

# 信号与系统— MATLAB 综合实验之连连看<sup>1</sup>

谷源涛

gyt@tsinghua.edu.cn

二〇一五年七月六日

<sup>1</sup>本文为清华大学电子工程系三字班本科生课程“MATLAB 高级编程与工程应用”的教学课件。请勿转载和过度引用。

# 目录

<b>第一章 制作自己的连连看</b>	<b>1</b>
第一节 引言	1
第二节 设计和制作游戏	1
1.2.1 游戏规则和流程	1
1.2.2 数据结构	1
1.2.3 判定两个块是否可消除	2
1.2.4 判定游戏可否继续进行	3
第三节 “外挂”模式	3
第四节 科学问题	3
第五节 练习题	3
<b>第二章 攻克别人的连连看</b>	<b>5</b>
第一节 系统概述	5
第二节 处理图像数据	7
2.2.1 图像分块	7
2.2.1.1 计算分块大小	9
2.2.1.2 确定左侧空白和上方空白的大小	10
2.2.2 寻找相同的图像块	10
2.2.2.1 高通滤波	11
2.2.2.2 度量图像分块的相似性	12
2.2.2.3 相似对排序或自动聚类	13
第三节 其他模块	13
2.3.1 获取图像	13
2.3.2 自动按键	13
第四节 练习题	13

# 第一章 制作自己的连连看

## 第一节 引言

本章引导大家设计和制作一款经典连连看游戏，并实现一种自动游戏的“外挂”模式。本章内容主要涉及编程知识，和“信号与系统”关系不大，但本章内容将为下一章奠定基础，所以也必须全部掌握。

## 第二节 设计和制作游戏

### 1.2.1 游戏规则和流程

不考虑各种花样和道具，连连看游戏的基本流程是：

1. 载入或动态生成游戏区域；
2. 循环直到满足退出条件
  - (a) 判定是否有可消除的块，如果没有则随机交换全部或某些块的位置直到有可消除的块；
  - (b) 玩家选择两个块；
  - (c) 电脑判断这两个块是否可消除；
  - (d) 如果是，消除这两个块；如果否，回到2b。

退出条件包括但不限于：所有块全部消除；玩家主动退出。

为了增强竞赛感或增加趣味性，有些游戏增加了时间限制；另一些游戏则在步骤2d之后增加了某些特定处理，比如：如果有空位的话，所有块向某个方向（或中心）移动；延长时间；等等。另外，游戏区域可能不是矩形，其间可能有若干个固化的无法消除的块。

### 1.2.2 数据结构

呈现在玩家面前的游戏区域由一个个图像块构成，但程序内部只要维护每个图像块的索引（数值）及其位置（二维数组）即可。例如，图1.1所示的游戏区域在程序里表示为

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 1 & 5 \\ 5 & 6 & 7 & 8 & 1 & 3 \\ 1 & 4 & 9 & 10 & 7 & 10 \\ 9 & 4 & 11 & 12 & 6 & 8 \end{bmatrix}$$





其中数字1表示，数字2表示，等等。当然也可以用其他数字或符号表示每个图像块，程序中只要维护好一个描述对应关系的常量表即可。另外，可以用0表示消掉块后的空位置。



图 1.1: 游戏区域示例

1.2.3 判定两个块是否可消除

上述流程中，要编程实现的“算法”含量最高的部分是步骤2c。  
连连看能够消除的模式包括三种，分别是直线连、拐弯连和拐俩弯连，如图1.2所示。

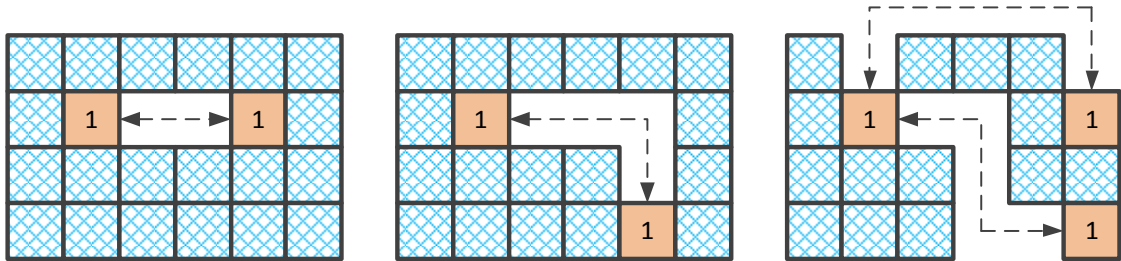


图 1.2: 可消除的模式

为判定直线连，可以先比较两个候选块是否有相同的横坐标（下标）或纵坐标；如有则进一步判定两者直连线经过的坐标上是否有块阻挡。  
为判定拐弯连，可分别以两个候选块为中心画十字，注意十字线遇到阻挡块就要终止；然后判断两个十字是否有交点。为判断拐俩弯连，可以在分别画十字的基础上，判定两个十字上的点之间是否有直连。这两种情况如图1.3所示。

