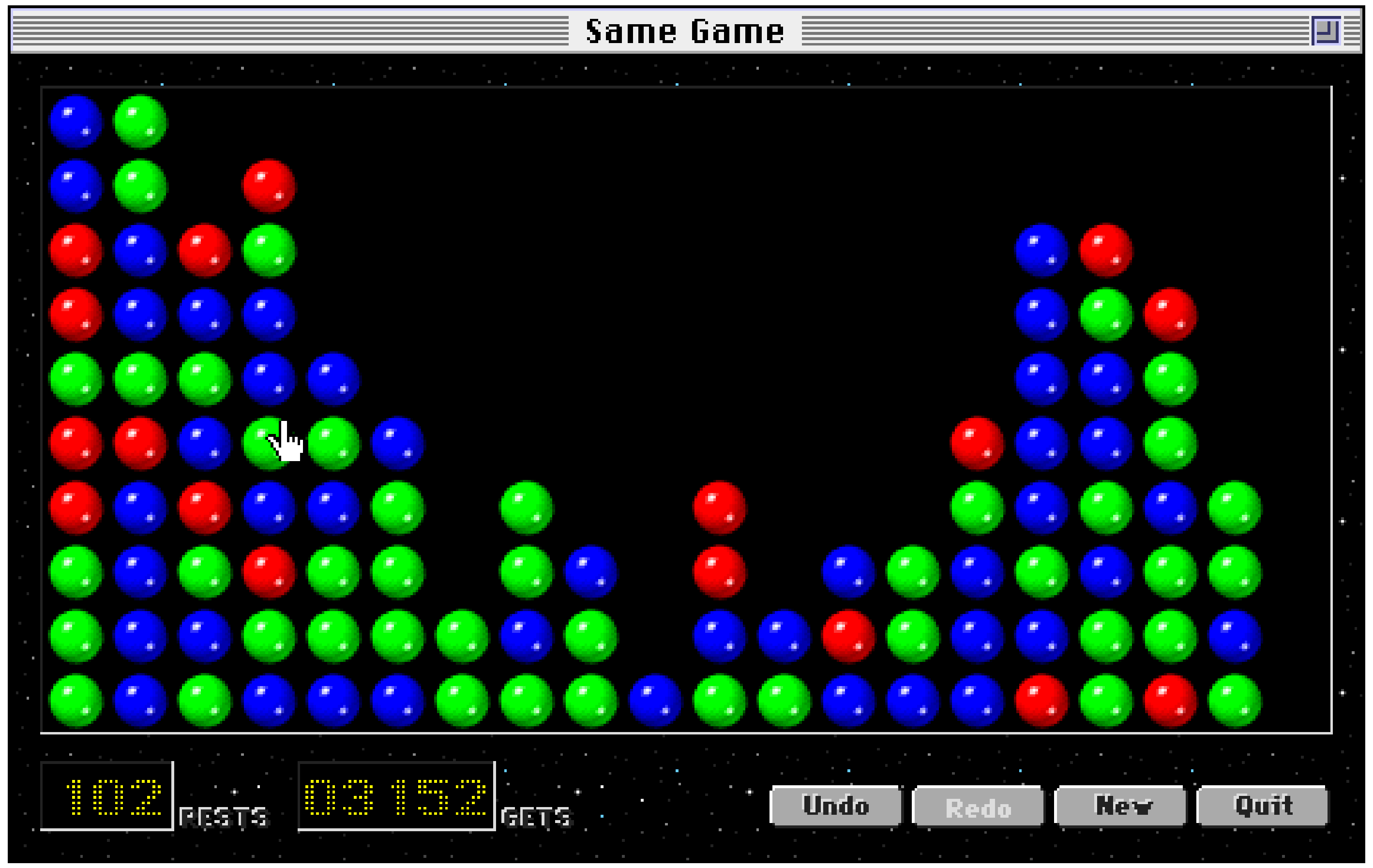
Final Defense

Tile-Matching Game Impletation using Matlab

Background

Background

A tile-matching video game is a type of puzzle video game where the player manipulates tiles in order to make them disappear according to a matching criterion. In many tile-matching games, that criterion is to place a given number of tiles of the same type so that they adjoin each other. That number is often three, and these games are called match-three games.



Objectives

Objectives

Our project aims to develop a match-two-type tile-matching game with a user-friendly graphical interface.

✓ **Basic Game**

Dvelop a game where two tiles can be eliminated after matching. The program can initialize chess board, deal with every step, and check whether the game ends.

✓ **User Interface**

Develop a user-friendly graphical interface full of colorful tiles. The UI should be user-friendly and easy-to-use.

✓ **Advanced**

Implement a solver tool that can automatically display the solution.

Implementation

Game Logic Implementation

Initialization

In the initialization, we take parameters about the chess and the game board. We randomly positions of different same-tile pairs. And if there is no conflicts, we put the pairs on the board.

