Università degli Studi di Napoli Federico II

Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione

Coverage control of a fleet of robots on a 1D and 2D space

Project report for Control of Complex Systems and Networks course

Abstract

This work addresses the coverage control problem for a fleet of mobile robots operating in both one-dimensional and two-dimensional spaces. The study begins by exploring a network of heterogeneous mobile sensors constrained to move on a circular path in a 1D space, implementing control laws to optimize the network's coverage while reducing the effect of bounded position measurement errors. The robustness of the controller is further tested under intermittent sensor readings, and while moving in a formation. Subsequently, the problem is extended to a 2D environment where robots adaptively position themselves in regions of high sensory interest using only local measurements. A consensus-based adaptation law is introduced to allow sensory information propagation across the network, resulting in more optimal coverage. Simulation results demonstrate the effectiveness of the implemented control laws in achieving near-optimal configurations.

Mario Valentino
P38000205

Contents

List of Figures

Chapter 1

One-dimensional coverage control

The coverage control problem for mobile sensor networks has received significant attention in recent years due to its wide-ranging applications, such as environmental monitoring, surveillance, and target tracking. In this chapter, we address the problem of a network of heterogeneous mobile sensors, where sensors have different maximum velocities and are constrained to move on a one-dimensional circular space. The goal is to optimize the network coverage by minimizing a coverage cost function. Specifically, we want to ensure the fastest possible arrival time from any sensor to any point on the circle.

The challenge becomes more complex when real-world conditions, such as position measurement errors, are considered. These errors can significantly affect the network ability to achieve an optimal configuration, as the sensors rely on their estimated positions to make movement decisions. In this work, the upper bounds of the sensors' measurement errors are known a priori, which allows for the design of estimation algorithms to mitigate the impact of inaccuracies in position measurements. We will also test the robustness of the control law when the sensors' measurements are intermittent, causing a delay in the response of each agent to changes in the environment.

Finally, we will test the performance of a pinning control on the closed loop system to find out if it's possible to move the sensors while keeping their formation.

The chapter is structured as it follows: in Section 1, we give the mathematical formulation of the problem as given by Song et al. in [1]. In Section 2, we implement their proposed control law. Finally, we test its robustness in Section 3 by using intermittent distance readings and introducing pinning control. Section 4 will draw conclusions on the performed experiments.

1.1 Problem formulation

The one-dimensional formulation of the problem goes as follows [1]: consider a mobile sensor network consisting of n sensors, constrained to move along a circle. We will assume discrete-time first order dynamics:

$$q_i(k+1) = q_i(k) + u_i(k) \tag{1.1}$$

The sensors are heterogeneous, meaning they possess different maximum velocities λ_i, known by all sensors in the network. The objective is to optimize the coverage of the network by minimizing a coverage cost function. The cost function is defined as the largest arrival time from the sensor network to any point on the circle. Formally, the cost function $T(q_1, q_2, \ldots, q_n)$ can be expressed as:

$$T(q_1, q_2, \ldots, q_n) = \max_{q \in S} \min_{i \in \{1,2,\ldots,n\}} \frac{d(q_i, q)}{\lambda_i} \tag{1.2}$$

Where:

- S is the set of all points on the circle.
- $d(q_i, q)$ is the shortest distance on the circumference between sensor i and point q.
- λ_i is the maximum velocity of sensor i.

This function represents the largest time for any sensor to reach a point on the circle, which is what we want to minimize.

The sensor positions are initialized randomly along the unit circle. The network topology is assumed to be fixed and each node is neighbor with its immediate clockwise and counter clockwise node at the initial positions. Formally, we can write the neighboring set as:

$$\mathcal{N}_i = \{i-1,\, i+1\} \tag{1.3}$$

Now and throughout all the work, the indices $i+1$ and $i-1$ are handled in a circular manner. This means that when $i = n$, the expression $i+1$ refers to the first sensor (i.e., $i+1 = 1$), and when $i = 1$, the expression $i-1$ refers to the last sensor (i.e., $i-1 = n$). This circular indexing ensures that the control law applies consistently across all sensors, regardless of their position on the circumference.

To account for real-world conditions, measurement errors in the sensor positions are considered. Specifically, each sensor has a bounded position measurement error, where only the upper bounds δ_i of the measurement errors are known in advance. The measured distance of sensor i from sensor j at time instant k is expressed as:

$$d_{i,j}(k) = d^*_{i,j}(k) + e_{i,j}(k) \quad \forall i,\, \forall j \in \mathcal{N}_i \tag{1.4}$$

The term $d^*_{i,j}(k)$ is the real distance between the two nodes. The implementation of a function that computes such distance on a circumference requires accounting for the periodic nature of the circle. Specifically, we compute the counterclockwise distance to the *next* node, treating it as a positive value, and the clockwise distance to the *previous* node, treating it as a negative value. In this implementation, the measurement error $e_{i,j}(k)$ is defined as a sinusoidal function s.t. $|e_{i,j}(k)| \leq \delta_i$. The expression used is:

$$e_{i,j}(k) = \left(1 + 3\sin\left(\omega_i k + 10(j-i)\phi_i\right)\right)\frac{\delta_i}{4} \quad \forall i,\, \forall j \in \mathcal{N}_i \tag{1.5}$$

where:

- ω_i is the frequency associated with sensor i,
- ϕ_i is the phase shift associated with sensor i,
- δ_i is the upper bound of the measurement error for sensor i.

The task is to design distributed control laws that can drive the sensor network to an optimal configuration, minimizing the coverage cost function while accounting for the measurement errors. The control law used in this work was introduced by Song et al. in [2], and then enriched with distance estimation law in [1] to deal with the measurement errors.

$$u_i(k) = \lambda_i \,\mathrm{sat}_{\pm 1}\left(\tilde{u}_i(k)\right) \tag{1.6}$$

$$\tilde{u}_i(k) = \eta_i \left[(\lambda_{i-1} + \lambda_i)\tilde{d}_{i,i+1}(k) + (\lambda_i + \lambda_{i+1})\tilde{d}_{i,i-1}(k)\right] \tag{1.7}$$

η_i is a positive control parameter. The term $\tilde{d}_{i,j}(k)$ is an estimate of the distance from i to j. It is computed as:

$$\tilde{d}_{i,j}(k) = \frac{\check{d}_{i,j}(k) + \hat{d}_{i,j}(k)}{2}, \quad j \in \mathcal{N}_i \tag{1.8}$$

i.e. the average between the upper bound $\hat{d}_{i,j}(k)$ and the lower bound $\check{d}_{i,j}(k)$ of the estimation. The estimation bounds are updated according to the following iterative equations:

$$\check{d}_{i,j}(k+1) = \max\{\bar{d}_{i,j}(k+1) - \bar{\delta}_{i,j}, \check{d}_{i,j}(k) + u_j(k) - u_i(k)\} \tag{1.9}$$

$$\hat{d}_{i,j}(k+1) = \min\{\bar{d}_{i,j}(k+1) + \bar{\delta}_{i,j}, \hat{d}_{i,j}(k) + u_j(k) - u_i(k)\} \tag{1.10}$$

where:

$$\bar{d}_{i,j}(k) = \frac{d_{i,j}(k) - d_{j,i}(k)}{2} \tag{1.11}$$

$$\bar{\delta}_{i,j} = \frac{\delta_i + \delta_j}{2} \tag{1.12}$$

$$\check{d}_{i,j}(0) = d_{i,j}(0) - \bar{\delta}_{i,j}, \tag{1.13}$$

$$\hat{d}_{i,j}(0) = d_{i,j}(0) + \bar{\delta}_{i,j}$$

By denoting $\bar{e}_{i,j}(k) = \frac{e_{i,j}(k) - e_{j,i}(k)}{2}$, Song at al. proved that with sufficiently low gains η_i, and if $\bar{e}_{i,j}(k)$ reaches its minimum and maximum value in finite time steps, the control law in Equation 1.6 will converge to 0 as time goes to infinity. Furthermore, the coverage cost function in Equation 1.2 reaches its minimum $T^* = 2\pi / \sum_{i=1}^{n} \lambda_i$ if and only if [2]:

$$\frac{\bar{d}(q_i, q_{i+1})}{\lambda_i + \lambda_{i+1}} = \frac{\bar{d}(q_j, q_{j+1})}{\lambda_j + \lambda_{j+1}}, \quad \forall i, \forall j \in \mathcal{N}_i \tag{1.14}$$

Since the system under analysis is subject to measurement errors, the nodes will not reach perfectly the consensus value in Equation 1.14. However, if additionally $\sum_{i=1}^{n} \delta_i < 2\pi \min_{i \in \mathcal{I}_n} \frac{\lambda_i + \lambda_{i+1}}{2\lambda_M + \sum_{i=1}^{n} \lambda_i}$, with $\lambda_M = \max_i \lambda_i$, the network of sensors will be driven to a static configuration in a neighborhood of T^* [2].

The control in 1.6 is saturated to $\pm\lambda_i$, meaning that the sensors can move clockwise and counter-clockwise with a velocity up to their maximum velocity. However, in a real setting of a fleet of robots moving along a line, it may be important to design the control law so that the initial robots order is preserved, effectively preventing collisions. In this case, the saturation of each control input is between 0 and the maximum velocity λ_i. With this adjustment, it is proved by Song et al. that the previous theoretical results still hold, and the order is preserved during the task. The expected downside is a slower convergence to a larger neighborhood of T^* as the simulations will show in Section 2.

1.2 Control law implementation

The control law is implemented using the Simulink scheme represented in Figure 1.1. At each step, every agent gets a noisy distance reading from its neighbor. Then, the distance reading is processed by the estimator block, shown with more detail in Figure 1.2. The filtered distance readings are then used to apply the control law to each agent.

The parameters used for the simulations are listed in table 1.1.

