第五次课程作业

图像匹配问题

1. 问题描述

已知一个任意形状，查找在大图像中最接近的形状位置。

输入：一个小图形状和一张大图

输出：最接近的形状在大图中的位置

假设：

（1）已知形状与目标形状有一定的形变。

（2）形状与大图像均为二值图像，图中有多个形状。

要求：

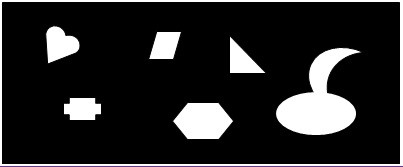
建立数学模型，并编写程序实现；

查找精度高，抗噪能力强（大图像可能含有噪声），速度快。

例如，小图形状如下：



大图形状如下：



1. 问题分析

* 图像在计算机中可以二值矩阵来表示；
* 将源图像与目标图像分别二值化，我们的目的是：在目标图像中找到与源图像匹配的区域。
* 在目标图像中，以其中一个像素点为左上角，取与源图像等大小的子图，计算其与模板的误差值；逐像素地移动模板子图像的原点像素，在每一次移动的过程中计算每个子图与模板的误差值，在所有能够取到的子图中，找到与模板图最相似的子图（误差值最小）作为最终匹配结果。
* 根据结果进行优化，不断完善模型。

1. 模型假设

* 形状与目标形状有一定的形变。
* 形状与大图像均为二值图像。
* 图中有多个形状。
* 大图像中存在一定噪声。

1. 模型构成

|  |  |
| --- | --- |
| **符号** | **含义** |
| Src | 大小为m1xn1的源图像矩阵 |
| Des | 大小为m2xn2的目标图像矩阵 |
| m1、n1 | Src行数、列数 |
| m2、n2 | Des行数、列数 |
| K(i,j) | 在搜索图**S**中，以**(i,j)**为左上角，取**MxN**大小的子图 |
| D(i,j) | K(i,j)与T(x,y)的差异值 |
| w | 旋转角度 |
| r\_src | 旋转w度的源图像 |
| (x\_maxpos,y\_maxpos) | 最大匹配点 |
| fin | 带有fin字样的变量代表最终结果 |

1. 模型建立与求解

5.1源图形优化

* 根据Matlab提供的imread函数将源图像与目标图像分别读入为矩阵src，des。
* 再使用im2bw函数分别将矩阵src，des二值化。白色为1、黑色为0。
* 根据题目给出的案例，源图像的周围带有一圈白色的线条，显然这些线条的存在会对于图像匹配造成干扰，我们要把它们去掉。
* 大概思路是：从上下左右依次找到白边终止行（列）求出交点后进行裁剪。
* 给出下面的算法：

