# Distributed Autonomous Systems 2024-25

## Course Project

This project assignment is designed to give the opportunity to apply the theoretical concepts learned during the course. Students are allowed to get inspiration from any source they prefer, such as the tutorial papers in [1, 2, 3].

## Task 1 – Multi-Robot Target Localization

In multi-robot target localization problems, a team of robots measures (noisy) locations of targets, aiming to cooperatively estimate their locations. As shown in Figure 1, a fleet of $N \in \mathbb{N}$ robots makes noisy observations of a target vehicle. Each robot is equipped with sensors that can measure the targets' locations within a limited area. The objective of the task is to implement a distributed strategy that allows the fleet to localize the targets in a cooperative fashion.

we want to simulate a framework in which N number of observers is much larger w.r.t. T number of targets! (N>>T)

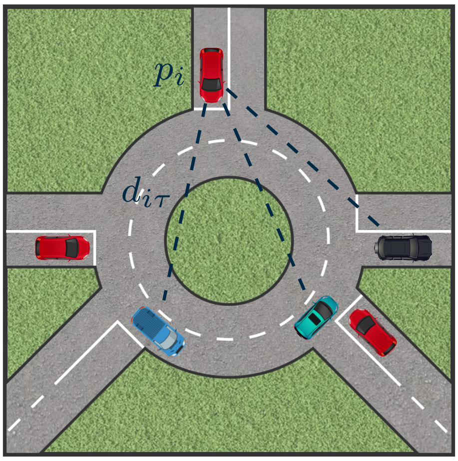


Figure 1: Example of multi-robot target localization problem. A team of red vehicles observing a pool of three targets (black, blue, and green vehicles). The blue, dashed lines represent the noisy measurements made one vehicle in the team.

### Task 1.1 – Distributed consensus optimization

it must be used in part 1.2!

In this task, you are required to implement a distributed optimization algorithm to solve a consensus optimization problem. This task will be the basis for the cooperative localization problem.

*i)* As a preliminary task, implement the *Gradient Tracking* algorithm to solve a consensus optimization problem in the form

$$\min_{z} \sum_{i=1}^{N} \ell_i(z)$$

with decision vector $z \in \mathbb{R}^d$, $d > 1$, and quadratic functions $\ell_i : \mathbb{R}^d \mapsto \mathbb{R}$, for all $i = 1, \ldots, N$. You can extend the scalar case $d = 1$ provided during the coding lectures.

*ii)* Run a set of simulations to test the effectiveness of the implementation. Moreover, provide a set of solutions that includes different weighted graph patterns (e.g., cycle, path, star) whose weights are determined by the Metropolis-Hastings method. Finally, for each simulation, plot the evolution of the cost value and of the norm of the gradient of the cost function at the current solution estimates.

**Hint.** Use a logarithmic scale on the $y$-axis when plotting both the cost function and its gradient square norm. Be sure to plot the norm of the sum of the local gradients.

## Task 1.2 – Cooperative multi-robot target localization → static scenario

In this task, leveraging the work done in Task 1.1, you are required to implement the *Gradient Tracking* algorithm to enable the fleet of robots to cooperatively localize the targets. A pictorial representation of the scenario is given in Figure 1.

creating scenario

*i)* Generate $N \in \mathbb{N}$ robot locations (e.g., red vehicles in Figure 1), denote by $p_i \in \mathbb{R}^d$.

*ii)* Generate $N_T \in \mathbb{N}$ target positions and then, for each robot $i$, provide noisy measurements of its distance $d_{i\tau} \in \mathbb{R}_{\geq 0}$ from each target $\tau \in 1, \dots, N_T$.

implementing solver (cntrllr)

*iii)* Implement the *Gradient Tracking* algorithm adopting the following local cost functions, for all $i = 1, \dots, N$.

for 1 T we need at least three N!

$$\ell_i(z) := \sum_{\tau=1}^{N_T} \left( d_{i\tau}^2 - \|z_\tau - p_i\|^2 \right)^2$$

where $z = \text{col}(z_1, \dots, z_{N_T}) \in \mathbb{R}^{dN_T}$ is the optimization variable. Specifically, the $\tau$-th block-component of $z$ denotes the estimated $\tau$-th target position $z_\tau \in \mathbb{R}^d$, which the robots should agree upon.

simulate!

*iv)* Provide comprehensive simulations in which the robots localize multiple targets.

*v)* For each simulation, plot *(i)* the evolution of the cost function and *(ii)* the evolution of the norm of the gradient of the cost function across the iterations (use semi-logarithmic scale on the $y$-axis).

Check quality? Compare with centralized solution!

CoPilot myDadons!

## Task 2 – Aggregative Optimization for Multi-Robot Systems

Consider a team of $N$ robots. Denote the position of each robot $i$ at iteration $k \in \mathbb{N}$ with $z_i^k \in \mathbb{R}^d$ and denote with $z^k \in \mathbb{R}^{dN}$ the stack vector of the locations of the whole team. Consider the case of a two-dimensional environment, resulting in $\boxed{d = 2}$.