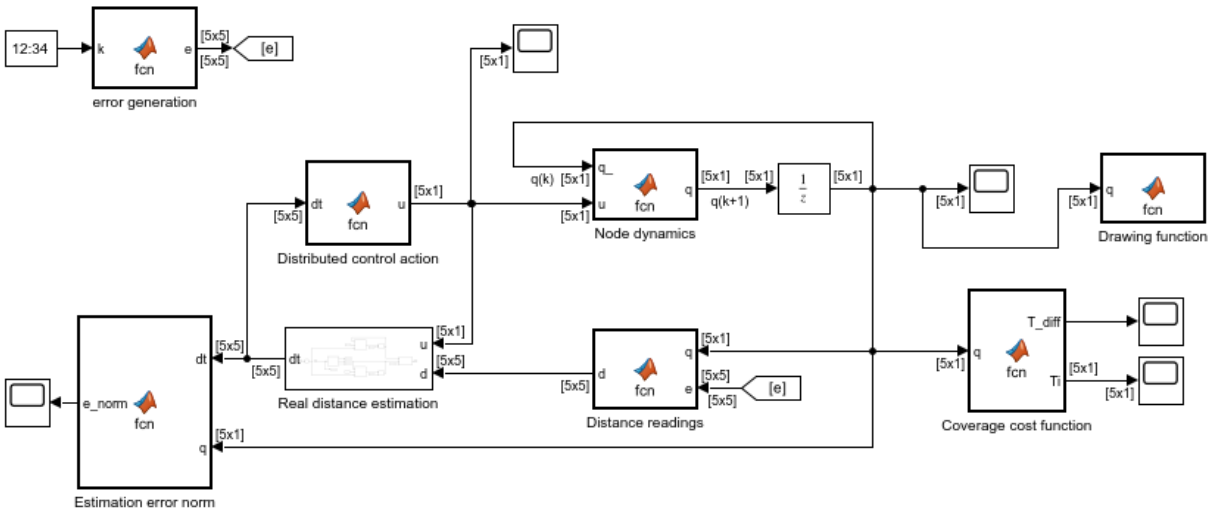

Figure 1.1: Simulation scheme for the control law with distance estimation

Parameter	Description	Value
n	Number of sensors	5
λ	Maximum velocity of each sensor	$\text{rand}(1,n) \cdot 4.5 + 0.5$
δ	Upper bound on measurement error	$\text{deg2rad}(\text{rand}(1,n) \cdot 36)$
ω	Error frequency	$\text{rand}(1,n) \cdot 2$
ϕ	Error phase shift	$\text{rand}(1,n) \cdot \pi$
η	Control gains	$0.005 \cdot \text{ones}(1,n)$

Table 1.1: Parameters used in the simulations.

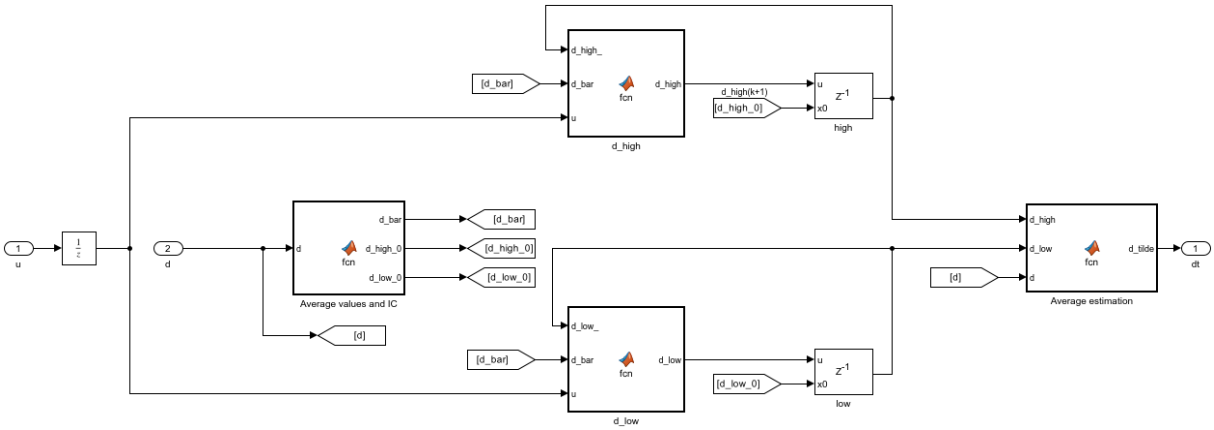

Figure 1.2: Distance estimation block

First, the stability of the closed loop system is assessed when no measurement error is present and without order preservation. Figure 1.3 shows that the closed loop system effectively achieves consensus value in Equation 1.14, therefore the cost function is minimized.

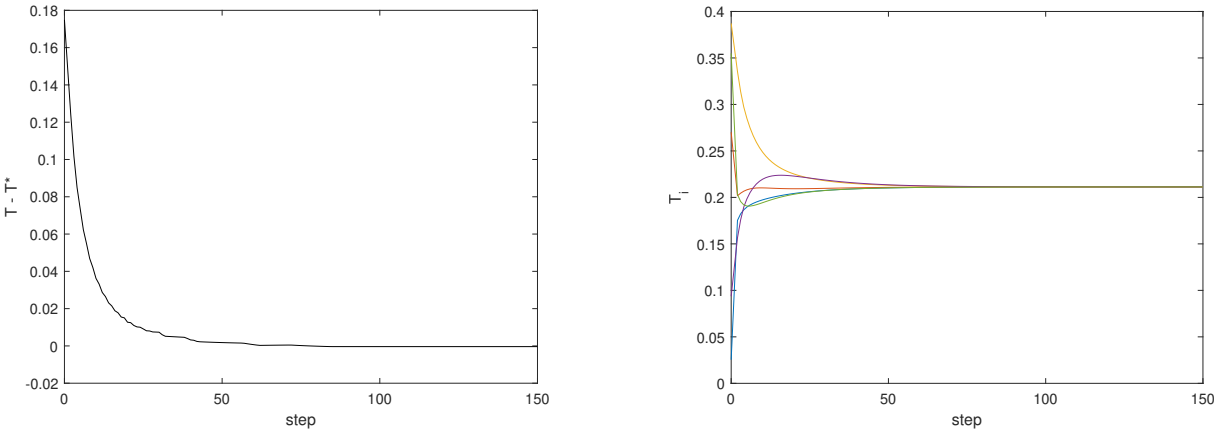

Figure 1.3: Optimal value and consensus achieved with no measurement error

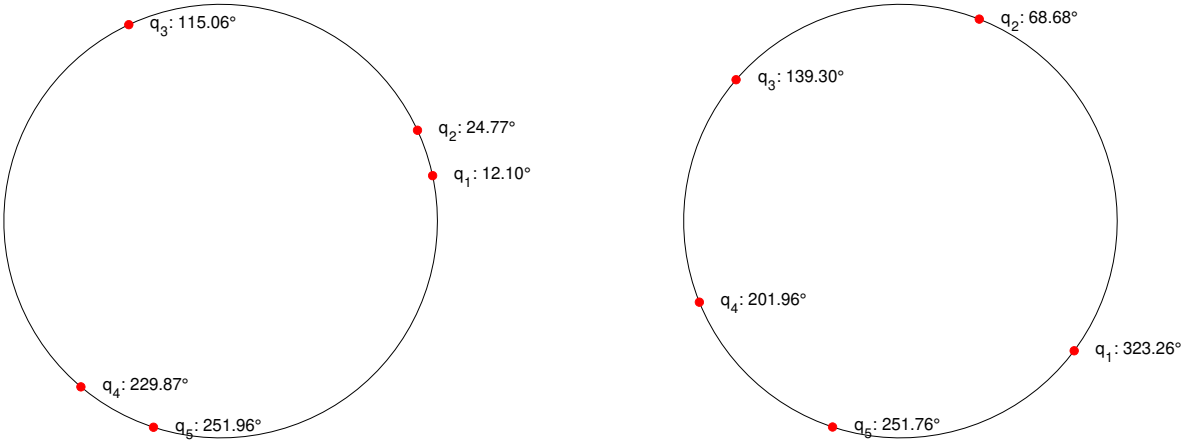

Figure 1.4: Initial (left) and final (right) sensor positions with no measurement error

The following Figures show the simulations on the system when measurement errors are applied. As expected, consensus is not perfectly achieved but the control law without order preservation is capable of bringing the agents to a near optimal configuration, and converges to zero in finite time.

