Modeling and Control for Cyber-Physical Systems

Project Activity II

1st Alessio Giorgetti
Politecnico di Torino
Matricola: 345637
S345637@studenti.polito.it

2nd Anna Roma
Politecnico di Torino
Matricola: 345819
anna.roma@studenti.polito.it

3rd Stefano Giulianelli
Politecnico di Torino
Matricola: 347911
stefano.giulianellli@studenti.polito.it

***Abstract*—This report presents the design and evaluation of distributed control strategies for a network of magnetic levitation systems modeled as a cyber-physical system. Each agent estimates its own state using either a cooperative or local observer and applies a distributed control law based on neighbourhood information. The goal is to achieve consensus and track the reference trajectory provided by a leader node.**

I. Introduction

This report focuses on a **cooperative dynamic regulator** framework for the synchronization of multi-agent systems, where a **leader node** S_0 directs the behaviour of a network modeled as a **directed graph**. Each **agent node** S_i is represented by a Linear Time Invariant (LTI) dynamical system and exploits information shared through the communication network to perform coordinated tasks assigned by the leader.

The leader node LTI system S_0 is described by equation [1]:

$$\begin{cases} \dot{x}_0 = Ax_0 - Bu_0 \\ y_0 = Cx_0 \ , \end{cases} \tag{1}$$

which includes the term u_0, representing the *state feedback*. The leader includes the feedback directly in its model because it is required to follow a reference trajectory and remain stable on its own. Therefore, it is assumed that the leader already applies its own local control, such as:

$$u_0 = -K_0 x_0 \quad \Rightarrow \quad \dot{x}_0 = (A - BK_0)x_0$$

This local control is predefined to ensure the leader's autonomous stability, providing a reliable reference for the synchronization of the follower agents.

The follower nodes S_i, with $i = 1, \dots, N$, are modeled as LTI systems as follows:

$$\begin{cases} \dot{x}_i = Ax_i + Bu_i \\ y_i = Cx_i + Du_i \ , \end{cases} \tag{2}$$

where:

- $x_i \in \mathbb{R}^n,\ y_i \in \mathbb{R}^p$ and $u_i \in \mathbb{R}^m,\ \forall i \in \{0, 1, 2, \dots, N\}$
- A describes the internal dynamics of the system

$$\begin{bmatrix} 0 & 1 \\ 880.87 & 0 \end{bmatrix}$$

- B describes how the input influences the evolution of the status

$$\begin{bmatrix} 0 \\ -9.9453 \end{bmatrix}$$

- C defines the observer influence

$$\begin{bmatrix} 708.27 \\ 0 \end{bmatrix}$$

- D represents the direct effect of the input u_i on the output y_i

$$\begin{bmatrix} 0 \end{bmatrix}.$$

  In our case, there is no direct effect, meaning that the input u_i only acts on the dynamic of the system, trough the matrices (A, B) and not directly on the output.
- Finally, the triple (A, B, C) is sterilizable and detectable.

At this stage, follower agents do not include the local feedback term $u_i = -K_0 x_i$ in their model: this control is introduced later during the cooperative regulator design and it is not part of the initial physical system.

We also assume that the agent's state is not directly measurable because not all of its state variables (position and velocity) might be observable through sensors. This means that the output matrix $C \neq I$. As a result, each agent must estimate its own state.

The main goal is to design distributed control strategies such that the multi-agents system performs cooperative tracking problems. In this way we try to drive the global disagreement error to zero, ensuring that all agents follow the leader's reference behaviour.

