Nonlinear Dynamics and Systems Theory, **21** (3) (2021) 303–314

Trajectory Tracking of Coordinated Multi-Robot Systems using Nonlinear Model Predictive Control

H. Purnawan, S. Subchan *, D. Adzkiya and T. Asfihani

Control and Optimization Group, Modeling and Simulation Laboratory, Department of Mathematics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Received: November 10, 2020; Revised: June 3, 2021

Abstract: This paper proposes a centralized and decentralized nonlinear model predictive control (NMPC) for multiple robots in the trajectory tracking problem with collision avoidance. The kinematic model of mobile robot is employed to implement these concepts. The path of each robot is constructed such that there exist some intersections between some paths of the robots. Additionally, the initial conditions and parameters of the model are taken so that the robots will collide on the intersections of their paths. Based on the simulation results, both centralized and decentralized schemes can avoid collision between one robot and another one by satisfying the inequality constraints. All solutions of the optimization problems in both schemes are feasible as well, so this indicates that local minimum solutions are found. According to the simulations, the decentralized scheme is better than the centralized scheme in terms of the computational complexity and error tracking.

Keywords: *nonlinear optimization; multiple robots dynamics; nonlinear model predictive control; centralized and decentralized schemes.*

Mathematics Subject Classification (2010): 65K10, 70E60, 90B15, 49M37.

1 Introduction

During the last years, the research interests in robotics area have grown exponentially from some publications (refer to [1–3]), but are not limited to those. Nowadays, to employ a robot for many kinds of tasks is very common in various fields such as agriculture, logistics, and even service for some of Covid-19 patients in hospital [4–6]. In those applications, it is expected that the robot can be navigated in different situations and environments [7]. The strategy can be done by controlling its position, so that it can

* Corresponding author: mailto:subchan@matematika.its.ac.id

move from its current position to the final destination. This concept is usually called the point stabilization control. When the robot has to follow the given path, it is commonly known as the trajectory tracking.

To address many ways in controlling the robot, several techniques have been implemented such as the PID controller [8], sliding mode control [3], sliding PID [9] and others. Two mentioned methods specifically do not use multivariable controls, whereas most robots are a system with multivariable inputs and states usually called the multi-input-multi-output (MIMO) type. Especially for these cases, we have to design many controllers for each input and state by assuming a decoupling control. Through this approach, the system can be treated as the single-input-single-output (SISO) type, so a single loop controller can be developed as well [10].

One of the controllers having a capability to deal with the MIMO systems is a model predictive control (MPC). A version of the MPC for nonlinear systems is a nonlinear model predictive control (NMPC). Because of its ability to handle the limitation of a system, this controller becomes the most popular control method in some sectors such as chemical processes [11], autonomous vehicles [12], and others. As time goes by, the NMPC was also implemented for multi-object systems. Some approaches were proposed, namely, centralized [13], decentralized [14,15], and distributed [13]. Based on the previous study, in this paper, we implement the centralized and decentralized scheme-based NMPC in trajectory tracking, where each path of the robots has some intersections. The aim is to examine our proposed method by defining collision avoidance constraint in the optimization problem of NMPC.

This paper is constructed as follows. The kinematic model of nonholonomic robots in terms of a MIMO system is defined in Section 2. Next, Section 3 explains the centralized and decentralized scheme-based NMPC for multi-agent systems. In this section, the collision avoidance constraint is defined to avoid collision among the robots. Then, Section 4 contains some discussions based on our simulation results. Lastly, Section 5 is the conclusion and development of the future research.

2 Kinematic Model of Mobile Robot

Kinematics is the fundamental study that learns about how an object can move from one position to other positions without considering the forces working on the object. In order to design mobile robots for some purpose, it is essential to understand the mechanical behavior of a robot [16]. The behavior of a mobile robot can be expressed in the following mathematical model [16]:

$$\begin{aligned} \dot{x} &= v\cos\psi, \\ \dot{y} &= v\sin\psi, \\ \dot{\psi} &= \omega, \end{aligned} \tag{1}$$

where (x, y) represents the position of a mobile robot in the X-axis and Y-axis, respectively, ψ is the heading angle of a mobile robot, v and ω, respectively, denote the linear and angular velocity.

