数字图像分析第五次实验报告

院　　系：信息科学技术学院

学　　号：SA19023005

姓　　名：景军元

授课教师：李厚强、周文罡

课程助教：欧阳剑波、刘一衡、林丰

实验名称：局部特征匹配及几何校验

一、基本原理

1、SIFT 描述子

（1）方向归一化

方向旋转不变：主方向（dominant orientation）对齐：

根据 image patch 的梯度主方向，对 patch 进行旋转，将 patches 修正到一个 canonical orientation（主方向），该主方向用于对 image patch 进行旋转对齐。

（2）描述子

将 patch 基于主方向旋转对齐后，将其均分为$k \times k$的子块，对每个子块，采用基于梯度方向的直方图（8 维）表示特征，最后 SIFT 描述子维度：$k \times k \times 8 = 128\ (k = 4)$。

2、局部视觉特征匹配

基于距离比值的准则（Distance ratio criterion）：

给定一幅图像的 SIFT 特征，计算它和另一幅图像的所有 SIFT 特征之间的距离。如果最小距离和第二小的距离之间的比值小于 0.80，就认为这两个特征之间进行了正确的匹配。

3、基于 Spatial Coding 的几何校验

（1）建立空间编码映射

考虑已匹配特征之间的相对位置关系，有：

$$GX(i,j) = \begin{cases} 0, & x_i < x_j \\ 1, & x_i \geq x_j \end{cases} \quad GY(i,j) = \begin{cases} 0, & y_i < y_j \\ 1, & y_i \geq y_j \end{cases}$$

（2）比较两图匹配点的空间编码，计算不一致度：

$$V_x(i,j) = GX_1(i,j) \oplus GX_2(i,j)$$
$$V_y(i,j) = GY_1(i,j) \oplus GY_2(i,j)$$

