Robust PID consensus-based formation control of nonholonomic mobile robots affected by disturbances

Matin Mansouri
Master of Engineering Student
Concordia University
Montreal, Canada
matin.mansouri23@gmail.com

Sourena Morteza Ghasemi
Master of Engineering Student
Concordia University
Montreal, Canada
sourenaghasemi1997@gmail.com

***Abstract*— In this study, we introduce a unique controller designed to universally address the challenges of achieving position and orientation consensus in formations of multiple nonholonomic vehicles, represented as differential drive robots experiencing external disturbances. This controller utilizes a straightforward Proportional Integral Derivative (PID) configuration. To comply with the Brockett stabilization theorem for nonholonomic systems, we have integrated a time-varying persistency of excitation term within the angular velocity dynamics. Our approach marks the inaugural solution to the robust formation issue employing a smooth, time-varying controller. Simulation results are presented to substantiate the theoretical aspects of our approach.**

Keywords—PID controller, consensus, nonholonomic robots, robust controller, Formation control

I. INTRODUCTION

In recent years, the coordination and formation control of autonomous mobile robots have garnered significant attention within the robotics community due to their extensive applications in surveillance, search and rescue, and autonomous transportation. Among various control strategies, consensus-based formation control has emerged as a prominent approach for ensuring that a group of robots achieves a common objective through localized interactions among them. This project extends upon the pioneering work by Romero et al. [1], a novel Proportional Integral Derivative (PID) controller addressing the global position and orientation consensus-based formation problem for multiple nonholonomic vehicles modeled as differential drive robots subjected to external disturbances.

The primary motivation behind this study stems from the inherent challenges associated with the nonholonomic constraints of mobile robots, which limit their motion to certain directions, making their control a nontrivial problem especially in the presence of external disturbances such as slipping or skidding. The original work by Romero et al. [1] is ground-breaking as it proposes the first solution to the robust formation problem using a smooth time-varying controller, which is necessary to satisfy the Brockett condition [3]—a fundamental theorem in control theory determining the local controllability of nonlinear systems. This theorem underscores the importance of understanding the system's dynamics and constraints in designing effective control strategies, a crucial aspect addressed in our project.

This report aims to replicate the results obtained by Romero et al. [1] through simulation in MATLAB and to provide a comprehensive analysis of the controller's performance under various scenarios. By doing so, we seek not only to validate the effectiveness of the proposed PID consensus-based formation control strategy but also to explore its limitations and potential areas for further improvement. The following sections will delve into the problem formulation, detail the controller design as per the original study, present simulation results, and conclude with a discussion on the findings and future research directions.

Notation: $R := [-\infty, \infty], R > 0 := (0, \infty], R \geq 0 := [0, \infty]$. $\|x\|$ stands for the standard Euclidean norm of vector $x \in R^n$. I_n represents the identity matrix of size n × n. 1_k stands for a column vector of size k with all entries equal to one. $\otimes$ represents the standard Kronecker product. For any function $f : R \geq 0 \rightarrow R$, the $\mathcal{L}_\infty$-norm is defined as $\|f\|_\infty := sup_{t\geq 0} |f(t)|$, $\mathcal{L}_2$-norm as $\|f\|_2 := \left(\int_0^\infty |f(t)|^2 \, dt\right)^{1/2}$. The $\mathcal{L}_\infty$ and $\mathcal{L}_2$ spaces are defined as the sets $\{f : R \geq 0 \rightarrow R : \|f\|_\infty < \infty\}$ and $\{f : R \geq 0 \rightarrow R^n : \|f\|_2 < \infty\}$, respectively. The set N¯ is defined as $\bar{N} := \{1, \ldots, N\}$, where N is a positive natural number.