Model (1) is a nonlinear system and it can be written in a state space form as follows:

$$\dot{\boldsymbol{\chi}} = \boldsymbol{f}(\boldsymbol{\chi}, \boldsymbol{u}), \tag{2}$$

where $\boldsymbol{\chi} = [x, y, \psi]^T$ denotes the state variables and $\boldsymbol{u} = [v, \omega]^T$ indicates the input or control variables for the system. Equation (2) is a continuous-time system, so it can be converted into a discrete-time system using a discretization method. Based on the Euler approach, equation (2) is stated as

$$\boldsymbol{\chi}(k+1) = \boldsymbol{\chi}(k) + T_s \boldsymbol{f}(\boldsymbol{\chi}(k), \boldsymbol{u}(k)), \tag{3}$$

where T_s is the sampling time used to transform the continuous-time system into a discrete-time system.

3 Nonlinear Model Predictive Control

The NMPC is an optimization-based method for nonlinear systems [17]. Generally, the difference between the MPC and the NMPC lies in their systems. However, it is not limited to those, since the MPC can be implemented on the nonlinear systems by using the linearization method around operating points (refer to [18]).The idea of the NMPC is to predict the future system behavior by utilizing the model (3). The aim of the NMPC is to minimize an objective function subject to its constraints, so it is possible to obtain the optimal inputs along the prediction horizon.

Assuming that the prediction horizon is equal to the control horizon at every discrete-time k, the optimization formula of the NMPC can be written as follows:

$$V_N(k) = \sum_{i=1}^{N} \|\boldsymbol{r}_i(k) - \hat{\boldsymbol{y}}_i(k)\|^2_{\boldsymbol{Q}} + \|\boldsymbol{u}_{i-1}(k)\|^2_{\boldsymbol{R}} \tag{4}$$

subject to

$$\begin{aligned}
&\boldsymbol{\chi}_{i+1}(k) = \boldsymbol{f}_d(\boldsymbol{\chi}_i(k), \boldsymbol{u}_i(k)), \quad i = 0, 1, \cdots, N-1,\\
&\hat{\boldsymbol{y}}_i(k) = \boldsymbol{g}(\boldsymbol{\chi}_i(k)), \quad i = 1, 2, \cdots, N,\\
&\boldsymbol{\chi}_0(k) = \boldsymbol{\chi}(k),\\
&\boldsymbol{\chi}^{min} \le \boldsymbol{\chi}_i(k) \le \boldsymbol{\chi}^{max}, \quad i = 0, 1, \cdots, N,\\
&\boldsymbol{u}^{min} \le \boldsymbol{u}_i(k) \le \boldsymbol{u}^{max}, \quad i = 0, 1, \cdots, N-1,
\end{aligned}$$

where V_N is a quadratic cost function, N is the prediction horizon, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are the weighting matrices being positive semi-definite and positive definite, where their sizes are appropriate to the dimension of outputs and inputs, $\boldsymbol{r}$ is the reference or desired value, $\hat{\boldsymbol{y}}$ is the predicted outputs of the system, $\boldsymbol{\chi}^{min}$, $\boldsymbol{\chi}^{max}$, $\boldsymbol{u}^{min}$, $\boldsymbol{u}^{max}$ are the lower and upper bounds for the states and inputs, respectively, and the notation in (4) is given as follows:

$$\begin{aligned}
\|\boldsymbol{r}_i(k) - \hat{\boldsymbol{y}}_i(k)\|^2_{\boldsymbol{Q}} &= [\boldsymbol{r}_i(k) - \hat{\boldsymbol{y}}_i(k)]^T \boldsymbol{Q} [\boldsymbol{r}_i(k) - \hat{\boldsymbol{y}}_i(k)],\\
\|\boldsymbol{u}_{i-1}(k)\|^2_{\boldsymbol{R}} &= (\boldsymbol{u}_{i-1}(k))^T \boldsymbol{R} \boldsymbol{u}_{i-1}(k).
\end{aligned}$$

By solving the optimization problem (4), we obtain a sequence of optimal inputs $\{\boldsymbol{u}^*_0(k), \boldsymbol{u}^*_1(k), \cdots, \boldsymbol{u}^*_{N-1}(k)\}$, but only the first element of the sequence, i.e., $\boldsymbol{u}^*_0(k)$, is injected to the system.

To implement the NMPC for multi-robot systems, it is important to formulate the constraints appropriately to avoid collision among the agents. The following approach is used to manifest the concept.