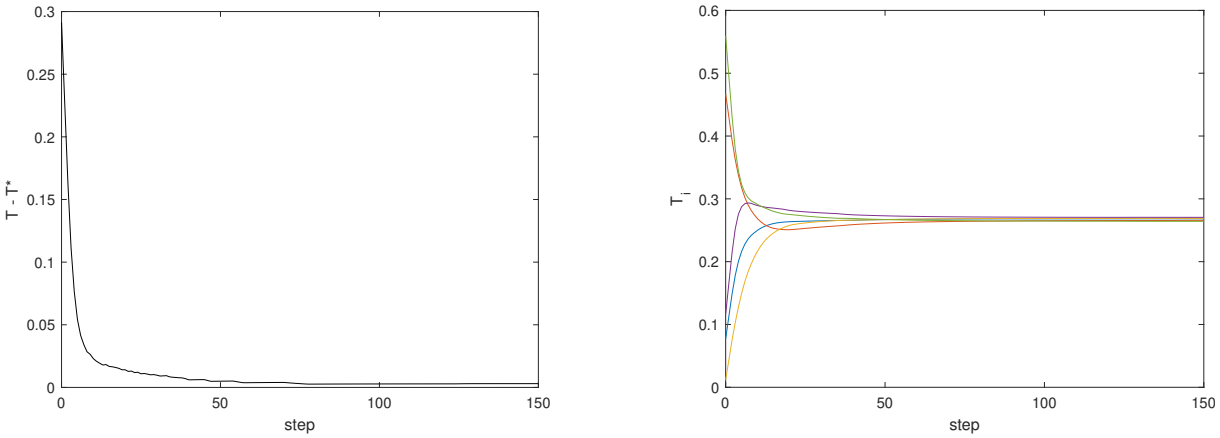

Figure 1.5: Optimal value and consensus almost achieved with measurement error and no order preservation

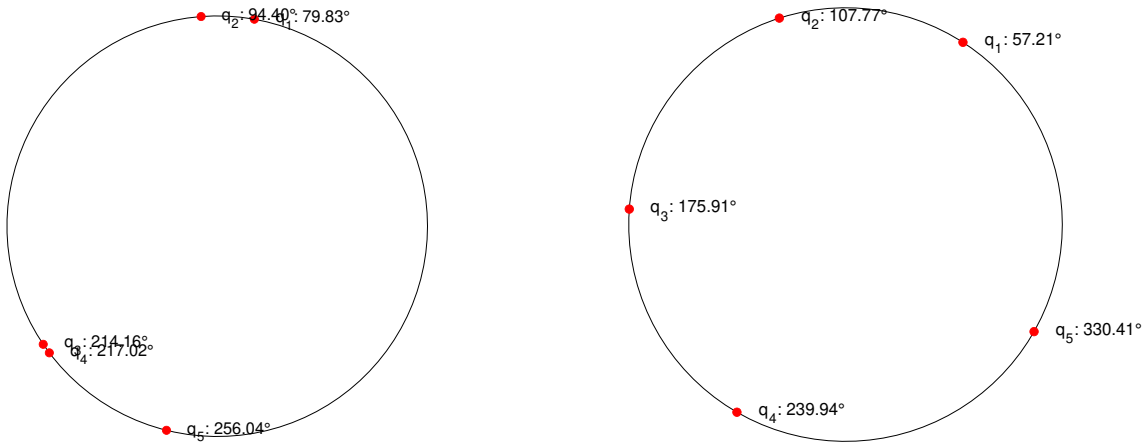

Figure 1.6: Initial (left) and final (right) sensor positions with measurement error and no order preservation

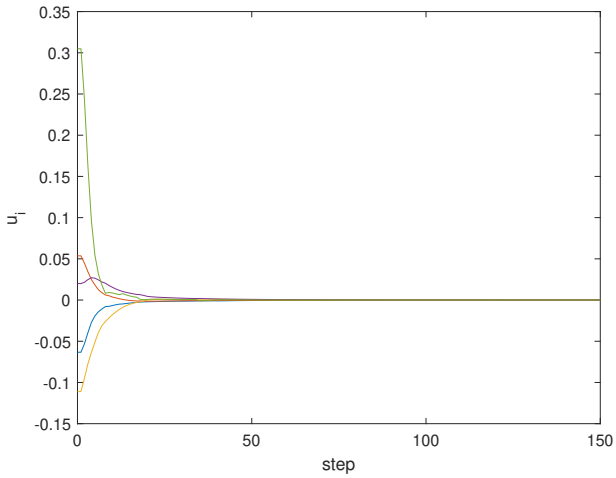

Figure 1.7: Control actions with measurement error and no order preservation

Finally, we test the control law with order preservation. The control law is still capable of achieving a near optimal configuration. However, the settling time and the steady state error are greater (note that the time scale in the following figures are larger).

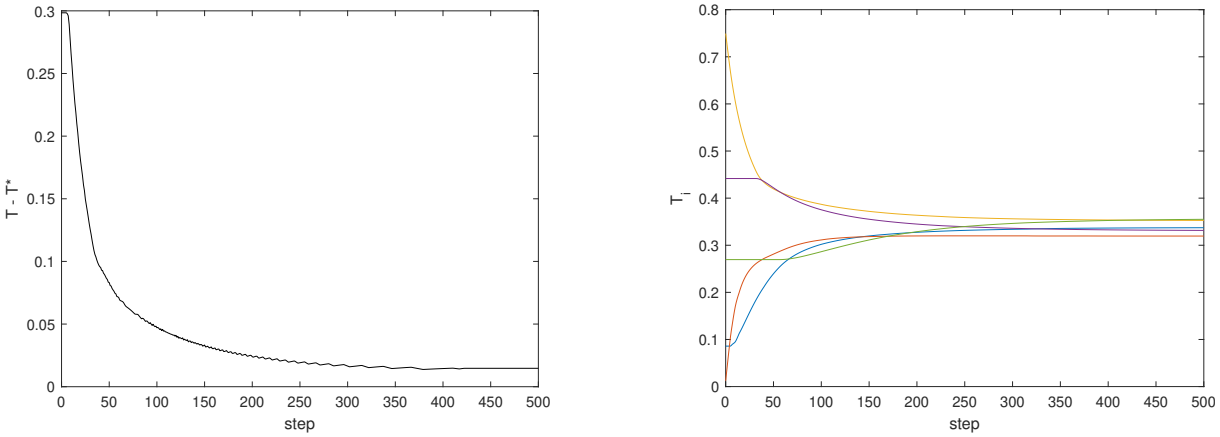

Figure 1.8: Optimal value and consensus almost achieved with measurement error and order preservation

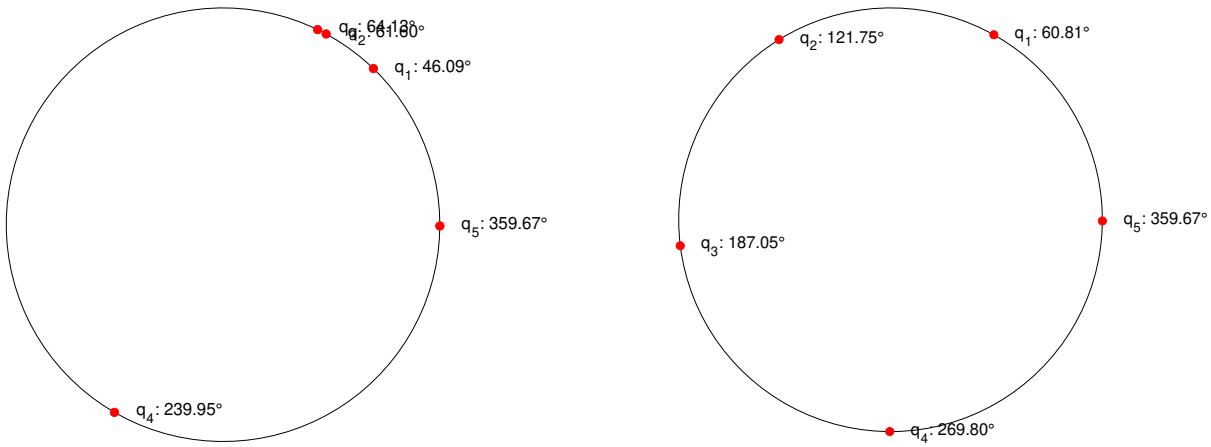

Figure 1.9: Initial (left) and final (right) sensor positions with measurement error and order preservation

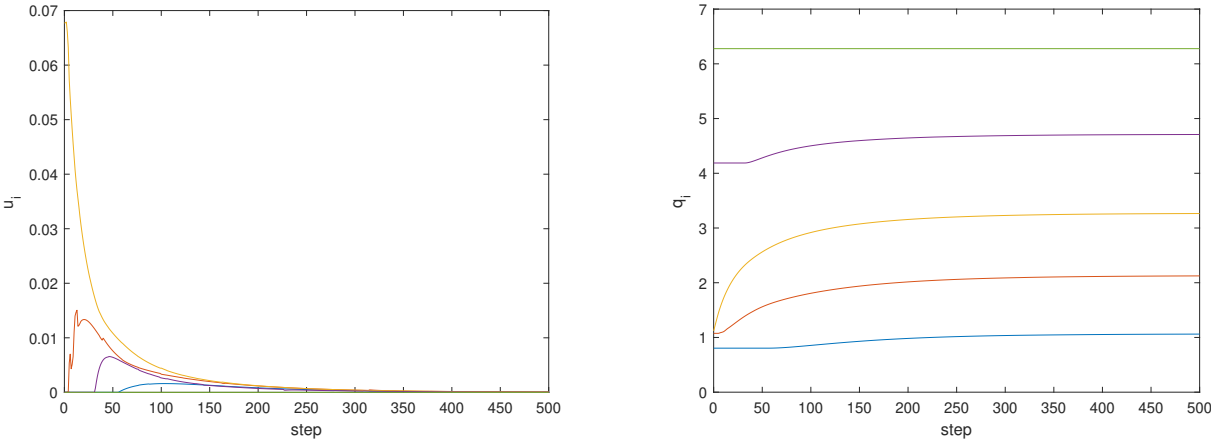

Figure 1.10: Control actions (left) and sensors' positions (right) with measurement error and order preservation

1.3 Intermittent distance readings

In a real setting, the information coming from the sensors can be unavailable at every time instant. To examine how the control law in Equation 1.6 copes with such real-world imperfections, we perform simulations where the distance readings from each sensor are updated intermittently rather than continuously. Specifically, each sensor updates its distance reading every ν steps. The following simulations show the robustness for low values of ν: when the distance readings are accurate and free from measurement errors, the control law remains robust for low values of ν, specifically when $\nu < 4$ the system converges to the optimal formation, even though some oscillatory behavior appears. This suggests that the system can tolerate some delay in sensor updates without significant degradation in performance.

As we increase ν beyond 4, the robustness of the controller becomes highly dependent on the initial positions of the sensors. When sensors are initially evenly distributed along the circumference, the system can handle higher values of ν, as shown in the simulation in Figure 1.11. The reason is that in an even distribution, agents require smaller adjustments to reach their target positions. Consequently, they move at slower velocities, which helps mitigate the adverse effects of delayed distance updates. On the other hand, Figure 1.13 shows how when starting from a more clustered initial condition, the system loses stability at lower values of ν.

