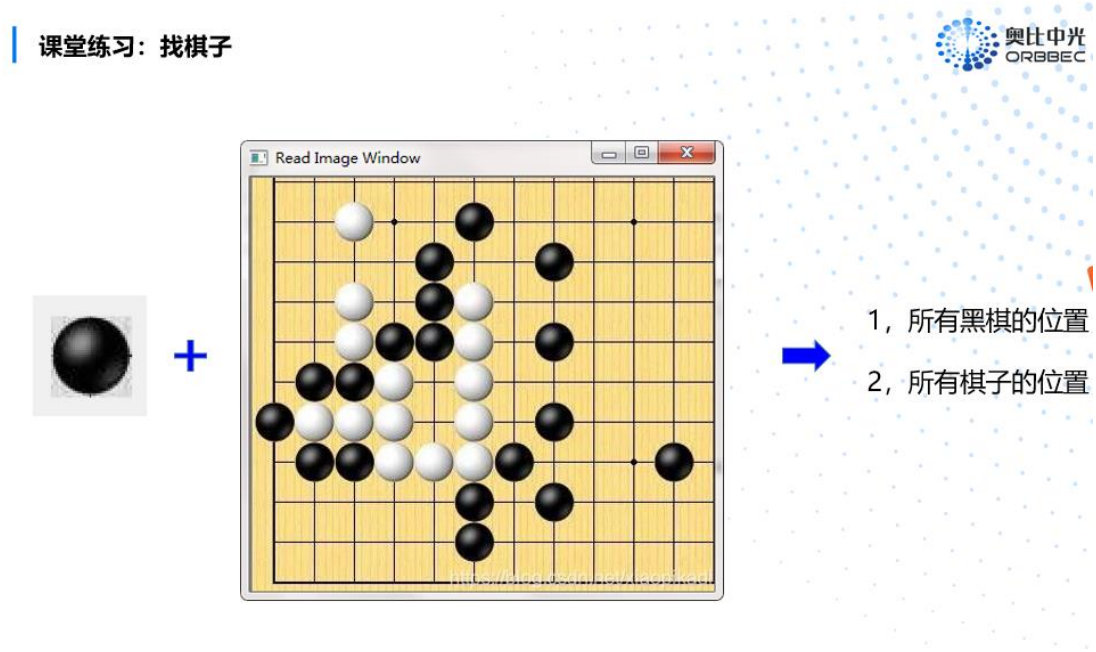
作业 3（2022 年 7 月 13 日）

19200300004 黄铭瑞

作业内容

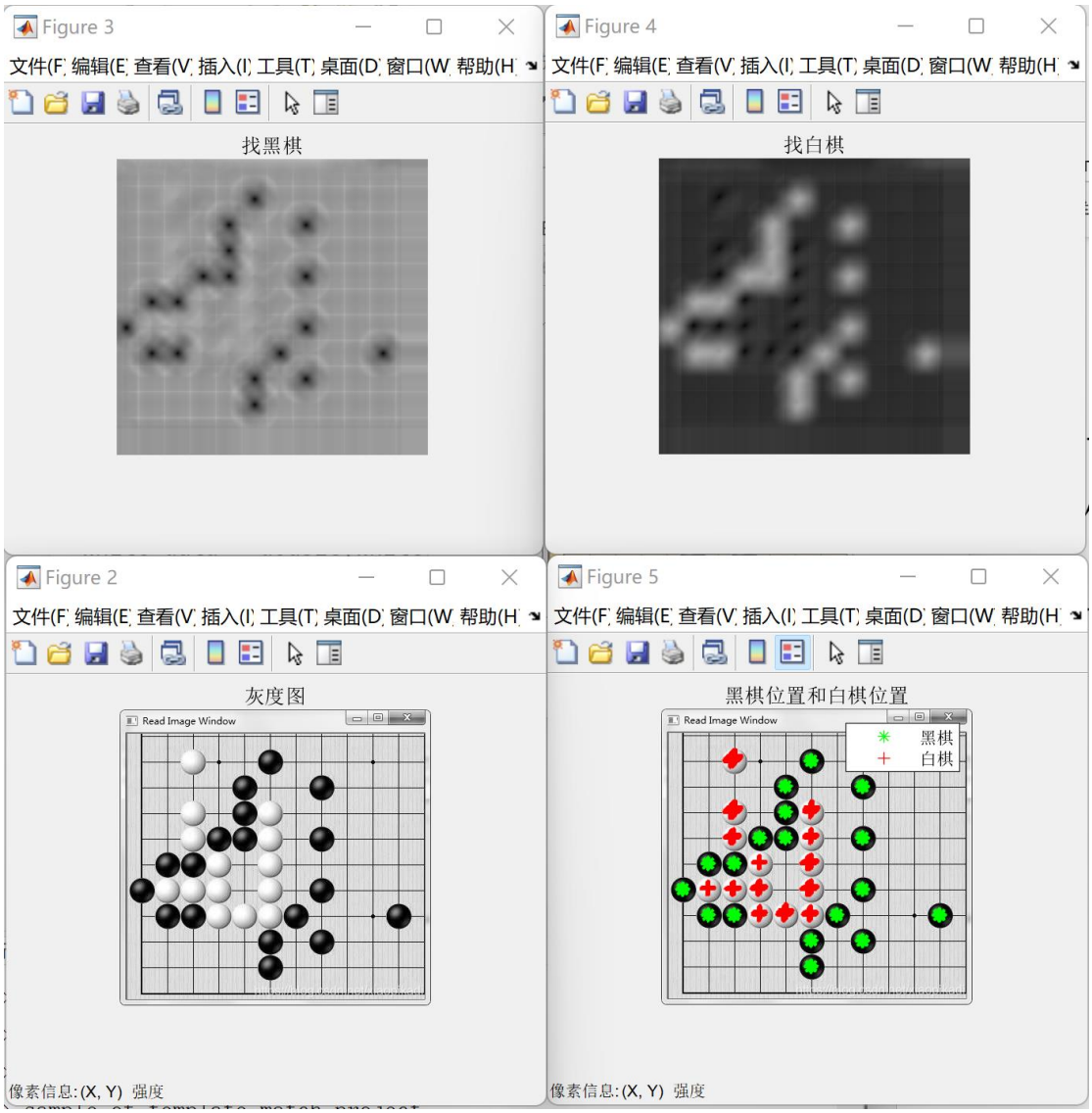
1.找棋子



2. 完成 RANSAC 直线检测与圆检测；

作答

1. 首先读取图像的灰度图，再获取黑白两棋的模板，经过两个 for 循环，确定出当前循环的窗口坐标，该窗口的值和模板的值相减取绝对值，元素求和并赋给输出图像。



2.

基本思路如下，首先生成 80 个随机散点，然后再添加 20 个沿圆分布的散点。尝试 1000 次，每次随机选取 3 个点，根据这三个点拟合出圆，并计算出圆心和半径。计算各个点到圆心的距离，如果和半径的差值小于容差值，添加到选择点中。1000 次尝试后，选出选择点最多的那一次，绘制出那一次的选择点和根据那一次的圆心和半径拟合圆。拟合方法为最小二乘法。