（3）查找并删除几何位置最失配的点，直到几何完全匹配：

$$S_x(i) = \sum_{j=1}^{N} V_x(i,j)$$

$$S_y(i) = \sum_{j=1}^{N} V_y(i,j)$$

二、实验内容

1、问题重述

（1）给定一对图像，利用提取好的 SIFT 特征文件，根据 distance ratio criterion 计算图像间的局部特征匹配关系；

（2）基于上述初步匹配结果，实现 spatial coding 方法，进行匹配校验，确定几何不一致的匹配；

（3）将几何一致的匹配和不一致的匹配在图像上画出来，分别用绿色和红色进行区分。

2、函数实现

（1）spatialCoding.m

①源代码

```
%% Display SIFT features of two images
clear;
close all;
clc;

%% Load two images and their SIFT features
src_1 = './test images/Mona-Lisa-73.jpg';
src_2 = './test images/Mona-Lisa-489.jpg';
ext1 = '.sift';    % extension name of SIFT file
ext2 = '.sift';    % extension name of SIFT file
siftDim = 128;

%% load image
im_1 = imread(src_1);
im_2 = imread(src_2);

%% load SIFT feature
% SIFT
% feature 文件格式：binary 格式，开头四个字节（int）为特征数目，后面逐个为 SIFT
特征结构体，每个 SIFT 特征包含 128D 的描述子（128 个字节）
%  和  [x, y, scale, orientation]的 16 字节的位置、尺度和主方向信息（float）

featPath_1 = [src_1, ext1];
featPath_2 = [src_2, ext2];

fid_1 = fopen(featPath_1, 'rb');
featNum_1 = fread(fid_1, 1, 'int32'); %  文件中 SIFT 特征的数目
SiftFeat_1 = zeros(siftDim, featNum_1);
paraFeat_1 = zeros(4, featNum_1);
for i = 1 : featNum_1 %  逐个读取 SIFT 特征
      SiftFeat_1(:, i) = fread(fid_1, siftDim, 'uchar'); %先读入 128 维描述子
      paraFeat_1(:, i) = fread(fid_1, 4, 'float32');         %再读入[x, y, scale, orientation]信息
end
fclose(fid_1);

fid_2 = fopen(featPath_2, 'rb');
featNum_2 = fread(fid_2, 1, 'int32'); %  文件中 SIFT 特征的数目
SiftFeat_2 = zeros(siftDim, featNum_2);
paraFeat_2 = zeros(4, featNum_2);
for i = 1 : featNum_2 %  逐个读取 SIFT 特征
```

```
    SiftFeat_2(:, i) = fread(fid_2, siftDim, 'uchar'); %先读入 128 维描述子
    paraFeat_2(:, i) = fread(fid_2, 4, 'float32');        %再读入[x, y, scale, orientation]信息
end
fclose(fid_2);

%% normalization
SiftFeat_1 = SiftFeat_1 ./ repmat(sqrt(sum(SiftFeat_1.^2)), size(SiftFeat_1, 1), 1);
SiftFeat_2 = SiftFeat_2 ./ repmat(sqrt(sum(SiftFeat_2.^2)), size(SiftFeat_2, 1), 1);

%% Display SIFT feature matching on RGB image
[row, col, cn] = size(im_1);
[r2, c2, n2] = size(im_2);
imgBig = 255 * ones(max(row, r2), col + c2, 3);
imgBig(1 : row, 1 : col, :) = im_1;
imgBig(1 : r2, col + 1 : end, :) = im_2; %%  将两幅图像拼成了一副大图像，左右排列
paraFeat_2(1, :) = paraFeat_2(1, :) + col; %  第二幅图像中的 SIFT feature 的列坐标要修改
mtchdPnt1=[];
mtchdPnt2=[];
for i = 1 : featNum_1
    feat = sqrt(sum((SiftFeat_2 - SiftFeat_1(:, i)).^2));
    [B, I] = sort(feat);
    if B(1)/B(2) < 0.8
        mtchdPnt1 = [mtchdPnt1; i];
        mtchdPnt2 = [mtchdPnt2; I(1)];
    end
end
hold off;
GX1 = zeros(size(mtchdPnt1,1),size(mtchdPnt1,1));
GY1 = zeros(size(mtchdPnt1,1),size(mtchdPnt1,1));
for i = 1 : size(mtchdPnt1, 1)
    for j = i+1 : size(mtchdPnt1, 1)   %根据几何关系，GX 和 GY 均为反对称，因此只需计
算一半
        if paraFeat_1(1, mtchdPnt1(i))>=paraFeat_1(1, mtchdPnt1(j))
            GX1(i,j) = 1;
        else
            GX1(j,i) = 1;
        end
        if paraFeat_1(2, mtchdPnt1(i))>=paraFeat_1(2, mtchdPnt1(j))
            GY1(i,j) = 1;
        else
            GY1(j,i) = 1;
        end
    end
end
```

```
GX2 = zeros(size(mtchdPnt2,1),size(mtchdPnt2,1));
GY2 = zeros(size(mtchdPnt2,1),size(mtchdPnt2,1));
for i = 1 : size(mtchdPnt2, 1)
    for j = i+1 : size(mtchdPnt2, 1)
        if paraFeat_2(1, mtchdPnt2(i))>=paraFeat_2(1, mtchdPnt2(j))
            GX2(i,j) = 1;
        else
            GX2(j,i) = 1;
        end
        if paraFeat_2(2, mtchdPnt2(i))>=paraFeat_2(2, mtchdPnt2(j))
            GY2(i,j) = 1;
        else
            GY2(j,i) = 1;
        end
    end
end
VX = xor(GX1,GX2);
VY = xor(GY1,GY2);
SX = sum(VX,2);
SY = sum(VY,2);
unmtchdPnt1 = [];
unmtchdPnt2 = [];
tag = 0;
while sum(SX)~=0 || sum(SY)~=0
    if tag
        [M, I] = max(SX);
    else
        [M, I] = max(SY);
    end
    VX(I,:) = [];
    VX(:,I) = [];
    VY(I,:) = [];
    VY(:,I) = [];
    unmtchdPnt1 = [unmtchdPnt1; mtchdPnt1(I)];
    unmtchdPnt2 = [unmtchdPnt2; mtchdPnt2(I)];
    mtchdPnt1(I) = [];
    mtchdPnt2(I) = [];
    SX = sum(VX,2);
    SY = sum(VY,2);
    tag=~tag;
end
figure(1); imshow(uint8(imgBig)); axis on;
hold on;
for i = 1 : size(mtchdPnt1,1)
```

```
        figure(1);
        hold on;
        plot([paraFeat_1(1, mtchdPnt1(i)), paraFeat_2(1, mtchdPnt2(i))], [paraFeat_1(2,
mtchdPnt1(i)), paraFeat_2(2, mtchdPnt2(i))], 'g');
end
for i = 1 : size(unmtchdPnt1,1)
        figure(1);
        hold on;
        plot([paraFeat_1(1, unmtchdPnt1(i)), paraFeat_2(1, unmtchdPnt2(i))], [paraFeat_1(2,
unmtchdPnt1(i)), paraFeat_2(2, unmtchdPnt2(i))], 'r');
end
hold off;
```

