# 《连连看助手》需求文档（V8 修订版）

## 一、项目概述

本项目《连连看助手》是一款基于图像识别与视觉叠加技术的智能辅助工具，通过实时分析游戏棋盘画面，自动计算并可视化展示下一步最佳消除提示线。本程序采用 Python 编写，核心组件包括 OpenCV、PyQt5、PyAutoGUI、NumPy 等。所有图像识别与匹配都完全由 **OpenCV + PyQt5 + PyAutoGUI** 实现

## 核心模块及功能

系统由以下五个核心模块组成：  
1. 控制面板（Dialog）：负责用户输入，包括ROI坐标、颜色优先级、运行控制、调试选项。  
2. 图像识别模块（Recognizer）：完成屏幕截图、棋盘分割、模板颜色匹配。  
3. 分析与提示模块（Analyzer）：穷举模拟交换，分析可能形成的3连或4连组合并输出提示线。  
4. 叠加层（Overlay）：负责在屏幕上绘制半透明颜色填充、文字标签与白色提示线。  
5. 配置与日志模块（ConfigManager）：保存用户设置，控制调试日志输出与防抖逻辑。

主要功能：

1. 自动识别屏幕中固定区域（8×8 棋盘）的颜色布局。  
2. 支持判断横向、纵向、L 形、T 形、≥5 连等多种可消除组合。  
3. 自动分析所有可能的交换操作，得出最优解并绘制提示线。  
4. 提示线支持规模优先策略与颜色优先级机制。  
5. 可视化叠加层实时显示识别结果与提示路径，辅助调试。

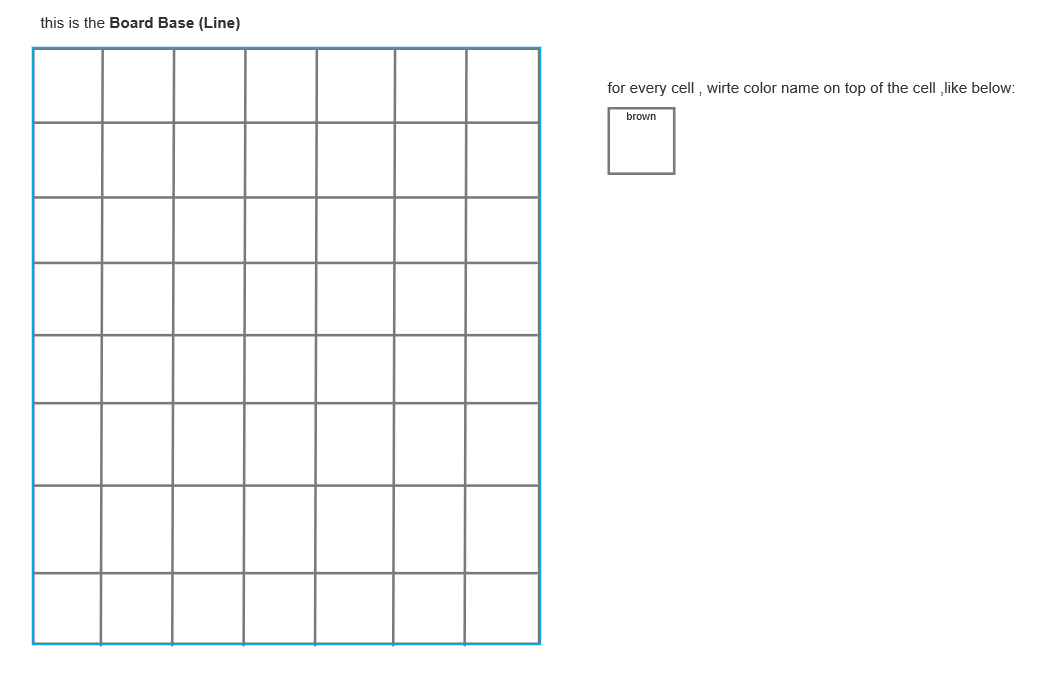
## 三、运行环境

· 游戏分辨率：1920×1080（窗口模式）。  
· 棋盘固定大小：932×932 像素。  
· 棋盘左上角坐标默认值：(494,100)，可在控制面板调整。  
· 运行依赖：opencv-python、numpy、pyautogui、PyQt5。

## 四、核心功能与算法

### 辅助调试功能（棋盘识别 + 可视化叠加层）

此功能主要用于开发与测试阶段，帮助开发者与玩家验证识别算法的准确性与绘制效果，可在程序界面中选择是否开启（默认开启，未来可改为可选项）。  
  
· **可视化叠加层**：

可视化叠加层是为了辅助我调试程，是帮助我在制作程序的过程中识别我的程序是否能在关键步骤识别棋盘的判断依据。采用 PyQt5 的透明窗口实现覆盖显示。该层中内容及顺序如下：  
 - **小球颜色识别基盘**（底层）：半透明颜色块填充（30% 透明度），显示识别到的颜色。  
 - **棋盘基座线**（中层）：红色棋盘基座线（3px 常亮），显示一个 8×8 的网格线，这个网格线中有64个小格子（记为：cell）  
  
