图像配准实验报告

自动化 65 姚润昭 2160504132

摘要

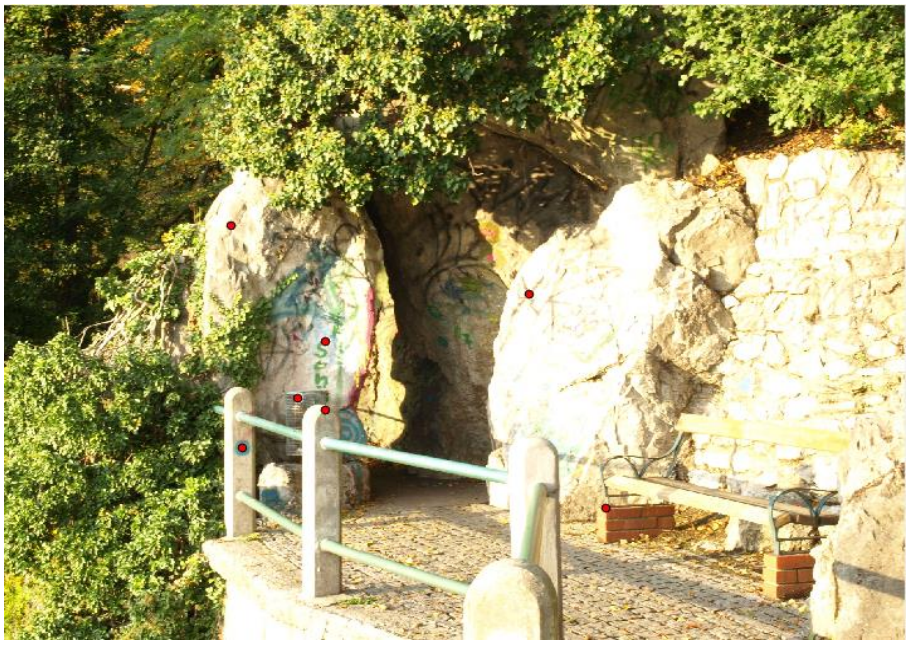
在 A、B 两幅图采集对应的七个点的基础上，求得了变换矩阵 H，之后再用前向映射、反向映射以及系统函数三种方法实现了图像匹配。

题目要求：

要求根据已给的两幅图像，在各幅图像中随机找出 7 个点，计算出两幅图像之间的转换矩阵 H，并且输出转换之后的图像。

注：已给图像分别为 Image A 和 Image B。

1. 手动标点



图一　Image A 手动标点

图二　Image B 手动标点

2. 输出两图七点坐标

```
imageB =                imageA =

   1.0e+03 *               1.0e+03 *

    0.6338    1.3966        1.1939    1.6914
    1.0054    1.3247        0.9701    1.9050
    0.9055    1.2528        1.3058    1.7423
    1.0733    1.0370        2.4398    2.1644
    0.8376    0.4616        2.1296    1.2439
    1.9204    1.0570        0.9244    0.9489
    1.9844    2.0319        1.3058    1.4473
```

3. 计算转换矩阵（根据公式：$\mathbf{H} = \mathbf{Q}\mathbf{P}^T(\mathbf{P}\mathbf{P}^T)^{-1} = \mathbf{Q}\mathbf{P}^\dagger$ ）

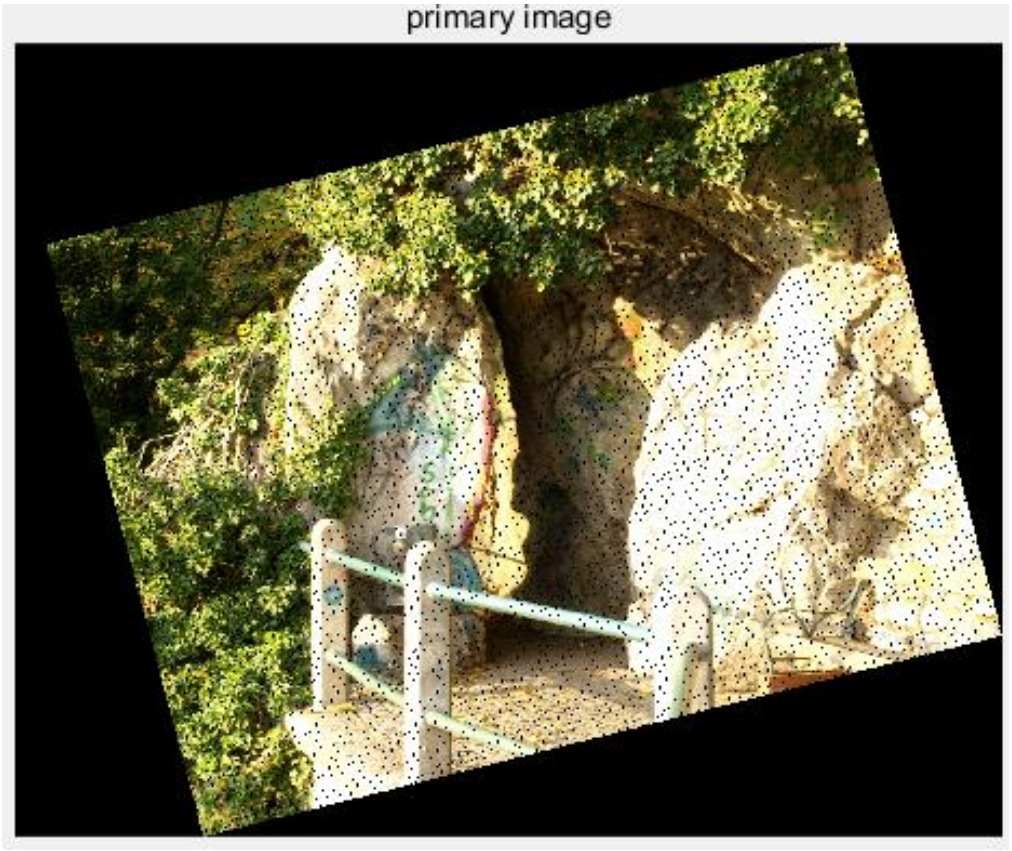
```
H =

    0.9563   -0.2441  180.7239
    0.2546    0.9688 -694.3662
         0         0    1.0000
```