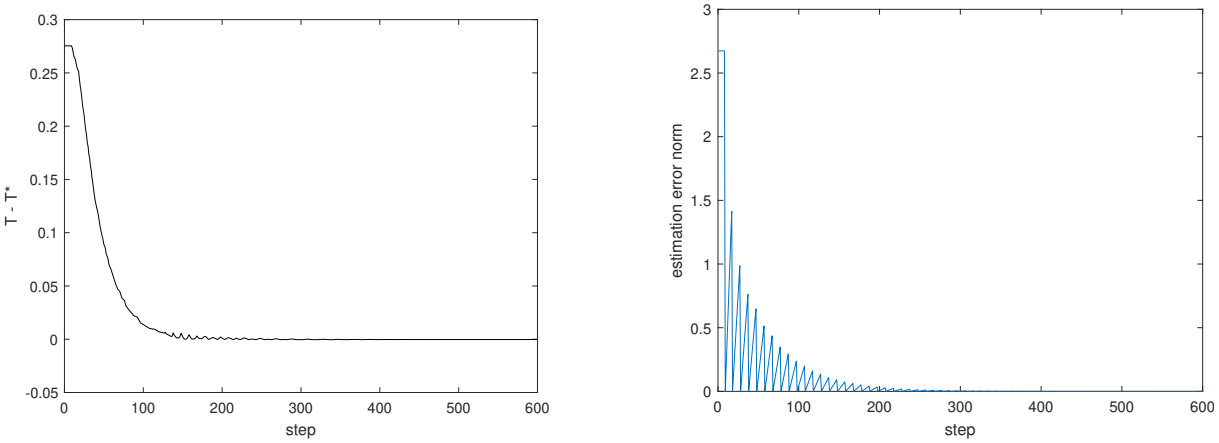

Figure 1.11: Optimal value achieved and estimation error converging to zero when $\nu = 10$ and no measurement error

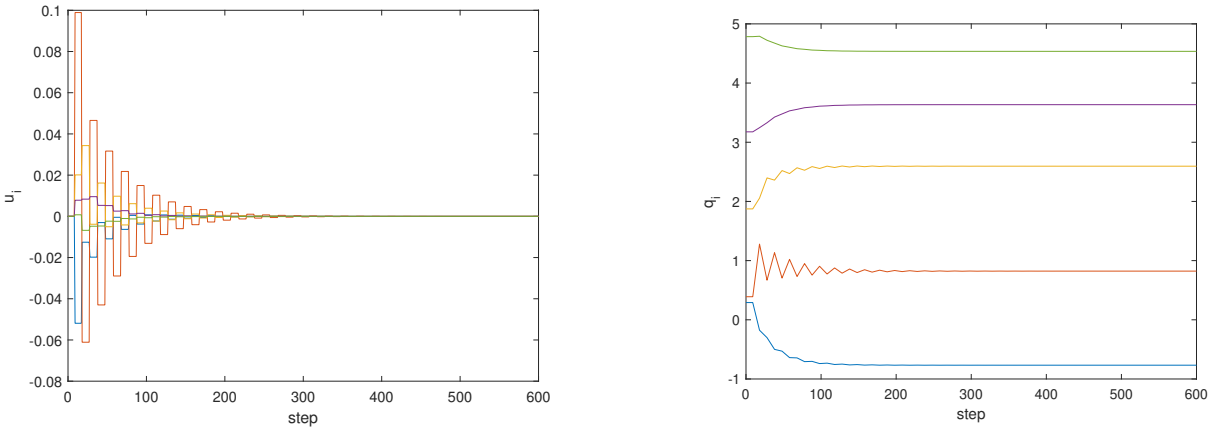

Figure 1.12: Control input (left) and sensor positions (right) when $\nu = 10$ and no measurement error

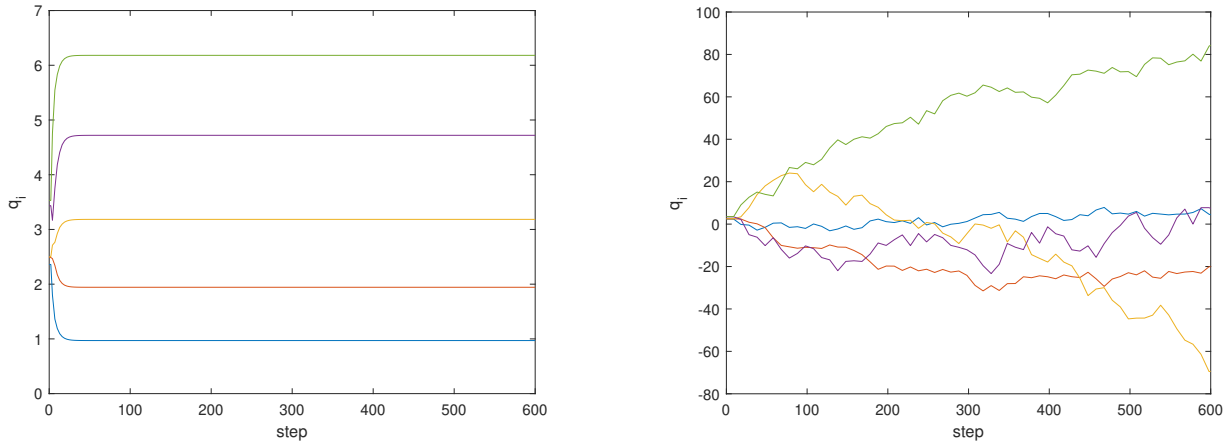

Figure 1.13: System evolution with $\nu = 3$ (left) and $\nu = 10$ (right) when the sensors start close to each other

When an intermittent distance reading is coupled with measurement errors as large as they were chosen in Table 1.1, the controller and the estimator are not capable of reliably stabilizing the system with values of $\nu > 1$. The following Figures show the emerging of persistent oscillations with $\nu = 2$ and no order preservation.[1]

[1]Note that here the combined requirement of robustness to updating the distance reading every second step and measurement errors are quite demanding. The measurement errors according to Table 1.1 have an amplitude up to 36 degrees, so there is no wonder that the system soon loses stability!

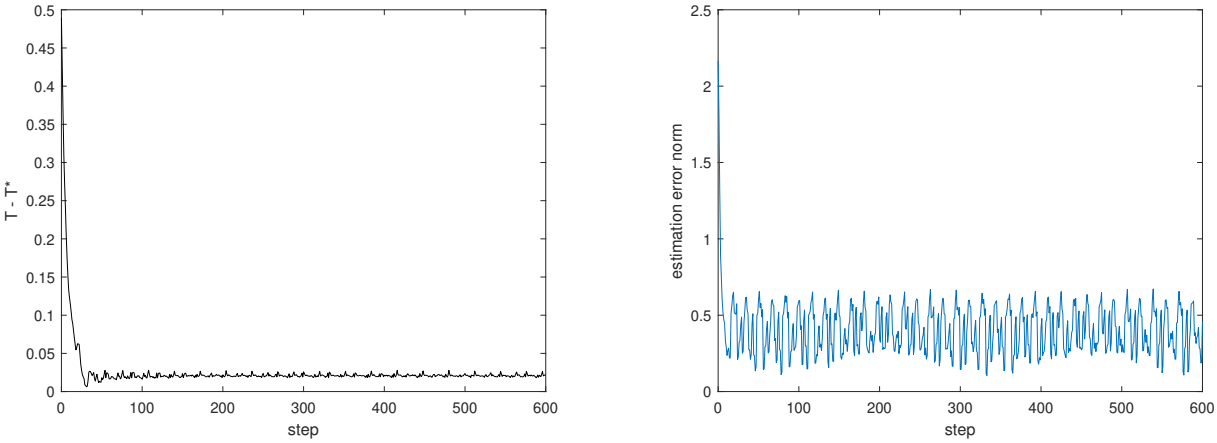

Figure 1.14: Emerging of persistent oscillations in the cost function and estimation error when $\nu = 2$ with measurement error

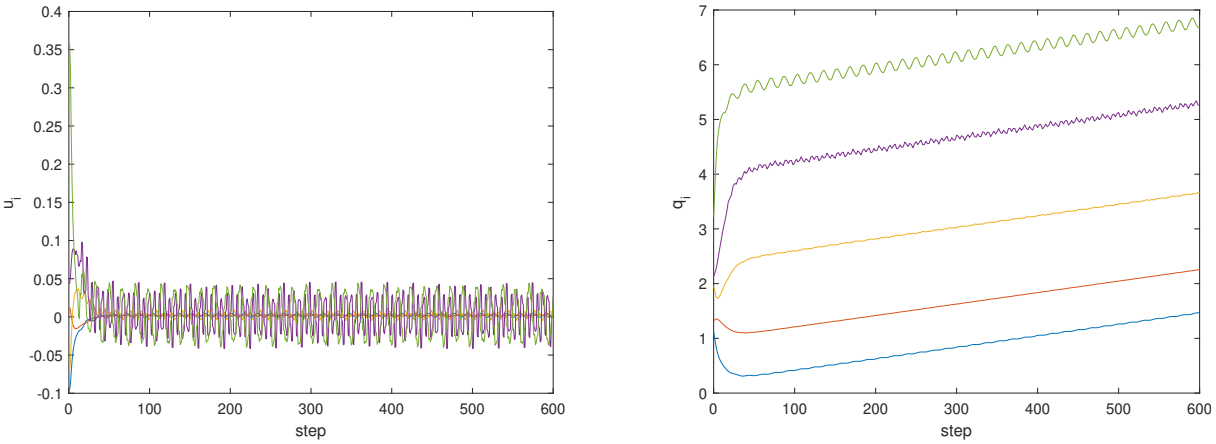

Figure 1.15: Control input (left) and sensor positions (right) when $\nu = 2$ with measurement error