Deal with Steps

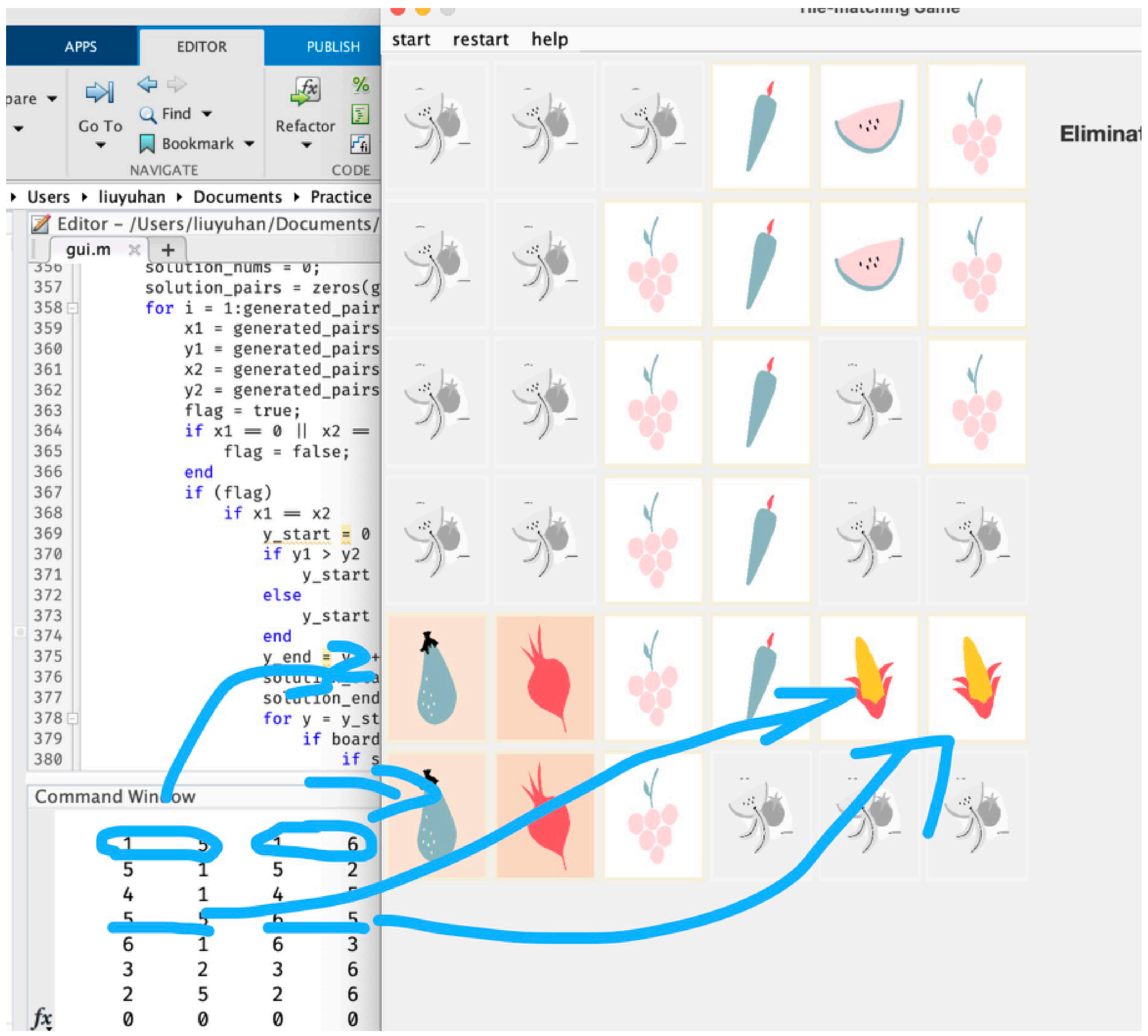
To deal with every step in the game, we checked whether the step is valid by checking whether they are in the same columns or rows. If so, we eliminate the chess.

Check Game Over

We use a loop to check every chess. If there is no chess pairs that can be solved, we output that the game is over. Otherwise the game is still alive. We also check whether there is single chess.

Solver

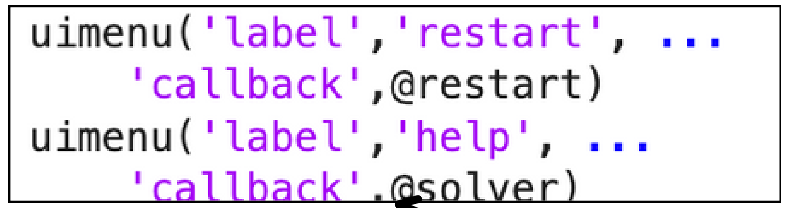
Since we know all the information about the generated same tile pairs, we can use a loop to check whether there are still chance to solve this pair. In this way, we can output the pairs that can be solved in the correct way without interrupting solving other pairs.