Collision avoidance among the robots: to ensure safety during the trajectory execution, it is necessary to formulate the following inequality constraints using the Euclidean distance in optimization problem (4). By considering the discrete system for each prediction window, we define the constraint as follows:

$$\sqrt{(x_i^j(k)-x_i^l(k))^2+(y_i^j(k)-y_i^l(k))^2} \geq \frac{1}{2}(r_d^j+r_d^l),\ j \neq l,$$
$$\forall j, l = 1, 2, \cdots, N_r,\ \forall i = 0, 1, \cdots, N, \tag{5}$$

where $(x_i(k), y_i(k))$ is the position of robots at time k, r_d is the diameter of robots, N_r represents the number of robots, and the superscript denotes the robot index.

To design the NMPC controller for multiple robots, some schemes that can be utilized are the centralized and decentralized schemes. The explanation about the schemes is discussed in the following subsection.

3.1 Centralized NMPC

The centralized scheme is a classical control scheme, where the whole system is controlled by one controller [13]. Since we treat all robots as a single entity, the system can be written as a large-scale system as follows:

$$\bar{\boldsymbol{\chi}}(k+1) = \begin{bmatrix} \boldsymbol{f}_d^1(\boldsymbol{\chi}^1(k), \boldsymbol{u}^1(k)) \\ \boldsymbol{f}_d^2(\boldsymbol{\chi}^2(k), \boldsymbol{u}^2(k)) \\ \vdots \\ \boldsymbol{f}_d^{N_r}(\boldsymbol{\chi}^{N_r}(k), \boldsymbol{u}^{N_r}(k)) \end{bmatrix} = \bar{\boldsymbol{f}}_d(\bar{\boldsymbol{\chi}}(k), \bar{\boldsymbol{u}}(k)), \tag{6}$$

where the state and input variables are defined as $\bar{\boldsymbol{\chi}}(k) = [\boldsymbol{\chi}^1(k), \boldsymbol{\chi}^2(k), \cdots, \boldsymbol{\chi}^{N_r}(k)]^T$ and $\bar{\boldsymbol{u}}(k) = [\boldsymbol{u}^1(k), \boldsymbol{u}^2(k), \cdots, \boldsymbol{u}^{N_r}(k)]^T$, respectively. These changes also influence the formula of objective function and constraints for our optimization problem. The new objective function and constraints can be stated as

$$\bar{V}_N(k) = \sum_{i=1}^{N} \left\|\bar{\boldsymbol{r}}_i(k) - \bar{\hat{\boldsymbol{y}}}_i(k)\right\|_{\bar{\boldsymbol{Q}}}^2 + \|\bar{\boldsymbol{u}}_{i-1}(k)\|_{\bar{\boldsymbol{R}}}^2 \tag{7}$$

subject to

$$\begin{aligned}
&\bar{\boldsymbol{\chi}}_{i+1}(k) = \bar{\boldsymbol{f}}_d(\bar{\boldsymbol{\chi}}_i(k), \bar{\boldsymbol{u}}_i(k)), \quad i = 0, 1, \cdots, N-1, \\
&\bar{\hat{\boldsymbol{y}}}_i(k) = \bar{\boldsymbol{g}}(\bar{\boldsymbol{\chi}}_i(k)), \quad i = 1, 2, \cdots, N, \\
&\bar{\boldsymbol{\chi}}_0(k) = \bar{\boldsymbol{\chi}}(k), \\
&\bar{\boldsymbol{\chi}}^{min} \leq \bar{\boldsymbol{\chi}}_i(k) \leq \bar{\boldsymbol{\chi}}^{max}, \quad i = 0, 1, \cdots, N, \\
&\bar{\boldsymbol{u}}^{min} \leq \bar{\boldsymbol{u}}_i(k) \leq \bar{\boldsymbol{u}}^{max}, \quad i = 0, 1, \cdots, N-1, \\
&\sqrt{(x_i^j(k)-x_i^l(k))^2+(y_i^j(k)-y_i^l(k))^2} \geq \frac{1}{2}(r_d^j+r_d^l),\ j \neq l, \forall j, l = 1, 2, \cdots, N_r, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \forall i = 0, 1, \cdots, N,
\end{aligned}$$

where $\bar{\boldsymbol{r}}(k) = [\boldsymbol{r}^1(k), \cdots, \boldsymbol{r}^{N_r}(k)]^T$, $\bar{\hat{\boldsymbol{y}}}(k) = [\hat{\boldsymbol{y}}^1(k), \cdots, \hat{\boldsymbol{y}}^{N_r}(k)]^T$ and the weighting matrices $\bar{\boldsymbol{Q}}$ and $\bar{\boldsymbol{R}}$ are defined as $\bar{\boldsymbol{Q}} = \mathrm{diag}(\boldsymbol{Q}, \boldsymbol{Q}, \cdots, \boldsymbol{Q})$ and $\bar{\boldsymbol{R}} = \mathrm{diag}(\boldsymbol{R}, \boldsymbol{R}, \cdots, \boldsymbol{R})$ with respect to the size of $\bar{\hat{\boldsymbol{y}}}$ and $\bar{\boldsymbol{u}}$, respectively.