1.4 Pinning control

In this section, we apply pinning control to the sensors implementing the coverage control law. The aim is to check if robots are capable of moving while keeping their formation. We introduce a pinner node q_s that evolves with the same dynamics as Equation 1.1:

$$q_s(k+1) = q_s(k) + u_s(k) \tag{1.15}$$

The pinned node is chosen as $j = \arg\min_i \lambda_i$: the of the slowest node in the network. The pinner node q_s starts at the same position as the pinned node q_j and moves at constant speed along the circle:

$$q_s(0) = q_j(0) \tag{1.16}$$
$$u_s(k) = 0.01 \tag{1.17}$$

The control law in Equation 1.6, with the addition of the pinning control term on the pinned node, becomes:

$$u_i(k) = \lambda_i \, \mathrm{sat}_{\pm 1} \left(\tilde{u}_i(k) - \rho_i K (q_i(k) - q_s(k)) \right) \tag{1.18}$$

where ρ_i is a flag to apply the pinning control law only to the pinned node, and K is a control parameter set to $K = 0.1/\lambda_j$.

$$\rho_i = \begin{cases} \rho_i = 1 & \text{if } i = j \\ \rho_i = 0 & \text{if } i \neq j \end{cases} \tag{1.19}$$

In the following simulations, with the choice of parameters listed in Table 1.1 and the control law in Equation 1.6, the sensors are able to keep a nearly optimal configuration while rotating according to the pinner node dynamic. The estimation law is still capable of reducing the effect of the measurement error. However, Figure 1.18 shows how the steady state value of $T - T^*$ increases with higher pinner velocity u_s.

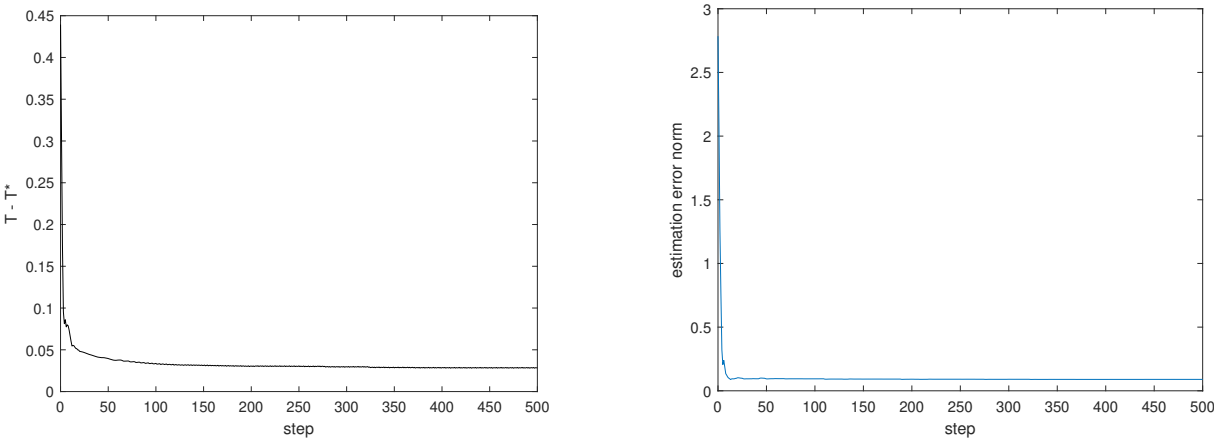

Figure 1.16: Optimal value almost achieved, and distance estimation error while rotating in formation at constant speed

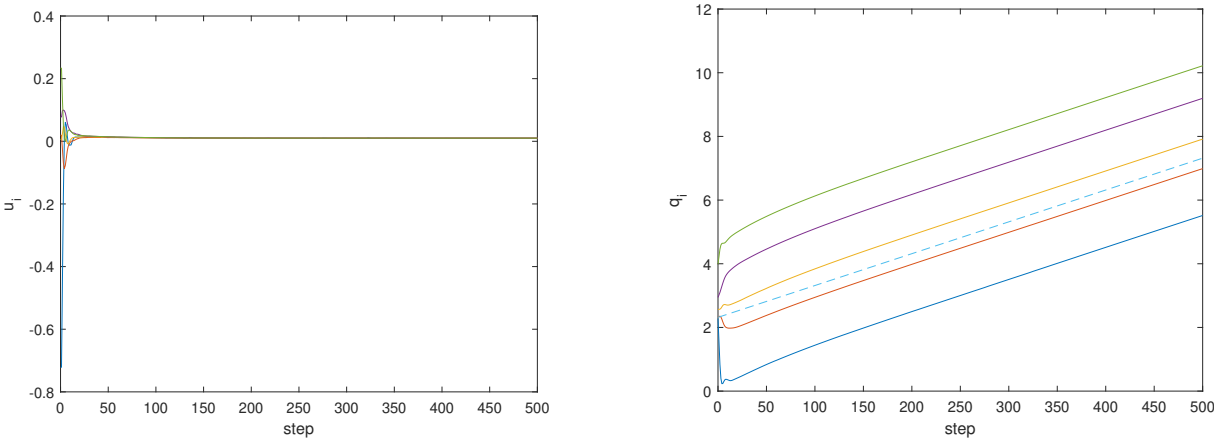

Figure 1.17: Control input (left) and sensors' positions (right) while rotating in formation at constant speed. The dashed line represents the pinner node

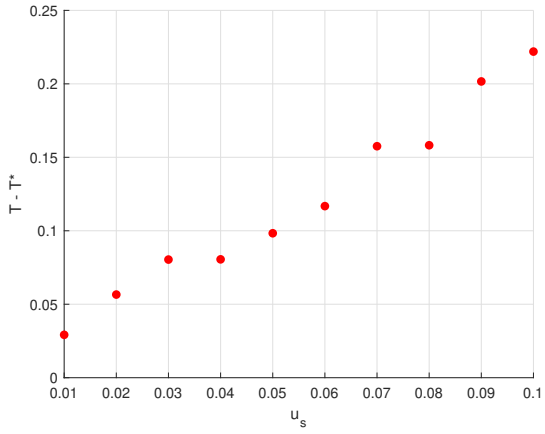

Figure 1.18: Optimal formation error when increasing the pinner node's velocity u_s

1.5 Conclusion

We addressed the coverage control problem for a network of heterogeneous mobile sensors operating in a one-dimensional circular space. The study focused on implementing distributed control laws to minimize the coverage cost function, representing the largest arrival time from the sensor network to any point on the circle. The problem was formulated by considering real-world complexities such as measurement errors and intermittent sensor readings, which were mitigated using estimation algorithms that only required knowledge of the upper bounds of the errors.

Simulations demonstrated the effectiveness of this control law in driving the sensor network to a near-optimal configuration, even in the presence of measurement errors. Further, we examined the robustness of the control law under intermittent sensor readings, revealing that the system could tolerate delays in distance updates up to a certain limit, depending on the initial sensor positions. Lastly, pinning control was introduced to test the ability of the sensors to maintain formation while in motion, with results indicating that near-optimal formation could be preserved, albeit with an increase in steady-state error

as the pinner velocity increased.

Overall, the findings show that the control law is robust and capable of achieving near-optimal coverage in various real-world conditions. Future work could explore extending these results to more complex, such as time varying, network topologies.

Chapter 2

Two-dimensional coverage control

After analyzing the problem of coverage control in a one dimensional space, we move a step forward to a different scenario, looking at the problem proposed by Schwager et al. [3]. In this formulation, robots move within a bounded convex region guided by exact sensor measurements to minimize a cost function representing sensing efficiency. As in Chapter 1, the controller must be able to deal with some nonlinearity: the sensing function that describes the importance of each point of the region is unknown to the robots. First, a basic estimation law will be used. It will lead to locally true approximations of the sensory function by each robot. Then, a coupling term will be added to the estimation law that will propagate the information of each sensor in the network, leading to a globally true approximation of the sensory function. The chapter is structured as it follows: we will give the problem formulation in Section 1; then, Section 2 will show the performance of the basic controller; finally, a consensus based term will be added in Section 3. The conclusions will be drawn in Section 4.

2.1 2D Problem formulation

We will consider n robots whose dynamic is continuous time. Their position $p_i \in \mathbb{R}^2$ is described by the following equation:

$$\dot{p}_i = u_i \tag{2.1}$$

The robots move in a boundend convex environment $Q \subset \mathbb{R}^2$. We assume the robots to be able to compute their Voronoi partitions $V_i = \{q \in Q \mid \|q - p_i\| \leq \|q - p_j\|, \forall j \neq i\}$. The robots aim to adaptively cover areas of high sensory interest, where the sensory function $\phi : Q \to \mathbb{R}_{>0}$ represents the importance of different areas in Q. It is static and it's unknown to the robots. However, they have accurate information about the value of the sensory function in their exact position.

The cost function for sensing over the region Q is defined as:

$$H(p_1, \ldots, p_n) = \sum_{i=1}^{n} \int_{V_i} \frac{1}{2} \|q - p_i\|^2 \phi(q)\, dq \tag{2.2}$$

Here, the sensory values for each position in the region are weighted by the unreliability of the robot's sensors: $\frac{1}{2}\|q-p_i\|^2$, meaning that the more the sensor is far away from the point q, the more its information is unreliable. The objective is to minimize the cost function H.

The mass, first moment, and centroid of the Voronoi region V_i are defined as follows:

$$M_{V_i} = \int_{V_i} \phi(q)\, dq \tag{2.3}$$

$$L_{V_i} = \int_{V_i} q\phi(q)\, dq \tag{2.4}$$

$$C_{V_i} = \frac{L_{V_i}}{M_{V_i}} \tag{2.5}$$

A locally optimal coverage configuration (or near optimal configuration) occurs when each robot is positioned at the centroid of its Voronoi region, $p_i = C_{V_i}$ [4]. We only look for local minima of H because finding a global minima is an NP-hard problem that is difficult to solve online.