II. PROBLEM FORMOLATION

We are examining a group of N nonholonomic vehicles represented as differential drive robots operating in the xy-plane with three degrees of freedom, first of all we have to check Brockett's theorem [3] because our system is non holonomic and non-linear. The Brockett's condition states that a continues-time-invariant must controlled by an input which is fully described the behavior of the system which shows by equation (7): two translations and one rotation, while experiencing constant disturbances. Each robot is characterized by coordinates $z_i = [x_i, y_i]^T \in R^2$ for translation and $\theta i \in R$ for rotation. The structure of these robots is illustrated in Figure 1. Assuming non-slip conditions (pure rolling), the dynamics of the ith robot can be described by the equation:

$$\begin{aligned} \dot{z}_\iota &= \phi(\theta_i) v_i, \\ \phi(\theta_i) &= \begin{bmatrix} \cos(\theta_i) \\ \sin(\theta_i) \end{bmatrix}, \\ \dot{\theta}_i &= \omega_i, \\ \begin{bmatrix} \dot{v}_\iota \\ \dot{\omega}_\iota \end{bmatrix} &= M_i{}^{-1} B_i \tau_i + \delta_i, \end{aligned} \tag{1}$$

The variables v_i and w_i represent the linear and angular velocities, respectively. The inertia matrix $M_i := diag(m_i, I_i) \in R^2$ is defined with I_i representing the moment of inertia and m representing the mass. The input

matrix B_i is given by $B_i = \frac{1}{r_j}\begin{bmatrix} 1 & 1 \\ 2R_i & -2R_i \end{bmatrix}$ where R_i is distance between z_i and the wheels that have radius r_i. The control signal for the wheels is denoted as $\tau_i \in R^2$ and $\delta_i = col(\delta_{vi}, \delta_{wi}) \in R^2$ represents a constant unknown disturbance, Tzafestas [4].

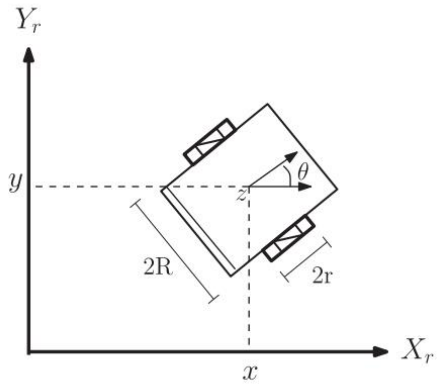


Figure 1: Schematic of a differential wheeled mobile robot. [2]

The communication network between the mobile robots is represented by the Laplacian matrix $L := [\ell_{ij}] \in R^{N\times N}$, whose elements is.

$$\ell_{\{ij\}} = \begin{cases} \sum_{k\in N_i} a_{ik} & i = k, \\ -a_{ik} & i \neq k, \end{cases} \quad (2)$$

where $i \in \bar{N}$ and N_i is set of transmitting information to the i_{th} robot.

In this project, the assumption bellow is imposed on the interconnection of the agents.

A1 The interconnection graph is undirected, static and connected.

Let $\bar{z}_i \in R^2$ be defined $\bar{z}_i := z_i - z_d$, where $z_{di} = [x_{di}, y_{di}]^T$ represents the desired (constant) position of the i_{th} robot relative to the center of a specified desired formation pattern. The goal of this study is to develop a decentralized controller τ_i, meeting the criteria specified in A1, to solve the following problem:

(FP) Formation Problem. The positions z_i of all robots asymptotically converge to the positions specified by a desired formation, with all robots agreeing on the center z_c of this formation. Simultaneously, the orientations θ_i of the robots asymptotically reach a consensus value θ_c. Additionally, the velocities of the robots converge to zero. Therefore, there exist values $z_c \in R^2$ and $\theta_c \in R$ such that:

$$\lim_{t\to\infty} \bar{z}_i(t) = z_c;\ \lim_{t\to\infty} \theta_i(t) = \theta_c;\ \lim_{t\to\infty} v_i(t) = 0;$$
$$\lim_{t\to\infty} \omega_i(t) = 0;\ \forall\, i \leq N$$

The center of the formation z_c and the consensus orientation θ_c are unknown and are determined by the interconnection topology of the robots and their initial positions.

There are two primary challenges in addressing the problem **(FP)**: (a) the robots are subject to constant disturbances; and (b) the nonholonomic system (1) can only be stabilized at a point using either non-smooth or time-varying feedback methods, Brockett [3].

To overcome these challenges, we propose a robust controller designed as a smooth, time-varying scheme. This controller incorporates a simple PID-like structure along with a time-varying persistency of excitation term.