4. 配准过程

1）前向映射（将原图像矩阵像素点映射到配准图像像素点）

建立一个空矩阵 new_B，根据 H 矩阵，将图像 B 每个像素的坐标映射到空矩阵上，填入像素值，由于离散点映射的非连续性，new_B 图像中存在很多零像素点。如图 primary image 所示：



再用最邻近插值的方法，对图片中的零像素点进行插值，得到最终的图像，如图 final img 所示。

2）反向映射（将配准图像的像素点映射到原图像像素点上）

由于前向影射，存在零像素点的问题，需要插值，比较繁琐且耗时长，故采用反向映射得到如下图片：

3）利用 matlab 自带的函数 imtransform（）进行配准得到了如下图像 rotated img：

5. 代码：

1）获取七个点的坐标打印并保存

```
function mouse_click_track(imgpath)
  global i;
  global A;
    i=0;
    A=zeros(7,2);
    im=imread(imgpath);
    imshow(im);
    hold on;
    set(gcf,'WindowButtonDownFcn',@MouseClickFcn)

    function MouseClickFcn(src,event)
    global i;
    global A;
    i=i+1;
    point_c=get(gca,'CurrentPoint');
    x=point_c(1,1);
    y=point_c(1,2);
    A(i,1)=x;
    A(i,2)=y;
    if(i==7)
        A
        save imageB.mat -ascii A
```

```
    end
    plot(x,y,'ko', 'MarkerSize', 5,  'MarkerFaceColor', 'r');
```

2）前向映射

将矩阵 ImgB 坐标点对应的像素值映射到空矩阵上：

```
load imageB.mat -ascii;
load imageA.mat -ascii;
imgA=imread('C:\Users\DELL\Desktop\配准\image A.jpg');
imgB=imread('C:\Users\DELL\Desktop\配准\image B.jpg');
B=[imageB,[1;1;1;1;1;1;1]]';
A=[imageA,[1;1;1;1;1;1;1]]';
H=A*(B')*inv(B*B');                %计算H

for i=1:2089
  for j=1:2785
     a=H*[i;j;1];
        new_img(500+round(a(1,1)),800+round(a(2,1)),:)=imgB(i,j,:);
     end                    %是因为映射过程中round（a（1,1））可能为负数，所以
                                加上一个较大的整数，保证对应点是一个正整数
end
imwrite(new_img,'new_img.bmp')
imshow(new_img);
```

对映射得到的图像 primary image 进行最邻近差值：

```
clc;clear;
H=[ 0.9563   -0.2441  180.7239
    0.2546    0.9688 -694.3662
   -0.0000    0.0000    1.0000];
new_img=imread('new_img.bmp');
subplot(1,2,1)
imshow(new_img);
title('primary image')
for i=2:2670
    for j=2:3330
    if(new_img(i,j,:)==0)
        ori=inv(H)*[i-500;j-800;1];
        if(ori(1,1)>0&&ori(1,1)<2090&&ori(2,1)>0&&ori(2,1)<2786) %确保
                                        是图片中的黑点，而不是四周的黑色区域
            if(new_img(i+1,j,:)~=0)
                new_img(i,j,:)=new_img(i+1,j,:);break;end
```

```
            if(new_img(i-1,j,:)~=0)
             new_img(i,j,:)=new_img(i-1,j,:);break;end
            if(new_img(i,j+1,:)~=0)
                new_img(i,j,:)=new_img(i,j+1,:);break;end
            if(new_img(i,j-1,:)~=0)
                new_img(i,j,:)=new_img(i,j-1,:);break;end
        end
    end
end
end
subplot(1,2,2)
imshow(new_img)
title('final image')
```

3）反向映射

```
H_=inv(H);
new_img=uint8(zeros(2678,3336,3));
for i=1:2678
   for j=1:3336
   t=H_*[i-500;j-800;1];
   t1=round(t(1,1));
   t2=round(t(2,1));
   if(t1>0&&t1<2090&&t2>0&&t2<2786)
       new_img(i,j,:)=imgB(t1,t2,:);
   end
   end
end
subplot(1,2,1)
imshow(imgB)
subplot(1,2,2)
imshow(new_img)
title('backward img')
```

4）matlab 自带函数配准

```
tform=cp2tform(imageB,imageA,'linear conformal');
Iout=imtransform(imgB,tform); %用matlab自带函数进行配准
figure
subplot(1,2,1),imshow(imrotate(Iout,180));
subplot(1,2,2),imshow(imgA);
```

6. 心得体会

在选取坐标点时要尽可能精确，并且分布在整个图片中，以便得

到的数据更有代表性。运用三种配准方法得到的图片视觉上差别不大，具有较好的效果。在前向映射和反向映射中都采用了最近邻插值的方法，采用双线性和三次差值效果应该会更好。粗略估计三种方法的运行时间如下表

方法	**时间（s）**
前向映射	60（映射）+25（插值）
反向映射	25
Imtransform（）（matlab 自带函数）	6

可见差距很明显，尤其是前向映射中许多的循环和判断花费时间，有待于优化。反向映射比前向映射代码简练运行时间短，但是据 matlab 自带函数仍有较大差距。