AI61201: Visual Computing with AI/ML

Programming Assignment 5: Feature Design (20 Marks)

Due Date: *October 20 (by 9 PM IST)*

Instructions: *Complete the two tasks given, keeping the following point in mind:*

- *Vectorized implementations are possible for most operations, and using for or while loops to access individual pixels is strongly discouraged due to its inefficiency. You can use NumPy to perform the mathematical operations.*

Task 1: Harris Corner Detection (10 Marks)

1. **Implementing Harris Corner Detector (4 Marks)**
    - Implement the Harris Corner Detector algorithm from scratch.
2. **Applying the Algorithm to `boxes.png` (1 Marks)**
    - Apply the implemented algorithm to detect corners in `boxes.png`.
3. **Visualization of Detected Corners (2 Marks)**
    - Visualize the detected corners by marking them on the image.
4. **Report: Number of Detected Corners (1 Mark)**
    - Report the total number of corners detected in the image.
5. **Comparison with OpenCV's implementation (2 Marks)**
    - Compare your results with OpenCV's implementation of Harris corner detector and explain any differences observed

(Note: You can directly use OpenCV for operations like gradient computation, Gaussian weighting etc. There is no need to implement the corresponding convolution operations from scratch. However, all the steps for Harris corner detector must be implemented).

Task 2: Hough Transform for Circle Detection (10 Marks)

1. **Canny Edge Detection (2 Marks)**
    - Use OpenCV's **Canny edge detector** to generate the edge map from `clocks.png`. Show the edge map.
2. **Implementing Hough Transform for Circle Detection (4 Marks)**
    - Implement the Hough Transform for circle detection from scratch using the edge map obtained in the first step.
3. **Applying the Hough Transform to Detect Clocks (1 Marks)**
    - Apply the implemented algorithm to detect clocks in the `clocks.png` image. How many clocks are detected by the algorithm you implemented?

4. **Visualization of Detected Circles (1 Marks)**
    - Visualize the detected circles by annotating them on the image.

(Note: since Hough transform is computationally intensive, you can resize the given image to a smaller resolution like 128 ⨉ 128 and use it for steps 1-4).

5. **Comparison with OpenCV's implementation (2 Marks)**
    - Compare your results with OpenCV's Hough Circle Transform and explain any differences. Report the number of clocks detected by OpenCV in this case.

Link to download the inputs for all tasks -
https://drive.google.com/drive/folders/1tlznd4xsf5AFgKsWgdJgUmRbPFqQM96M?usp=sharing

Submission Guidelines

1. *The content that you submit must be your individual work.*
2. *Submit your code in .py as well as in .ipynb file format. Both these file submissions are required to receive credit for this assignment.*
3. *Ensure your code is well-commented and easy to follow. You can write your answers and explanations using text cells in the jupyter notebook files wherever required.*
4. *The files should be named as "<roll_number>_assignment_5". For example, if your roll number is 20MI3AI18, the code file names will be 20MI3AI18_assignment_5.py and 20MI3AI18_assignment_5.ipynb. You should place all these files within a single zip file and upload it to Moodle as 20MI3AI18_assignment_5.zip.*
5. *All submissions must be made through Moodle before the deadline. The submission portal will close at the specified time, and submissions via email would not be accepted.*

TA for this assignment: Sourabh Choudhary

If you have any queries regarding Assignment 5, please email at jaihindsourabh@gmail.com .