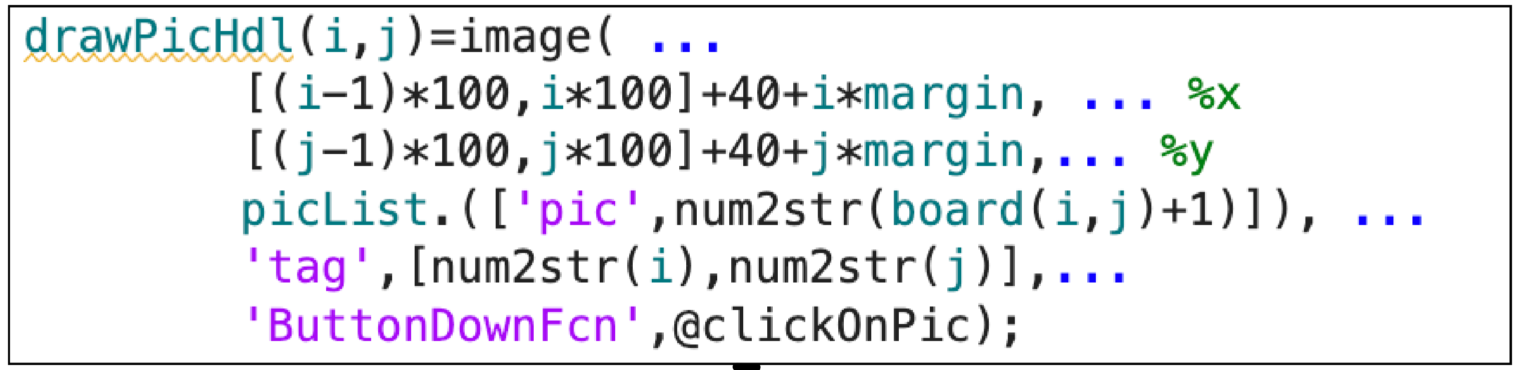
```
    solution_nums = 0;
    solution_pairs = zeros(g
    for i = 1:generated_pair
        x1 = generated_pairs
        y1 = generated_pairs
        x2 = generated_pairs
        y2 = generated_pairs
        flag = true;
        if x1 == 0 || x2 ==
            flag = false;
        end
        if (flag)
            if x1 == x2
                y_start = 0
                if y1 > y2
                    y_start
                else
                    y_start
                end
                y_end = y
                soluti
                solution_end
                for y = y_st
                    if board
                        if s
```

```
     1     5     1     6
     5     1     5     2
     4     1     4     5
     5     5     6     5
     6     1     6     3
     3     2     3     6
     2     5     2     6
     0     0     0     0
```