*for(从上到下扫描)*

*if（整行求和小于n1）*

*记录所在行；*

*else*

*跳转下一行；*

*for(从左到右扫描)*

*if（整列求和小于m1）*

*记录所在列；*

*else*

*跳转右一行；*

*for(从下到上扫描)*

*if（整行求和小于n1）*

*记录所在行；*

*else*

*跳转上一行；*

*for(从右到左扫描)*

*if（整列求和小于m1）*

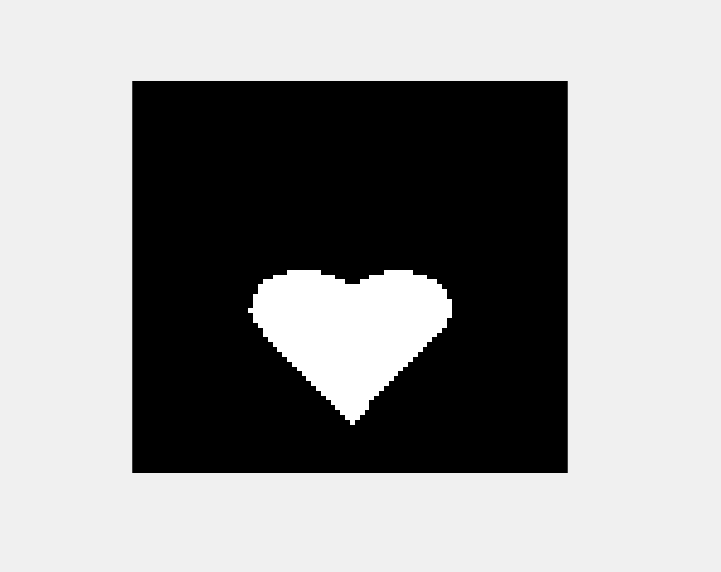
*记录所在列；*

*else*

*跳转左一行；*

*裁剪；*

* 运行算法之后，白色线条去除完毕



* 如果源图像中，图形占比太小（心形），也会影响我们的匹配成功率。
* 所以我们要将图形“扣”出来。
* 大概的思路是，从上下左右逐行（列）扫描图像，找到第一条不全黑的行（列），保留它前面的9(测试得出选取9像素，在旋转时不会失真)行像素，其余全部去掉。
* 根据白色为1、黑色为0的特点，给出下面的算法：