The digraph $\mathcal{G} = \{\nu,\ \epsilon\}$ represents the shared information through the communication network (nodes $\{1, \dots, N\}$), between the nodes $\nu = \{v_1, v_2, \dots, v_N\}$ and the weighted arcs ϵ. We define $a_{i,j} > 0$ the weights of the arc (v_i, v_j) and they are collected in the Adjacency matrix $\mathcal{A} = \{a_{i,j} \neq 0,\ i \neq j \iff$ node a_i receives informations from node $a_j\}$. The augmented graph $\overline{\mathcal{G}} = (\overline{\nu}, \overline{\epsilon})$ represents connection between the follower nodes' graph $\mathcal{G}$ and the leader node v_0. In this way, the control law for each agent highly depends on the graph topology, as the edges connecting the nodes represent the communication links between agents. Consequently, the execution of tasks and the behaviour of each agent are strongly influenced by how the nodes

exchange information within the network.

In our project, we use a waterfall topology, a cascading structure in which agents are organized hierarchically. Choosing appropriate weights for the edges is not trivial: we have tested several configurations to determine which leads to better convergence.

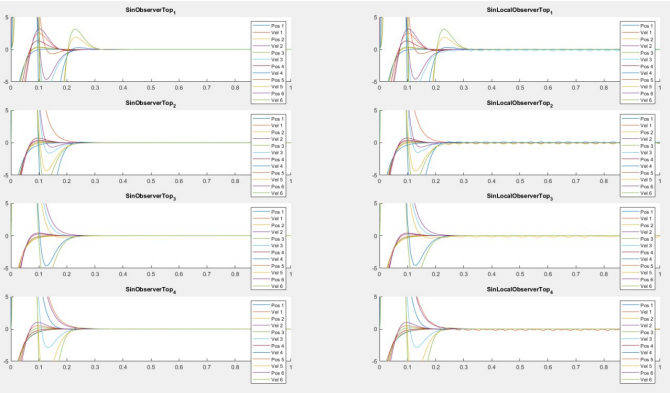

Fig. 1: Comparison of different topologies

Among the various configurations tested, the third topology shown in Figure [1] appears to converge to the leader position and velocity in the shortest amount of time, with less overshoot in position compared to the others

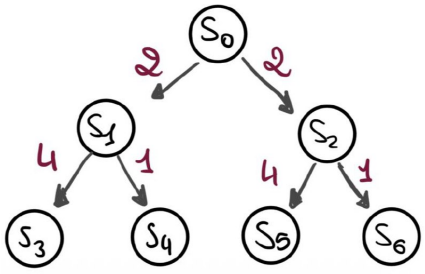

Fig. 2: Waterfall topology

This leads us to adopt the topology in Figure [2] that appears to be the most effective so far, the one in which the agents seem to follow the leader more closely. It is clear that in our configuration we respect the *Assumption 3.1* from [1]:

> The augmented graph $\overline{\mathcal{G}}$ contains a **spanning tree** with the root node being the leader node. In other words, there is a directed path (not necessarily unique) from the leader node to every follower node.

With this assumption we can try to solve the cooperative tracking problem.

II. Task 1: Distributed Regulator Based on a Distributed Neighbourhood Observer Structure

In the first task, we solve the problem in which each agent's control law can only rely on the information received from its neighbours, according to the chosen topology, using a fully distributed control scheme.

Problem formulation

We first define the **neighbourhood tracking error** of node i. It represents the difference between its estimated state and a combination of estimated states of its neighbouring nodes, indicating how much it deviates from the group behaviour:

$$\hat{\epsilon}_i = \sum_{j=1}^{N} a_{i,j}(\hat{x}_j - \hat{x}_i) \; + g_i(\hat{x}_0 - \hat{x}_i). \tag{3}$$

The Sliding Variable Feedback (**SVFB**) control protocol for each node i is based on the error $\hat{\epsilon}_i$ and designs a control law that ensures convergence and stability using only local information from neighbour nodes:

$$u_i = cK\hat{\epsilon}_i \ , \tag{4}$$

where $c \in \mathbb{R}_+$ is the **coupling gain** that controls how strongly each agent reacts to the differences with its neighbours and $K \in \mathbb{R}^{m \times n}$ is the **feedback gain** matrix and defines how an agent computes its control input based on local tracking errors. We also introduce the local output estimate $\hat{y}_i = C\hat{x}_i$, calculated using the estimated state $\hat{x}_i$. Based on this, we can express the **local output estimation error** as the difference between the actual output and its estimate:

$$\tilde{y} = y_i - \hat{y}_i \tag{5}$$

With these elements in place, we now have all the ingredients to construct the **neighbourhood output estimation error** for node i, following the same reasoning used in Task 1 (Equation [3]):

$$\xi_i = \sum_{j=1}^{N} a_{i,j}(\tilde{y}_j - \tilde{y}_i) + g_i(\tilde{y}_0 - \tilde{y}_i) \tag{6}$$

