匹配几何校验实验报告

[redacted]

■ 实验代码

Set path and parameters

```
clear;

close all;

clc;

% src_1 = './test images/37967br1.jpg';

% src_2 = './test images/791.jpg';

% src_1 = './test images/4.jpg';

% src_2 = './test images/Apollo-266.jpg';

src_1 = './test images/771.jpg';

src_2 = './test images/305.jpg';

% src_1 = './test images/Apollo-49.jpg';

src_2 = './test images/Apollo-266.jpg';


ext = '.sift';     % extension name of SIFT file

siftDim = 128;

maxAxis = 400;
```

Load image

```
im_1 = imread(src_1);

if max(size(im_1)) > maxAxis

   im_1 = imresize(im_1, maxAxis / max(size(im_1)));

end
```

```
im_2 = imread(src_2);

if max(size(im_2)) > maxAxis

   im_2 = imresize(im_2, maxAxis / max(size(im_2)));

end
```

Load SIFT feature from file

```
featPath_1 = [src_1, ext];

featPath_2 = [src_2, ext];



fid_1 = fopen(featPath_1, 'rb');

featNum_1 = fread(fid_1, 1, 'int32');

SiftFeat_1 = zeros(siftDim, featNum_1);

paraFeat_1 = zeros(4, featNum_1);

for i = 1 : featNum_1

   SiftFeat_1(:, i) = fread(fid_1, siftDim, 'uchar');

   paraFeat_1(:, i) = fread(fid_1, 4, 'float32');

end

fclose(fid_1);

fid_2 = fopen(featPath_2, 'rb');

featNum_2 = fread(fid_2, 1, 'int32');

SiftFeat_2 = zeros(siftDim, featNum_2);

paraFeat_2 = zeros(4, featNum_2);

for i = 1 : featNum_2

   SiftFeat_2(:, i) = fread(fid_2, siftDim, 'uchar');

   paraFeat_2(:, i) = fread(fid_2, 4, 'float32');

end

fclose(fid_1);
```

```
%%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);
```

Check match based on distances between SIFT

```
normVal = mean(sqrt(sum(SiftFeat_1.^2)));

matchInd = zeros(featNum_1, 1);

matchDis = zeros(featNum_1, 1);

validDis = [];

gridDisVec = [];

ic = 0;

for i = 1 : featNum_1

   tmpFeat = repmat(SiftFeat_1(:, i), 1, featNum_2);

   d = sqrt(sum((tmpFeat - SiftFeat_2).^2)) / normVal; % L2 distance

   matchDis(i) = min(d);

   [v, ind] = sort(d);

   if v(1) < 0.4

      matchInd(i) = ind(1);

      ic = ic + 1;

      validDis(ic, 1 : 3) = [v(1), v(2), v(1) / v(2)];

      tmp = (SiftFeat_1(:, i) - SiftFeat_2(:, ind(1))).^2;

      tmp2 = reshape(tmp(:), 8, 16);

      gridDisVec(ic, 1 : 16) = sqrt(sum(tmp2));

   end

end

% figure; stem(matchDis); ylim([0, 1.2]);

% figure; stem(matchDis(matchInd > 0)); ylim([0, 1.2]);
```

Show the local matching results 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;

np = 40;

thr = linspace(0,2*pi,np) ;

Xp = cos(thr);

Yp = sin(thr);

paraFeat_2(1, :) = paraFeat_2(1, :) + col;

figure(3); imshow(uint8(imgBig)); axis on;

hold on;

matchCount = 0;

record_Q_M = []; QM_index = 0;

for i = 1 : featNum_1

   if matchInd(i) > 0
```

```
       matchCount = matchCount + 1;

       xys = paraFeat_1(:, i);

       xys2 = paraFeat_2(:, matchInd(i));

       figure(3);

       hold on;

       plot(xys(1) + Xp * xys(3) * 6, xys(2) + Yp * xys(3) * 6, 'r');

       plot(xys2(1) + Xp * xys2(3) * 6, xys2(2) + Yp * xys2(3) * 6, 'r');

       hold on; plot([xys(1), xys2(1)], [xys(2), xys2(2)], '-b', 'LineWidth', 0.8);
```

记录所有的特征匹配点，并删除重复坐标点

