20241224影像處理期末報告

**1. 題目 :** 基於 OpenCV 的棋盤與棋子檢測系統

**2. Introduction**

**2.1 動機**

從課上的object detection 練習 (tic-tac-toe 跟 tetris block)發想，  
我認為西洋棋盤與棋子也是一個值得做出來的主題。

**2.2 目的**

開發一個能夠準確檢測棋盤並識別棋子位置的系統，成功輸出棋盤狀態(有哪些格子有棋子,以0跟1所表示)。

**3.** 文獻探討

* 1. **Chessboard Detection**
* 方法：使用 OpenCV 的 cv2.findContours 和 cv2.approxPolyDP 來檢測棋盤輪廓並將其逼近為四邊形。

**3.2 Perspective Transformation**

* 方法：採用 cv2.getPerspectiveTransform，將檢測到的四邊形映射為正方形，確保棋盤能被精確地對齊。

**3.3 Grid Division**

* 方法：將棋盤分為 8x8 網格，每個網格作為檢測棋子的區域（ROI）。.

**3.4 Handling Lighting Conditions**

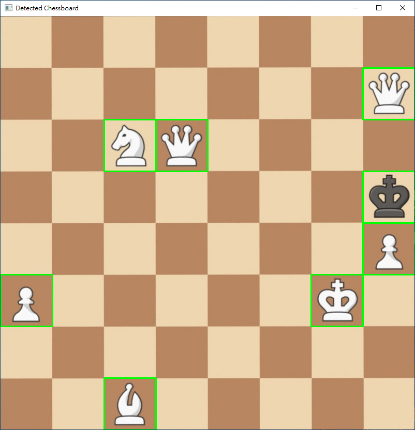
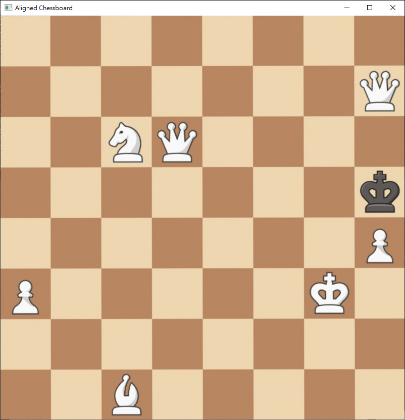
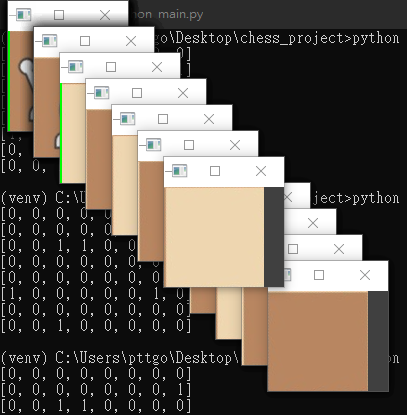
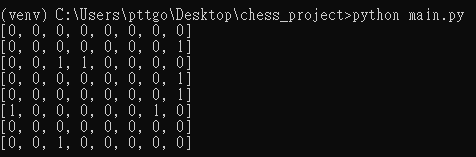
* 方法：使用Adaptive Thresholding，應對光線不均情況並準確檢測棋盤和grid markings。.

**4.** **我的方法**

1. **Chessboard Detection:**
   * 使用 cv2.findContours 檢測棋盤的四邊形結構，參考課堂範例。
2. **Perspective Transformation:**
   * 根據四點對應完成透視變換，實現棋盤的準確對齊。.
3. **Chess Piece Detection:**
   * 邊緣檢測：使用cv2.Canny和Adaptive Thresholding以加強圖像重建能力。
   * 圖像處理：結合segmentation與color thresholding進行穩健的檢測。.

#### 5. 實驗結果

**5.1 檢測結果展示**

* 原始影像(有傾斜,有非棋盤物)及棋盤偵測
* 透視變換、棋盤劃分及棋子檢測
* 輸出成表格狀

**5.2 結果分析**

**棋盤部分:  
對於一些在chess.com上的chessboard style可以有正確的結果，如NEO、棕色、冰洋等，相反的對於那些兩種顏色相似/邊界不明顯的)還不適配**

**棋子部分:  
能正確的判斷哪些方格內(將圖片分成8\*8之後的格子)有物品(棋子)**

**5.3 改進空間**

* **處理光線變化**：  
  強化對光線影響的處理及降噪能力。
* **提升檢測能力**：  
  採用深度學習模型進行多模態分類，以提升棋子檢測效果。
* **增加功能:  
  增加辨識棋子種類的功能(white\_pawn,black\_rook之類的)，將棋盤辨識+棋種辨識之後  
  以表格的方式印出哪格有什麼棋子(C3 wp, A1 br之類的)。**

**6.** 參考資料

1. OpenCV 官方文件: [https://docs.opencv.org](https://docs.opencv.org/)

#### ****相關論文****

* **“**[End-to-End Chess Recognition](https://arxiv.org/abs/2310.04086)**" by** Athanasios Masouris, Jan van Gemert
* “[Determining Chess Game State from an Image](https://arxiv.org/abs/2104.14963)” **by** Georg Wölflein, Ognjen Arandjelović

1. 課堂教材

* 1.11, Contours - Advanced Properties
* 1.9, Thresholding