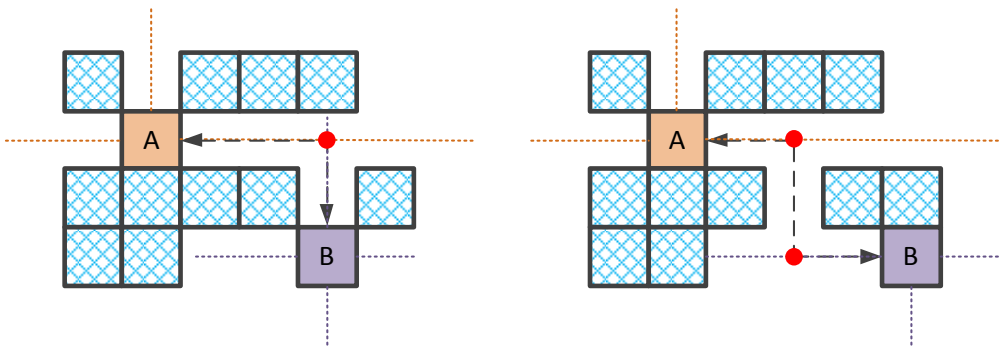


图 1.3: 判断块是否可消除的方法

当然可能有其他更好的判定方法，同学们可以自由设计。

### 1.2.4 判定游戏可否继续进行

每当玩家成功消掉两个块后，需要判定游戏是否可以继续玩，即是否还存在两个可消除的块。如果存在则游戏继续，否则随机交换所有或部分块的位置直到有可消除的块。

## 第三节 “外挂”模式

如果盯了好久也未能找到一对可消除的块，不耐烦的玩家往往会使用“道具”。道具可以是提示两个可消除的块的位置，也可以是炸掉任意两个相同的块，即使它俩不能相连。

比道具更进一步的是“外挂”模式，即游戏自动一对对地消除所有的块。

就实现而言，“外挂”模式一点不难，只要依次判定两两块是否可以消除，找到一对后消除，再行循环继续判断即可。但设计一个低复杂度算法却不简单，同学们可以试试。

## 第四节 科学问题

由“外挂”模式出发，很容易想到几个和连连看相关的理论难题：

1. 给定一个游戏区域，如何快速判断它可否完全消除？
2. 给定一个可完全消除的游戏区域，它的消除方法是否唯一？
3. 是否存在一类有意义的<sup>1</sup>游戏区域，它的所有消除顺序都能实现完全消除。
4. 图像块的种类和游戏区域的尺寸在上述问题中起到什么作用？例如，为保证恒存在可完全消除的游戏区域，在图像块的种类和游戏区域的尺寸都趋于无穷大的过程中，两者的相对关系服从什么规律吗？
5. ....

上述问题可能落入图论和组合数学的范畴，我们不打算解决。就我所知，大部分连连看游戏也没有试图解决上述问题，它们要么载入一个固定的游戏区域，要么生成一个随机区域，一旦发现无法消除，就随机交换图像块直到有可消除的块。

下面提一个可以解决的问题：如何生成一个有意义的游戏区域，并保证它是可完全消除的？

再进一步，如何定义“游戏难度”，并根据指定的难度生成一个有意义的游戏区域。

## 第五节 练习题

1. 在MATLAB环境下，设置当前路径为linkgame，运行linkgame（打开linkgame.fig 或右键linkgame.p点“运行”），熟悉游戏。（上述程序已经过MATLAB2010b以上版本的测试。）

