连连看实验报告

郭一隆 (2013011189)

September 3, 2015

Contents

List of Figures

List of Tables

List of Source Codes

1 原创性

本次连连看大作业由本人独立思考完成，如有雷同，纯属巧合。

2 制作自己的连连看

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。

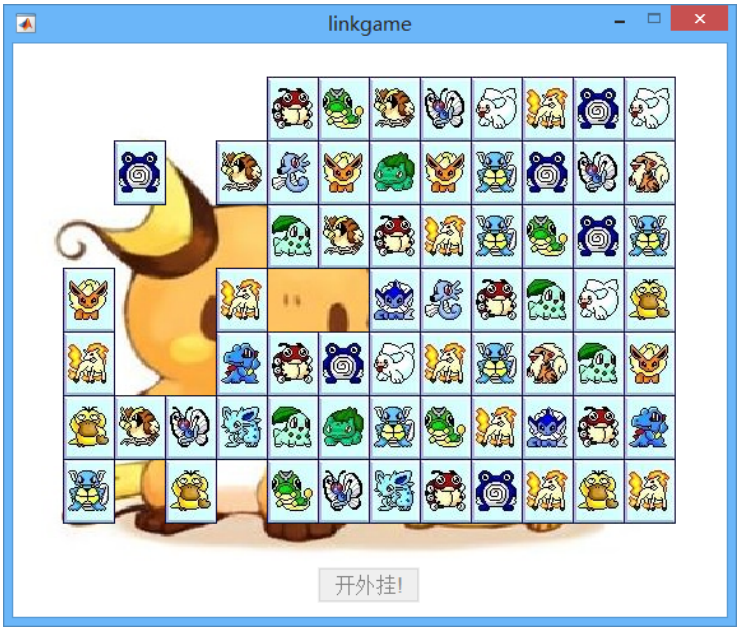

Figure 2.1: 运行连连看

2. 注意 linkgame 目录下有个detect.p，它的功能时检测块是否可以消除。现将其删掉，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。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。

   算法实现分为以下几个步骤：

   (a) **画十字 adjcross**：在给定点周围空白处画十字，遇到块则停止。

```
function [I,J] = adjcross(mtx,x,y)
% return index vector of can-reach blocks of mtx(x,y) on both directions
% notice: x >= 2 && y >= 2 && all(mtx >= 0)

% get vectors of four directions
left = mtx(x,1:y-1);
right = mtx(x,y+1:end);
up = mtx(1:x-1,y);
down = mtx(x+1:end,y);

% get zero run length adjacent to mtx(x,y)
lz = find(cumsum(left,'reverse')==0);
rz = find(cumsum(right,'forward')==0);
uz = find(cumsum(up,'reverse')==0);
dz = find(cumsum(down,'forward')==0);

I = [x*ones(1,length([lz,rz])),uz.',x+dz.'];
J = [lz,y+rz,y*ones(1,length([uz;dz]))];

end
```

Listing 2.1: `detect.m(adjcross)`

(b) **判断是否可直连 `canlink0`**：先判断横纵坐标是否在同一直线，再确认路径上是否有障碍。

```
function bool = canlink0(mtx,x1,y1,x2,y2)
% return 1 if it's a direct link (no turns)

if x1 == x2 && ~any(mtx(x1,min(y1,y2)+1:max(y1,y2)-1))
    bool = 1;
elseif y1 == y2 && ~any(mtx(min(x1,x2)+1:max(x1,x2)-1,y1))
    bool = 1;
else
    bool = 0;
end

end
```

Listing 2.2: `detect.m(canlink0)`

(c) **判断是否可用不超过一个直角的连线连接 `canlink1`**：选取两目标点之一作为起点，画十字；对十字上的点进行遍历，检查是否存在可与另一目标点直连的点。

```
function bool = canlink1(mtx,x1,y1,x2,y2)
% return 1 if the turns of link path <= 1

if canlink0(mtx,x1,y1,x2,y2)
    bool = 1;
    return
end

% grow the cross of origin
[I,J] = adjcross(mtx,x1,y1);

for n = 1:length(I)
    i = I(n); j = J(n);
    if canlink0(mtx,i,j,x2,y2)
        bool = 1;
        return
    end
end

bool = 0;

end
```

Listing 2.3: `detect.m(canlink1)`

(d) **判断可连性 canlink2**：选取两目标点之一作为起点，画十字；对十字上的点进行遍历，检查是否存在可与另一目标用不超过一个直角的连线连接的点。

```
function bool = canlink2(mtx,x1,y1,x2,y2)
% return 1 if these two blocks can link!

if canlink1(mtx,x1,y1,x2,y2)
    bool = 1;
    return
end

% grow the cross of origin
[I,J] = adjcross(mtx,x1,y1);

for n = 1:length(I)
    i = I(n); j = J(n);
    if canlink1(mtx,i,j,x2,y2)
        bool = 1;
        return
    end
```

```
116  end
117
118  bool = 0;
119
120  end
```

Listing 2.4: detect.m(canlink2)

再次运行 linkgame，detect.m功能正常。

```
1   function bool = detect(mtx, x1, y1, x2, y2)
2
3       [m,n] = size(mtx);
4
5       % add surrounding zeros
6       mtx = [0,zeros(1,n),0;
7           zeros(m,1),mtx,zeros(m,1);
8           0,zeros(1,n),0];
9
10      origin = mtx(x1+1,y1+1);
11      target = mtx(x2+1,y2+1);
12
13      if origin == target && canlink2(mtx,x1+1,y1+1,x2+1,y2+1)
14          bool = 1;
15      else
16          bool = 0;
17      end
18
19  end
```

Listing 2.5: detect.m(main)

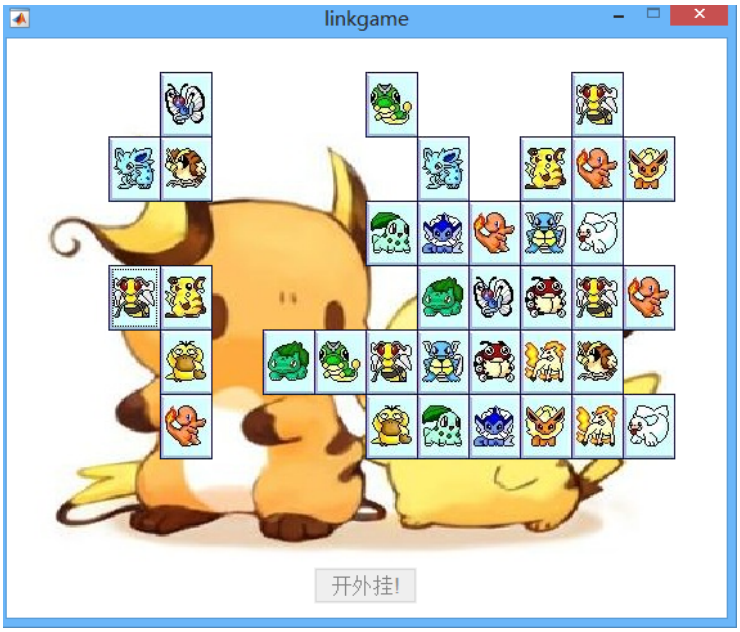

Figure 2.2: 重写 `detect.m` 后运行 `linkgame`