```
        if QM_index == 0

            record_Q_M = [record_Q_M;xys(1),xys(2),xys2(1),xys2(2),xys(3),xys2(3)];

        else

            if ( ~(length(find(record_Q_M(:,1)== xys(1))) * length(find(record_Q_M(:,2)==
xys(2)))))

                record_Q_M = [record_Q_M;xys(1),xys(2),xys2(1),xys2(2),xys(3),xys2(3)];

            end

        end

        QM_index = QM_index + 1;
```

```
    end

end

figure(3);

title(sprintf('Total local matches : %d (%d-%d)', length(find(matchInd)),  featNum_1 ,
featNum_2));    % 所有的匹配点

hold off;
```

几何校验（Spatial_Verification.m）

```
Spr = 3;   % k = 0,...,Spr-1

Wrong_Matched_Point  = Spatial_Verification( record_Q_M , Spr);
```

红线——几何不一致的匹配

```
figure(4);imshow(uint8(imgBig)); axis on;

hold on;

for i = 1 : length(Wrong_Matched_Point)

   xys = Wrong_Matched_Point(i , [1 2 5]);

   xys2 = Wrong_Matched_Point(i , [3 4 6]);

   figure(4)

   hold on;

   plot(xys(1), xys(2), 'rs');

   plot(xys2(1), xys2(2), 'rs');

   hold on; plot([xys(1), xys2(1)], [xys(2), xys2(2)], '-r', 'LineWidth', 0.8);

end

title(sprintf('几何不一致的匹配'));

hold off;
```

蓝线——几何一致的匹配

```
ALL_Point = [record_Q_M;Wrong_Matched_Point] ;

% Delete  Rows

[temp_Point,mm,nn] = unique(ALL_Point,'rows');

cc = tabulate(nn);

Right_Matched_Point = ALL_Point(mm(cc(cc(:,2)==1,1)),:);


figure(5);imshow(uint8(imgBig)); axis on;

hold on;

[m_Right_Matched_Point,n_Right_Matched_Point] = size(Right_Matched_Point);

for i = 1 : m_Right_Matched_Point
```

```
    xys = Right_Matched_Point(i , [1 2 5]);

    xys2 = Right_Matched_Point(i , [3 4 6]);

    figure(5)

    hold on;

    plot(xys(1), xys(2), 'bo');

    plot(xys2(1), xys2(2), 'bo');

    hold on; plot([xys(1), xys2(1)], [xys(2), xys2(2)], '-b', 'LineWidth', 0.8);

end

title(sprintf('几何一致的匹配'));

hold off;
```

○ Spatial_Verification.m

```
function Wrong_Matched_Point = Spatial_Verification( record_Q_M ,  Spr)

FUNCSTART = 1; Wrong_Matched_Point = []
while (FUNCSTART)

  [SpN,SpM] = size(record_Q_M) ;  % i = 1,...,SpN
  Sp_Theta = [0:1:Spr-1].*(pi / (2*Spr));

  ALLQ_x_i_k = zeros(SpN,length(Sp_Theta));
  ALLQ_y_i_k = zeros(SpN,length(Sp_Theta));

  ALLM_x_i_k = zeros(SpN,length(Sp_Theta));
  ALLM_y_i_k = zeros(SpN,length(Sp_Theta));

  for Spk=1:Spr
     ALLQ_x_i_k(:,Spk) = cos(Sp_Theta(Spk)).*(record_Q_M(:,1)) -
                              sin(Sp_Theta(Spk)).*(record_Q_M(:,2));
     ALLQ_y_i_k(:,Spk) = sin(Sp_Theta(Spk)).*(record_Q_M(:,1)) +
                              cos(Sp_Theta(Spk)).*(record_Q_M(:,2));
  end

  for Spk=1:Spr
     ALLM_x_i_k(:,Spk) = cos(Sp_Theta(Spk)).*(record_Q_M(:,3)) -
                              sin(Sp_Theta(Spk)).*(record_Q_M(:,4));
     ALLM_y_i_k(:,Spk) = sin(Sp_Theta(Spk)).*(record_Q_M(:,3)) +
                              cos(Sp_Theta(Spk)).*(record_Q_M(:,4));
  end
```