As we anticipated in the previous section, the goal is to design a cooperative observer that provides an estimate $\hat{x}_i(t)$ of the state vector x_i, such that

$$\lim_{t \to \infty} \hat{x}_i(t) = x_i \ , \ \forall i = 1, \ldots, N \ . \tag{7}$$

For this reason, we combine Equations [4] and [6] to derive the **cooperative distributed observer** of each node i, which directly depends on the information from the nodes that communicate to node i:

$$\dot{\hat{x}}_i = A\hat{x}_i + cBK\hat{\epsilon}_i - cF\xi_i - BK_0\hat{x}_i \ , \tag{8}$$

where the terms have the following roles:

- $A\hat{x}_i$: represents the natural dynamics of the estimated state.
- $cBK\hat{\epsilon}_i$: a cooperative correction term exploiting the differences between the estimates of neighboring nodes, encouraging consensus.
- $-cF\xi_i$: a classical observer correction (similar to a Luenberger observer) based on local output estimation errors, where F is the observer gain matrix.
- $-BK_0\hat{x}_i$: a local stabilizing feedback acting on the state estimate, ensuring each agent maintains stability independently of cooperation.

The local feedback term $-BK_0\hat{x}_i$ acts on the estimated state of node i to ensure individual stability regardless of communication with neighbours. The matrix gain K_0 is designed such that $A - BK_0$ is Hurwitz stable, meaning each agent remains stable even if communication fails or is delayed. This term decouples the agent's intrinsic stability from the cooperative dynamics driven by $cBK\hat{\epsilon}_i$ and $-cF\xi_i$, providing a robust baseline feedback. Thus, including $BK_0\hat{x}_i$ in the observer guarantees reliable local stability while supporting the overall cooperative control strategy. It is also important to mantain the followers node dynamic equal to the one of the leader node.

Now that we have a cooperative way to estimate the state variables of all the nodes we can obtain the complete **local closed-loop dynamics** of the agent i

$$\dot{x}_i = Ax_i + cBK\epsilon_i - BK_0x_i \tag{9}$$

In this framework, there is no external reference signal; instead, the leader acts as the reference for the follower agents. The leader dynamics is designed by appropriately choosing the matrix K_0 so that it generates the desired trajectory - whether sinusoidal, constant or ramp. Once the leader and agent nodes were fully implemented, we conducted various simulations using different leader trajectories to test the robustness of the cooperative dynamic regulator.

Experimental results

As a first test, we plotted the positions of the agents (Figure [3]), comparing their states with the leader in three different scenarios: constant, ramp, and sinusoidal trajectories. From the graph, it can be observed that all agents are able to track the leader's position, although not instantaneously. It is also noticeable that the agents which are "further" from the leader in the communication topology tend to reach convergence more slowly, suggesting the presence of a communication delay. This behaviour is consistent across all three scenarios and it is particularly evident in the constant trajectory case.

