ASSIGNMENT #01

COMPUTER VISION (FALL-24)
BS-AI

Due Date: 15 Sept 2024

Question 1: Image Stitching using SIFT on COIL-20 Dataset

In this assignment, you will implement the Scale-Invariant Feature Transform (SIFT) algorithm for image stitching using the **COIL-20 dataset**. The COIL-20 dataset consists of grayscale images of 20 different objects, each captured from 72 different angles, spaced at 5-degree intervals. This dataset is commonly used for object recognition and computer vision tasks. You can download the dataset from the following link:
https://www.kaggle.com/datasets/cyx6666/coil20

Problem Statement:
Your task is to apply the SIFT algorithm to stitch together 15 images of **each object** in the COIL-20 dataset. Instead of using all 72 images, you will select 15 images captured from different angles for each object that will best form a coherent and logical panorama.
Requirements:

- **Stitch All Objects**: For each object in the COIL-20 dataset, you must select 15 images taken from different angles (out of the available 72 angles).
- **Image Stitching**: Use SIFT to detect features and align the selected images to create a panorama or stitched image for each object.
- **Objective**: Create as clear and logical a panoramic image as possible, ensuring smooth transitions between the images.

Output:
Submit stitched panoramas for all 20 objects and the code file

Question 2: Build a Chess Game using Image Processing:

Part 1: Detect Lines on the Chessboard Using Hough Transforms

1. Use the Hough Line Transform to detect the grid lines of the given chessboard image.
2. Draw the detected lines on the image to visualize the grid.

Part 2: Counting the Number of Boxes on the Chessboard

1. Use the intersection points of the detected lines from Part 1 to calculate the number of boxes (squares) on the chessboard.

Part 3: Placing Chess Pieces on the Chessboard

1. Place chess pieces on the board by replacing the pixels in each chess square with the pixels of the respective chess piece.
2. Use the locations of the intersections calculated in Part 2 to determine where each chess piece should be placed.

Part 4: Applying Transformations to the Chessboard

1. Apply transformations to the chessboard to make the game look dynamic. Use projections and other transformations to simulate a 3D effect for the board.

Output:
Submit your Python code along with the final images showing the chessboard with pieces placed and any transformations applied.

Note: Submit a single zip folder containing both folders of questions.