III. Control Design

To address the formation control problem outlined in the journal article, we propose a controller design methodology inspired by the concepts presented therein. Our approach integrates the inner control loop proposed in the article with our own simulation results to achieve effective formation control and disturbance rejection.to do so, we define two errors which is cartesian position and orientation errors which respectively is:

$$e_{zi} := \sum_{j\in N_i} a_{ij}\left(\bar{z}_i - \bar{z}_j\right); \quad e_{\theta i} := \sum_{j\in N_i} a_{ij}\left(\theta_i - \theta_j\right) \quad (3)$$

A. Inner Control Loop

The first step in our controller design is to implement the inner control loop proposed in the journal article. This control loop, represented by Equation (4), facilitates the regulation of individual robot movements within the formation. By defining the signal u_i as the new input control, we establish a framework for controlling both position and orientation errors of the agents.

$$\tau_i = B_i^{-1} M_i u_i = \frac{r}{2}\begin{bmatrix} m_i & \frac{I_i}{2R_i} \\ m_i & -\frac{I_i}{2R_i} \end{bmatrix}\begin{bmatrix} u_{vi} \\ u_{wi} \end{bmatrix}, \quad (4)$$

B. Control Laws

Utilizing the inner control loop in conjunction with our simulation model, we derive the control laws governing the dynamics of the robots' positions and orientations. Equation (5) captures the dynamics of each robot's position $\bar{z}_i$ and orientation θ_i in relation to the desired formation configuration. Additionally, the control laws incorporate terms to address disturbances, ensuring robust performance in real-world scenarios.

$$\begin{aligned} \dot{z}_i &= \phi(\theta_i) v_i, \\ \dot{\theta}_i &= \omega_i, \end{aligned} \quad (5)$$

IV. Main Results

From equation (5), it is observed that the angular velocity and acceleration are decoupled from the linear velocity and acceleration. Specifically, the resulting dynamics of angular motion form a linear double integrator, which is underactuated. This characteristic allows for the use of a linear PID-like controller to ensure that orientations converge to a consensus agreement point.

Therefore, let $\dot{v}_i$ be defined as:

$$\dot{v}_i = -p_{vi}\varphi(\theta_i)^T e_{zi} - d_{vi} v_i - k_{vi}\tilde{\sigma}_{vi}$$

$$\tilde{\sigma}_{vi} := \hat{\sigma}_{vi} + \frac{1}{k_{vi}}\sigma_{vi}$$
$$\dot{\hat{\sigma}}_{vi} = p_{vi}\phi(\theta_i)^T e_{zi} - r_{vi}v_i \quad (6)$$
$$\sigma_{vi} = \frac{1}{m_i r_i}(\delta_{vi} + \delta_{\omega i})$$

and $\dot{\omega}_i$ be defined as:

$$\dot{\omega}_i = -p_{\omega i}e_{\omega i} - d_{\omega i}\omega_i - k_{\omega i}\tilde{\sigma}_{\omega i} + \alpha_i(t, \theta_i, e_{zi})$$
$$\tilde{\sigma}_{\omega i} := \hat{\sigma}_{\omega i} + \frac{1}{k_{\omega i}}\sigma_{\omega i}$$
$$\dot{\hat{\sigma}}_{vi} = p_{\omega i}e_{zi} - r_{\omega i}v_i \quad (7)$$
$$\sigma_{wi} = \frac{2R_i}{I_i r_i}(\delta_{vi} - \delta_{\omega i})$$
$$\alpha_i(t, \theta_i, e_{zi}) = k_{\alpha i}f_i(t)\phi_i(\theta_i)^T e_{zi}$$

Romero et al. [1] proved that control gains will satisfy Globally Asymptotically stability of equilibrium for close loop system. The control gains derived as:

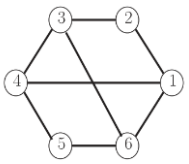
$$d_{vi} = 2d'_{vi} + r_{vi}, k_{vi} = d'_{vi} + r_{vi}$$
$$d_{\omega i} = 2d'_{\omega i} + r_{\omega i}, k_{\omega i} = d'_{\omega i} + r_{\omega i} \quad (8)$$

V. Simulation

In this section, we conduct numerical simulations, Euler Method, with MATLAB to confirm the effectiveness of the suggested controller. The setup used for the simulation, involved a network of six differential-drive mobile robots arranged as depicted in Figure 2. On the left side of Figure 2, the communication topology and the desired hexagonal formation pattern can be observed. The distances x_{di} and y_{di} are specified in the table on the right side, along with their respective disturbances. All robots share identical model parameters $: m = 19\,kg, I = 3\,kg\,m^2, R = 0.3\,m, and\ r = 0.05\,m$. For more detailed information about the model, refer to Shojaei et al. [5].

