Chessboard Analyzer Project Documentation

Objective

The primary objective of this project is to analyze a chessboard image, identify and annotate its black and white squares, and calculate the count of each type of square. This involves correcting the perspective of the chessboard, dividing it into a grid, and applying image processing techniques to classify the squares.

Approach

The Chessboard Analyzer employs a systematic approach to process a chessboard image and annotate its squares. The workflow is divided into three key stages:

1. Point Selection and Homography Transformation

- The first step is to interactively select four points on the chessboard that represent its corners (top-left, top-right, bottom-left, and bottom-right).
- A **homography transformation** is then applied using the selected points. This corrects the perspective distortion in the image, producing a straightened, bird's-eye view of the chessboard.
- The transformed image ensures uniform grid dimensions and facilitates the subsequent analysis of the board.

2. Preprocessing

- Preprocessing involves preparing the image for analysis by enhancing its contrast and binarizing it. This is done in two steps:
  - **Contrast Limited Adaptive Histogram Equalization (CLAHE):**
    - CLAHE enhances local contrast, making the intensity differences between black and white squares more distinct.
  - **Thresholding:**

        - Using Otsu's thresholding, the image is converted to binary form, where black and white pixels represent the black and white squares respectively.
- This step ensures that the chessboard squares are easy to segment and classify.

3. Grid Method for Square Classification

- The straightened image is divided into an **8x8 grid**, where each cell represents a square of the chessboard.
- For each grid cell:
    - The region of interest (ROI) is extracted, and the ratio of black pixels to the total number of pixels in the cell is calculated.
    - A threshold (50%) is used to classify the cell as either a black or white square.
- The detected squares are then annotated with red borders for black squares and green borders for white squares, creating a visual representation of the analysis.

Preprocessing: Key Steps and Purpose

1. **Contrast Enhancement:**
    a. CLAHE is applied to enhance the local contrast of the image. This step helps address variations in lighting across the chessboard, ensuring consistent detection of square colors.
    b. Without CLAHE, parts of the image under shadow or glare might produce incorrect classifications.
2. **Thresholding:**
    a. The image is binarized using Otsu's thresholding method, which automatically determines the optimal threshold value based on the image's intensity histogram.
    b. This step simplifies the image into two distinct regions: black squares and white squares, which are essential for the grid-based analysis.

Main Algorithm: Grid Method for Square Detection

The core algorithm used for detecting and annotating the chessboard squares is the **grid method**. Here's a detailed breakdown of how it works:

1. **Image Division:**
    a. After applying the homography transformation, the chessboard image is uniformly divided into an 8x8 grid. Each grid cell corresponds to a square on the board.
    b. The height and width of each grid cell are calculated based on the overall dimensions of the transformed image.
2. **Pixel Analysis:**
    a. For each grid cell, the number of black pixels (intensity = 0) is counted.
    b. The ratio of black pixels to the total number of pixels in the cell is computed.
3. **Classification:**
    a. A threshold value (0.5 or 50%) is used to classify the square:
        i. **Black square:** If the black pixel ratio exceeds 50%.
        ii. **White square:** Otherwise.
    b. This approach ensures accurate classification regardless of slight variations in lighting or texture.
4. **Annotation:**
    a. After classification, the grid cell is annotated with a colored border:
        i. **Red** for black squares.
        ii. **Green** for white squares.
    b. The annotations provide a clear visual representation of the results.
5. **Counting Squares:**
    a. As the grid is processed, the counts of black and white squares are maintained and displayed at the end of the analysis.

Workflow Explanation

Here's a step-by-step walkthrough of the workflow implemented in the code:

1. User Input (Point Selection)

- The user selects four points on the image by clicking on the corners of the chessboard. These points are used to compute the homography transformation matrix.

2. Perspective Correction

- Using the selected points, the chessboard is straightened into a standard 400x400 pixel output image using OpenCV's `cv2.warpPerspective()` function.

3. Preprocessing

- The straightened image undergoes CLAHE and Otsu's thresholding to prepare it for grid-based analysis.

4. Grid Analysis

- The image is divided into an 8x8 grid, and each square is classified based on the ratio of black pixels. The classifications are annotated on the image.

5. Output

- The annotated image is displayed using Matplotlib and saved to the specified output directory.
- The counts of black and white squares are printed to the terminal.

Results

When the program is executed:

- The chessboard is successfully analyzed, and the black and white squares are annotated in the output image.
- The program outputs the count of black and white squares to the terminal, providing an objective measure of the detection results.