 - **提示线**（顶层）：白色提示线（5px 常亮），指示当前最优或并列最优消除位置，提示线绘制后，玩家无需猜测下一步，直接根据线条指示移动即可。

### 4.1 小球颜色识别算法

图像模板匹配方案:

对每种小球准备模板图，模板存放于程序目录下的templates文件夹，包括red、blue、green、yellow、purple、brown、skull七种图像；运行时对每个格子进行小图四配(使用OpenCV的cv2.matchTemplate在RGB模式下进行颜色匹配，取代早期使用第三方库Airtest，实现独立的识别模块)。

注意： 小球颜色的识别是后续判定提示线的核心依据，如果这一步没有做好，那么自然会引起下一步提示线绘制发生错误

**识别流程：**  
1. 将截图分割成8×8单元格；  
2. 每个cell与所有模板匹配，取最大置信度；  
3. 若置信度波动小于阈值（防抖机制），沿用上一帧结果。

优化建议：在模板匹配前可加入预处理，如裁剪模板外圈10%、CLAHE对比度均衡、轻微模糊，以增强光照鲁棒性。

#### 小球颜色映射与显示

在抓取的小球与模板图像进行匹配后，会在【小球颜色识别基盘】中渲染成指定的颜色，例如：红色小球会匹配red.png，然后在小球颜色识别基盘显示为一个填充色#FF0000透明度30%的正方形色块，该色块可以快速，明显的帮助我看到和判断当前小格子内，是否正确的识别出了每个小球的颜色

| **小球类型** | **模板图像** | **填充颜色** | **透明度** |
| --- | --- | --- | --- |
| 红色 | red.png | #FF0000 | 30% |
| 蓝色 | blue.png | #0000FF | 30% |
| 绿色 | green.png | #00FF00 | 30% |
| 黄色 | yellow.png | #FFFF00 | 30% |
| 紫色 | purple.png | #8000FF | 30% |
| 棕色 | brown.png | #7B3F00 | **60%** |
| 骷髅 | skull.png | #CCCCCC | 30% |

备注：

考虑到模板图像中是有背景色的（来自于游戏内的棋盘底色，分为深灰和浅灰2种），所以这里默认裁去模板图像外圈 **10%，降低干扰，具体为：**自动处理模板和格子图：**裁掉外圈 10% 灰边** → 灰度归一化 → 轻度模糊 → **CLAHE 对比增强** → 统一尺寸后匹配。

存储位置：

小球识别时的颜色参照模板存储在：Python代码当前所在的目录的Template文件夹下，目录结构如下：

连连看/

├── match\_overlay\_v11.py

├── templates/

│ ├── red.png

│ ├── blue.png

│ ├── green.png

│ ├── yellow.png

│ ├── purple.png

│ ├── brown.png

│ └── skull.png

└── match\_overlay\_config.json

### 4.2 消除判定规则

系统支持以下可消除组合类型：  
1. 横向或纵向连续 ≥3 个相同颜色的格子。  
2. L 形组合（如：(1,1)-(1,2)-(1,3)-(2,3)-(3,3) 或其镜像）。  
3. T 形组合（如：(1,2)-(2,1)-(2,2)-(2,3)-(3,2) 或其旋转变体）。  
4. ≥5 连长线（如横向5连、竖向5连）。  
所有这些形状均需被识别为有效消除结构。  
  
棋盘扫描策略：  
· 穷举所有相邻格（水平与垂直方向各一次）交换。  
· 对每次交换模拟生成新棋盘。  
· 计算交换后形成的所有可能消除组合。

### 4.3 提示线绘制算法

提示线的绘制逻辑是本程序的核心模块，其目的是在屏幕中标出“移动后形成的可消除区域”。  
算法步骤如下：  
1. 枚举所有可能交换的两个相邻格（向右与向下）。  
2. 模拟交换并生成新的棋盘状态。  
3. 在新棋盘中查找所有 ≥3 连的消除组合，并自动将相连区域（L/T）合并为一个整体组件。  
4. 为所有候选组合计算规模（长度或总格数）。  
5. 确定最大规模 `max\_len`：  
 - 若 `max\_len = 3` → 仅绘制颜色优先级最高的组合。  
 - 若 `max\_len ≥ 4` → 同时绘制所有 `len == max\_len` 的组合，可跨颜色。

绘制规则：  
· 白色线条（5px 不透明），常亮显示。  
· 若形成多个同规模组合（例如三组4连），则同时显示三条提示线。  
· 当检测到 T 或 L 形结构时，自动合并成一条完整连线。  
· 绘制线条位置基于“移动后形成的消除区域”，而非移动路径。  
· 当棋盘无有效解时，保持上一帧结果不变。