②函数说明

首先，根据 distance ratio criterion，设置阈值为 0.8，计算所有匹配点。之后分别计算两张图的 GX 和 GY，由于几何对称性，矩阵反对称，因此只需计算一半的值。最后迭代删除几何不匹配的点，直到完全匹配。

三、实验结果及分析

1、相关图像对

先考虑具有一定相关性的图像对之间的匹配，如图 1 所示为 Mona-Lisa-73 和 Mona-Lisa-489 的匹配情况，图中共有 40 条匹配，其中 25 条为几何一致的匹配，以绿色标注；15 条为几何不一致的匹配，以红色标注。

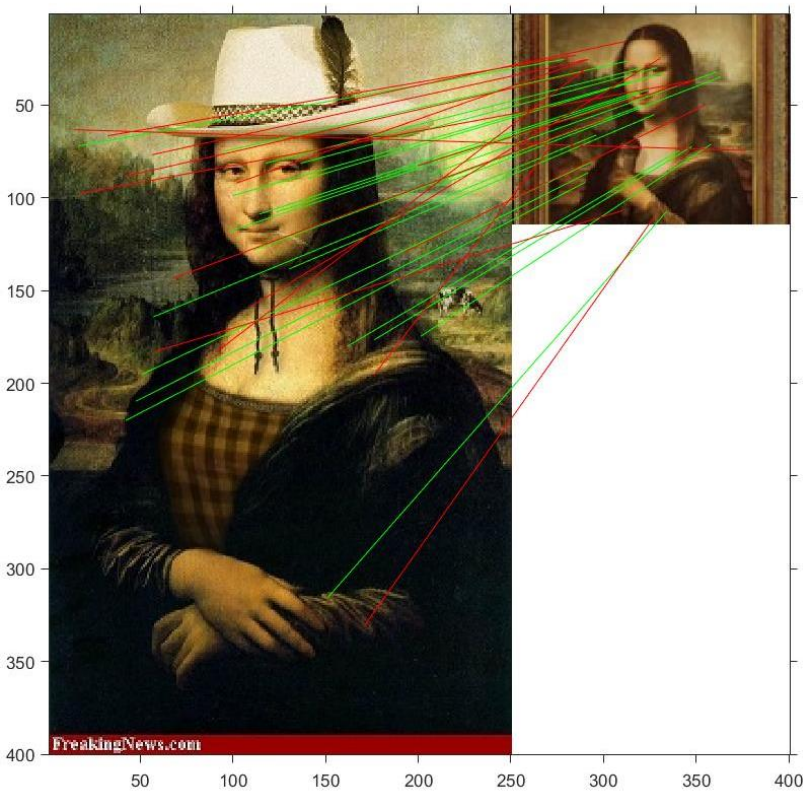

图 1. Mona-Lisa-73 与 Mona-Lisa-489 匹配结果

如图 2 所示为 Mona-Lisa-73 和 Mona-Lisa-452 的匹配结果，共有 69 条匹配，其中 47 条几何一致，以绿色标注；22 条几何不一致，以红色标注。