```
for k = 1:Spr
    for i = 1:SpN
        for j = 1:SpN
            if ALLQ_x_i_k(i,k) < ALLQ_x_i_k(j,k)
                GX_temp(i,j) = 0;
            else
                GX_temp(i,j) = 1;
            end

            if ALLQ_y_i_k(i,k) < ALLQ_y_i_k(j,k)
                GY_temp(i,j) = 0;
            else
                GY_temp(i,j) = 1;
            end
        end
    end
    GXQ{k} = GX_temp;
    GYQ{k} = GY_temp;
end


for k = 1:Spr
   for i = 1:SpN
       for j = 1:SpN
            if ALLM_x_i_k(i,k) < ALLM_x_i_k(j,k)
                GX_temp(i,j) = 0;
            else
                GX_temp(i,j) = 1;
            end

            if ALLM_y_i_k(i,k) < ALLM_y_i_k(j,k)
                GY_temp(i,j) = 0;
            else
                GY_temp(i,j) = 1;
            end
       end
   end
    GXM{k} = GX_temp;
    GYM{k} = GY_temp;
end


for k = 1:Spr
```

```
    VX{k} = xor(GXQ{k},GXM{k});
    VY{k} = xor(GYQ{k},GYM{k});
end


SX = VX{1}; SY = VY{1};
for k = 2:Spr
    SX = SX + VX{k};
    SY = SY + VY{k};
end

% Condition of Quit
if  ((sum(sum(SX))==0) && (sum(sum(SY))==0))
    FUNCSTART = 0;
else
    sum_row_SX = sum( SX , 2 );
    sum_row_SY = sum( SY , 2 );

    IDSX = find(sum_row_SX==max(sum_row_SX));
    IDSY = find(sum_row_SY==max(sum_row_SY));

    if (max(sum_row_SX) >= max(sum_row_SY) )
        Serch_ID = IDSX(1);
    else
        Serch_ID = IDSY(1);
    end

    Wrong_Matched_Point = 
[Wrong_Matched_Point;record_Q_M(Serch_ID,:)];
    record_Q_M(Serch_ID,:) = [];
end