---

<sup>1</sup>即不是那些显然的规则排列。

2. 注意linkgame目录下有个detect.p，它的功能是检测块是否可以消除。现在请你把它移动到其他文件夹或删除！然后把linkgame\reference 目录下的detect.m复制到linkgame目录下。detect.m文件中是detect函数，函数以图像块的索引矩阵与要判断的两个块的下标为输入，如果两个块能消掉则输出1，否则输出0。请根据文件中的注释提示，实现判断块是否可以消除的功能。写完后再次运行linkgame，检验游戏是否仍然可以正确运行，当你的程序的判断结果有误时，在游戏界面右下角会有提示。（注意：当detect.p 文件存在时，detect.m 文件将不会被执行，所以测试时一定要移走detect.p文件。）
3. 你一定发现了“外挂”模式，是不是很有趣？逐一自动消除所有的块的功能是由link 目录的omg.p实现的。现在请你把它也删掉！然后把link\reference目录下的omg.m复制到link目录下。omg.m文件的注释中对输入输出变量做了详细说明，请以这个文件为基础，实现逐一自动消除所有块的功能。（同上题要移走omg.p文件。）写完后再次运行linkgame，检验自动模式是否正确。（在自动点击过程中可按F12终止。）
4. （选做）不限于正文中讨论的内容，同学们自由发挥，愿意做些什么都好。

## 第二章 攻克别人的连连看

本章引导同学们用“信号与系统”知识和MATLAB编程技术，设计和实现一个软件，可以模拟玩家，自动玩现有的连连看游戏。本章介绍的思路不仅适用于连连看，对其他小型益智游戏也有参考价值。

### 第一节 系统概述

我们最终希望实现一个机器人，趴在电脑前，手拿鼠标玩连连看，如图2.1所示。



图 2.1: 终极目标：机器人手握鼠标玩连连看

技术太超前了往往被质疑是弄虚作假。好吧，我们做一个很糙很机器的版本，如图2.2所示。



图 2.2: 普通目标：机器人手握鼠标玩连连看

其实我们没有必要刻意做出人形的机器，只要做个专门玩连连看的就好，如图2.3所示。这个版本具备了模拟连连看玩家的基本要素：视觉传感器、信息处理中心和执行机构，系统示意如图2.4所示。



图 2.3: 现实目标：机器手操作鼠标玩连连看

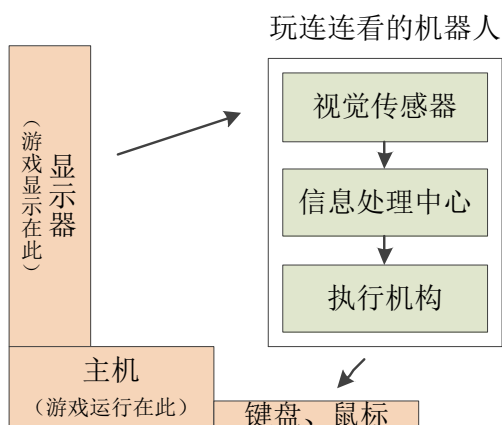


图 2.4: 现实目标：系统结构示意图

鉴于本课程的定位——MATLAB编程与应用——我们最终把课程要求定位如图2.5所示。



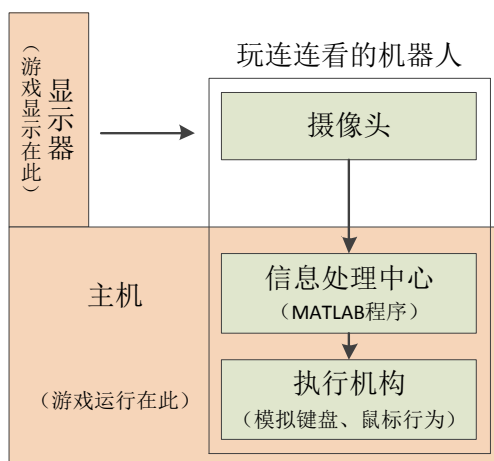


图 2.5: 课程目标：系统结构示意图

可见，上述系统中，需要我们设计和实现的主要技术模块包括

1. 采集屏幕为MATLAB可处理的图像数据；
2. 处理图像数据，寻找可消除的图像块，确定鼠标点击位置；
3. 模拟鼠标点击。

上述主要技术模块中，1可用MATLAB提供的Image Acquisition Toolbox编程实现，3可借助一款著名的工具“按键精灵”实现。我们需要重点解决的，也是最有“信号与系统”技术含量的，就是2（处理图像数据部分）。

## 第二节 处理图像数据

处理图像部分可进一步细分为

1. 识别游戏区域，将图像分块；
2. 寻找相同的图像块；
3. 寻找可消除的图像块。

其中3可以用上一章已经实现的技术解决，本章只要完成1和2即可。

### 2.2.1 图像分块

图像分割（Image Segmentation）是一个非常重要也相对成熟的研究领域。我们不打算介绍经典的图像分割技术，而仅仅针对连连看的具体问题，基于“信号与系统”的核心知识介绍一种原理简单且效果还不错的方法。