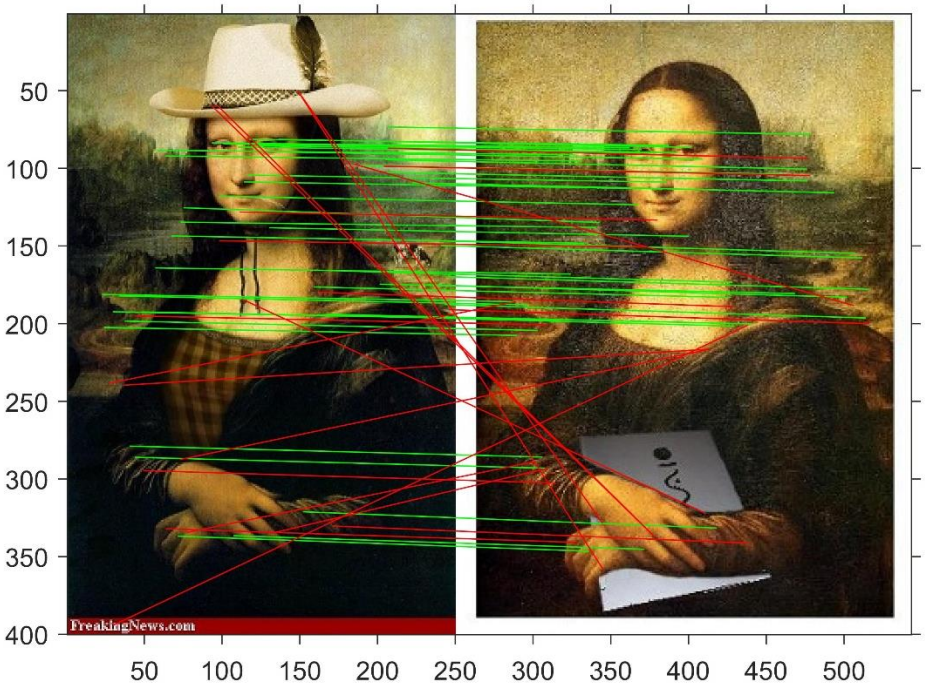

图 2. Mona-Lisa-73 与 Mona-Lisa-452 匹配结果

如图 3 所示为 Mona-Lisa-73 和 Mona-Lisa-653 的匹配结果，共有 72 条匹配，其中 54 条几何一致，以绿色标注；18 条几何不一致，以红色标注。

图 3. Mona-Lisa-73 与 Mona-Lisa-653 匹配结果

上述图片的匹配效果均较好，可以看出，在图片相似度较高时，该方法能获得一个不错的匹配结果。

接下来再做几对相关图像对的匹配，如图 4 所示为 Apollo-18 和 Apollo-49 的匹配结果，共有 102 条匹配，其中 37 条几何一致，以绿色标注；65 条几何不一致，以红色标注。

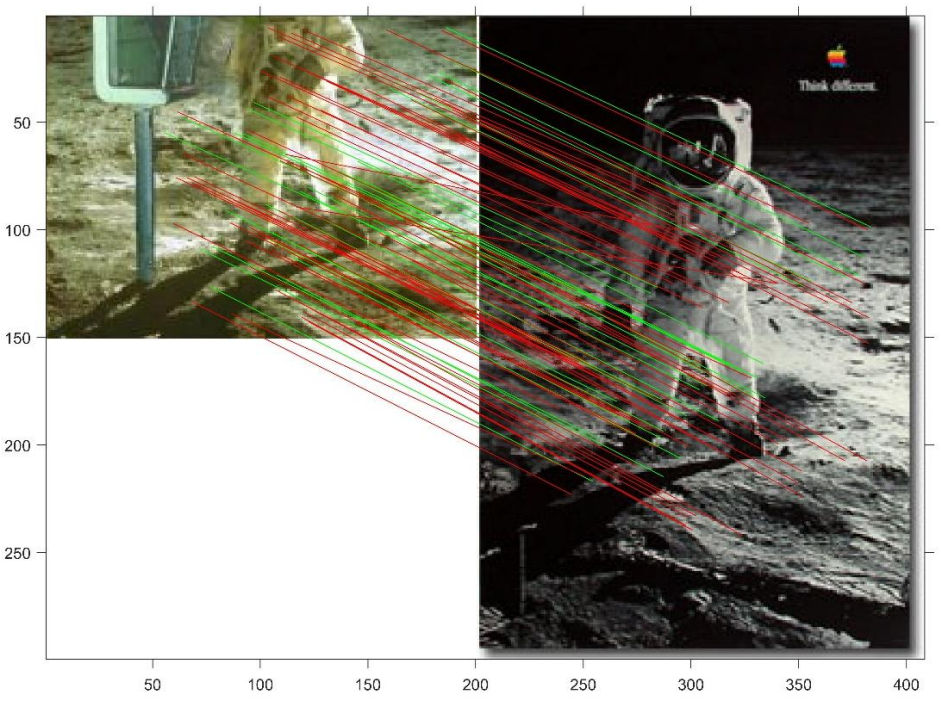

图 4. Apollo-18 与 Apollo-49 匹配结果

如图 5 所示为 Apollo-49 和 Apollo-266 的匹配结果，共有 25 条匹配，其中 17 条几何一致，以绿色标注；8 条几何不一致，以红色标注。