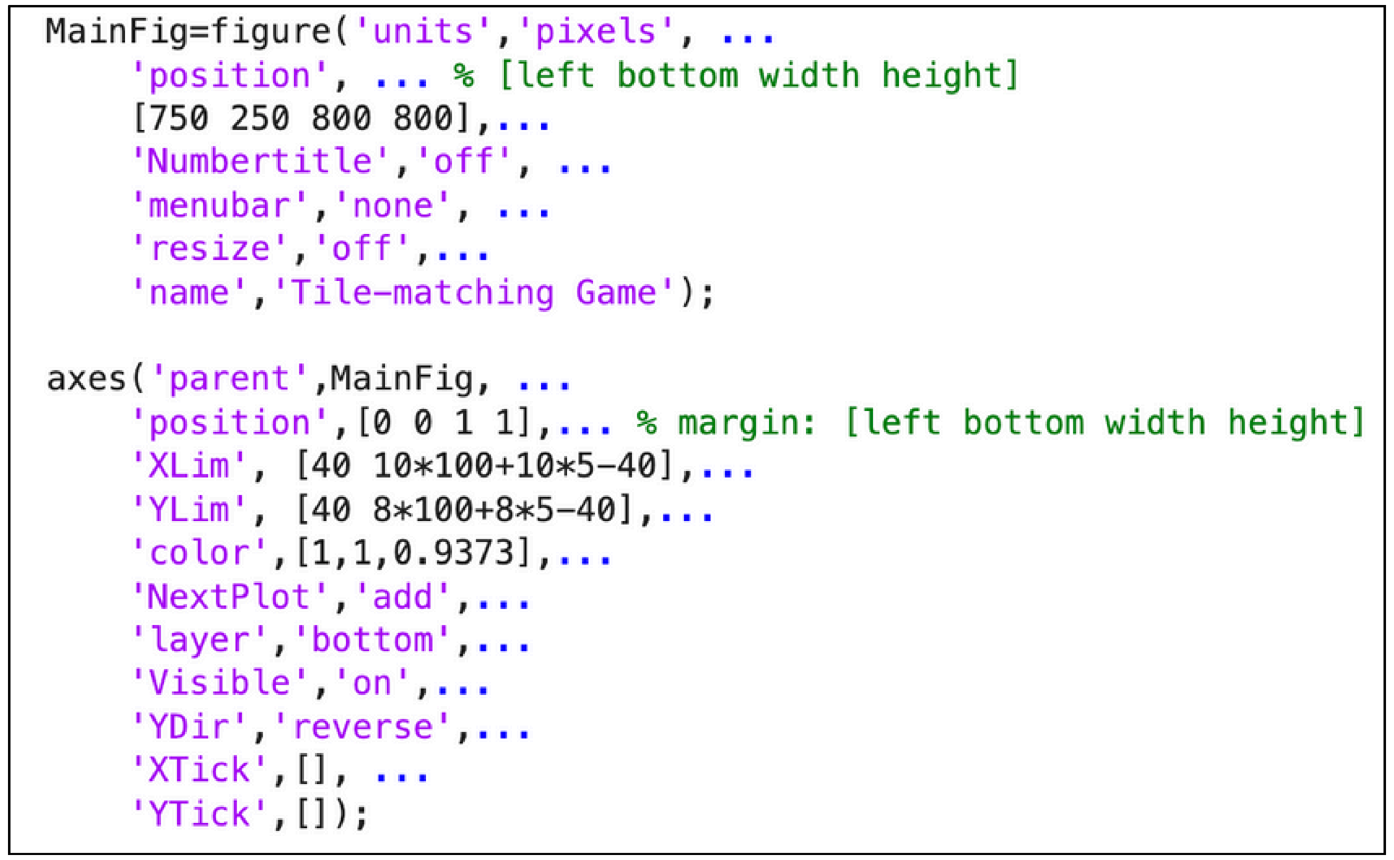
GUI



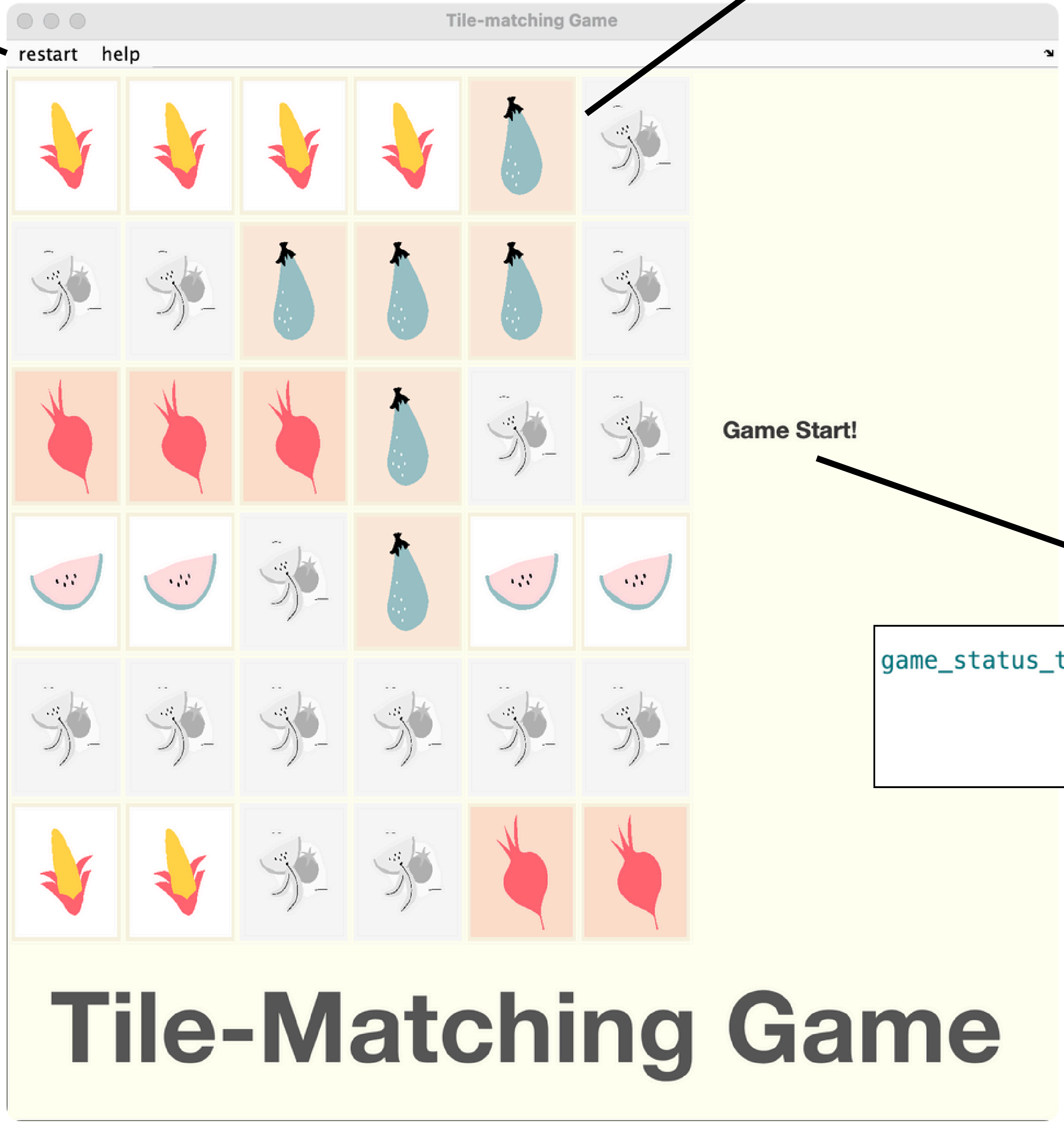
```
uimenu('label','restart', ...
    'callback',@restart)
uimenu('label','help', ...
    'callback',@solver)
```

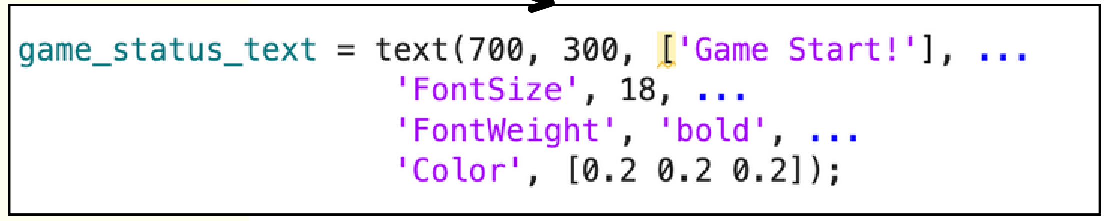


```
drawPicHdl(i,j)=image( ...
        [(i-1)*100,i*100]+40+i*margin, ... %x
        [(j-1)*100,j*100]+40+j*margin,... %y
        picList.(['pic',num2str(board(i,j)+1)]), ...
        'tag',[num2str(i),num2str(j)],...
        'ButtonDownFcn',@clickOnPic);
```