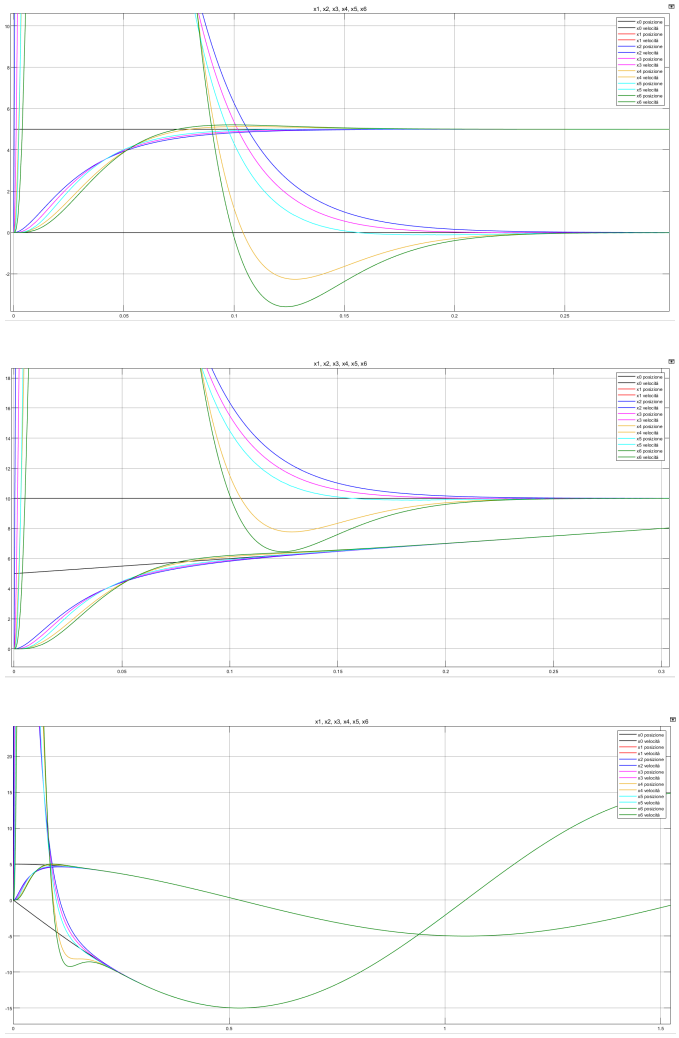

Fig. 3: Position estimation performances in different scenarios.

III. Task 2: Distributed regulator based on local observers

Problem Formulation

Now, we assume that the agent's estimated state is not computed by a cooperative observer. Instead, the estimate for each node is obtained locally using only the local output estimation error $\tilde{y}_i$. This local estimate is then used to compute the **local closed-loop dynamics** of each agent: by substituting the neighbourhood estimation error (Equation [6]) in Equation [8] with the local output estimation error defined in Equation [5].

$$\dot{\hat{x}}_i = A\hat{x}_i + cBK\hat{\epsilon}_i - cF\tilde{y}_i - BK_0\hat{x}_i \ . \tag{10}$$

Similarly to the cooperative observer in Equation [8], we observe that the feedback term is also present here. This term plays the same crucial role described in the previous section: regardless of whether the observer is local or cooperative, the feedback ensures the essential stabilization of the agent's dynamics.