图 5. Apollo-49 与 Apollo-266 匹配结果

可以看出，对于大图和大图的局部匹配，在色彩和对比度差异较大时，匹配结果并不理想，但是色彩接近时还是能获得一个较好的匹配结果。

再看几组匹配结果，如图 6 所示为 Disney-00524 和 Disney-00550 的匹配结果，共有 24 条匹配，其中 8 条几何一致，以绿色标注；16 条几何不一致，以红色标注。

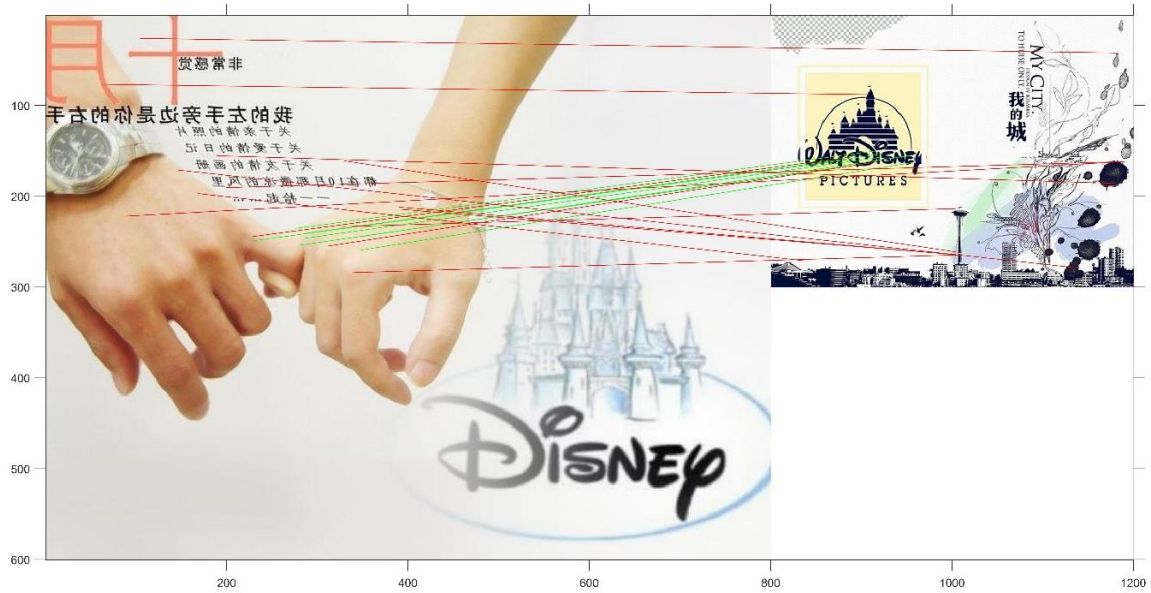

图 6. Disney-00524 与 Disney-00550 匹配结果

如图 7 所示为 Disney-00524 和 Disney-00550 的匹配结果，共有 104 条匹配，其中 81 条几何一致，以绿色标注；23 条几何不一致，以红色标注。