The robots approximate the sensory function using a basis function model:

$$\hat{\phi}_i(q,t) = \mathcal{K}^T(q)\hat{a}_i(t) \tag{2.6}$$

where $\mathcal{K} : Q \to \mathbb{R}^m_{>0}$ is a vector of continuous basis functions, and $\hat{a}_i(t)$ is a time-varying parameter vector that is adjusted using an adaptation law. Each robot has its own parameters estimate. The mass, first moment, and centroid computed using the estimated parameters are indicated with $\hat{M}_{V_i}$, $\hat{L}_{V_i}$, $\hat{C}_{V_i}$.

When evaluating the sensory function estimate $\hat{\phi}$, we say that it is:

- a Globally True Approximation: The robot's sensory function approximation $\hat{\phi}_i(q)$ matches the true sensory function $\phi(q)$ for all $q \in Q$.
- a Locally True Approximation: The robot's sensory function approximation $\hat{\phi}_i(q)$ matches the true sensory function $\phi(q)$ over a subset of the environment.

A key assumption is that there exists an unknown ideal parameter vector a such that $\phi(q) = K(q)^T a$, ensuring that the robots can learn an accurate approximation of the sensory function.

The control law proposed by Schwager et al. in [3] is the following:

$$u_i = K(\hat{C}_{V_i} - p_i) \tag{2.7}$$

For each robot, the adaptation law that allows the estimation of the parameters for the computation of $\hat{C}_{V_i}$ is based on the measurement of the sensory function at their current position. It involves the quantities $\Lambda_i(t)$ and $\lambda_i(t)$, defined as:

$$\Lambda_i(t) = \int_0^t w(\tau)K_i(\tau)K_i^T(\tau)\, d\tau \tag{2.8}$$

$$\lambda_i(t) = \int_0^t w(\tau)K_i(\tau)\phi_i(\tau)\, d\tau \tag{2.9}$$

where $w(\tau) \geq 0$ is a data collection weighting function, $\mathcal{K}_i(\tau)$ represents the value of the basis functions at the robot's position at time τ, and $\phi_i(\tau)$ is the sensory measurement taken by robot i at time τ. These quantities are computed using the following differential equations, starting from zero initial conditions:

$$\dot{\Lambda}_i = w(t)K_iK_i^T \tag{2.10}$$

$$\dot{\lambda}_i = w(t)K_i\phi_i \tag{2.11}$$

By defining:

$$F_i = \frac{\int_{V_i} \mathcal{K}(q)(q-p_i)^T dq K \int_{V_i} (q-p_i)\mathcal{K}(q)^T dq}{\int_{V_i} \hat{\phi}_i(q) dq} \tag{2.12}$$

The pre-adaptation law for the parameter vector $\hat{a}_i$ is given by:

$$\dot{\hat{a}}_{\text{pre}_i} = -F_i \hat{a}_i - \gamma(\Lambda_i \hat{a}_i - \lambda_i) \tag{2.13}$$

where $\gamma > 0$ is a learning rate gain. A potential issue with the control law arises when the parameter vector $\hat{a}_i$ approaches zero, leading to a singularity in the calculation of the estimated centroid $\hat{C}_{V_i}$, as it involves the inverse of $\hat{M}_{V_i}$. To address this issue, a mechanism to prevent the parameters from dropping below a minimum value is used. This is accomplished using a projection operation that modifies the pre-adaptation law. The final adaptation law is given by:

$$\dot{\hat{a}}_i = \Gamma \left(\dot{\hat{a}}_{\text{pre}_i} - I_{\text{proj}_i} \dot{\hat{a}}_{\text{pre}_i} \right) \tag{2.14}$$

where $\Gamma \in \mathbb{R}^{m \times m}$ is a diagonal, positive-definite adaptation gain matrix, and I_{proj_i} is a projection matrix defined element-wise as:

$$I_{\text{proj}_i}(j) = \begin{cases} 0 & \text{if } \hat{a}_{i,j} > a_{\min}, \\ 0 & \text{if } \hat{a}_{i,j} = a_{\min} \text{ and } \dot{\hat{a}}_{\text{pre}_{i,j}} \geq 0 \\ 1 & \text{otherwise.} \end{cases} \tag{2.15}$$

Here, $a_{\min}$ is the lowest value of a_i, that we assume is known a priori. This projection ensures that $\hat{a}_{i,j}$ does not fall below $a_{\min}$, thereby avoiding the singularity in the calculation of $\hat{C}_{V_i}$.

2.2 Simulation results

The following simulations are performed using the parameters in Table 2.1. The sensory function $\phi(q)$ is parametrized as a linear combination of a set of Gaussian basis functions, through the coefficients a. Specifically, the features vector $\mathcal{K}(q)$ consists of m Gaussian functions centered at different locations in the environment. For a set of $m = 9$ Gaussians, each component $\mathcal{K}_j(q)$ of the features vector is defined as:

$$\mathcal{K}_j(q) = \frac{1}{2\pi\sigma_j^2} \exp\left(-\frac{(q-\mu_j)^2}{2\sigma_j^2} \right) \tag{2.16}$$

where $\sigma_j = 0.18$ and μ_j represents the center of the j-th Gaussian function. In the simulation, the environment Q (the unit square) is divided into a 3×3 grid, and each μ_j is chosen to be at the center of one of the nine grid squares. With the choice of the parameters a in Table 2.1, only the bottom left and the top left gaussians contribute significantly to the value of $\phi(q)$. In Figure 2.3, the red dots represent the peaks of the function $\phi(q)$. In accordance to Schwager et al. [3], the data weighting function used to update Λ_i and λ_i is:

$$w(t) = \begin{cases} \|\dot{p}\| & t < T \\ 0 & t \geq T \end{cases} \tag{2.17}$$

where $T = 10$ to ensure that Λ_i and λ_i remain bounded.

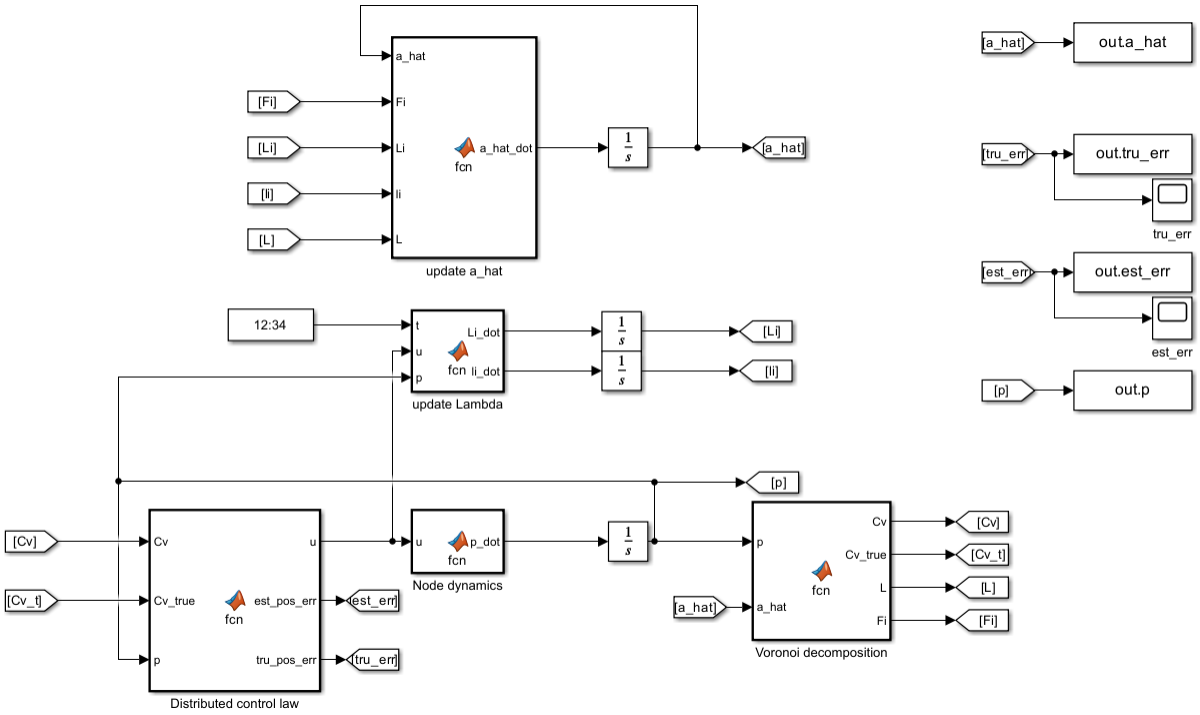

Figure 2.1: Simulation scheme for the 2D adaptive control law

Parameter	Description	Value
n	Number of robots	20
x_0, y_0	Initial positions	rand$(n, 1)$ each
Q	Environments bounds	$0 \leq x \leq 1; 0 \leq y \leq 1$
$a_{\min}$	Minimum parameter value	0.1
a	Real parameters vector	$[100; a_{\min} \cdot \text{ones}(1, 7); 100]$
$\hat{a}_0$	Initial parameter estimate	$a_{\min} \cdot \text{ones}(9, n)$
K	Control gain matrix	$[3, 0; 0, 3]$
Γ	Projection adaptation matrix	eye(9)
γ	Learning rate	100

Table 2.1: Parameters used in the simulations.