首先设定一些条件（先验知识），后文技术将主要在这些条件下工作。

1. 矩形游戏区域的采集比较端正，无残缺且空白不大（空白宽度/高度不超过一个图像块的宽度/高度）；
2. 游戏区域像素点足够多，比如每个图像块有 $30 \times 30$ 像素点以上。

比如已经采集到了图像，如图2.6所示。将其变为灰度图，如图2.7所示<sup>1</sup>。

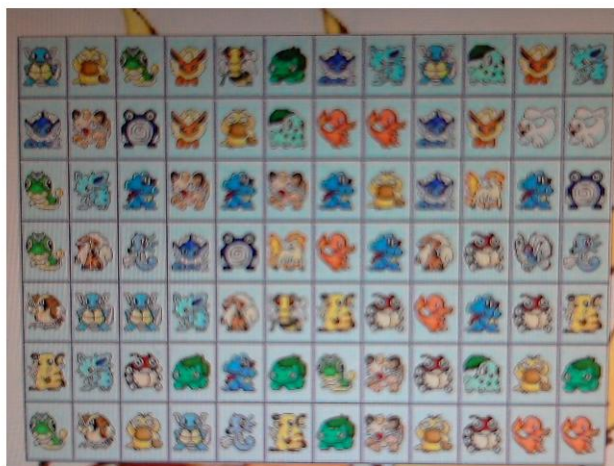


图 2.6: 原始彩色图像

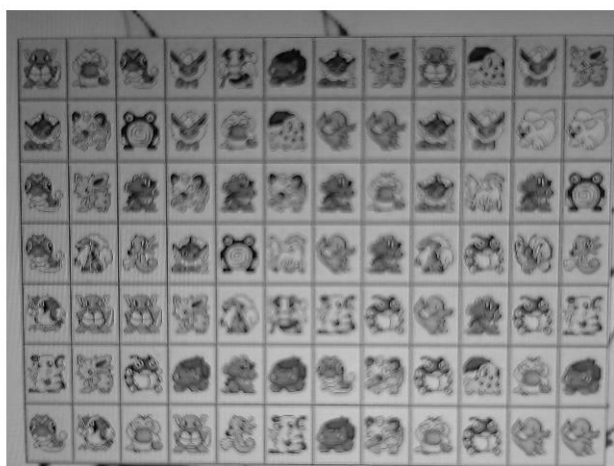


图 2.7: 灰度图像

图像分割的目的是提取出每个小的图像块，结合游戏规则和上述假设，我们只需要确定四个值：每个图像块的宽度 $W_B$ 、高度 $H_B$ ，左侧空白宽度 $X_S$ 和上方空白高度 $Y_S$ ，如图2.8所示。