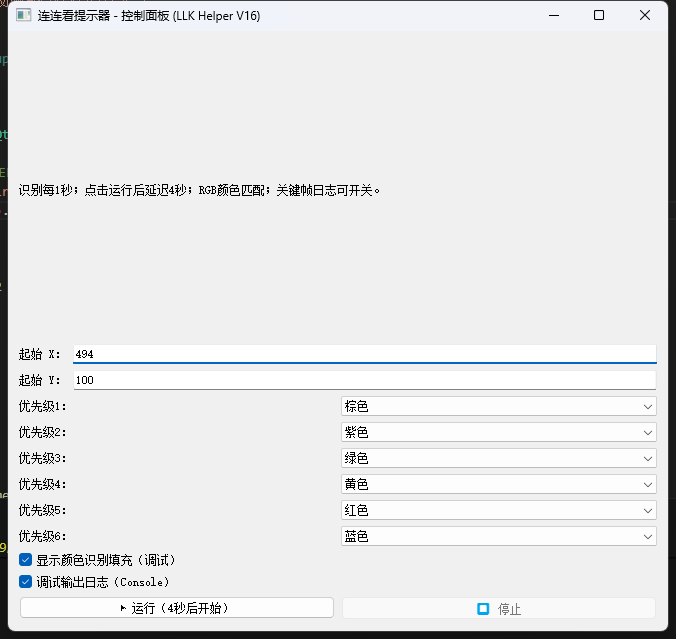
示例说明：  
假设当前棋盘为 3×3：  
(1,1) 红 (1,2) 绿 (1,3) 蓝  
(2,1) 黄 (2,2) 绿 (2,3) 紫  
(3,1) 绿 (3,2) 红 (3,3) 紫  
若将 (3,1) 的绿色小球移动至 (3,2)，可形成纵向三连 (1,2)-(2,2)-(3,2)。  
程序将绘制一条白色提示线贯穿这三个格子的位置。

### 4.4 颜色优先级机制

用户可在控制面板中自定义颜色优先级：  
默认顺序如下：  
优先级1：棕色  
优先级2：紫色  
优先级3：绿色  
优先级4：黄色  
优先级5：红色  
优先级6：蓝色  
当消除规模 = 3 时，程序根据优先级选择绘制最高优先颜色的提示线。

### 4.5 用户交互与控制界面

主界面功能：  
· ROI起始坐标输入框，允许用户自定义截图的起始坐标，默认是（494,100）。  
· 6 个颜色优先级下拉框，允许用户自定义提示线绘制时的优先级。  
· checkBox：

1. 显示颜色识别填充（默认开启，可关闭），该功能用于打开识别小球时的颜色填充以及每个Cell顶部的文字，可以直观的看到识别的是否正确
2. 调试输出日志（Console），用于在vscode的console中打印出关键步骤信息，方便调试。  
   · 底部按钮：运行 / 停止程序按钮  
   真题界面如下图，虽然有点丑，但是基本功能都有  
     
     
   运行逻辑：  
   1. 点击“运行”→ 程序最小化，2 秒后自动启动叠加层。  
   2. 叠加层置顶，实时捕获与绘制提示线。  
   3. 当用户点击任务栏重新打开窗口时，叠加层自动 lower()，以便可操作“停止”按钮。  
   4. 点击“停止”→ 停止刷新并隐藏叠加层。

### 4.6 配置文件

配置文件名：match\_overlay\_config.json（自动生成与脚本同目录）  
内容示例：  
{  
 'roi\_x': 490,  
 'roi\_y': 100,  
 'priority\_cn': ['棕色','紫色','绿色','黄色','红色','蓝色'],  
 'show\_color\_overlay': true  
}  
每次点击“运行”时自动更新。

## 五、显示参数

|  |  |  |
| --- | --- | --- |
| 元素 | 颜色 / 样式 | 备注 |
| 棋盘基座线 | 红色线条 3px | 常亮，固定 8×8 网格 |
| 提示线 | 白色线条 5px | 常亮，显示符合条件的消除组合 |
| 颜色识别填充 | 对应颜色，30%透明 | 可关闭，用于调试棋盘识别准确性 |

## 六、性能与刷新机制

· 刷新周期：300ms。  
· 识别与绘制均在主线程执行。  
· 单次计算最多模拟 128 次交换。  
· CPU 占用 5%~10%，性能稳定。

## 七、未来优化方向

1. 模板预处理：裁剪、CLAHE、模糊以增强稳定性；  
2. 动态刷新率：允许300~1000ms范围调节；  
3. 自动ROI：识别棋盘区域自动定位；  
4. 热键ROI微调：Ctrl+Alt+方向键；  
5. 配置可视化：面板显示保存状态；  
6. 模块结构化：拆分Recognizer、Analyzer、Visualizer以便维护。

**八、结论附录： 项目进度里程碑**

这里记录所有的里程碑版本：

1. **LLK\_Helper\_V16.py**：  
    这是迭代了16次后的版本，2025年10月16日晚形成，OpenCV正确识别了所有小球，并且在每隔cell上都显示出了正确的识别结果，因此，提示线也被正确的绘制了。这个版本是一个从头到尾跑通的稳定版本，但是还有诸多缺陷，比如：

* 不能灵活的去掉**小球颜色识别基盘**（底层）
* 识别过程中，叠加层会遮挡控制面吧（Dialog窗口），导致后续的停止、运行操作等不好点击。需要手动的从任务栏关闭应用，这很傻。。
* 

1. 其他迭代版本，还没有进行呢