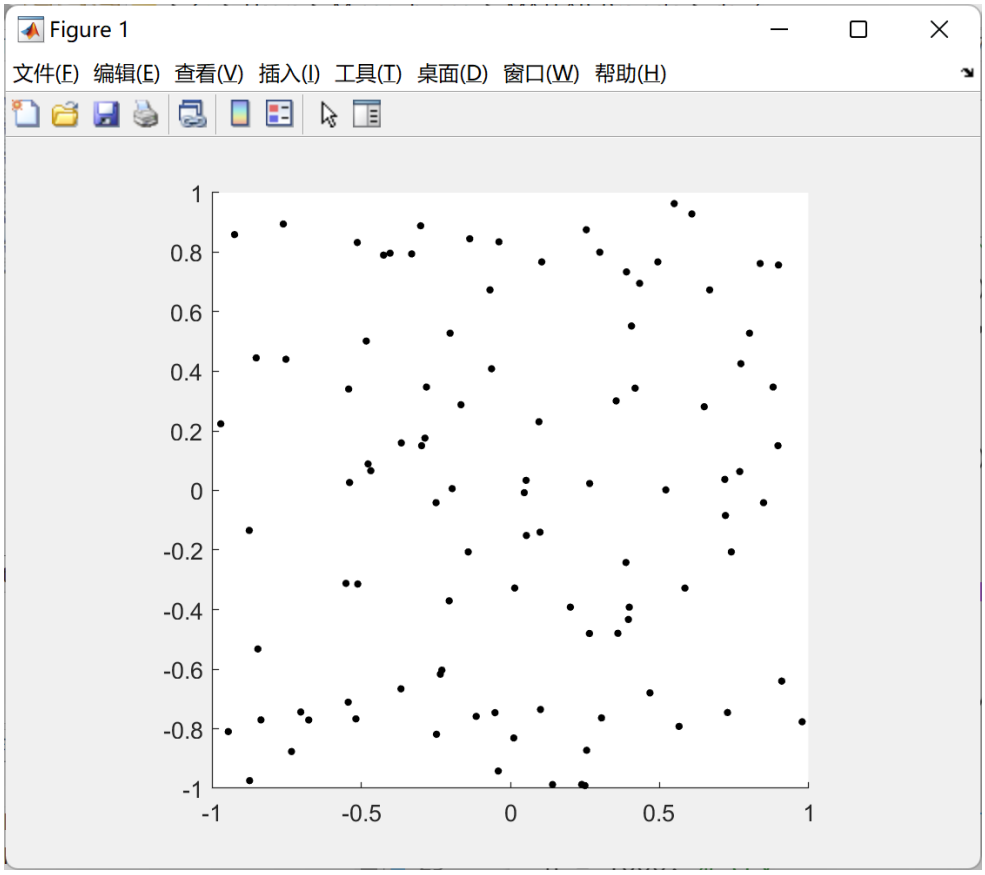


Figure 1
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1

（散点图）

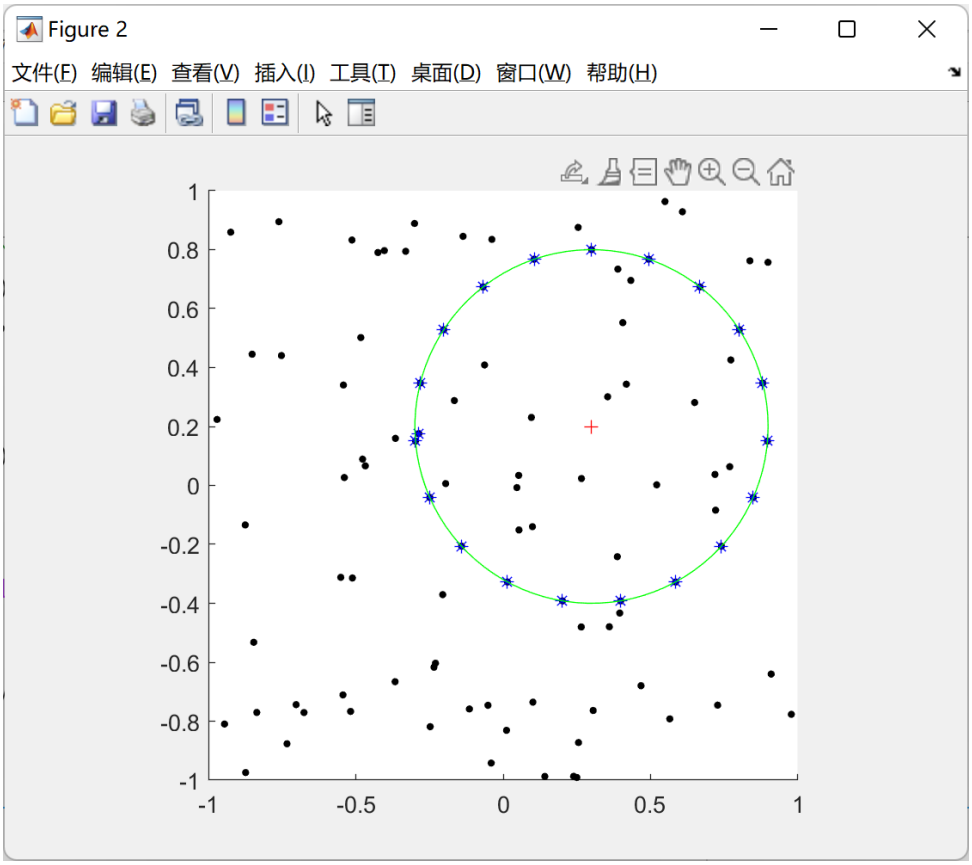


Figure 2
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1

（拟合图）

附录

模板匹配代码

```
clear all;
close all;

I= imread('../images/go.png');
imshow(I);
impixelinfo;
img=rgb2gray(I);
black = img(250:283,343:376);
white = img(50:83, 78:111);
[m,n] = size(img);
img_out1=uint8(zeros(m, n));  % 结果矩阵初始化
img_out2=uint8(zeros(m, n));

img_data = double(img);   % 数据类型转换
black_data = double(black);  % 数据类型转换
white_data = double(white);

for j= 1: m-34+1
    for i= 1: n-34+1
        %找黑棋
        window_data = img_data(j:j+34-1,i:i+34-1);% 确定
img_data 在当前循环对应的窗口坐标;
        black_abs_data = abs(window_data-black_data);   % 提取出
window_data 窗口 与 black_data 相减，取绝对值，;
        img_out1(j:j+33,i:i+33) =
sum(black_abs_data(:))/1000; % 把所有元素相加求 sum, 并赋值给
img_out 的相应格子。
        %找白棋
        white_abs_data = abs(window_data-white_data);
        img_out2(j:j+33,i:i+33) = sum(white_abs_data(:))/1000;

    end
end
figure;
imshow(img);
impixelinfo;
title('灰度图');
```

```
figure;
imshow(img_out1);
title('找黑棋')
figure;
imshow(img_out2);
title('找白棋')
figure;
imshow(img);
hold on;
impixelinfo;
[r,c] = find(img_out1 < 40);
plot(c+15, r+15, 'g*');
[r,c] = find(img_out2 < 30);
plot(c+15, r+15, 'r+');
title('黑棋位置和白棋位置');
legend('黑棋','白棋');
```