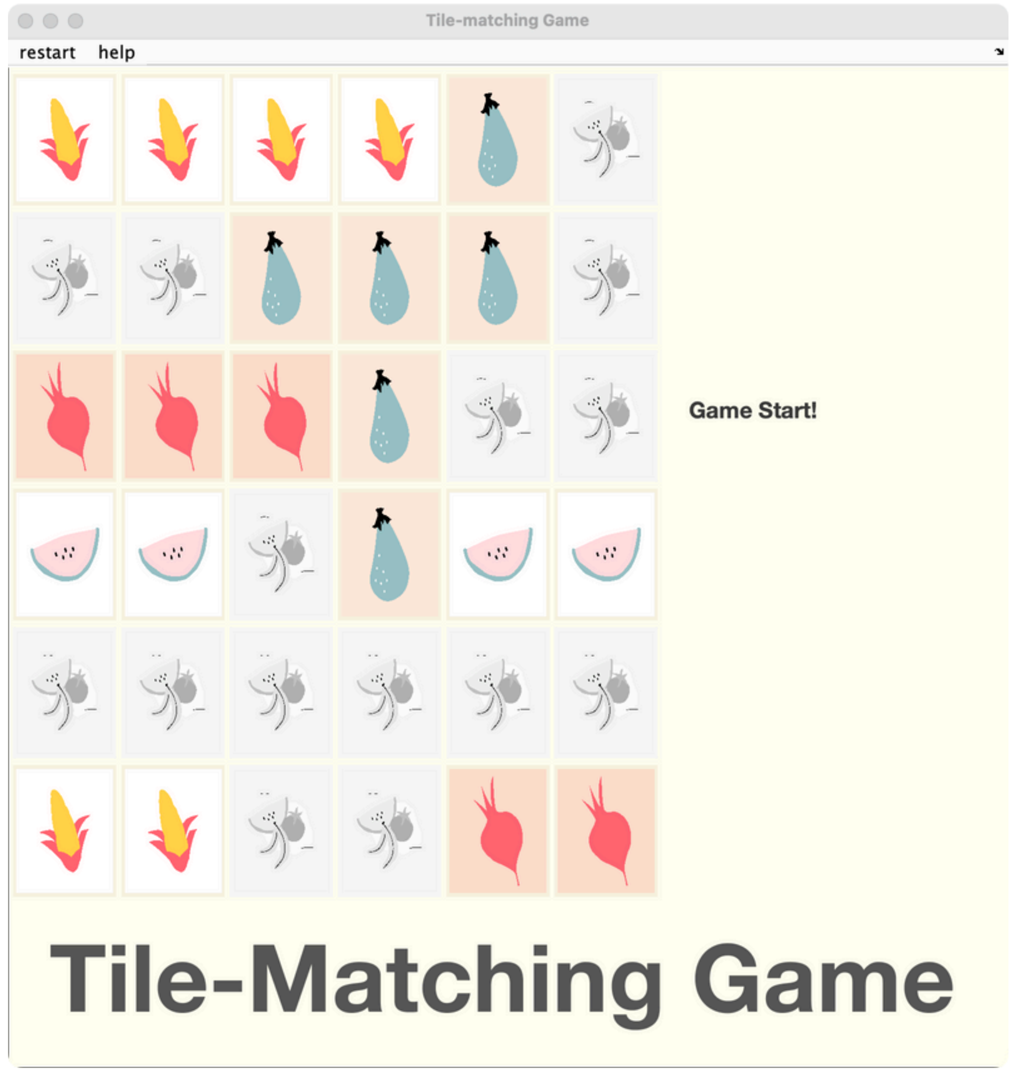
```
 MainFig=figure('units','pixels', ...
     'position', ... % [left bottom width height]
     [750 250 800 800],...
     'Numbertitle','off', ...
     'menubar','none', ...
     'resize','off',...
     'name','Tile-matching Game');

 axes('parent',MainFig, ...
     'position',[0 0 1 1],... % margin: [left bottom width height]
     'XLim', [40 10*100+10*5-40],...
     'YLim', [40 8*100+8*5-40],...
     'color',[1,1,0.9373],...
     'NextPlot','add',...
     'layer','bottom',...
     'Visible','on',...
     'YDir','reverse',...
     'XTick',[], ...
     'YTick',[]);
```





```
game_status_text = text(700, 300, ['Game Start!'], ...
                   'FontSize', 18, ...
                   'FontWeight', 'bold', ...
                   'Color', [0.2 0.2 0.2]);
```