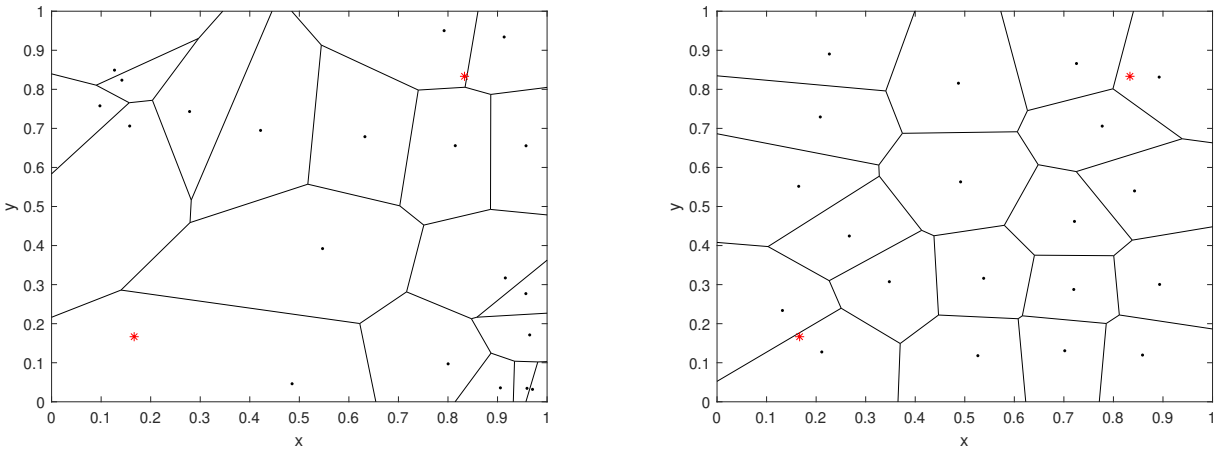

Figure 2.2: Initial (left) and final (right) positions of the agents

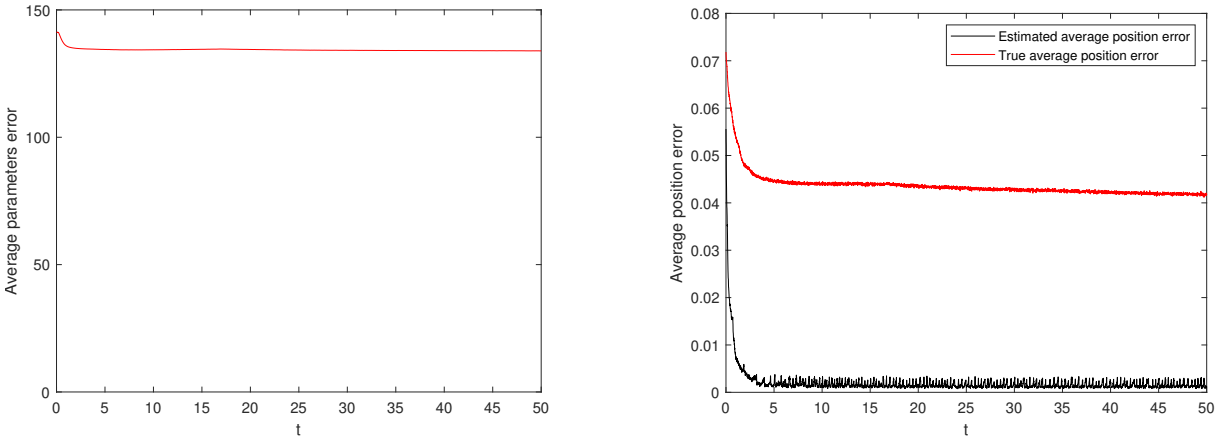

Figure 2.3: Parameters error (left) and position error (right)

The approximation of the sensory function $\phi(q)$ by each robot converges locally along its trajectory. However, the robots fail to reach a globally true approximation of the sensory function across the entire environment. This limitation is due to the lack of parameter coupling among the robots, which restricts the ability of individual robots to learn the sensory distribution outside their immediate regions. The parameters estimate $\hat{a}_i$ of each robot does not converge to a common value. Each robot adapts its parameters independently based on its local sensory measurements, resulting in varying estimates across the network. The final configuration of the robots shows that they aggregate around areas of high sensory interest. Although the estimated error $\|\hat{C}_{V_i} - p_i\|$ converges to zero, as shown in Figure 2.3, such configuration is only suboptimal because of the discrepancy between the estimated centroids and the true centroids positions C_{V_i}.

The use of a skew symmetric component in the control gain K can make the trajectories more rich, improving the exploration and reducing the parameter estimation error [5]. In the following simulations, the control gain is set to $K_1 = [3,\ 1;\ -1,\ 3]$ and to $K_3 = [3,\ 3;\ -3,\ 3]$ to show the effects of an increase of a skew symmetric component. With reference to Figure 2.3, with the skew symmetric component the robots are able to reduce the true position error. Still, as shown in Figure 2.5, such improvement is

limited and does not guarantee convergence to zero of the estimation error.

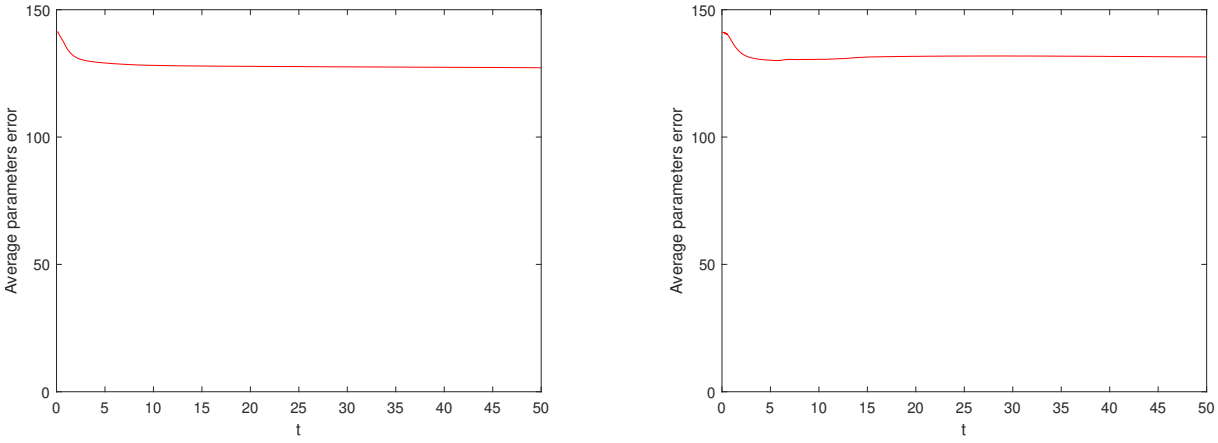

Figure 2.4: Parameters errors with K_1 (left) and K_3 (right)

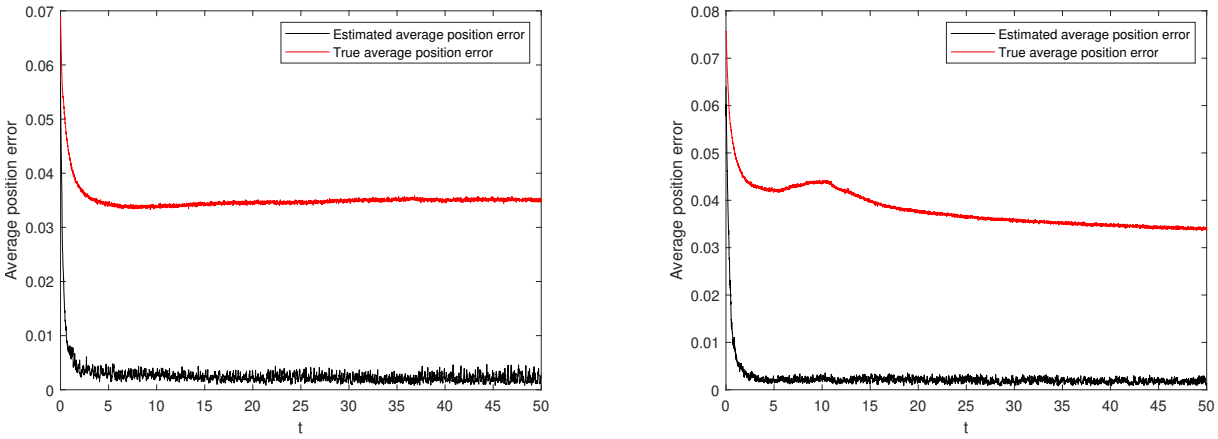

Figure 2.5: True and estimated position errors with K_1 (left) and K_3 (right)