RANSAC 直线检测代码

```
clear
close all
clc

%% 生成 60 个随机点，然后添加 11 个点的直线，打乱点的顺序
Points = rand(60,2);
line = 0:0.1:1;
y = -0.5 * line + 0.8 + (rand(1,11)-0.5)/50;  % try

Points = [Points; cat(1, line, y)'];
scatter(Points(:,1), Points(:,2), 10, 'k', 'filled');
hold on
grid on
daspect([1 1 1]);

Points(:,3) = rand(size(Points,1), 1);
Points = sortrows(Points, 3);

X = Points(:, 1);
Y = Points(:, 2);

%% 尝试 1000 次
n = 1000; % try
tol = 0.02; % 容差值
```

```
for i = 1 : n
    choose = randperm(length(X));  % 所有样本点随机排序
    choose = choose(1:2);   % 随机选取 2 个样本点
    choose_x = X(choose);
    choose_y = Y(choose);

    % 1，根据这 2 个样本点，生成直线方程（待完成）。。。
    t = polyfit(choose_x, choose_y, 1);

    % 2，根据容差值 tol，结合直线方程生成容差带，并统计落在容差带内的点的个数（待完成）。。。
    all_distance = abs(t(1)*X-Y+t(2))/sqrt(t(1)^2+(-1)^2);
    choose = all_distance < tol;
    find_t(i,:) = t;
    choose_num(i) = sum(choose);
    choose_point{i} = choose;

end

%%3，迭代结束后，找出有效样本点数最多的容差带，并显示输出其对应的有效样本点，以及对应的直线（待完成）。。。
[m,index] = max(choose_num);
t = find_t(index, :);
choose = choose_point{index};
choose_x = X(choose);
choose_y = Y(choose);


plot(choose_x, choose_y, 'b*', choose_x, polyval(t,choose_x), 'r-');
legend('所有点', '内点', '最终拟合直线')
hold on
grid on
daspect([1 1 1]);
```

RANSAC 圆检测代码

```
clear
close all
clc

%% 生成 80 个随机点，然后添加 20 个点圆
```

```
Points = 2*(rand(80,2) - 0.5);
scatter(Points(:,1),Points(:,2), 10, 'ko', 'filled');
hold on;
daspect([1 1 1]);

theta = linspace(0, 2*pi, 20);
x = 0.6*sin(theta) + 0.3;
y = 0.6*cos(theta) + 0.2;

scatter(x, y, 10, 'ko', 'filled');
savefig('./scatter_circle');

Points = [Points; cat(1, x, y)'];
X = Points(:, 1);
Y = Points(:, 2);

%% 尝试 1000 次
n = 1000; % try
tol = 0.02; % 容差值
for i = 1 : n
    choose = randperm(length(X));  % 所有样本点随机排序
    choose = choose(1:3);   % 随机选取 3 个样本点
    choose_x = X(choose);
    choose_y = Y(choose);

    [x0, y0, R] = circlefit(choose_x, choose_y);
    % 点到圆心的距离
    all_distance = sqrt(abs((X-x0).^2 + (Y-y0).^2));
    choose = abs(all_distance - R) < tol;

    choose_num(i) = sum(choose);
    choose_point{i} = choose;
    r(i) = R;
    center_x(i) = x0;
    center_y(i) = y0;
end

[m_num,index] = max(choose_num);
choose = choose_point{index};
R = r(index);
center_x = center_x(index);
center_y = center_y(index);
choose_x = X(choose);
choose_y = Y(choose);
```

```
% 绘制结果
openfig('./scatter_circle.fig');
plot(choose_x, choose_y, 'b*');
alpha=linspace(0,2*pi,100);
plot(center_x+R*cos(alpha),center_y+R*sin(alpha),'g-');
plot(center_x, center_y, 'r+');


%% 最小二乘法拟合圆
function [xc,yc,R]=circlefit(x,y)
% CIRCLEFIT fits a circle in x,y plane
% x^2+y^2+a(1)*x+a(2)*y+a(3)=0
n=length(x);
xx=x.*x;
yy=y.*y;
xy=x.*y;

A=[sum(x) sum(y) n;sum(xy) sum(yy) sum(y);sum(xx) sum(xy)
sum(x)];
B=[-sum(xx+yy);-sum(xx.*y+yy.*y);-sum(xx.*x+xy.*y)];
a=A\B;
xc = -0.5*a(1);
yc = -0.5*a(2);
R = sqrt(-(a(3)-xc^2-yc^2));
end
```