*for(从上到下扫描)*

*if（整行求和不等于0）*

*记录所在行；*

*else*

*跳转下一行；*

*for(从左到右扫描)*

*if（整列求和不等于0）*

*记录所在列；*

*else*

*跳转右一行；*

*for(从下到上扫描)*

*if（整行求和不等于0）*

*记录所在行；*

*else*

*跳转上一行；*

*for(从右到左扫描)*

*if（整列求和不等于0）*

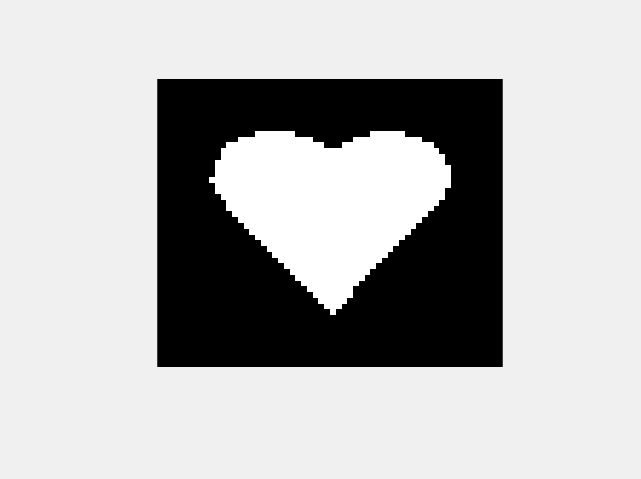
*记录所在列；*

*else*

*跳转左一行；*

*裁剪；*

* 运行算法之后，目标图形被扣出。



5.2图形匹配

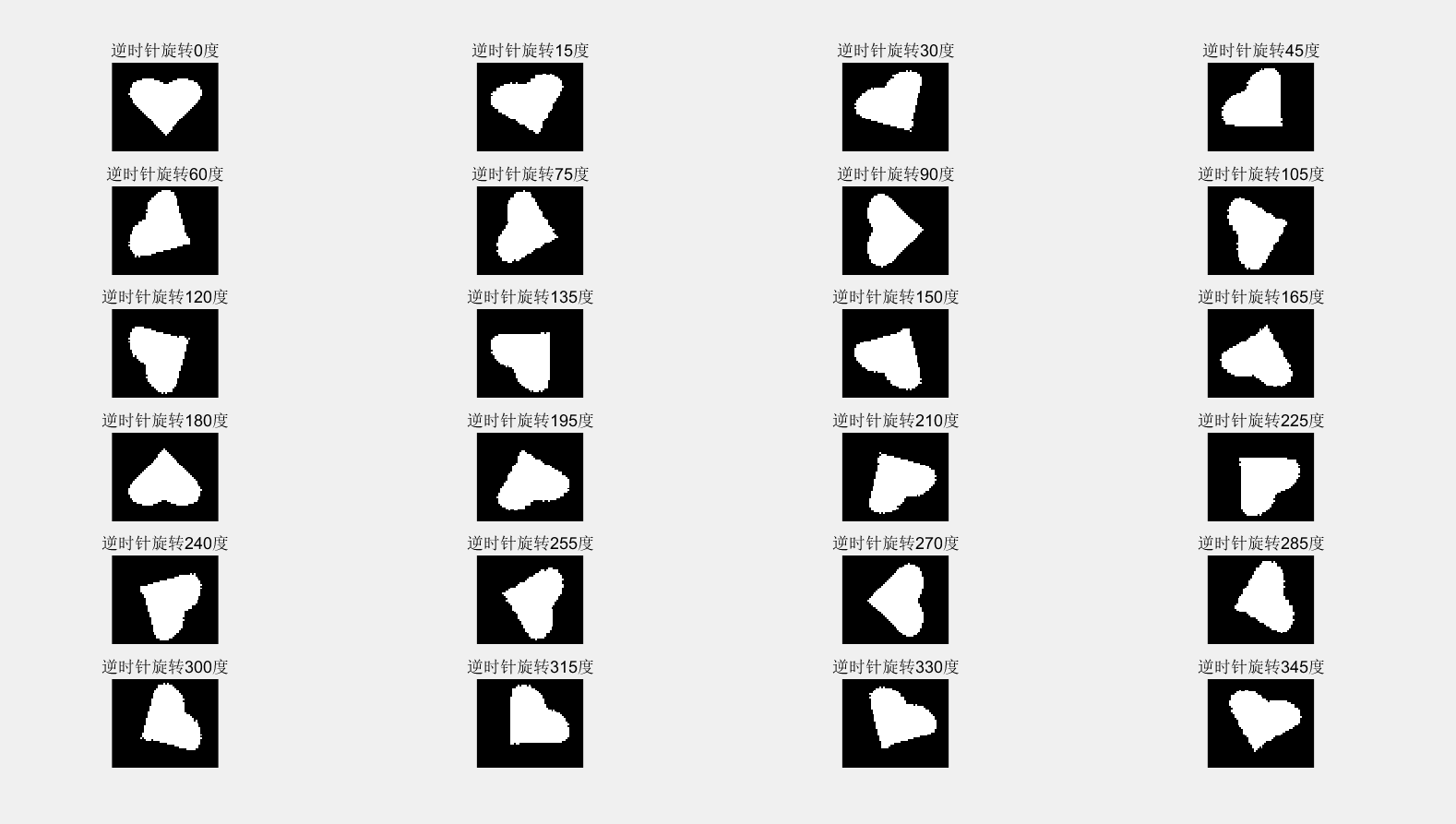
* 扣出图形之后还存在着目标图形被旋转的问题。
* 与几个同学讨论过之后决定用以下算法来解决该问题：

*for(将源图像旋转w（15，30，45度…）)*

*用旋转后的新图像去匹配;*

*找出这些匹配中的最小误差的最小值;*

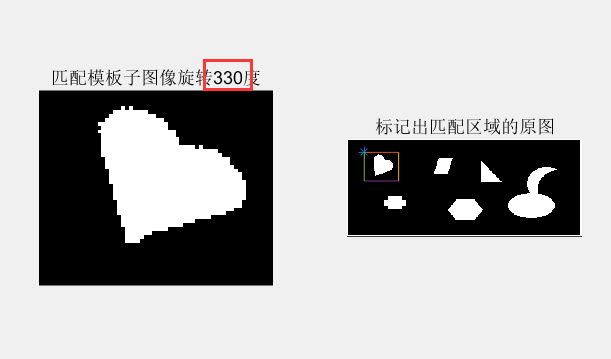
* 给出旋转后用来匹配的模板：



* 误差值的计算，参考于网上的算法：
* 设目的图片的des(x,y)的大小为m2\*n2，模板图像src(x,y)的大小为m1\*n1,则des与src的相关表示为：
* 其中，x=0,1,2…n-n0,y=0,1,2…m-m0,计算c(x,y)的过程就是在图像des(x,y) 中逐像素地移动模板子图像src(x,y) 的原点像素，在每一次移动的过程中根据上式计算每个像素位置的相关。对式（1）的向量表达式进行归一化后如下式所示：