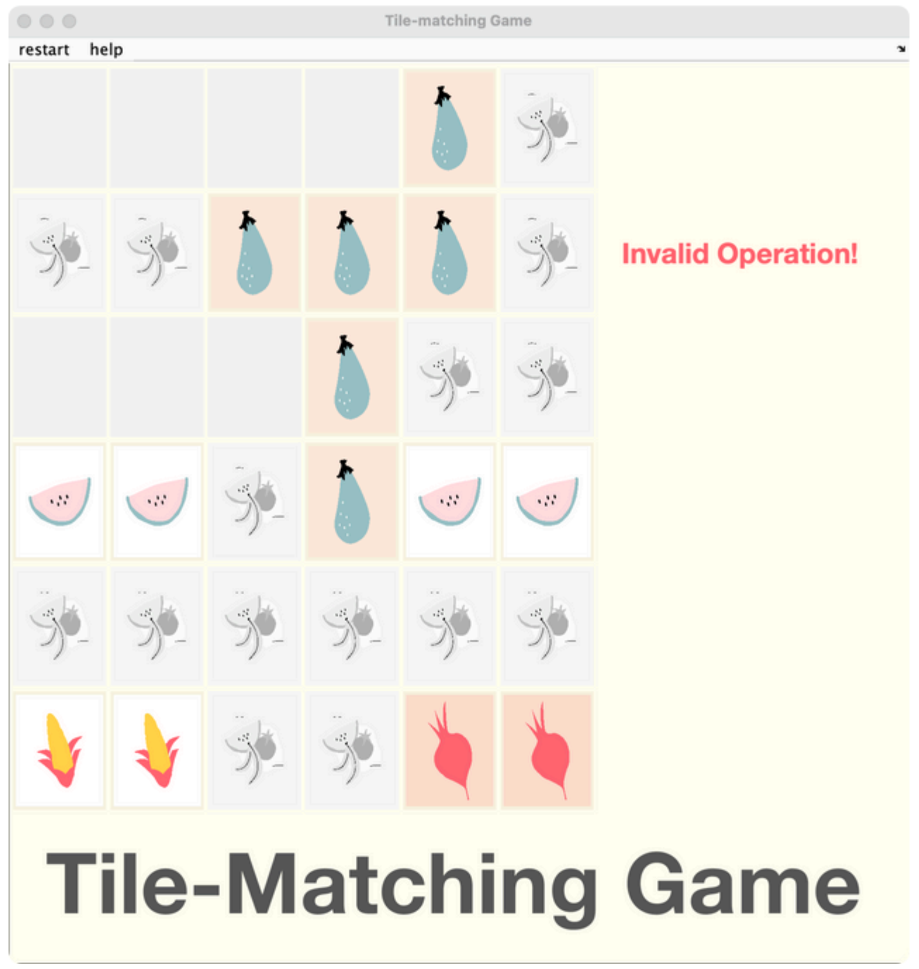
GUI



Game Start!

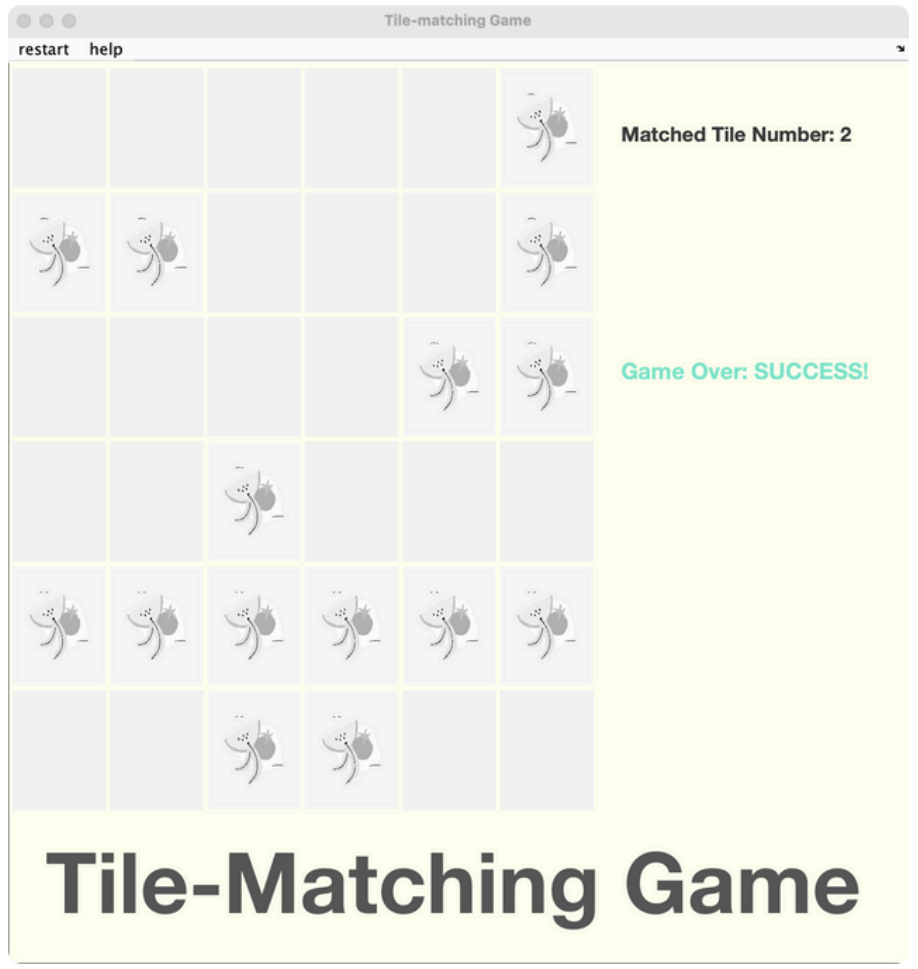
Tile-Matching Game

When playing the game, after every match, there will be some messages showing the game status. All messages of the game status will be displayed in the right part.



