

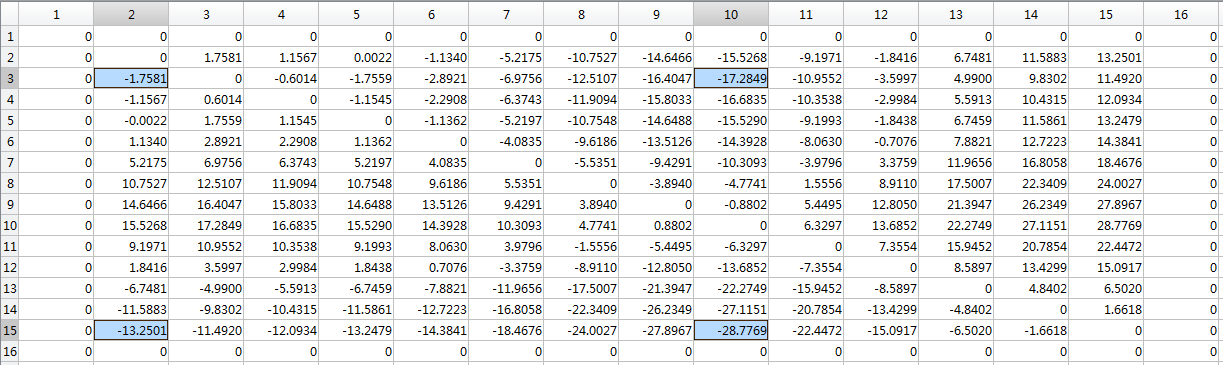
Trajectory of 220th agent.



Absoluta angle of every point (vector O)



Absolute angle of every three point (vector A)



Matrix U (I make the 0 margin to find local max(min) value conveniently). Notice the local max(min) value with blue, correspond to row and line 2,3,10,15. As the existence of 0 margin, the real row and line should be [1,2,9,14].

Then [1,2,9,14]’s corresponding absolute angle in Vector A will be the vector S (S ⊂ A):

S = [-3.6395,-5.3976,11.8873,-16.8896].

I think you have understood with above part. And the following part is what I neglect in paper.

Filter Process:

Next we need to make sure the every difference between two neighbor elements in S is bigger than 6 degree threshold (with enough difference to represent fitting-line’s angle), and delete others (e.g. S2). Then we get the vector G (G ⊂ S) :

G = [-3.6395 ,11.8873,-16.8896].

(PS: if we make threshold smaller, we will get more elaborate fitting-lines. Otherwise, we will get less precise result.)

Classify process:

After getting vector G, I use it to represent fitting angle. Traverse Vector O from first element, if it’s closer to G1, then it belong to the first team of fitting-line. If it’s closer to G2, it belong to the second team of fitting-line. Same process for G2 and G3. (Actually, I use the team amount to classify them. But I can’t remember detailed process, and I will check the code again.)

After all the fitting-line is clear, we can confirm where the subgoals are.

Actually the process is with respect to angle all the time.

1. Find local max(min) angle;
2. Filter invalid angles and get the represent angles for fitting;
3. Classify points to different team of fitting-line by their absolute angle.

I think this explanation will be helpful for you to understand the whole fitting process.