**鱼群算法解决太阳影子定位问题**

**臧泽林**

# 代码块1

% 本代码使用鱼群算法解决二维的搜索问题

% 根据图片中的影子长度的信息对图片拍摄的地点进行确定

% 首先建立数学模型，描述影子与经纬度的关系

% 然后通过搜索算法对拍摄的位置进行确定

% 输入的数据时储存于q2.txt文件中的不同时间的影子的长度，与时间日期数据

% @author: Jon Zang，杭州，15168307480，zangzelin@gmail.com

% date:2016-9-23

clc

clear all

%% 读入数据

xlsfile = 'q2.txt'; %打开文件，输入数据

data =xlsread(xlsfile);

NET = [];

NET.FEN= data(:,1);

NET.MIAO= data(:,2);

NET.X= data(:,3);

NET.Y= data(:,4);

%% 求解信息的初始化

init\_shadow();

%用来验证信息初始化是否成功的代码

% for i = 60:150

% for j = 3:60

% point = [i j];

% foodlevel(i,j) = getfoodlevel(point , NET);

% end

% end

%% 鱼群参数设置

%鱼的数量

fish.N = 20;

%鱼的活动范围，搜索区间二维

fish.d1max = 180;

fish.d2max = 180;

fish.d1min = 0 ;

fish.d2min = 0;

%鱼的移动步长

fish.step = 0.8;

fish.maxpation = 15;

fish.feelbound = 10;

%% 主程序

%放置人工鱼

X1=rand(fish.N,1)\*(fish.d1max-fish.d1min )+fish.d1min;%横坐标

X2=rand(fish.N,1)\*(fish.d2max -fish.d2min)+fish.d2min;%纵坐标

for i = 1:fish.N

WUCHA(i,1) = getfoodlevel( [X1(i,1) X2(i,1)],NET ) ;

end

X=[X1 X2 WUCHA ];%初始位置记录

fish.fishlist=X;%当前位置

%觅食

figure();

plot(109 ,19,'\*');

hold on

axis([0,180,0,180]);

for i = 1:50

wuchahe = 0;

for j = 1:fish.N

location = fish.fishlist(j,1:2);%导入人工鱼的位置

wucha =fish.fishlist(j,3) ;%计算误差

%更新人工鱼的位置

[newlocation , wucha1 ]= findindfood( location,fish.maxpation, fish.step ,fish.feelbound ,NET, wucha);

%在图中描绘此此收敛

if i > 1

if newlocation ~= [0 0]

arrow(location,newlocation-location);

end

end

hold on

if newlocation ~= [0 0]

%对此次收敛进行记录

fish.fishlist(j,1:2) = newlocation;

fish.fishlist(j,3) = wucha1;

end

wuchahe =wucha1+wuchahe;

end

%记录误差

wuchajilu(i) = wuchahe;

end

%%

%画图

plot(109 ,19,'\*');

hold on

axis([0,180,0,180]);

for j = 1:fish.N

location = fish.fishlist(j,1:2);

wucha =fish.fishlist(j,3) ;

plot(location(1),location(2),'dr'); hold on

end

# 代码块2

function [ newlocation,wucha] = findindfood( location,maxpation, step,feelbound ,NET, wucha)

% 人工鱼的觅食函数，输入人工渔的位置location，耐心maxpation，步长step，感知区域feelbound，网络参数NET，和当前误差wucha

% 返回觅食后的误差wucha和位置newlocation

newlocation = [ 0 0 ];

for i = 1: maxpation

newlocathion1 = location + rand(1,2)\*feelbound\*2-feelbound; %产生随机的位置

newfoodlevel = getfoodlevel( newlocathion1,NET );%测试产生的位置的适应度函数

if newfoodlevel<wucha && newlocathion1(1)>0 ... % 测试新的位置是否满足要求，如果满足要求则记录心的位置，不满足要求则重新产生新的位置，直到耐心用尽或者满足要求

&& newlocathion1(1)<180 && newlocathion1(2)>0 ...