### Task 2.1 – Distributed aggregative optimization

pure python code!

The goal of this task is to implement a distributed control algorithm that allows the robots to stay as close as possible to private targets, while keeping the fleet tight due to communication constraints. An illustrative example is shown in Figure 2.

this is what we want to figure out!

Define an appropriate potential function! (mabye more then one)

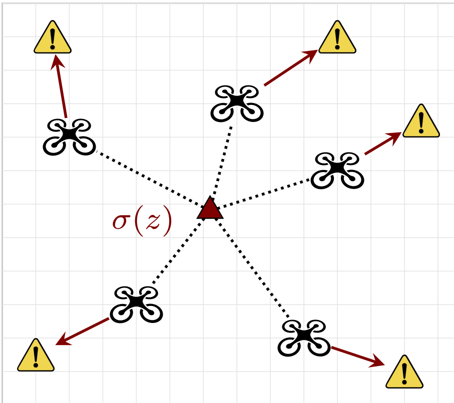


Figure 2: Example of the aggregative optimization scenario. Each robot wants to move toward a certain private target, keeping the fleet tight due to communication constraints.

This scenario can be formalized as the *Aggregative Optimization* problem

$$\min_{z \in \mathbb{R}^d} \sum_{i=1}^{N} \ell_i(z_i, \sigma(z)), \quad \text{with} \quad \sigma(z) := \frac{1}{N} \sum_{i=1}^{N} \phi_i(z_i)$$

where $\ell_i : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ and $\phi_i : \mathbb{R}^d \to \mathbb{R}^d$. In this scenario, robot $i$ is aware only of the decision variable $z_i$ and of the functions $\ell_i$ and $\phi_i$. The global aggregative variable $\sigma(z)$ can be seen as the barycenter of the team, and, hence, $\phi_i(z_i) = z_i$ for all $i = 1, \ldots, N$.

*i)* Implement the *Aggregative Tracking* algorithm by designing the most suitable cost functions $\ell_i$, which allows the robots to

   (a) stay close to fixed, known targets;

   (b) keep the fleet tight.

*ii)* Run a set of simulations showcasing the effectiveness of the distributed strategy.

*iii)* Plot the evolution of the cost function and of the norm of the gradient of the cost function across the iterations (use semi-logarithmic scale on the $y$-axis).

*iv)* Provide an animated visualization of the team behavior (you can use templates provided during lectures).

we must rely on 2.1 tsk code!

### Task 2.2 – ROS 2 implementation

This task amounts to the practical implementation in a ROS 2 environment of Task 2.1.

*i*) Starting from the ROS 2 package on distributed algorithms (e.g., average consensus) provided during coding lectures, create a ROS 2 package implementing the *Aggregative Tracking* algorithm.

*ii*) Run the same simulations of item *iii*) of Task 2.1, plotting the same quantities as done in item *iv*) of Task 2.1.

**Hint.** You can use RViz or similar tools for visualization of simulations in *ii*).

### Notes

1. Each group must be composed of at most 3 students.
2. Any other information and material necessary for the project development will be given during project meetings.
3. Each group (all members) must attend 2 mandatory meetings with the tutor of the course, in which the work progress must be shown.
4. The project report must be written in LaTeX and must follow the main structure of the provided template.
5. Any email for project support must have the subject:
   "[DAS2025] Group X: *support request*".
6. All the emails exchanged **must be cc-ed** to prof. Notarstefano, prof. Notarnicola, dr. Pichierri, and all the other group members.

### IMPORTANT: Instructions for the Final Submission

1. The final submission **deadline** is **7 days** before the exam date.
2. One member of the group must send an email with subject
   "[DAS2025] Group X: Submission"
3. The email **must** include a link to a **OneDrive** folder, shared with prof. Notarstefano, prof. Notarnicola, dr. Pichierri, and all the other group members.
4. The final submission folder must contain:

   - `README.md`
   - `report_group_XX.pdf`
   - `report` – a folder containing the LaTeX code and a `figs` folder (if any)
   - `code` – a folder containing the code relative to the main task, including `README.md`

## References

[1] G. Notarstefano, I. Notarnicola, A. Camisa *et al.*, "Distributed optimization for smart cyber-physical networks," *Foundations and Trends® in Systems and Control*, vol. 7, no. 3, pp. 253–383, 2019.

[2] A. Testa, G. Carnevale, and G. Notarstefano, "A tutorial on distributed optimization for cooperative robotics: from setups and algorithms to toolboxes and research directions," *arXiv preprint arXiv:2309.04257*, 2023.

[3] O. Shorinwa, T. Halsted, J. Yu, and M. Schwager, "Distributed optimization methods for multi-robot systems: Part 1 – A tutorial," *IEEE Robotics & Automation Magazine*, vol. 31, no. 3, pp. 121–138, 2024.