end

end
```

■ 实验结果

○ 相关图片

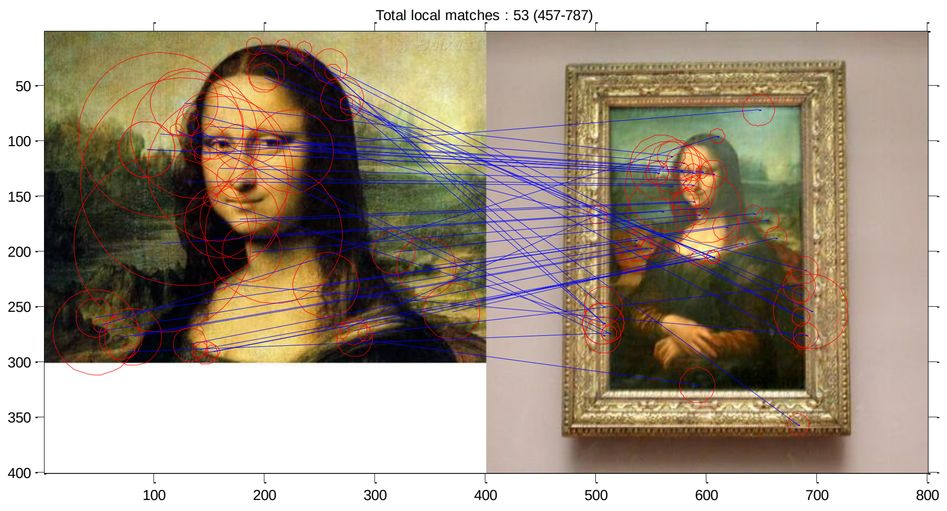

图 1　相关图像——所有匹配点

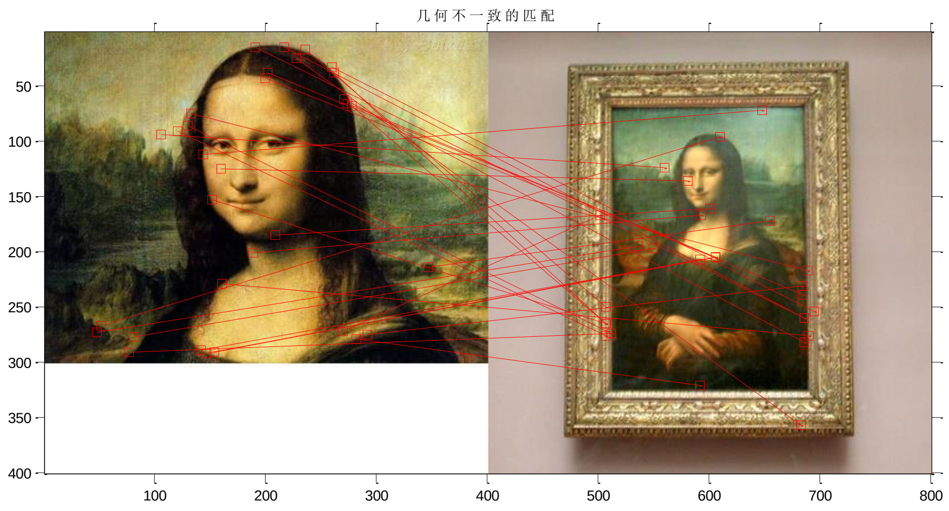

图 2　相关图像——几何不一致匹配点

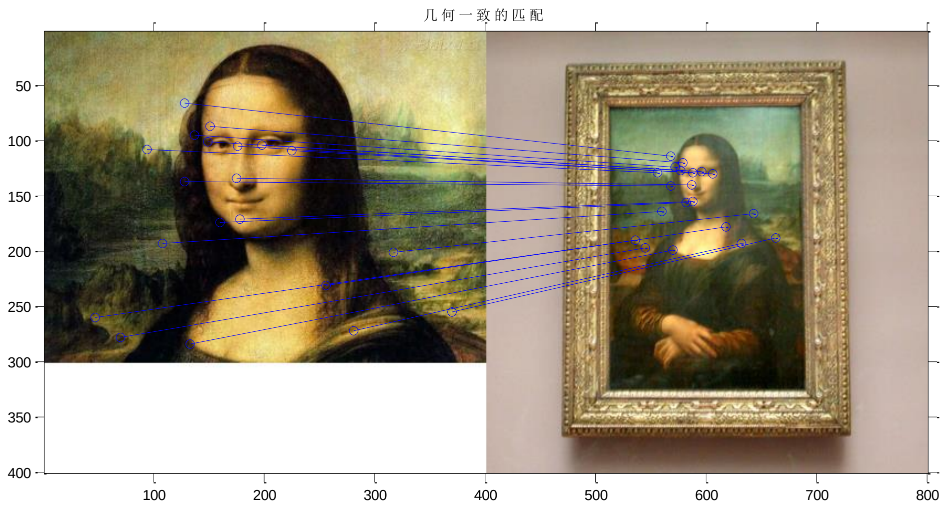