Invalid Operation!

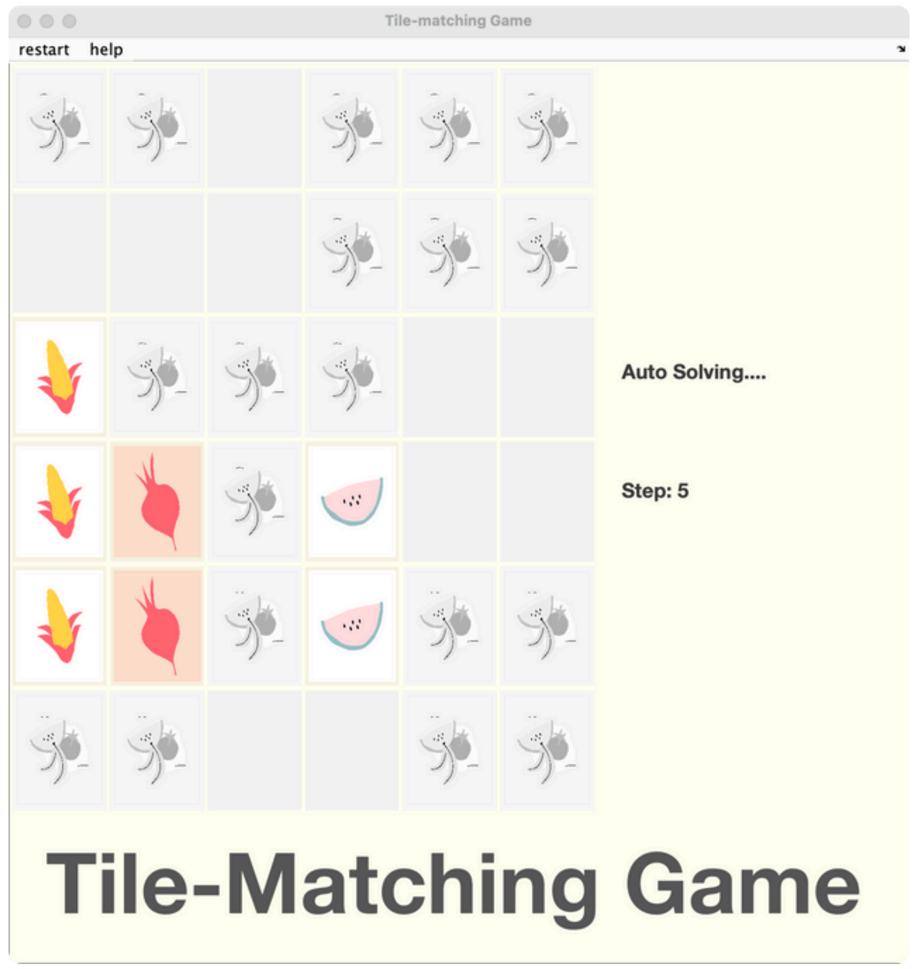
Tile-Matching Game



Matched Tile Number: 2

Game Over: SUCCESS!