Experimental results

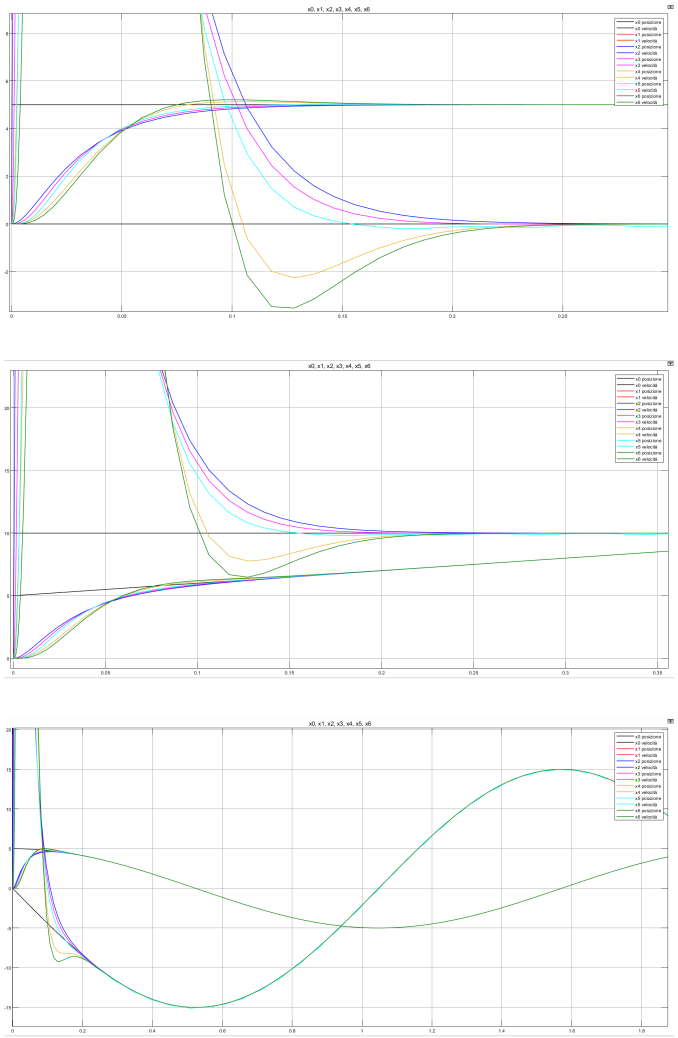

Fig. 4: Position estimation performances in different scenarios with local observer.

As in the experiment before we plotted the position of all the nodes of the system to see the tracking error with the leader. From the plots in Figure [4] we can observe a similar behaviour to that in Task 1: it is encouraging to see that all the nodes eventually converge to the leader's trajectory, which confirms that convergence is indeed achieved. However, this convergence is neither simultaneous nor instantaneous. In particular, node 5's velocity (shown in light blue) appears to behave more independently than the others and does not reach perfect convergence. Moreover, as observed in the previous case, nodes 4 and 6 tend to be the last to align with the leader's trajectory. This may be due to their relative position in the network topology, which could cause them to receive delayed or less direct information from the leader. To better understand the evolution of the system and these differences in node behavior, a more detailed comparison between the two algorithms used would be necessary.

IV. General Examination by adding noise

The most effective way to analyze which of the two observers performs better for our system is to calculate the **local disagreement error** for each node i, which measures, at each iteration k, the deviation of its state from the leader's state:

$$\delta_i(k) = x_i(k) - x_0(k) \ , \tag{11}$$

and the vector of **global disagreement error**, containing δ_i for all N nodes:

$$\delta = col(\delta_1, \delta_2, \dots, \delta_N) = x - \mathbf{1} \ \otimes x_0 \ . \tag{12}$$

As stated in the book [1], the goal of cooperative tracking is for all follower agents to converge to the state of the leader, that is, $\lim_{t\to\infty} \delta(t) = 0$, by being asymptotically stable.

From the plots in Figures [3] and [4], we expect the disagreement error to tend towards zero for both algorithms. Although this outcome is ideal by showing that both approaches achieve consensus, it does not allow us to clearly distinguish which method performs better in practice.

To address this, we introduce a **noise on the local output** y_i. This allows us to highlight differences between the two algorithms that may not affect coordination, but do influence estimation accuracy. The results of adding noise to the output measurements of each node are shown in Figure [5], where we calculated the local disagreement error for each observer.