<sup>1</sup>本章所有工作针对灰度图像进行。注意彩色到灰度的变化丢失了很多信息。利用彩色图像一定可以得到更高的精确度，鼓励同学们独立探索。

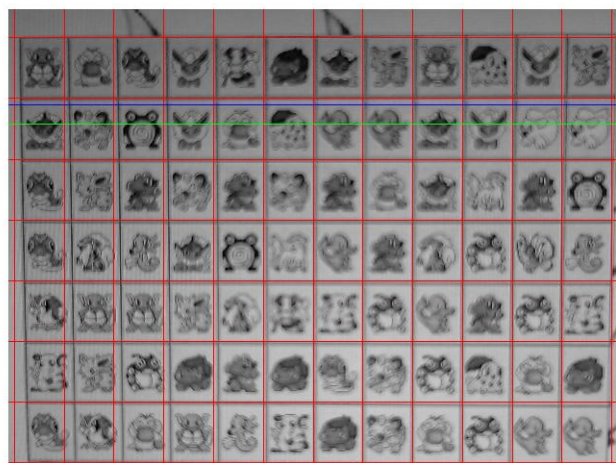


图 2.8: 灰度图像的分割结果

无需太认真的观测，即可发现其（和“信号与系统”有密切关系的）重要特征：周期性。下面我们就要用周期性对图像进行分割。

### 2.2.1.1 计算分块大小

在前述条件下，图像横、纵方向的周期即分别对应图像分块的宽和高。以横向为例，某两条水平线上的灰度值如图2.9 所示。

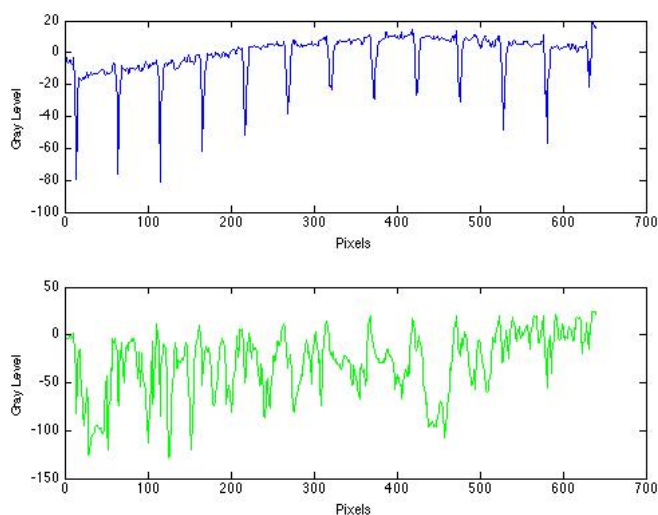


图 2.9: 某两条水平扫描线上的灰度值（按颜色分别对应于图2.8里的两条线，注意已调整直流分量）

某一条水平线的周期性强烈取决于该线上的内容：背景周期性明显，精灵则不明显。如果我们对所有水平线上的灰度值取均值（去除各行数据上的随机性影响，包括图像块的差异和摄像头的噪声），可得到图2.10（上）所示的波形，周期性明显且避免了偶然性因素。

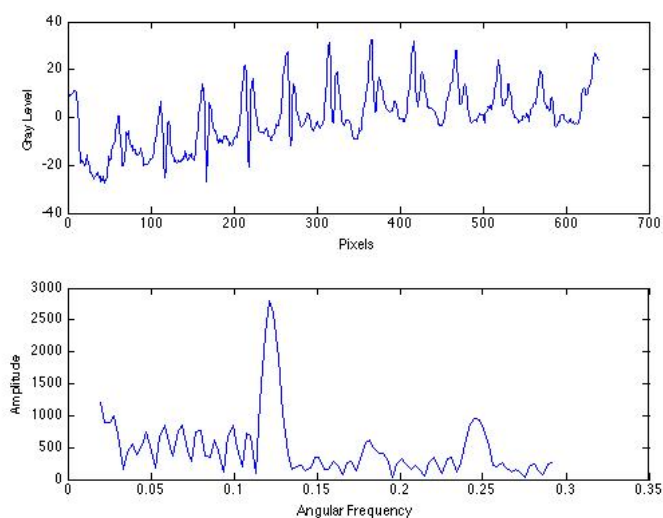


图 2.10: 水平扫描线的平均灰度值及其幅度谱

接下来利用傅里叶变换（具体可使用自定义的prefourier函数）得到其频谱如图2.10（下）所示（只画出了幅度谱），频谱峰值出现的位置对应其周期，即图像块宽度 $W_B = 51.78$ 。

### 2.2.1.2 确定左侧空白和上方空白的大小

我在“信号与系统”课上很少讲相位谱的用例，现在终于可以给一个了。这里又要用先验知识，即每个块的背景都是亮色（灰度值较大），而且精灵位于每个块的中心。图2.11中同时绘出了水平方向上灰度的均值及其基波，可见基波的最大值和图像块的边界对齐。现在想到如何计算左侧空白的大小了吗？对，用幅度谱峰值处对应的相位，可得到 $X_S = 6.87$ 像素点，如图2.11内左侧一条竖直线所示。