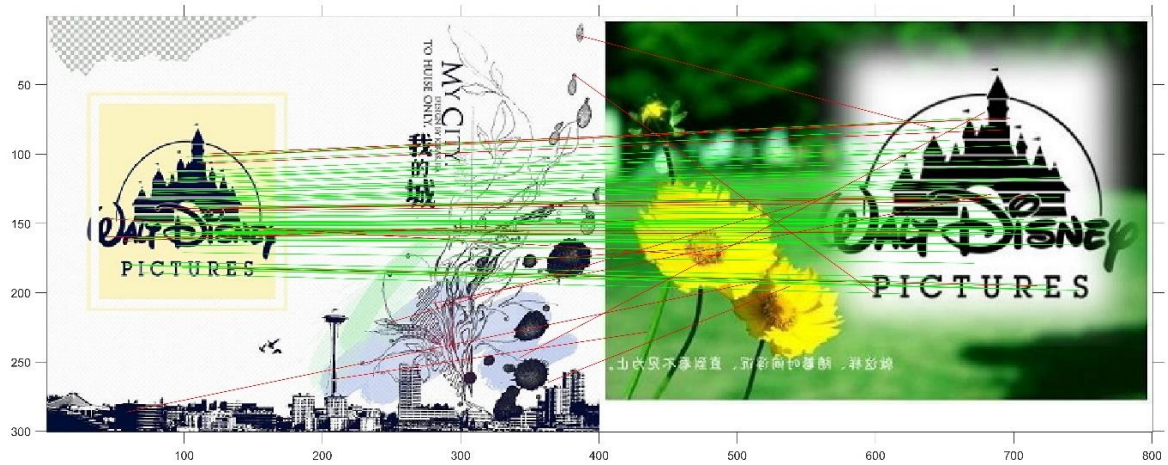

图 7. Disney-00550 与 Disney-00558 匹配结果

如图 8 所示为 Disney-00524 和 Disney-00558 的匹配结果，共有 42 条匹配，其中 21 条几何一致，以绿色标注；21 条几何不一致，以红色标注。

根据图 6、7、8 可以看出，这种局部匹配方法对于待匹配图片的纹理和色彩较敏感，即使边缘形状相似度高，但是色彩纹理不同的局部特征就不能匹配。这也说明局部匹配对于描述子的依赖性较高，如果描述子对于图片的特征不能完整地描述，则局部匹配就会导致失配。

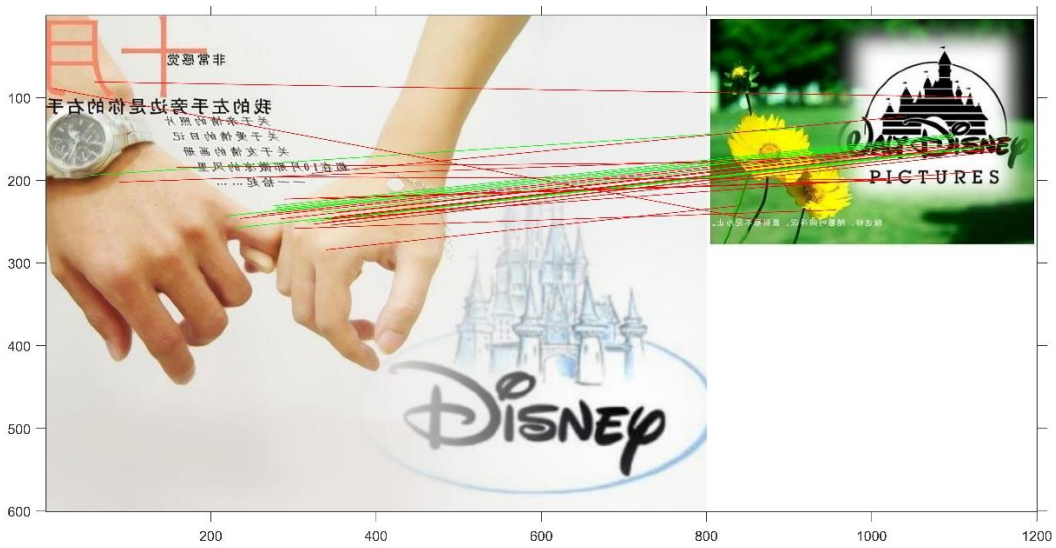

图 8. Disney-00524 与 Disney-00558 匹配结果

2、不相关图像对

再考虑不具有相关性的图像对之间的匹配，如图 9 所示为 Apollo-18 和 Mona-Lisa-73 的匹配情况，图中共有 2 条匹配，其中 1 条为几何一致的匹配，以绿色标注；1 条为几何不一致的匹配，以红色标注。实际上，这种情况下已经完全失配，因为只有一对匹配点时不能比较几何位置，不进行删除操作。