At each sampling time, Algorithm 3.1 is used to obtain the optimal input in the case of the centralized NMPC scheme.

Algorithm 3.1 The Centralized NMPC for Multi-Agent Systems.

1: Predict future outputs $\bar{\hat{\boldsymbol{y}}}_i(k),\ i = 1, 2, \cdots, N$, as a function of future inputs $\bar{\boldsymbol{u}}_i(k)$, $i = 0, 1, \cdots, N-1$, using the current states $\bar{\boldsymbol{\chi}}(k)$.

2: Solve the optimization problem (7) using a *fmincon* MATLAB function by utilizing an anonymous function to obtain the sequence of optimal inputs $\{\bar{\boldsymbol{u}}_0(k), \bar{\boldsymbol{u}}_1(k), \cdots, \bar{\boldsymbol{u}}_{N-1}(k)\}$.

3: Only the first element of the optimal input sequence $\bar{\boldsymbol{u}}_0(k)$ is applied to the systems by extracting it into $\{\boldsymbol{u}_0^1(k), \boldsymbol{u}_0^2(k), \cdots, \boldsymbol{u}_0^{N_r}(k)\}$ for each system.

4: Save the data and update the current states.

5: At time-$k+1$, repeat step 1.

The drawback of this scheme is the computational complexity that grows significantly when the number of agents increases, so it is not suitable for the case with too many agents. Next, the alternative scheme will be presented to control multi-agent systems.

3.2 Decentralized NMPC

In this architecture, the optimization problem is solved in parallel, i.e., each robot solves the optimization separately [15]. As in (5), to avoid collision among robots, each optimizer needs other robots' future states information. By using communication, this study assumes that all robots can share required information such as the current states and optimal inputs over the prediction horizon in the previous calculation at every sampling time [14]. If the communication is failed, the current and one-step previous states are used by each optimizer to estimate the last input commands of other robots, then it is used to predict other robots' future state information by reduplicating it over the prediction horizon [15]. The illustration of this method can be seen in [13, 15]. In the decentralized control, there are multiple NMPCs for each robot. Algorithm 3.2 shows the process to obtain the sequence of optimal inputs for each agent by using information from other agents.

Algorithm 3.2 The Decentralized NMPC for Multi-Agent Systems.

1: Guess the suitable initial inputs over the prediction horizon to check the inequality constraints (5) for collision avoidance by using current states information from other agents via communication.

2: Predict future outputs $\hat{\boldsymbol{y}}_i^j(k),\ i = 1, 2, \cdots, N$, as a function of future inputs $\boldsymbol{u}_i^j(k)$, $i = 0, 1, \cdots, N-1$, using the current states $\boldsymbol{\chi}^j(k)$ for the j-th agent.

3: Solve the optimization problem (4) by adding (5) for every agent using *fmincon* MATLAB by utilizing an anonymous function to obtain a sequence of optimal inputs $\{\boldsymbol{u}_0^j(k), \boldsymbol{u}_1^j(k), \cdots, \boldsymbol{u}_{N-1}^j(k)\}$.

4: Only the first element of the optimal input sequence $\boldsymbol{u}_0^j(k)$ is applied to the system of the j-th agent.

5: Save the data and update the current states.

6: At time-$k+1$, the obtained optimal input over the prediction horizon from (3) and updated current states are needed to check the inequality constraints by other agents via communication.

7: Repeat step 2.

Algorithm 3.2 is used by every agent to obtain its optimal inputs. This scheme is more efficient than the centralized one w.r.t both computational time and control design.