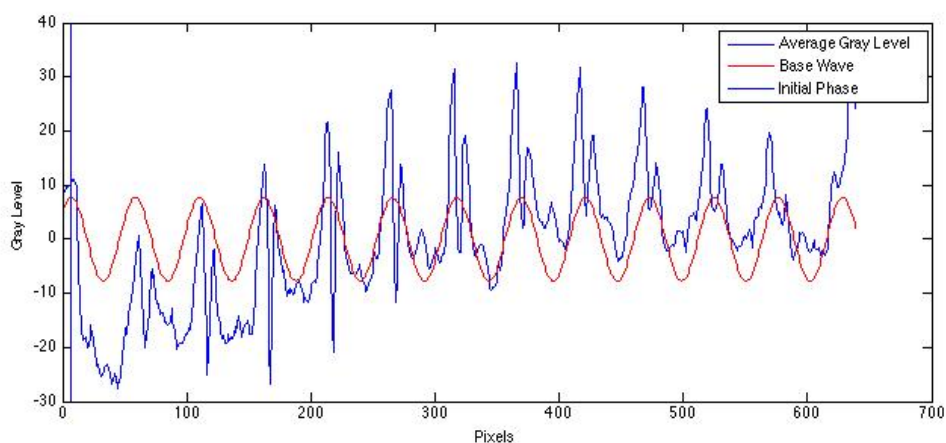


图 2.11: 用基波的相位判别分块偏移的原理

### 2.2.2 寻找相同的图像块

分割图像后要寻找相同的图像块。和第三次大作业的人脸检测类似，我们继续用“信号与系统”讲的内

积运算来度量图像分块的相似性。本次大作业将进一步利用匹配滤波技术展示内积运算在信号处理中的应用。

### 2.2.2.1 高通滤波

直观的看，每个图像分块区别于其他分块的重要特征是纹理。为了凸显纹理，可以用高通滤波器去除图像中缓慢变化的部分。为了设计一个二维的高通滤波器，我们可以简单地“旋转”一维滤波器的单位样值响应得到，如图2.12所示。