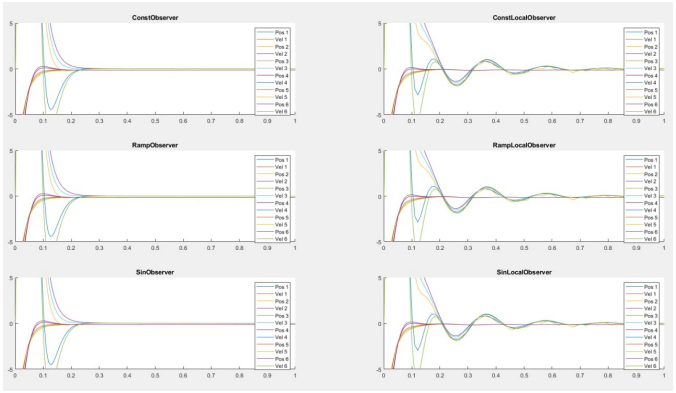

Fig. 5: Comparison of δ_i behavior with adding noise

Figure [5] compares the Global Observer (left) and the Local Observer (right) for each node, after applying the noise to each node in each scenario. Both observers show the local disagreement error on the state converging point wise to zero, as expected. However, the velocity disagreement error behaves differently. Although velocity is the derivative of the state,

$$\frac{d}{dt}(x - x_0) = \dot{x} - \dot{x}_0,$$

it describes the dynamics of the global disagreement error δ, so it can be rewritten in the form:

$$\dot{\delta} = A_c\delta + B_c\tilde{x} \ , \tag{13}$$

where:

- $A_c = \mathbf{I}_N \otimes A - c(L+G) \otimes BK$ describes the ideal closed dynamics of the system, in absence of noise or disturbances;
- $B_c = c(L+G) \otimes BK$ depends on the cooperative control law (u_i) and the topology of the graph $\mathcal{G}$;
- $\tilde{x} = \hat{x} - x$ is the global error vector of the estimate of the state, that highly depends on the observer.

Since we are working in the presence of measurement noise, it is important to analyze how each observer influences the error dynamics. While the matrix A_c remains structurally the same for both the local and global observer, the actual system response varies depending on the observer design. In noisy scenarios, in error dynamics we must also consider the additional term $B_c\tilde{x}$, which involves the state estimation error . This term captures the discrepancy between the estimated and true states and directly affects how accurately the system follows the expected behavior described by A_c.
In the case of the *global observer*, the noise effect is mitigated by the distributed nature of the estimation: information is shared between neighboring agents and estimation errors are smoothed out by cooperation. As a result, the real system dynamics stays close to the ideal behavior and the disagreement error tends to zero smoothly, demonstrating asymptotic stability.
In contrast, with a *local observer*, each node estimates the state based only on its own output measurements without exchanging information with neighbors. This isolation leads to a higher sensitivity to noise: the estimation error is not damped by the network and the noise propagates directly into the controller and then into the system dynamics. Consequently, the resulting disagreement error dynamics exhibits persistent oscillations, even if they still converge point wise to zero.
Following this logic, it highlights the importance of **inter-agent communication** in distributed control systems. When nodes cooperate, they collectively absorb and attenuate the impact of disturbances, resulting in more robust and stable global behavior. In realistic settings where noise is unavoidable, the use of a global (cooperative) observer can significantly enhance performance and convergence quality, making a strong case for cooperative estimation strategies.

V. General Examination by tuning parameters

Having said that, we have seen that, when noise is introduced, the two observers respond differently. We now aim to validate these observations and better understand how the parameters that have been chosen influence the estimation behavior.
So far, we have worked with the following set of parameters:

- $c = 1.5$ (from $u_i = cK\hat{\epsilon}_i$): this is the coupling gain used to implement the static *SVFB control protocol* for each node $i \in \mathcal{N}$. According to Theorem 3.1 in [1], for the global disagreement error to asymptotically converge to zero, it must hold that
$$c \geq \frac{1}{2\min_{i\in\mathcal{N}} \mathrm{Re}(\lambda_i)} .$$
In our case, this condition appears to be satisfied.

- $Q = \mathbf{I}_n$: this is the state weighting matrix used in the computation of the LQR or observer gain, assigning equal importance to all state components.

- $R = 1$: this is the measurement noise covariance or control effort weighting scalar, used to balance the trade-off between state estimation performance and control effort.

We then modify the parameters as follows:

Parameter	Original Value	Modification	New Value
c	1.5	$\times 50$	75
Q	$\mathbf{I}_n$	$\times 100$	$100 \cdot \mathbf{I}_n$
R	1	$\times 100$	100

In Figure [6] we tested these new parameters individually using the sinusoidal trajectory and compared the results between the two observers.