The control gains were selected as $p_{vi} = -10.5$, $p_{\omega i} = 0.04$, $d'_{vi} = 3.1, r_{vi} = 4.5$, $d'_{\omega i} = 0.052, r_{\omega i} = 0.44$, $k_{\alpha i} = 1.1$ and time-varying signal $f_i(t) = {}^5/_2 + 3\pi\,sin(2t) + \frac{2}{3}\,sin(3t)$.

The mobile robot's initial position is assigned randomly to show the generality of the work for any initial values. The values assigned for the graphs mentioned in the figure that shaped a hexagonal after the run were: $x_i(0) = [5\ 7\ 7\ 3\ 1\ 1]^T$, $y_i(0) = [2\ 5.5\ 3.5\ 2\ 3.5\ 5.5]^T$ and orientations (in rad) $\theta(0) = \left[\frac{\pi}{2}\ 0\ \frac{-\pi}{8}\ \frac{\pi}{8}\ \frac{-\pi}{8}\ \frac{-\pi}{8}\right]$ and we could see that we have a center z_c=[4.4254, 5.4023]. The handling of error states and the rejection of disturbances are demonstrated in Figure 5. From Figure 4 it can be seen that our agents reach the same consensus over time in both planes. Furthermore, it has been shown in Figure 6 and Figure 7, that torques converge to value of disturbances with a minus sign. Disturbances values can be seen at Figure 2.



Robots	1	2	3	4	5	6
x_{di}	2	1	-1	-2	-1	1
y_{di}	0	2	2	0	-2	-2
δ_{vi}	0.12	0.11	0.1	0.14	0.13	0.09
$\delta_{\omega i}$	0.07	0.06	0.065	0.075	0.05	0.055

Figure 2: Communication topology and hexagonal desired formation with the distances x_{di} and $y_{di}$and the disturbances. [1]

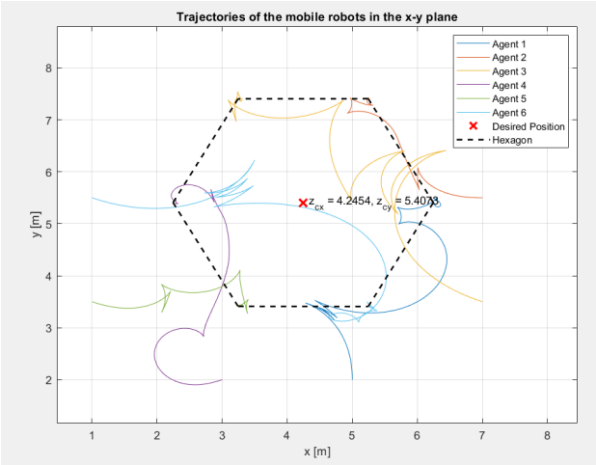


Figure 3: Trajectory of the mobile robots in the x-y plane.

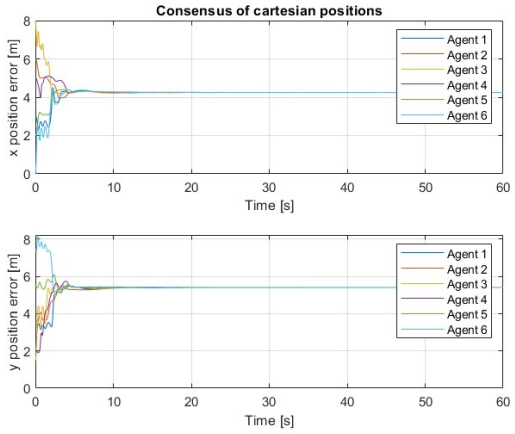


Figure 4:Consensus of the cartesian positions.