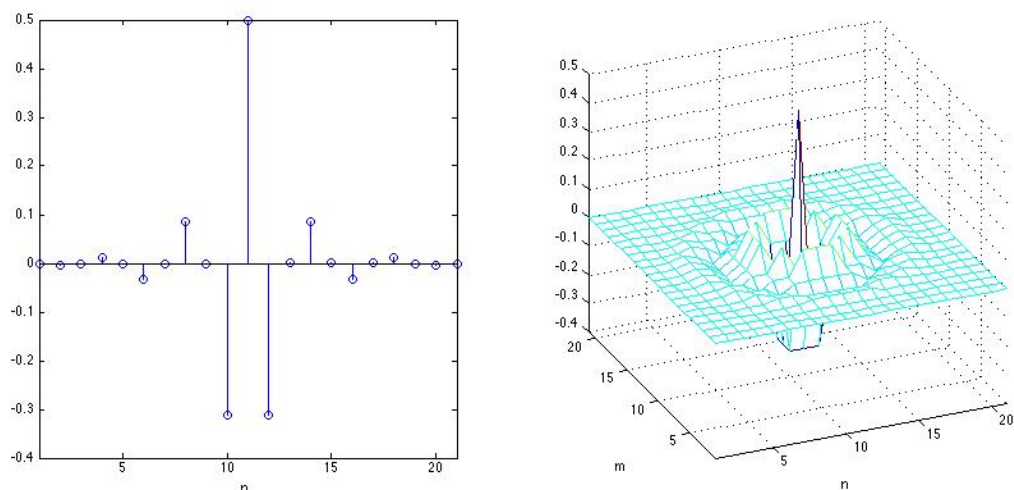


图 2.12: 一维和二维高通滤波器

高通滤波前后的效果如图2.13所示。

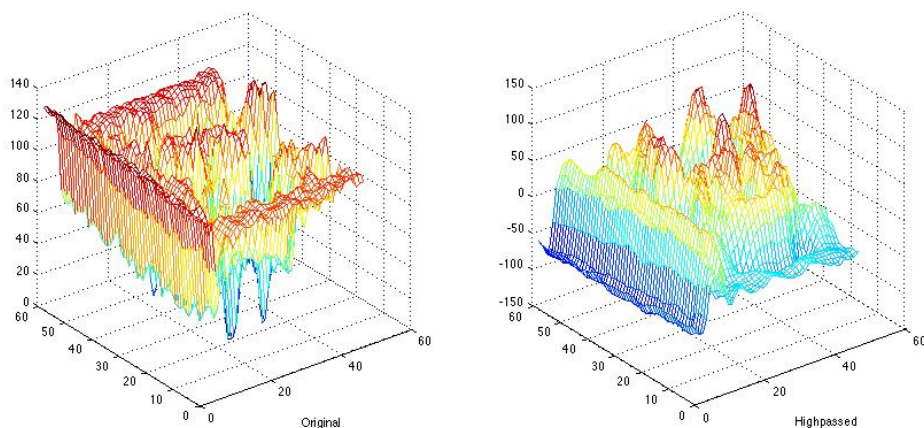


图 2.13: 高通滤波前后的数据



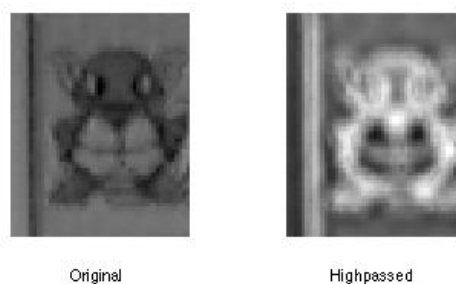


图 2.14: 高通滤波前后的图（为便于显示，高通后的图像加入了直流分量）

### 2.2.2.2 度量图像分块的相似性

回想课上讲的匹配滤波器，其核心操作是相关运算，就是把目标波形在接收信号上滑动，每滑动一格算一次内积（相关系数），在整个滑动范围寻找内积的最大值，用该值判定接收信号中是否存在目标波形；如果存在，最大值对应的位置就是目标波形出现的时间。

我们用完全相同的方法度量两个图像块的相似性。图2.15和图2.16分别展示了不同内容和相同内容的图像块的匹配滤波结果。

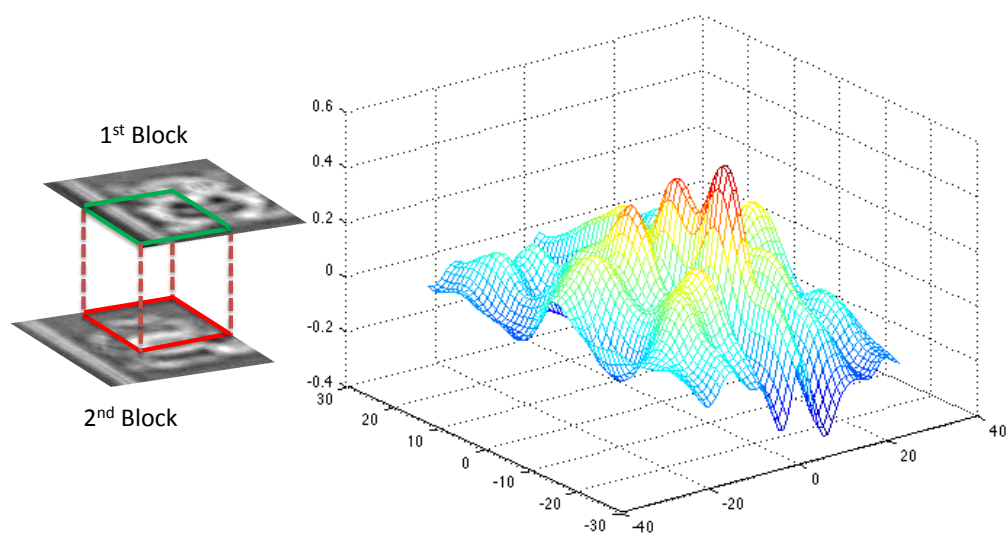


图 2.15: 不同内容图像块的二维匹配滤波示意图和结果（第1行的第1块和第2块）

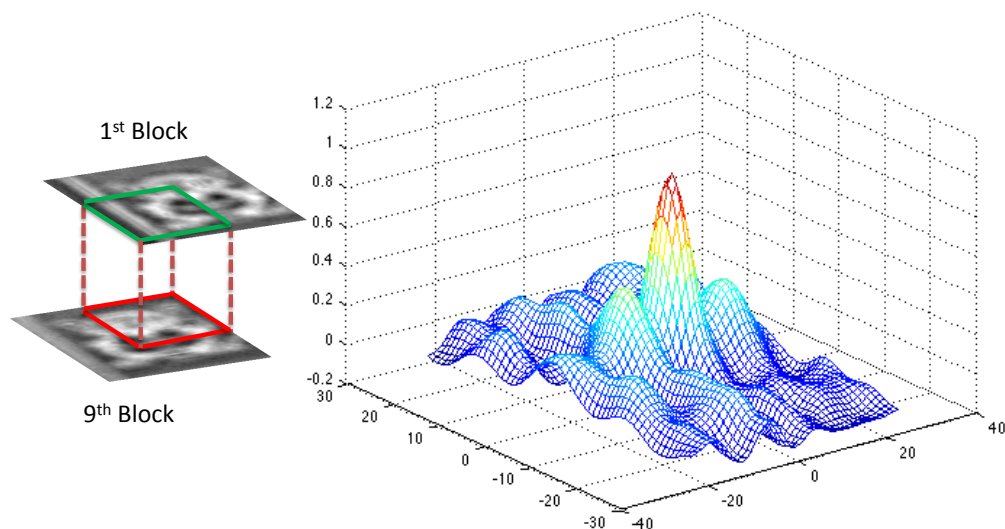


图 2.16: 相同内容图像块的二维匹配滤波示意图和结果（第1行的第1块和第9 块）

### 2.2.2.3 相似对排序或自动聚类

用上述方法度量所有图像分块两两之间的相似性。按照相似性由大到小排序，再逐一判别是否可消除（用上一章的方法），若可消除的话再控制鼠标点击，即完成了一次循环。逐次循环，直到把所有块消除。