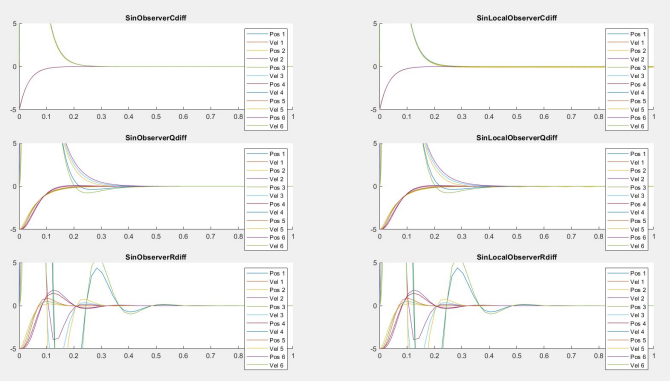

Fig. 6: Comparison of the two observers under sinusoidal trajectory with individual parameter modifications.

The plots show the disagreement error for position and velocity across the agents, under sinusoidal input. For both the coupling gain c and state penalty Q, the *local observer* shows significantly reduced oscillations compared to previous results in Figure [5]. However, after zooming in, we can observe some oscillations going toward zero, particularly for nodes 5 and 3. These nodes are more distant in the network (see Figure [2]) and are associated with higher edge weights compared to nodes 4 and 6, which are similarly distant but have lower weights. This may indicate, as previously suggested in Section [II], that the network topology significantly influences the accuracy of the state estimation for each node.
The Global Observer remains stable across all scenarios (as expected) and also demonstrates slightly faster convergence

when parameters Q and c are increased.
Finally, in the bottom row, increasing R ($\Rightarrow$ placing less trust in measurements) leads to worse performance in both observers: oscillations are increased and the convergence becomes slower and more unstable. We also observe that nodes farther from the leader tend to be more negatively affected, suggesting that the impact of large R is more severe for less connected or more peripheral nodes.

VI. Conclusions

Throughout this project, we have shown that the communication topology among agents plays a fundamental role in achieving cooperative tracking. Although we referred to our network structure as a *waterfall* topology (Figure [2]), it essentially functions as a *spanning tree* (as defined in [1]) with the leader at its root. Regardless of the name, this structure has proven essential in ensuring that all follower agents are able to track the reference trajectory generated by the leader.

We observed that the effectiveness of communication is influenced not only by the graph connectivity but also by the weights assigned to each edge, which represent the quality or strength of the information exchange. In particular, nodes farther from the leader tend to receive information with more delay. Moreover, among these distant nodes, those connected through higher-weighted edges experience worse estimation performance compared to those with lighter weights, suggesting that edge weighting significantly affects signal quality and system responsiveness.

Despite these limitations, the use of an appropriate observer allows for consistent and reliable information sharing. Even in topologies where certain nodes are disadvantaged in terms of position or connection weight, the system remains robust against noise. This robustness is particularly important in real-world applications, where noise is always present. Therefore, designing algorithms that can handle disturbances is critical for practical deployment.

Our analysis shows that the **global (cooperative) observer** is especially well-suited for such environments. It leverages inter-agent communication to mitigate the effects of noise and provides state estimates that closely track the ideal system behavior. As a result, it offers a more stable and accurate performance compared to its **local** counterpart.

In conclusion, communication is vital not only for information propagation across the network, but also for accurate state estimation at each node. Our results clearly demonstrate that cooperation among agents and shearing neighbors' information enhances overall system performance and resilience, validating the robustness of distributed observer, based control strategies in networked systems.

References

[1] F. L. Lewis, H. Zhang, K. Hengster-Movric, and A. Das, *Cooperative Control of Multi-Agent Systems: Optimal and Adaptive Design Approaches*, Springer, London, 2014.