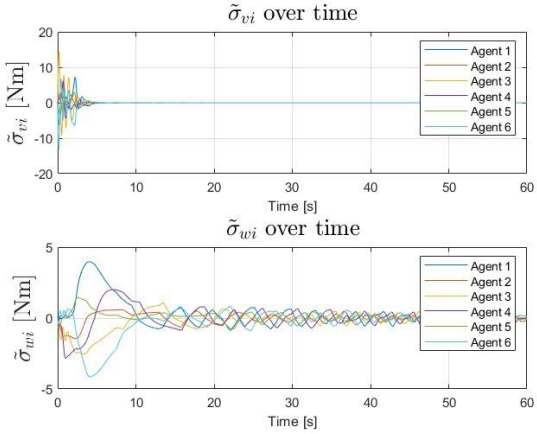


Figure 5: Behavior of the error state $\tilde{\sigma}_{vi}$ and $\tilde{\sigma}_{\omega i}$

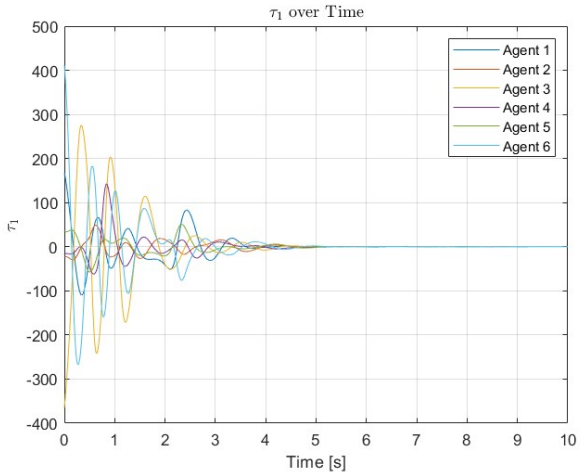


Figure 6: Behaviour of the input signals τ_1

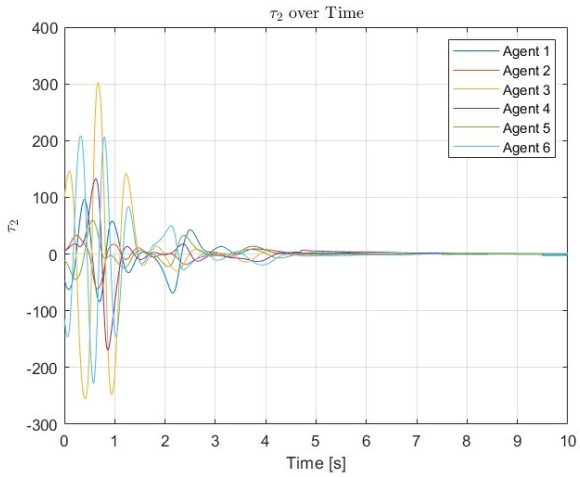


Figure 7: Behaviour of the input signals τ_2

VI. Conclusion

A new controller resembling a time varying PID system was introduced to address stabilizing nonholonomic mobile robots under perturbations. The innovative addition of integral action enabled the rejection of constant unknown disturbances, while the PD component coupled with the time-varying element ensured the convergence of all robot states. Future research will explore three main areas: (1) tackling the control challenge amidst time-varying communication delays, (2) developing an adaptive observer-based controller linked to robot speed, and (3) investigating a dynamic model that incorporates Coriolis and damping forces.

References

[1] Romero, J. G., Nuño, E., & Aldana, C. I. (2023). Robust PID consensus-based formation control of nonholonomic mobile robots affected by disturbances. International Journal of Control, 96(3), 791–799. https://doi.org/10.1080/00207179.2021.2015541

[2] J Emmanuel Nuño, Antonio Loría, Tonatiuh Hernández, Mohamed Maghenem, Elena Panteley, Distributed consensus-formation of force-controlled nonholonomic robots with time-varying delays, Automatica, Volume 120, 2020, 109114, ISSN 0005-1098, https://doi.org/10.1016/j.automatica.2020.109114.

