Decentralized multi-agent system in communication restricted environment

Pratyush Gupta
Supervisor : Professor Indranil Saha

Problem

1. n drones
2. No communication
3. Coordinated flight towards goal

Assumptions

1. Each drone is equipped with sensors to correctly calculate the position of the other drones
2. Each drone knows its own position
3. No other information is known
4. Each drone has the same goal region

Solution overview

1. Design a path planner such that it allows a drone to predict collisions
2. Then reduce the problem to a velocity assignment problem
3. Solve velocity assignment problem using NoRORCA

Step 1. Path planner

Drone position : D
Goal : G

Compute paths of the form D-S-G

Use any underlying algorithm for it(Eg. Modified RRT)

Step 2. Reduce to velocity assignment

Idea : Drone i computes path D_i - S_i - G
Then it assumes drone j computes D_j - S_i - G

Lets move to the board to justify this idea

Continued

Collision risk only arises when

D_1 cannot see S_2

OR

D_2 cannot see S_1

Assuming conditions hold for no collision

For a drone i:
Let $r_i = S_i - D_i$

Desired velocity is

$$V_i = C_i \left(r_i / \|r_i\|\right)$$

The velocity assignment problem

Each drone assumes that every other is also aiming for the same point that it is

So it can set **$V_j = C_j (r_j / ||r_j||)$**

Thus the problem reduces to computing C_j such that there is no collision

Why does this idea make sense?

Two drones that care actually on a collision course will end up at the same velocity assignment problem

Assuming the conditions specified earlier hold

Now we need a way to solve this problem such that the solutions computed by these two drones **agree**

Solving this problem

We set

$$v_i^{pref} = \alpha_0(d_i - s)$$

Then the optimization function becomes

$$\boxed{J = min \sum_i ||v_i - v_i^{pref}||^2}$$

Constraints

We want the constraints to be linear

Then the problem becomes a quadratic optimization problem

Can be approximately solved in polynomial time

Look to the board for the constraints

Do optimization algorithms work?

Say that we use an approximate algorithm. A solution x is epsilon-good if

$$|J(x) - J(x_{opt}| \leq \epsilon$$

Aim:

Show that two epsilon-good solutions will be close to each other

This is important to ensure the drones on a collision course can agree on a solution

Proof

The essential properties in use are:

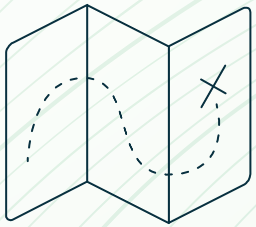
1. Hessian of objective is PD
2. Feasible region is convex
3. Objective has *considerable second derivative*

For the proof, look to the board

Finally

We have a complete algorithm to solve the problem we started with.

We will now continue with experiments to validate the theoretical framework



Questions