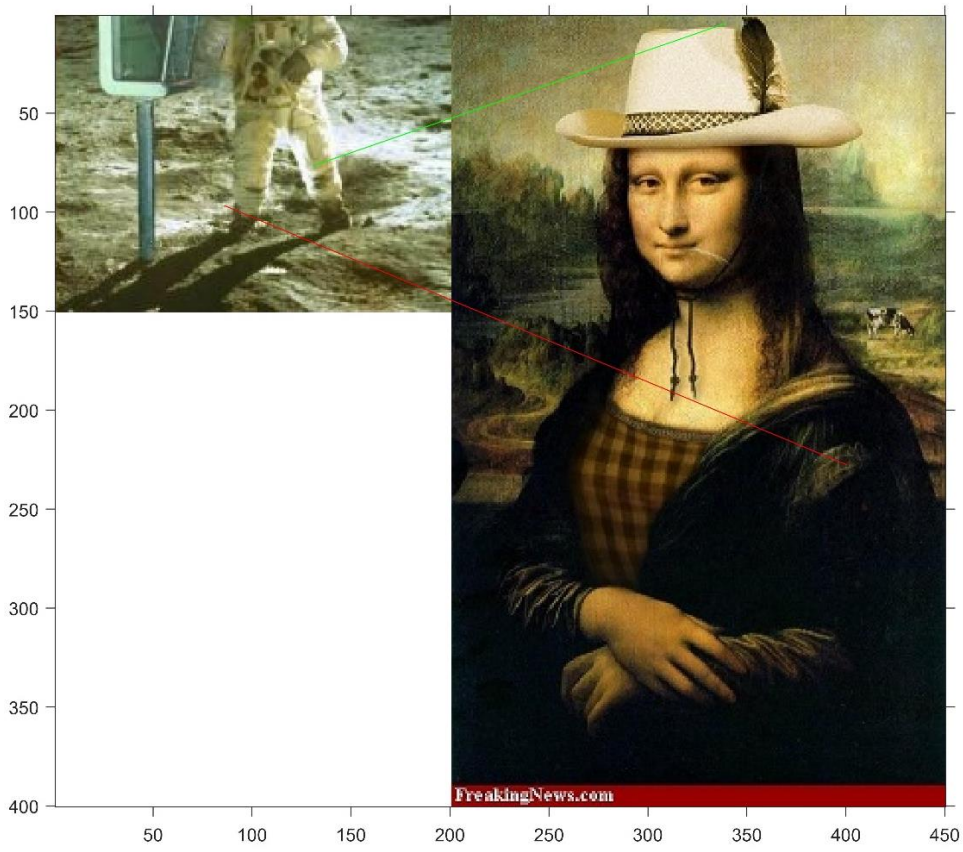

图 9. Apollo-18 与 Mona-Lisa-73 匹配结果

如图 10 所示为 Apollo-266 和 Disney-00524 的匹配结果，共有 38 条匹配，其中 6 条几何一致，以绿色标注；32 条几何不一致，以红色标注。可以看出，尽管采用基于距离比值的准则进行匹配的方法在色调接近的图片之间会造成大量误匹配，但是这种基于 spatial coding 的几何校验方法能较好的去除大量误匹配。