[3] Brockett, R. W. (1983). Asymptotic stability and feedback stabilization. In R. W. Brockett, R. S. Millman, & H. J. Sussmann (Eds.), Differential geometric control theory. Birkhauser

[4] Tzafestas, S. G. (2013). Introduction to mobile robot control. Elsevier.

[5] Shojaei, K., Shahri, A. M., Tarakameh, A., & Tabibian, B. (2011). Adaptive trajectory tracking control of a differential drive wheeled mobile robot. Robotica, 29(3), 391–402. https://doi.org/10.1017/S0263574710000202

Appendix:

```
clc
clear
close all

% Simulation parameters
T = 50; % Final time
dt = 0.001; % Time step
time_vector = 0:dt:T;

% Control gains
p_vi = 10;
p_wi = 0.06;
dprime_vi = 3.1;
r_vi = 4.5;
dprime_wi = 0.052;
r_wi = 0.44;
k_alphai = 1.1;
d_vi = 2*dprime_vi + r_vi;
k_vi = dprime_vi + r_vi;
d_wi = 2*dprime_wi + r_wi;
k_wi = dprime_wi + r_wi;

% Initial conditions
theta_0 = [pi/2 0 -pi/8 pi/8 -pi/8 -pi/8]; % Initial orientation
x_i_0 = 8 * rand(1, 6); % Initial position in x direction
y_i_0 = 8 * rand(1, 6); % Initial position in y direction
z_i_0 = [x_i_0; y_i_0]; % Initial z

% Initialize arrays
w = zeros(1,6);
v = zeros(1,6);
U = zeros(1,6);
u_vi = zeros(1,6);
u_wi = zeros(1,6);
v_dot = zeros(1,6);
w_dot = zeros(1,6);

% Desired States
x_d_i = [2 1 -1 -2 -1 1]';
y_d_i = [0 2 2 0 -2 -2]';
z_d_i = [x_d_i, y_d_i]';

% Model parameters
% All six agents are the same robot
m = 19; % Mass of robots (kg)
I = 3; % Moment of inertia (kg*m^2)
R = 0.3; %
r = 0.05; %

% Disturbance
delta_v_i = [0.12 0.11 0.1 0.14 0.13 0.09]'; % Velocity disturbance
delta_w_i = [0.07 0.06 0.065 0.075 0.05 0.055]'; % Angular velocity disturbance
zigma_wi = ((2*R)/(I*r))*(delta_v_i - delta_w_i);
zigma_vi = ((1)/(m*r))*(delta_v_i + delta_w_i);

% Laplacian matrix
N = 6; % Number of agents
g = graph();
```

```
% Add nodes
g = addnode(g, N);

% Define edges
edges = [1 2; 2 3; 3 4; 4 5; 5 6; 6 1; 4 1; 3 6];

% Add edges to the graph
g = addedge(g, edges(:, 1), edges(:, 2));

% Calculate Laplacian matrix
L1 = laplacian(g);
% Convert Laplacian matrix to full matrix
L = full(L1);
% Plot the topology
plot(g);
title('Our graph');

% Initialize errors to be 0
e_z = 0;
e_z_l = 0;
e_theta = 0;
e_theta_l = 0;

% Initialize arrays to store positions for plotting
x_history = zeros(6, length(time_vector));
y_history = zeros(6, length(time_vector));
z_x_bar_history = zeros(6, length(time_vector));
z_y_bar_history = zeros(6, length(time_vector));
theta_history = zeros(6, length(time_vector));
zigma_hat_vi = zeros(6, length(time_vector));
zigma_hat_wi = zeros(6, length(time_vector));
zigma_tilda_vi = zeros(6, length(time_vector));
zigma_tilda_wi = zeros(6, length(time_vector));
zigma_hat_dot_vi = zeros(6, length(time_vector));
zigma_hat_dot_wi = zeros(6, length(time_vector));
zigma_tilda_vi_history = zeros(6, length(time_vector));
zigma_tilda_wi_history = zeros(6, length(time_vector));
taw = zeros(2, N, length(time_vector));
taw1_history = zeros(6, length(time_vector));