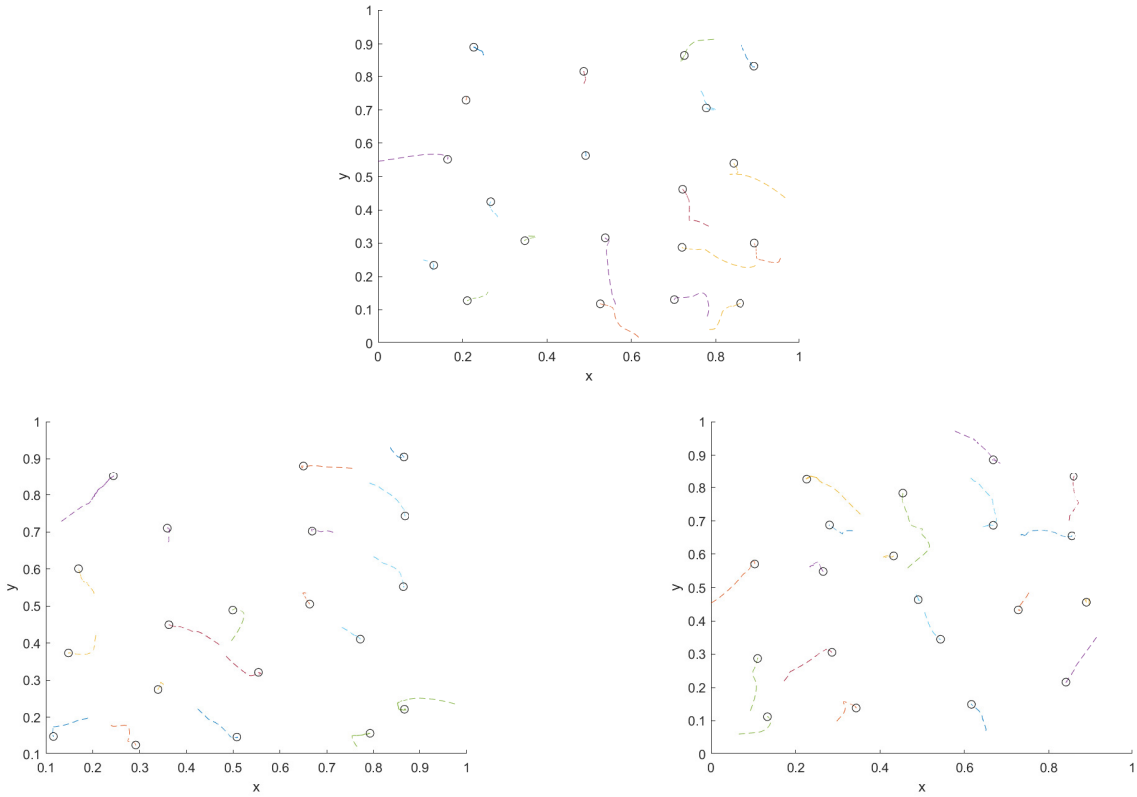

Figure 2.6: Trajectories with no skew symmetric component (top), K_1 (bottom left) and K_3 (bottom right). When a skew symmetric component is applied, the trajectories become more complex

2.3 Consensus Control Law

Since the issue with such an adaptation law is that each robot only finds a locally true approximation of the sensory function, Schwager et al. proved that by coupling the adaptation of the parameters between neighboring agents, consensus on the parameters is achieved, i.e. $\lim_{t\to\infty}(\hat{a}_i - \hat{a}_j) = 0$, $\forall i, j$, and with sufficiently rich trajectories of the robots, each robot converges to a globally true approximation of the sensory function $\phi(q)$ [3].
The consensus control law modifies the parameter adaptation law by adding a consensus term:

$$\dot{\hat{a}}_{\text{pre}_i} = -F_i\hat{a}_i - \gamma(\Lambda_i\hat{a}_i - \lambda_i) - \zeta\sum_{j=1}^{n} l_{ij}(\hat{a}_i - \hat{a}_j) \tag{2.18}$$

where $\zeta > 0$ is a consensus gain, and it is set to $\zeta = 80$ in our simulations. l_{ij} is a weighting factor that is positive if robots i and j are neighbors (i.e., they share a common edge in the Voronoi partition): l_{ij} is equal to the length of the shared Voronoi edge, or zero if robot i and j are not neighbors. The term $l_{ij}(\hat{a}_i - \hat{a}_j)$ couples the parameter estimates of neighboring robots, promoting the propagation of sensory information throughout the network.

By propagating sensory information throughout the network, the consensus control law allows robots to distribute themselves more effectively in areas of high sensory interest. This results in a tighter aggregation around these areas and an overall lower value of the cost function H. The parameter estimation error converges to zero, leading to the convergence of both the estimated position error and the true position error[1].

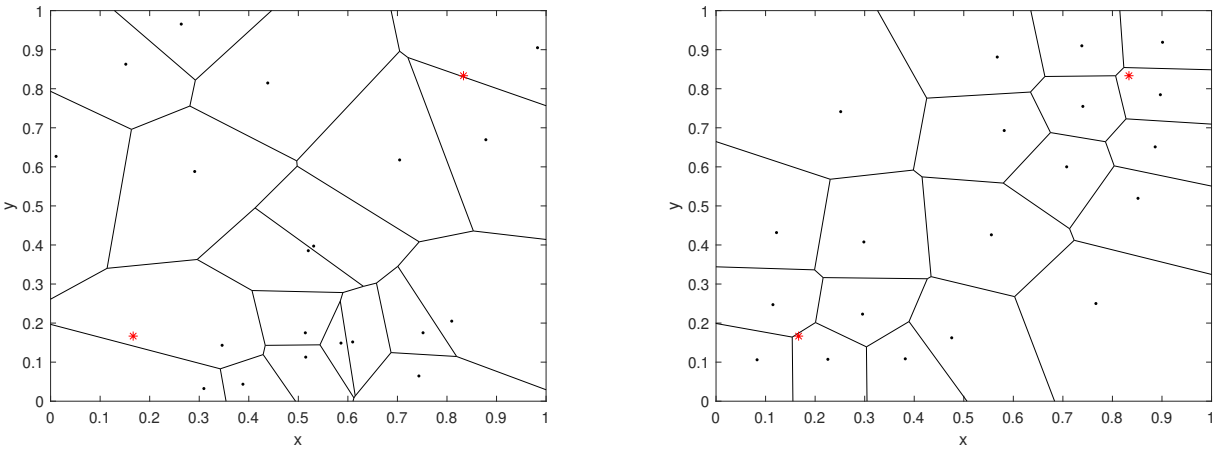

Figure 2.7: Initial (left) and final (right) positions of the agents with consensus controller

[1]Note that the small steady state errors and the high frequency oscillations of the position errors in both the consensus and the non-consensus controller are due to the numerical method used to approximate the surface integrals over the Voronoi regions for the computation of the centroids. We generate $\nu = 10000$ random points in the voronoi cell, and we discretize the integral by finding the values of $\phi(q)$ at the discrete points. The approximation would tend to the real value of the surface integral as $\nu \to \infty$, however this would surge the execution time since the integration is carried out for each Voronoi cell at each time instant. A trade off has been found between correctness of the solution and execution times

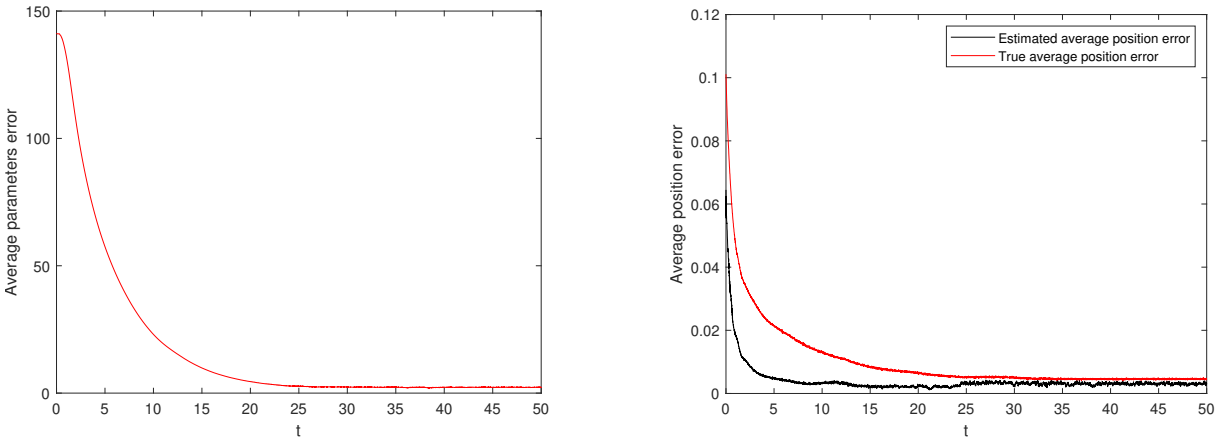

Figure 2.8: Parameters error (left) and position error (right) with consensus controller

2.4 Conclusions

In this chapter, we extended the coverage control problem to a two-dimensional environment as formulated by Schwager et al. [3]. The chapter demonstrated how a decentralized control strategy allows robots to adaptively position themselves in areas of high sensory interest using only local sensor measurements. Initially, a basic controller was introduced that relied solely on each robot's local sensory measurements. While this method allowed the robots to reach a near-optimal configuration, the lack of information sharing among the robots limited its effectiveness. Even with a skew symmetric component in the control gain matrix, that indeed improved the estimation of the parameters, the robots could only achieve locally true approximations of the sensory function, resulting in suboptimal coverage.

To address these limitations, a consensus-based adaptation law was added to couple the parameter estimates among neighboring robots. This consensus control law enabled the propagation of sensory information across the network, allowing each robot to learn a globally true approximation of the sensory function. As a result, the network of robots reached a more coordinated and optimal coverage configuration, effectively reducing the cost function H.

In conclusion, the results highlight the importance of information sharing in decentralized control strategies for multi-robot systems. Future work could explore the implementation of more complex sensory functions and environments, as well as the effects of dynamic changes in the sensory distribution on the performance of the proposed control laws.

Bibliography

[1] Cheng Song et al. “Coverage control for heterogeneous mobile sensor networks with bounded position measurement errors”. In: *Automatica* 120 (2020), p. 109118. ISSN: 0005-1098. DOI: https://doi.org/10.1016/j.automatica.2020.109118. URL: https://www.sciencedirect.com/science/article/pii/S0005109820303162.

[2] Cheng Song et al. “Coverage control for heterogeneous mobile sensor networks on a circle”. In: *Automatica* 63 (2016), pp. 349–358. ISSN: 0005-1098. DOI: https://doi.org/10.1016/j.automatica.2015.10.044. URL: https://www.sciencedirect.com/science/article/pii/S0005109815004446.

[3] Mac Schwager, Daniela Rus, and Jean-Jacques Slotine. “Decentralized, adaptive coverage control for networked robots”. In: *The International Journal of Robotics Research* 28.3 (2009), pp. 357–375.

[4] E Drezner. “Facility location: A survey of applications and methods”. In: *Journal of the Operational Research Society* 47.11 (1996), pp. 1421–1421.

[5] Mac Schwager et al. “A ladybug exploration strategy for distributed adaptive coverage control”. In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pp. 2346–2353.