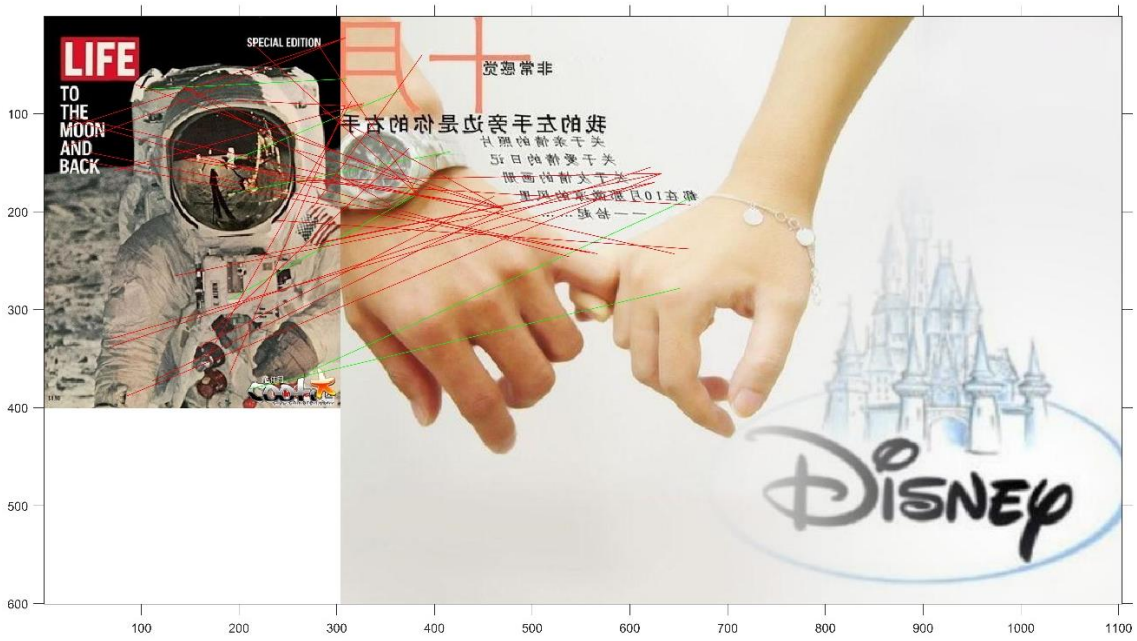

图 10. Apollo-266 与 Disney-00524 匹配结果

如图 11 所示为 Disney-00524 和 Mona-Lisa-452 的匹配结果，共有 13 条匹配，其中 3 条几何一致，以绿色标注；10 条几何不一致，以红色标注。同样可以看到，spatial coding 的方法去除了大部分失配点。

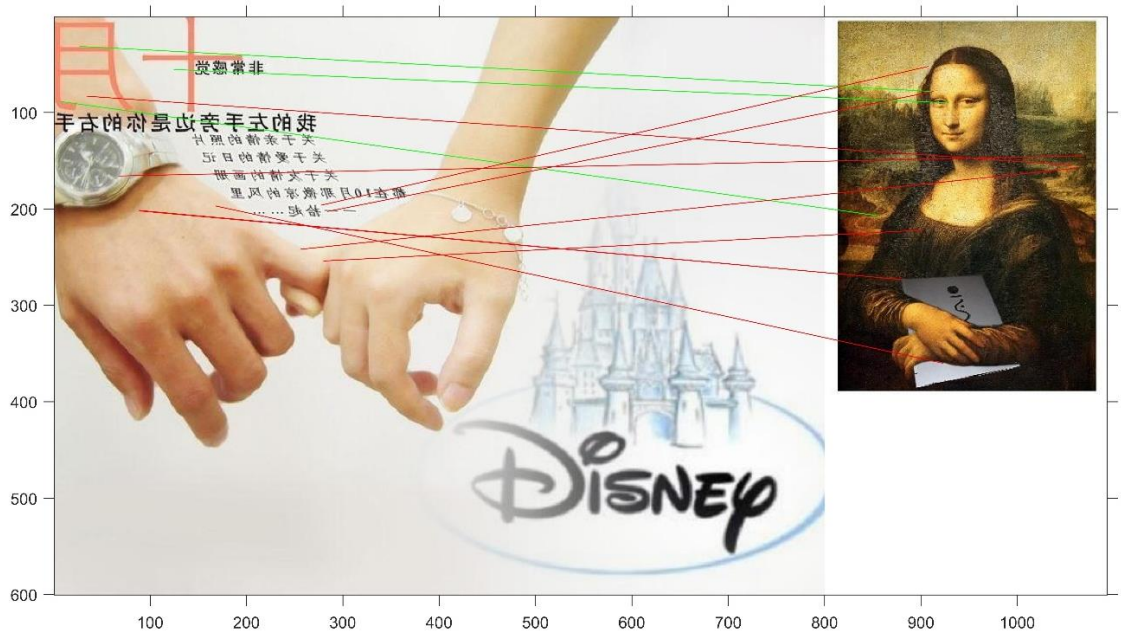

图 11. Disney-00524 与 Mona-Lisa-452 匹配结果

四、实验收获

1、了解了 SIFT 描述子的性质和特点。
2、实现了基于 spatial coding 进行局部特征匹配和几何校验的算法。
3、分析了算法中存在的一些问题，进一步理解了算法的应用限制。