taw2_history = zeros(6, length(time_vector));

for idx = 1:length(time_vector)
    t = time_vector(idx);

    z_bar = z_i_0 - z_d_i;
    f = (5/2) + 3*pi*sin(2*t) + ((2/3)*sin(3*t));

    for i = 1:N

        for j = 1:N % error loop
            % z error
            e_z_l = L(i,j)*(z_bar(:,i) - z_bar(:,j));
            e_z = e_z_l + e_z;
            % theta error
            theta_i_mod = mod(theta_0(:,i), 2*pi);
            theta_j_mod = mod(theta_0(:,j), 2*pi);

            % calculate theta error with constrained values
            e_theta_l = L(i,j)*(theta_i_mod - theta_j_mod);
            e_theta = e_theta_l + e_theta;
        end
```

```
        phi = [cos(theta_0(1,i)); sin(theta_0(1,i))];
        alpha = k_alphai*f*phi'*e_z;

        zigma_hat_dot_vi(1,i) = -p_vi*phi'*e_z + r_vi*v(1, i);
        zigma_hat_dot_wi(1,i) = p_wi*e_theta + r_wi*w(1, i);

        zigma_hat_vi(1,i) = zigma_hat_dot_vi(1,i)*dt + zigma_hat_vi(1,i);
        zigma_hat_wi(1,i) = zigma_hat_dot_wi(1,i)*dt + zigma_hat_wi(1,i);

        zigma_tilda_vi(1,i) = zigma_hat_vi(1,i) - (1/k_vi)*zigma_vi(i,1);
        v_dot(1,i) = p_vi*phi'*e_z - d_vi*v(1, i) - k_vi*zigma_tilda_vi(1,i);

        zigma_tilda_wi(1,i) = zigma_hat_wi(1,i) - (1/k_wi)*zigma_wi(i,1);
        w_dot(1,i) = -p_wi*e_theta - d_wi*w(1, i) - k_wi*zigma_tilda_wi(1,i) + alpha;

        % Update velocities
        v(1,i) = v(1,i) + v_dot(1,i)*dt;
        w(1,i) = w(1,i) + w_dot(1,i)*dt;

        % Taw i
        taw(:, :, idx) = r/2 .* [m*I/(2*R); m*-I/(2*R)] .* [v_dot; w_dot];

        % Update positions and orientations
        z_i_0(1,i) = z_i_0(1,i) + v(1,i)*cos(theta_0(1,i))*dt;
        z_i_0(2,i) = z_i_0(2,i) + v(1,i)*sin(theta_0(1,i))*dt;
        theta_0(1,i) = theta_0(1,i) + w(1,i)*dt;

        % Store variables for plotting
        x_history(i, idx) = z_i_0(1,i);
        y_history(i, idx) = z_i_0(2,i);
        theta_history(i, idx) = theta_0(1,i);
        z_x_bar_history(i, idx) = z_bar(1,i);
        z_y_bar_history(i, idx) = z_bar(2,i);
        zigma_tilda_vi_history(i, idx) = zigma_tilda_vi(1,i);
        zigma_tilda_wi_history(i, idx) = zigma_tilda_wi(1,i);
        taw1_history(i, idx) = taw(1,i);
        taw2_history(i, idx) = taw(2,i);

        % Avoid error accumulation
        e_z=0;
        e_theta=0;
        e_z_l=0;
        e_theta_l=0;
    end
end

% Define center of formation
z_c(1)=(z_i_0(1,1)+z_i_0(1,2)+z_i_0(1,3)+z_i_0(1,4)+z_i_0(1,5)+z_i_0(1,6))/6;
z_c(2)=(z_i_0(2,1)+z_i_0(2,2)+z_i_0(2,3)+z_i_0(2,4)+z_i_0(2,5)+z_i_0(2,6))/6;

% Plotting the results
figure;
colors = lines(N); % Generate N distinct colors
% markers = {'o', 's', 'd', '^', '>', '<'}; % Define N different markers
for i = 1:N
    plot(x_history(i,:), y_history(i,:), 'Color', colors(i,:));
    hold on;
end

% Plot the final desired positions
```

```
plot(z_c(1), z_c(2), 'rx', 'LineWidth', 2, 'MarkerSize', 10);
text(z_c(1), z_c(2), ['  z_{cx} = ', num2str(z_c(1)), ', z_{cy} = ', num2str(z_c(2))]);

% Connect points in z_i_0 with lines