图 3　相关图像——几何一致匹配点

○ 不相关图片

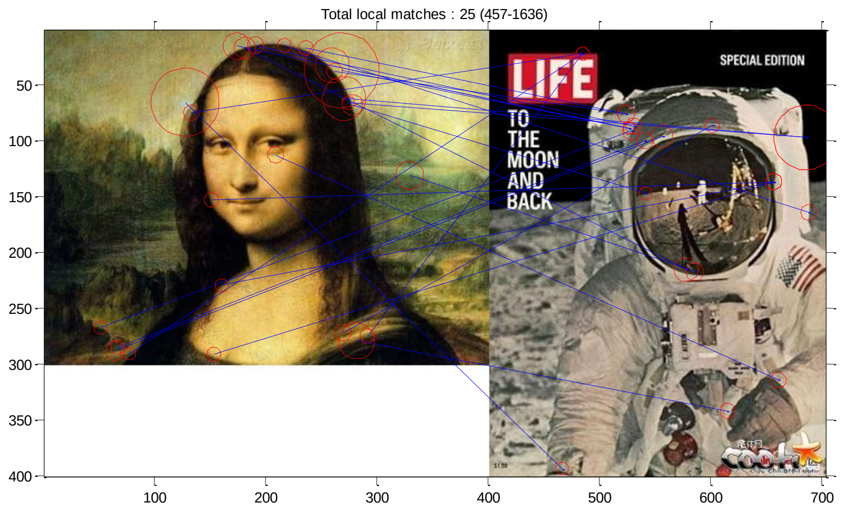

图 4　不相关图像——所有匹配点

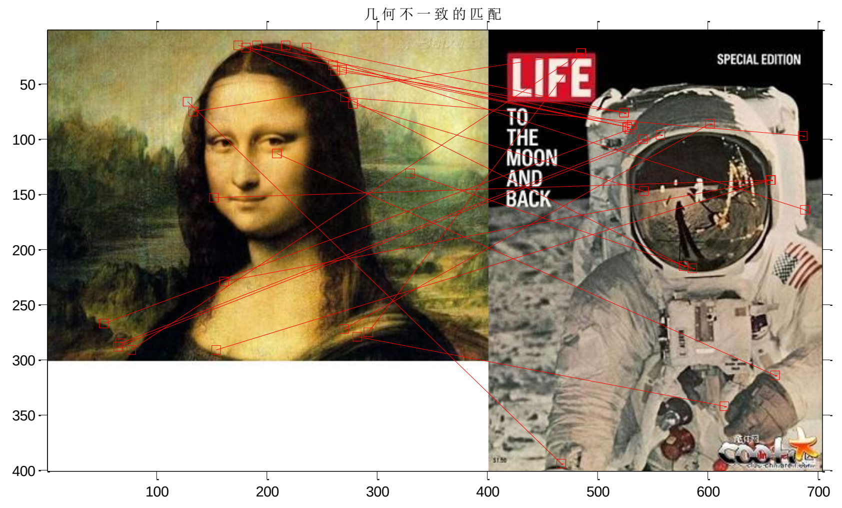

图 5　不相关图像——几何不一致匹配点

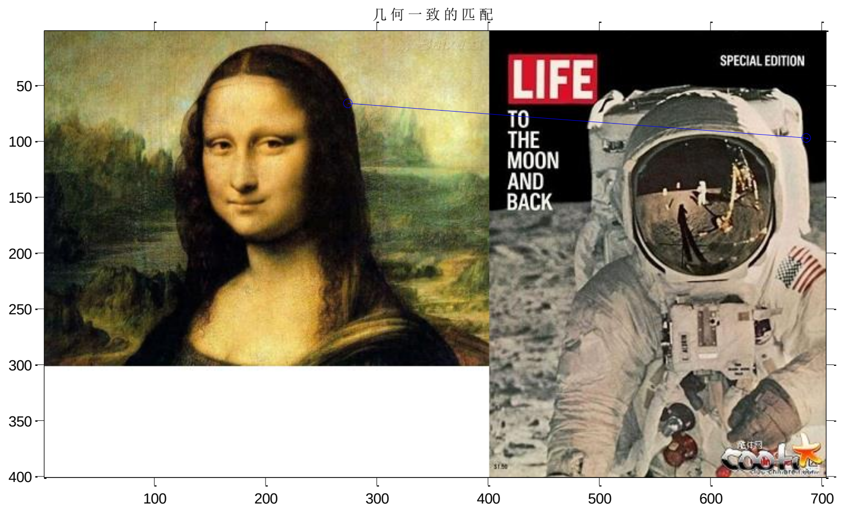

图 6　不相关图像——几何一致匹配点