IoT Project 2023

Distributed Drone Patrolling

Airborne Dynamics

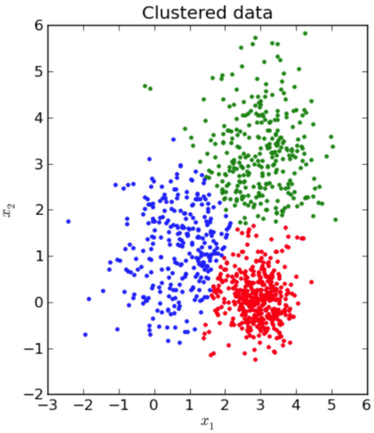
University of Rome "La Sapienza"

May 31st, 2023

Algorithm

Clustering

- *Divide et Impera*.
  - Divide the point cloud in clusters.
- One cluster per drone.
  - Clearly defined work load.
- Clusters are **disjoint**.
  - Might implement *Fuzzy Clustering* in the future.
- We chose **K-Means**.
  - Tests showed that it is the overall better method.



Priority Function

- Our distance metric D is made of two parts:

  1. d: euclidean distance between points.
  2. B: *bonus* decided by the current state of the destination target.

$$D = d\alpha - B\beta \tag{1}$$

- Given two points p_1 and p_2 we compute the euclidean distance d between them:

$$d = \sqrt{(p_{1,x} - p_{2,x})^2 + (p_{1,y} - p_{2,y})^2 + (p_{1,z} - p_{2,z})^2} \tag{2}$$

Priority Function (2)

- p_2: destination
- a: current age of information of p_2
- t: threshold of p_2
- w_a: age of information weight
- w_v: violation weight

- The **Bonus** will be:

$$A = a^2 w_a \tag{3}$$

$$V = (a - t)\, w_v \tag{4}$$

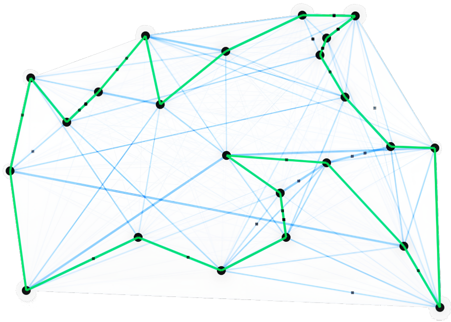
$$B = \begin{cases} A \cdot V, & \text{if } V > 0. \\ \frac{A}{|V|}, & \text{if } V < 0. \end{cases} \tag{5}$$

Ant-Colony

- This algorithm send out a number of ants to different paths on a computed graph, guided by the distance metric, and then returns the best one found.
- We use the *ant-colony* approach when:
  1. The cluster has outliers far away from the center of mass.
  2. The thresholds of the targets in the cluster are similar.
  3. The cluster is big.

Ant-Colony (2)

1. The *ants* **probabilistically** choose the next step of the path
2. Performs **backward updates** to said probability
    - Adding the inverse of the path's distance.
3. Repeat for a number of steps
4. the *best path* found has the **highest probability**.
5. *Example*:



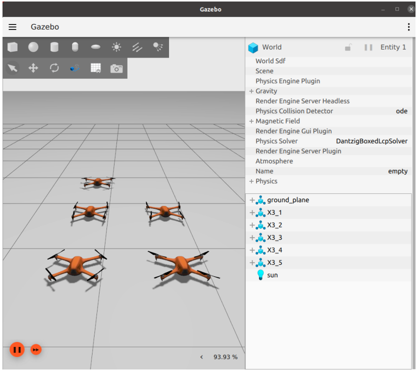
Greedy

- The greedy solution at **every step** evaluates the targets' priorities in the cluster
- Points the drone towards the **most urgent** one based on our dynamic distance metric.
- We choose *greedy* when:
  1. The wind is too strong.
  2. The cluster is small.
  3. The thresholds of the targets in the cluster are very different.

Drones

Movement

- Removed the *stop and rotate* phase
    - Much faster overall movement
- We assign the clusters based on the **starting position** of the drones
    - To avoid collisions after takeoff while trying to reach the clusters



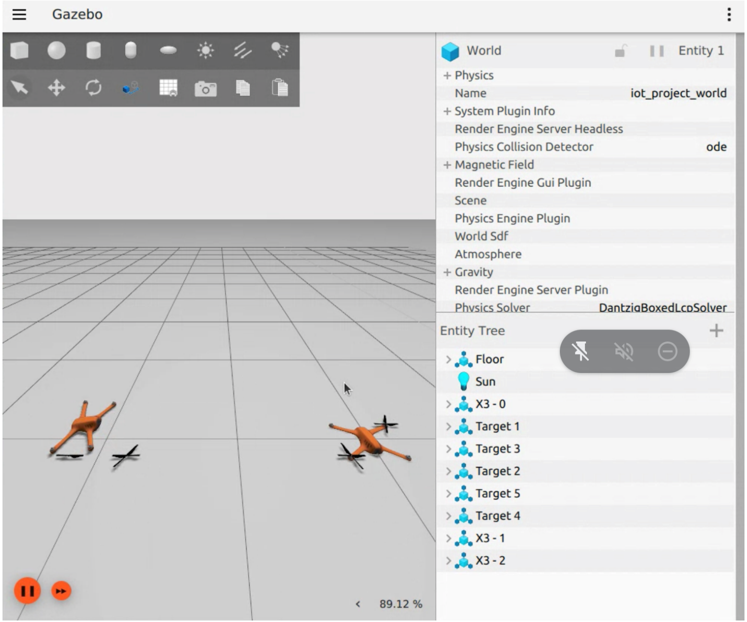
World Sdf	
Scene	
Physics Engine Plugin	
+ Gravity	
Render Engine Server Headless	
Physics Collision Detector	ode
+ Magnetic Field	
Render Engine Gui Plugin	
Physics Solver	DantzigBoxedLcpSolver
Render Engine Server Plugin	
Atmosphere	
Name	empty
+ Physics	

+ ground_plane
+ X3_1
+ X3_2
+ X3_3
+ X3_4
+ X3_5
sun

Wind

- Change in the distance precision
  - Smaller ϵ value
  - delay the *deceleration* phase
- Issues with *angular vector* in windy scenarios
  - Drones rotate and loose the orientation

Fin.



World Entity 1

+ Physics

Name iot_project_world

+ System Plugin Info

Render Engine Server Headless

Physics Collision Detector ode

+ Magnetic Field

Render Engine Gui Plugin

Scene

Physics Engine Plugin

World Sdf

Atmosphere

+ Gravity

Render Engine Server Plugin

Physics Solver DantzigBoxedLcpSolver

Entity Tree

- Floor
- Sun
- X3 - 0
- Target 1
- Target 3
- Target 2
- Target 5
- Target 4
- X3 - 1
- X3 - 2