for i = 1:size(z_i_0, 2)
    if i < size(z_i_0, 2)
        plot(z_i_0(1,[i,i+1]), z_i_0(2,[i,i+1]), 'k--', 'LineWidth', 1.5);
    else
        plot(z_i_0(1,[i,1]), z_i_0(2,[i,1]), 'k--', 'LineWidth', 1.5);
    end
end

title('Trajectories of the mobile robots in the x-y plane');
xlabel('x [m]');
ylabel('y [m]');
legend('Agent 1', 'Agent 2', 'Agent 3', 'Agent 4', 'Agent 5', 'Agent 6', 'Desired Position',
'Hexagon');

% Optionally, use xlim and ylim to set the limits of the plot
xlim([min(x_history(:)) - 1, max(x_history(:)) + 1]);
ylim([min(y_history(:)) - 1, max(y_history(:)) + 1]);

grid on;




% Plotting the x and y position errors over time
figure;
subplot(2,1,1);
for i = 1:N
    plot(time_vector, z_x_bar_history(i,:), 'Color', colors(i,:));
    hold on;
end
title('Consensus of cartesian positions');
xlabel('Time [s]');
ylabel('x position error [m]');
legend(arrayfun(@(x) ['Agent ' num2str(x)], 1:N, 'UniformOutput', false));
grid on;

subplot(2,1,2);
for i = 1:N
    plot(time_vector, z_y_bar_history(i,:), 'Color', colors(i,:));
    hold on;
end
xlabel('Time [s]');
ylabel('y position error [m]');
legend(arrayfun(@(x) ['Agent ' num2str(x)], 1:N, 'UniformOutput', false));
grid on;


% Plotting the zigma_tilda_vi and zigma_tilda_wi over time
figure;
subplot(2,1,1);
for i = 1:N
    plot(time_vector, zigma_tilda_vi_history(i,:), 'Color', colors(i,:));
    hold on;
end
title('$\tilde{\sigma}_{vi}$ over time','FontWeight', 'bold','FontSize', 16,
Interpreter="latex");
xlabel('Time [s]');
```

```
ylabel('$\tilde{\sigma}_{vi}$ [Nm]','FontWeight', 'bold','FontSize',
16,Interpreter="latex");
legend(arrayfun(@(x) ['Agent ' num2str(x)], 1:N, 'UniformOutput', false));
grid on;

subplot(2,1,2);
for i = 1:N
    plot(time_vector, zigma_tilda_wi_history(i,:), 'Color', colors(i,:));
    hold on;
end
title('$\tilde{\sigma}_{wi}$ over time','FontWeight', 'bold','FontSize',
16,Interpreter="latex");
xlabel('Time [s]');
ylabel('$\tilde{\sigma}_{wi}$ [Nm]','FontWeight', 'bold','FontSize', 16,
Interpreter="latex");
legend(arrayfun(@(x) ['Agent ' num2str(x)], 1:N, 'UniformOutput', false));
grid on;

% Extracting taw values for plotting
taw1 = squeeze(taw(1, :, :));  % Extract the first row (taw1) for all agents over time
taw2 = squeeze(taw(2, :, :));  % Extract the second row (taw2) for all agents over time

% Plotting taw1 and taw2 over time for each agent
figure;
% Plot taw1
for i = 1:N
    plot(time_vector, taw1(i,:), 'Color', colors(i,:));
    hold on;
end
title('$\tau_1$ over Time', Interpreter="latex");
xlabel('Time [s]');
ylabel('$\tau_1$', Interpreter="latex");
legend(arrayfun(@(x) ['Agent ' num2str(x)], 1:N, 'UniformOutput', false));
grid on;

figure;
% Plot taw2
for i = 1:N
    plot(time_vector, taw2(i,:), 'Color', colors(i,:));
    hold on;
end
title('$\tau_2$ over Time', Interpreter="latex");
xlabel('Time [s]');
ylabel('$\tau_2$', Interpreter="latex");
legend(arrayfun(@(x) ['Agent ' num2str(x)], 1:N, 'UniformOutput', false));
grid on;
```