&& newlocathion1(2)<180

newlocation= location+(newlocathion1-location)\*step; %计算新的位置

wucha = getfoodlevel( newlocation,NET );%计算误差

break;

end

end

end

# 代码块3

%编程时用作测试的函数，并没实际作用

point = [70,40]

for i = 1:21

cha = (point(1)-120)\*4;

shijiao =(min(i)+cha)/60\*15;

h(i) = asind(sind(point(2))\*sind(chiwei)+cosd(point(2))\*cosd(chiwei)\*cosd(shijiao));

%yingchang(i) = 3/tand(h(i));

banggao(i) = tand(h(i))\*yingchang2(i);

% f(i) = acosd((sind(h(i))\*sind(point(2))-sind(chiwei))/(cosd(h(i))\*cosd(point(2))));

end

m = mean(banggao);

for i =1:21

wucha(i) = abs(m/tand(h(i))-yingchang2(i));

end

wuchab= mean(wucha)

# 代码块4

function [ wuchab ] = getfoodlevel( point,NET )

%获取当前位置的适应度函数

% 输入网络参数和当前点的位置，返回适应度函数

for i = 1:21

%计算该参数在每个点的适应度，存入banggao矩阵

cha = (point(1)-120)\*4;

shijiao =(NET.min(i)+cha)/60\*15;

h(i) = asind(sind(point(2))\*sind(NET.chiwei)+cosd(point(2))\*cosd(NET.chiwei)\*cosd(shijiao));

%yingchang(i) = 3/tand(h(i));

banggao(i) = tand(h(i))\*NET.yingchang2(i);

% f(i) = acosd((sind(h(i))\*sind(point(2))-sind(chiwei))/(cosd(h(i))\*cosd(point(2))));

end

m = mean(banggao);%求取适应度的平均值

for i =1:21

wucha(i) = abs(m/tand(h(i))-NET.yingchang2(i));%计算总的误差并且返回

end

wuchab= mean(wucha);

end

# 代码块5

%% 时间变换

for i = 1:21

NET.min(i) = ((NET.FEN(i)-12)\*60+NET.MIAO(i));

end

%% 计算影长

for i = 1:21

a = [0,0];

b = [NET.X(i),NET.Y(i)];

NET.yingchang2(i) = juli(a,b);

end

%% 计算影角

for i = 1:21

b = [NET.X(i),NET.Y(i)];

NET.yingjiao(i) = acosd((1+NET.yingchang2(i)^2-juli([0,1],b))/(2\*NET.yingchang2(i)));

end

%% 参数计算

NET.N = 108;%4yue18ri

pi = 3.1415926;

NET.b=2\*pi\*(NET.N-1)/365/3.14\*180;

NET.delta=(0.006918-0.399912\*cosd(NET.b)+0.070257\*sind(NET.b)-0.006758\*cosd(2\*NET.b)+0.000907\*sind(2\*NET.b)-0.002697\*cosd(3\*NET.b)+0.00148\*sind(3\*NET.b));

NET.delta =NET.delta/3.14\*180;

NET.chiwei = NET.delta;%计算赤纬

# 代码块6

function arrow(P,V,color)

%二维空间中画箭头

%输入：P=[x0,y0],V=[a,b]

%将以P（x0，y0）为起点，以（x0+a，y0+b）为终点画出箭头

%可以进一步修改为三维空间到箭头,或者是以P为起始点,V为终点的箭头图像

color = 'k'; %设置颜色

x0 = P(1);y0 = P(2); %确定初始点

a = V(1); b = V(2); %确定结束点

l = max(norm(V), eps)/10; %计算长度

u = [x0 x0+a]; v = [y0 y0+b]; %计算向量

hchek = ishold;

plot(u,v,color) %画出箭头主要部分

hold on

h = l - min(.2\*l, .2) ;v = min(.2\*l/sqrt(3), .2/sqrt(3) ); %计算箭头的两端的起止

a1 = (a\*h -b\*v)/l;

b1 = (b\*h +a\*v)/l;

plot([x0+a1, x0+a], [y0+b1, y0+b], color) %画出箭头的一边

a2 = (a\*h +b\*v)/l;

b2 = (b\*h -a\*v)/l;

plot([x0+a2, x0+a], [y0+b2, y0+b], color) %画出箭头的另外一边

if hchek == 0

hold off

end