seven different local sums (LS1 … LS7). The kernel is designed to match the area of the face showing the eyes, the eye brows and the forehead.
    - The gray color refers to a neglection area (zeros), the white color refers to ones in the kernel and the black area refers to negative ones in the kernel.

**Note:** This kernel must nullify where the summation of LS1 and LS2 equates to zero, LS3 and LS4 their summation equates to zero, and LS5, LS6 and LS7 their summation equates to zero. Moreover, do not overlap the kernel areas.

# Assignment #1

## Thursday, 13th of November, 2025

**Modified Kernel Representation:**




P1 = (-0.5n,-0.5m)
P2 = (-0.05n,0)
P3 = (-0.5n,0)
P4 = (-0.05n,0.5m)
P5 = (0.05n,-0.5m)
P6 = (0.5n,0)
P7 = (0.05n,0)
P8 = (0.5n,0.5m)
P9 = (-0.325n,0.833m)
P10 = (-0.225n,2m)
P11 = (-0.1n,0.833m)
P12 = (0.1n,2m)
P13 = (0.225n,0.833m)
P14 = (0.325n,2m)

The values given here are computed relative to a reference point **x** shown in the figure above which is assumed to be located at (0, 0).

Where (m = 0.15n)

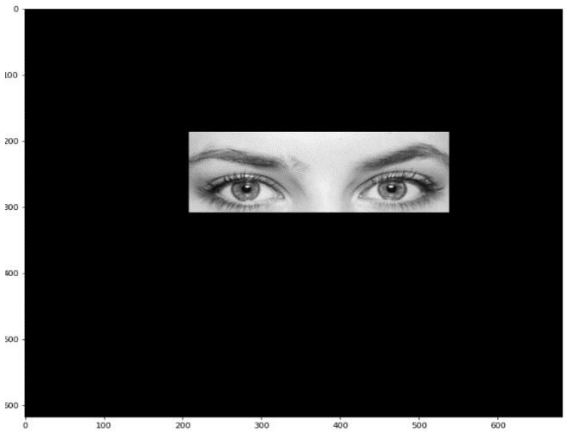
# Assignment #1

## Thursday, 13th of November, 2025

2. **ExtractDetectedEye:**
    - Input: Takes the image itself, the maximum position retrieved from DetectEye method and the kernel width size
    - Output: a 2D array representing the image drawn.
    - Description: This method is responsible to extract the eye area itself as detected from the first method. As we have the position of the maximum score, then draw around that position the same kernel size that detects the eye from the whole image. So, it is only retrieving the pixels of interest which are the eyes in the same range of the kernel size in terms of width and height.

**Test these functions on the following image with kernel size width equals to 330.**

**Here is the expected output after detecting the eye area with kernel size 330.**



# Assignment #1

## Thursday, 13th of November, 2025

**Show the output for the Second image with kernel size 150.**

**Also, show the output for the Third image with kernel size 250.**

# Assignment #1

## Thursday, 13th of November, 2025

---

**Submission Guidelines:**

1. **Teams:**
   - This assignment is should be done in groups of **3 to 4** students. **All students must be from the tutorial groups of the same TA.**

2. **Assignment Submission:**

You should submit the assignment through **submission form** mentioned **below taking into consideration** the following notes:

- You are asked to deliver a notebook (.ipynb/.py) showing your implementation of all the required functions representing the outputs for each.
- The .ipynb/.py file containing your assignment's implementation **(the notebook should be submitted showing the cells being run before and representing the output).**
- You can use the sample image given above in previous pages **(files provided with the assignment)** to test your code.
- The output for the second and third images given above in previous pages.
- The file should be uploaded on your drive and provide us with the link and make sure to **be accessible.**
- Submit the assignment through this form:
  https://forms.gle/vV9YBZRftNNJKfQy9

---

**Note that:**

- Copying code from other teams or ChatGPT is **totally prohibited**. A cheating detector will be used to confirm that. Any cheating case detected will be a **ZERO**.
- You **are not allowed** to use any **predefined functions for any of the requirements**.

- **The deadline to submit the assignment is on Thursday, 13th of November, 2025**