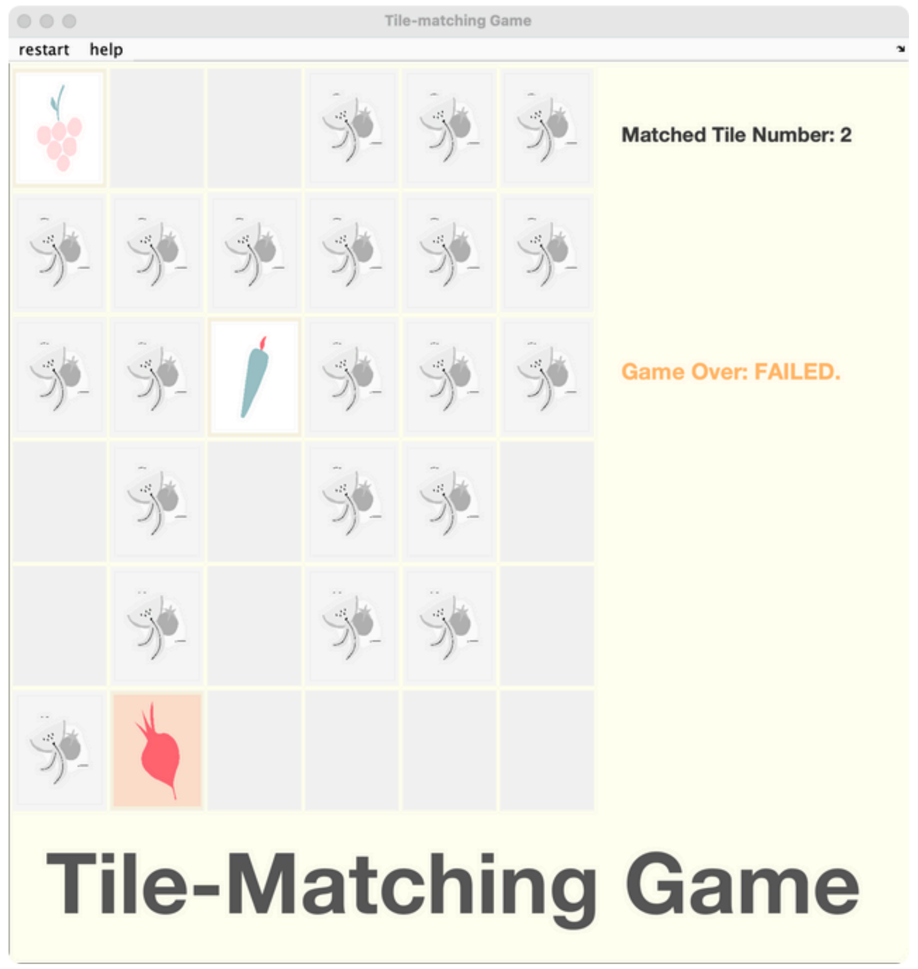
Tile-Matching Game



Auto Solving....

Step: 5

Tile-Matching Game

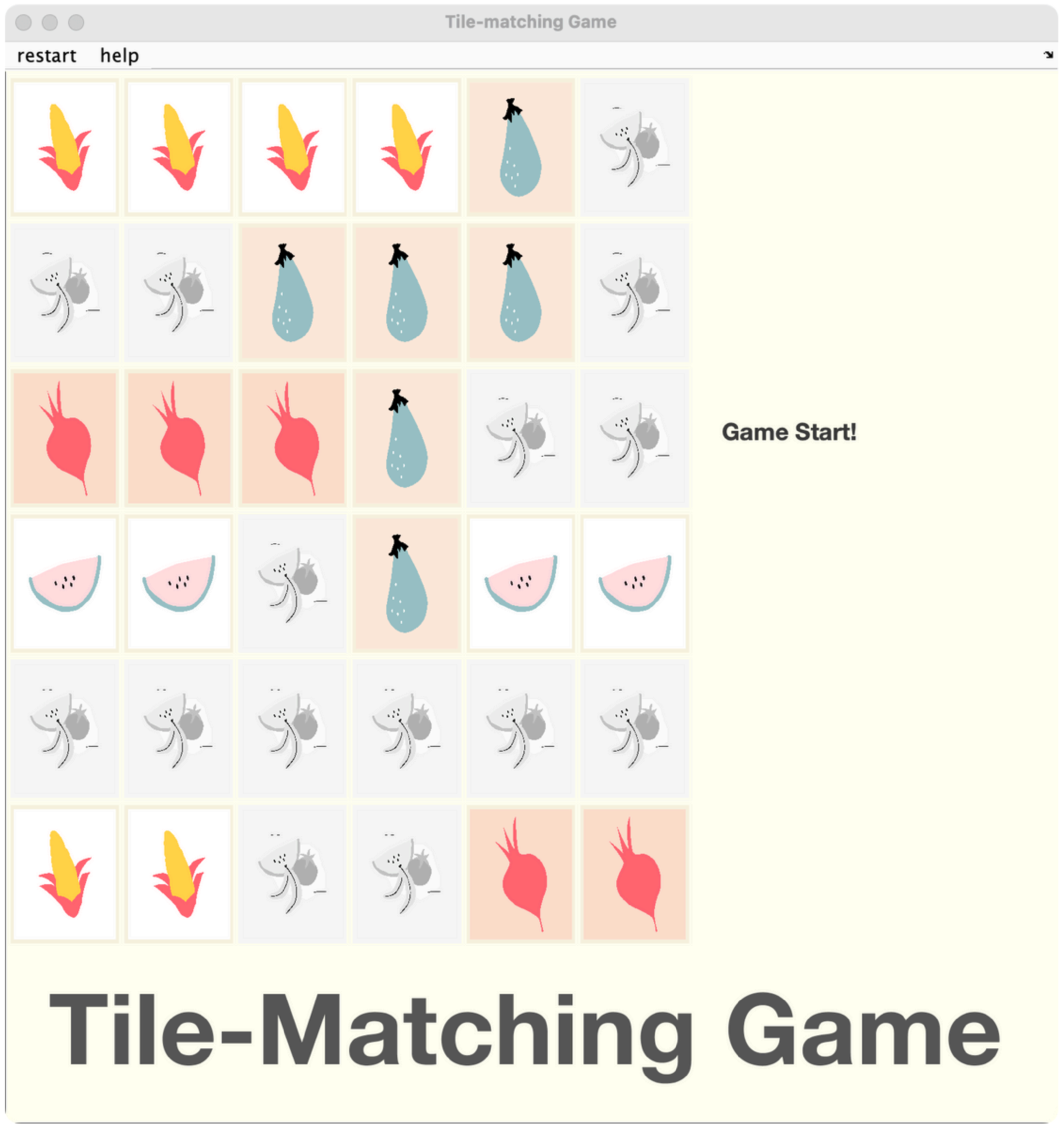


Matched Tile Number: 2

Game Over: FAILED.

Tile-Matching Game

Demo Presentation



Tile-Matching Game

Contribution

Name	Responsibilities
Fangmin Hou	Display; UI/UX Design and Implementation
Yuhan Liu	Game Logic and Solver Tool Implementation

Thanks for Listening