%% 计算路径平滑度函数

function [path\_smooth] = cal\_path\_smooth(pop, x)

[n, ~] = size(pop);

path\_smooth = zeros(1, n);

%循环计算每一条路径的平滑度

for i = 1 : n

single\_pop = pop{i, 1};

[~, m] = size(single\_pop);

%路径有m个栅格，需要计算m-1次

for j = 1 : m - 2

% 点i所在列（从左到右编号1.2.3...）

x\_now = mod(single\_pop(1, j), x) + 1;

% 点i所在行（从上到下编号行1.2.3...）

y\_now = fix(single\_pop(1, j) / x) + 1;

% 点i+1所在列、行

x\_next1 = mod(single\_pop(1, j + 1), x) + 1;

y\_next1 = fix(single\_pop(1, j + 1) / x) + 1;

% 点i+2所在列、行

x\_next2 = mod(single\_pop(1, j + 2), x) + 1;

y\_next2 = fix(single\_pop(1, j + 2) / x) + 1;

%path\_smooth(1, i) = path\_smooth(1, i) + abs(atan(abs(x\_now - x\_next1)/abs(y\_now - y\_next1))-atan(abs(x\_next2 - x\_next1)/abs(y\_next2 - y\_next1)));

%a2 = (x\_now - x\_next1)^2 + (y\_now - y\_next1)^2;

%b2 = (x\_next2 - x\_next1)^2 + (y\_next2 - y\_next1)^2;

c2 = (x\_now - x\_next2)^2 + (y\_now - y\_next2)^2;

%angle = (a2 + c2 - b2) / (2 \* sqrt(a2) \* sqrt(c2));

%若大于4小于等于8，说明此栅格与隔一个的栅格隔一行或一列且列或行相邻

if c2 < 8 && c2 > 4

path\_smooth(1, i) = path\_smooth(1, i) + 5;

%若大于1小于等于4，说明此栅格与隔一个的栅格为对角，也可能或同行或同列垮了一格

elseif c2 <= 4 && c2 > 1

path\_smooth(1, i) = path\_smooth(1, i) + 30;

%若等于1，说明此栅格与隔一个的栅格是上下或左右相邻，其路径不如直接从此格到邻格，显然冗余了。

elseif c2 <= 1

path\_smooth(1, i) = path\_smooth(1, i) + 5000;

%否则不设置值，也即值为0，此时此栅格与隔一个的栅格是正方形对角的关系，最好。

end

end

end