5.3模型求解

* 运行Matlab程序得到结果：



1. 模型分析

本模型根据图相匹配的基本原理—矩阵差异匹配。并在其基础上与几位同学共同探讨优化策略，并用Matlab代码实现，道理简单易懂，匹配准确度高，且对图片的压缩不敏感。该模型的缺点是枚举旋转各个角度的源图片，匹配速度存在欠缺。

参考博客：

https://blog.csdn.net/yi\_tech\_blog/article/details/70199021

1. 代码实现

* 主函数：

close all

clear all

clc;

src=imread('C:\Users\刘宏岩\Desktop\src.jpg');

src=im2bw(src);

des=imread('C:\Users\刘宏岩\Desktop\des.jpg');

des=im2bw(des);

src=rmwhite(src);

src=crop(src);

match(src,des);

* 去白边函数：

function f1=rmwhite(tmp)

[m1,n1]=size(tmp);

for i=1:m1

if(sum(tmp(i,:))<n1)

c1tmp=i;

break;

end

end

for i=1:n1

if(sum(tmp(:,i))<m1)

r1tmp=i;

break;

end

end

for i=m1:-1:1

if(sum(tmp(i,:))<n1)

c2tmp=i;

break;

end

end

for i=n1:-1:1

if(sum(tmp(:,i))<m1)

r2tmp=i;

break;

end

end

f1=tmp(c1tmp:c2tmp,r1tmp:r2tmp);

* 抠图函数：

function f2=crop(tmp)

[m1,n1]=size(tmp);

for i=1:m1

if(sum(tmp(i,:))~=0)

c1tmp=i-9;

if(c1tmp<1)

c1tmp=1;

end;

break;

end;

end;

for i=1:n1

if(sum(tmp(:,i))~=0)

r1tmp=i-9;

if(r1tmp<1)

r1tmp=1;

end;

break;

end;

end;

for i=m1:-1:1

if(sum(tmp(i,:))~=0)

c2tmp=i+9;

if(c2tmp>m1)

r1tmp=m1;

end;

break;

end;

end;

for i=n1:-1:1

if(sum(tmp(:,i))~=0)

r2tmp=i+9;

if(r2tmp>n1)

r2tmp=n1;

end;

break;

end;

end;

f2=tmp(c1tmp:c2tmp,r1tmp:r2tmp);

* 图像匹配函数：

function imgmatch=match(src,des)

Dmin=100000;

finiPos=1;

finjPos=1;

for w=0:15:350

r\_src=imrotate(src,w,'crop');

[m2,n2]=size(des);

[m1,n1]=size(r\_src);

result=zeros(m2-m1+1,n2-n1+1);

vec\_sub = double( r\_src(:) );

norm\_sub = norm( vec\_sub );

for i=1:m2-m1+1

for j=1:n2-n1+1

subMatr=des(i:i+m1-1,j:j+n1-1);

vec=double( subMatr(:));

result(i,j)=vec'\*vec\_sub / (norm(vec)\*norm\_sub+eps);

end

end

[iMaxPos,jMaxPos]=find( result==max( result(:)));

tmp=des(iMaxPos:iMaxPos+m1-1,jMaxPos:jMaxPos+n1-1);

D=sum(sum(abs(tmp-r\_src)));

if(D<Dmin)

Dmin=D;

finiPos=iMaxPos;

finjPos=jMaxPos;

finw=w;

finsrc=r\_src;

end

end

figure,

subplot(121);imshow(finsrc),title(['匹配模板子图像旋转',num2str(finw),'度']);

subplot(122);

imshow(des);

title('标记出匹配区域的原图'),

hold on

plot(finjPos,finiPos,'\*');

plot([finjPos,finjPos+n1-1],[finiPos,finiPos]);

plot([finjPos+n1-1,finjPos+n1-1],[finiPos,finiPos+m1-1]);

plot([finjPos,finjPos+n1-1],[finiPos+m1-1,finiPos+m1-1]);

plot([finjPos,finjPos],[finiPos,finiPos+m1-1]);

end