# ENPM808X - Midterm (Proposal 0)

Our methodology is to incorporate FCNN - based YOLO Algorithm to detect and track humans. Our application will use a pre-trained model, based on the COCO - 2017 dataset which has classifications for upto 80 objects. To track our progress, we will utilize the Agile Methodology implementing Multiple Backlogs (Product, Iterative I, Iterative II and Worklog).
The steps followed are as follows:

1) Capture an image.
2) Pre-process the image to obtain confidence-level and weights of each class.
3) Post-process the image to obtain the best class.
4) Append bounding box of the detected object onto the image.
5) Display the modified image.

The camera will run in a loop till interrupted by keyboard or the time-limit runs out. The camera **captures an image** which will enter the Preprocessing segment of the program.

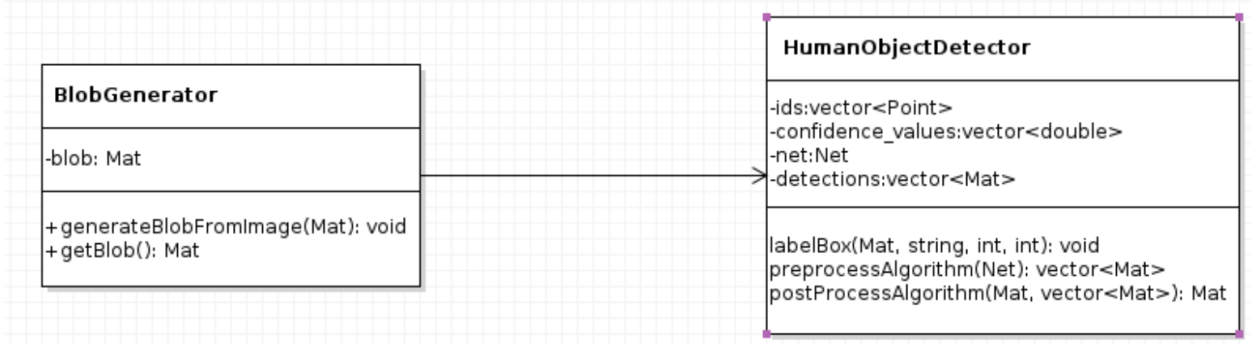
In the **Preprocessing step**, the image is converted to a blob. The blob forward-propagates into an OpenCV function, which returns an array [Order: (number_of_bounded_boxes, 85)]. Each bounded-box has 85 features out of which the first four features are X, Y (coordinates of center of bounding-box), W,H (Width and Height of bounding-box) and C(Confidence-Level). The remaining are class-weights of the bounding-box. The number of bounding-boxes are proportional to the size of the image-input.

The array can be assumed to be the output of the Neural-Network but it needs to be polished to obtain the best class-label. For this, we pass the output array into a **Post-Processing** segment wherein certain thresholds (Confidence and Score thresholds) are defined. Boxes having low confidences are considered to be bad detections and are therefore rejected.

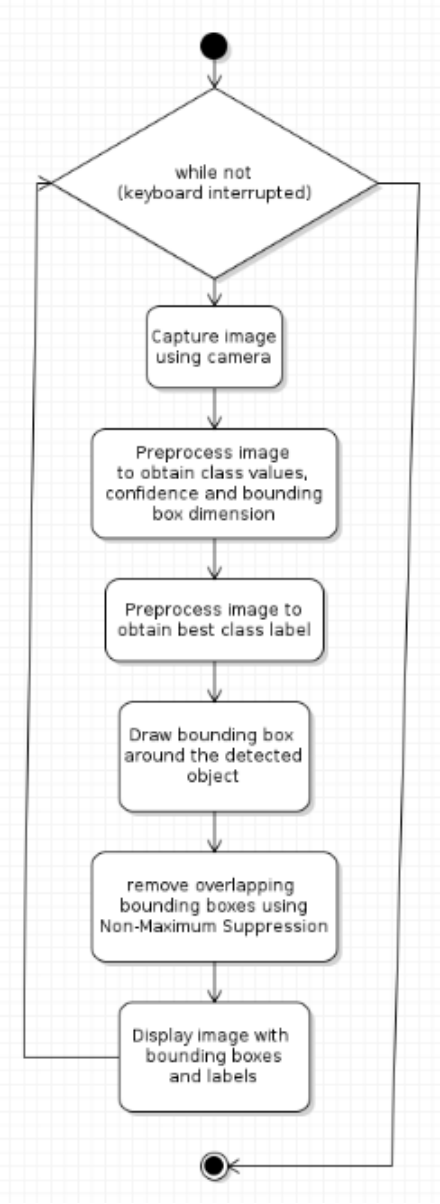
For the boxes with good detections, we find the id of class with maximum weight. If that class weight (or class score) is more than the score threshold, then we draw a bounding-box around the object with the class-label. There may be instances where multiple bounding boxes are drawn around the same object (Boxes overlapping). For this, we use a technique called Non-Maximum Suppression. The bounding-box is appended to the image and the modified image is then displayed on the screen.

The above video may not run at the desired frame-rate because the Human-Detection algorithm may take some time to detect and append bounding boxes onto each frame, therefore making the video slower than usual.

## UML Diagram:



## Activity Diagram:




## Sources:

*YOLO Algorithm for Object Detection:*
https://arxiv.org/pdf/2007.12099.pdf
*YOLOv5 pre-trained models:*
https://github.com/ultralytics/YOLOv5
*Code References:*
https://learnopencv.com/object-detection-using-yolov5-and-opencv-dnn-in-c-and-python/
https://www.youtube.com/watch?v=GmFe8ioTX3Q
docs.opencv.org