还有一种思路是利用相似性度量对所有图像块进行自动聚类，一次就把原始图像映射为一个由索引值组成的数组。这样就把玩别人的连连看变成了玩自己开发的连连看，继而完全套用上一章的方法。

我们无法回避的一个问题是：判断错了怎么办？换一对试试呗……真人玩还有看错的时候呢！

## 第三节 其他模块

为了方便同学，我们为“获取图像”和“自动按键”模块提供了参考代码。由于这部分内容超出了本课程的要求，我们不做深入讲解，感兴趣的同学也可以自主学习和实现。

### 2.3.1 获取图像

请参考MATLAB的Image Acquisition Toolbox。需要注意的是MATLAB2013及以上版本调用摄像头需要安装support package。具体安装方法可参考本链接。

### 2.3.2 自动按键

代码在“process/reference/linkgame.q”中，具体语法请参考“按键精灵”用户手册，代码只要稍加修改就可以用到其他的连连看游戏中去。

## 第四节 练习题

本章习题可以用上述介绍的方法完成，也可以用其他任意方法。

1. 在MATLAB环境下，将路径设置到process文件夹下。对游戏区域的屏幕截图（灰度图像）graygroundtruth进行分割，提取出所有图像分块。在一个figure中用subplot方式按照原始顺序绘出所有图像分块。
2. 对摄像头采集的图像（灰度图像）graycapture，参考第1题要求进行处理。讨论：和干净图像相比，被噪声污染的图像给分块带来了什么样的困难？
3. 在第2题基础上，计算所有图像分块的两两相似性，选出最相似的十对图像块。在一个figure中绘出，并显示其相似性度量值。（建议先利用graygroundtruth测试代码的正确性，然后再用graycapture完成本题。）
4. 在第3题基础上，找出相似度最大却不是同一种精灵的十对图像块。在一个figure中绘出，并显示其相似性度量值。讨论：这个结果和你的主观感受一致吗？
5. 在第3题基础上，将游戏区域映射为索引值的数组，并列出索引值和图像分块的对照关系表。讨论：你可以将全部图像分块正确映射到其索引值吗？哪些分块无法正确映射？为什么？
6. 在上述工作基础上，设计实现一个模拟的自动连连看。对摄像头采集的图像（灰度图像）graycapture进行分块并找出最相似的一对可消除分块后，将这图片上两个块的位置设为黑色或其他特定颜色（即模拟消除操作），并将图片展示在figure上。然后继续找出下一对可消除的分块并模拟消除，直至消除所有的分块或找不到可消除的分块对。设计一种方法验证并展示上述工作的正确性。
7. （选做）对摄像头采集的彩色图像colorcapture，重做第2至6题。注意不能简单将彩色图变换为灰度图后直接处理。你的方法如何利用颜色信息？利用颜色信息是否可大幅提高正确率？（提示：1、图像处理大作业里讲过的彩色直方图；2、对X-Y-RGB三维张量做匹配滤波。）
8. （选做）在上述工作基础上，设计实现一个真实的自动连连看。
  - 主程序process/linkgame.p已经完成，它首先采集摄像头的图像，然后调用并把图像传递给ai.m，再根据ai.m的输出参数step中指定的位置，利用按键精灵模拟鼠标点击，完成一次操作。
  - ai.m是需要你补全的文件。它的功能是处理输入的图像realcapture，找出一对可消除分块，并把要消除的块的下标保存到step中输出（没有可消除的块则step = -1），输入输出格式在文件注释中有详细解释。
  - 完成本题需要使用一个分立的摄像头（并且要保证摄像头找到的绝大部分是游戏区域，否则你之前的程序可能无法处理）。主程序linkgame.p通过user\_camera.m实时获取摄像头的图像，后者的输出参数realcapture就是送给ai.m的输入参数。

虽然本题是选做题，不计入成绩，但我非常希望同学们能够独立完成本题。完成后请录制一段自动游戏的视频，上传到学堂，让大家一起分享你成功的喜悦！