4 Simulation Results and Discussion

In this paper, both centralized and decentralized NMPC use the same prediction horizon N and sampling time T_s. The weighting matrices in the objective function and simulation time for each robot are given by $\boldsymbol{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\boldsymbol{R} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$ and $t_{end} = 400$ s. The initial conditions and references of all simulations are shown in Table 1, the diameter of all robots is assumed 0.3 m and the upper and lower bounds of the inputs are defined as $v \in [-0.6, 0.6]$ for linear velocity and for angular velocity $\omega \in [-\frac{\pi}{4}, \frac{\pi}{4}]$, respectively.

Table 1: The initial conditions and references of each robot.

Robot	Initial conditions	References	
		$x_{ref}(t)$	$y_{ref}(t)$
1	$[-2, 3.5, 0]^T$	$-1 + 2\sin(0.02t)$	$1 + 2\cos(0.02t)$
2	$[-2, -3.5, 0]^T$	$-1 + 2\sin(0.02t)$	$-1 - 2\cos(0.02t)$
3	$[2, 3.5, \pi]^T$	$1 - 2\sin(0.02t)$	$1 + 2\cos(0.02t)$
4	$[2, -3.5, \pi]^T$	$1 - 2\sin(0.02t)$	$-1 - 2\cos(0.02t)$

All simulations are performed in MATLAB 2019b on a computer with 8GB RAM and a core i5-1035G4. For the first running simulation, we use $T_s = 0.5$ s and $N = 8$ (equal to 4 s) with the initial conditions as in Table 1. The simulation result of the centralized scheme to follow the specified references can be seen in Figure 1 while for the decentralized scheme it is shown in Figure 2.

Figures 1 and 2 show all robots can follow their references using both centralized and decentralized schemes from the initial conditions of each robot. The tracking errors of each robot are caused by avoiding the collision among them, so these results are appropriate to our desire. From the figures, we can see the decentralized scheme is better than the centralized one if it is seen from the tracking error between their trajectories and references. The intersections among references are constructed to verify whether the proposed approach through the algorithms can work perfectly.

To ensure that the inequality constraints for collision avoidance are satisfied by each robot, we show the distance between one robot and other robots. The results of our simulation about this idea are presented in Figures 3 and 4 for each proposed scheme.

In Figures 3 and 4, the requested minimum distance so that the collision does not occur is a half of the total diameter of each robot, i.e., 0.3 m. The symbol of $D_{jl}(k)$ represents the distance between the i-th and j-th robot at time k computed by the following formula:

$$D_{jl}(k) = \sqrt{(x^j(k) - x^l(k))^2 + (y^j(k) - y^l(k))^2}, \quad j \neq l, \; j, l = 1, 2, 3, 4.$$

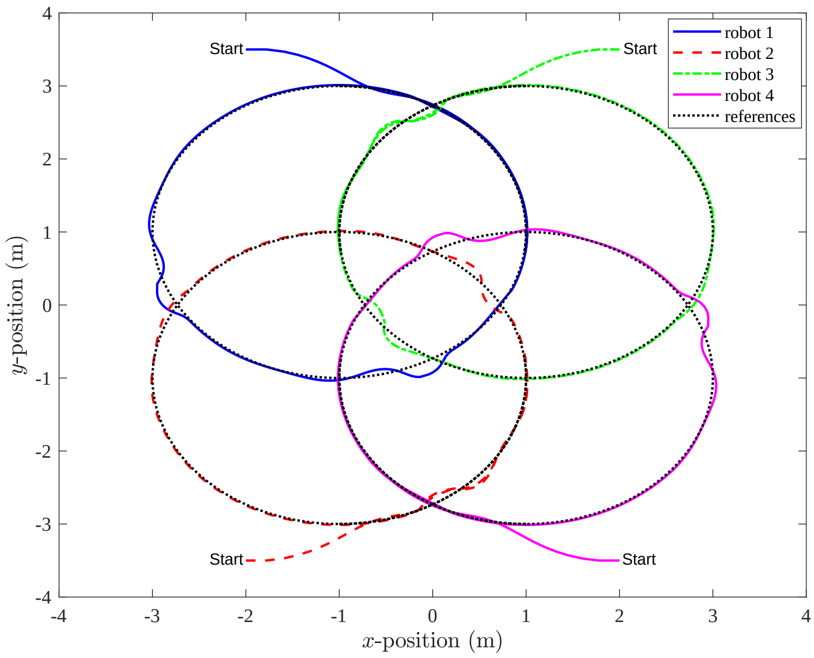

Figure 1: Trajectory Tracking for Each Robot using the Centralized NMPC.

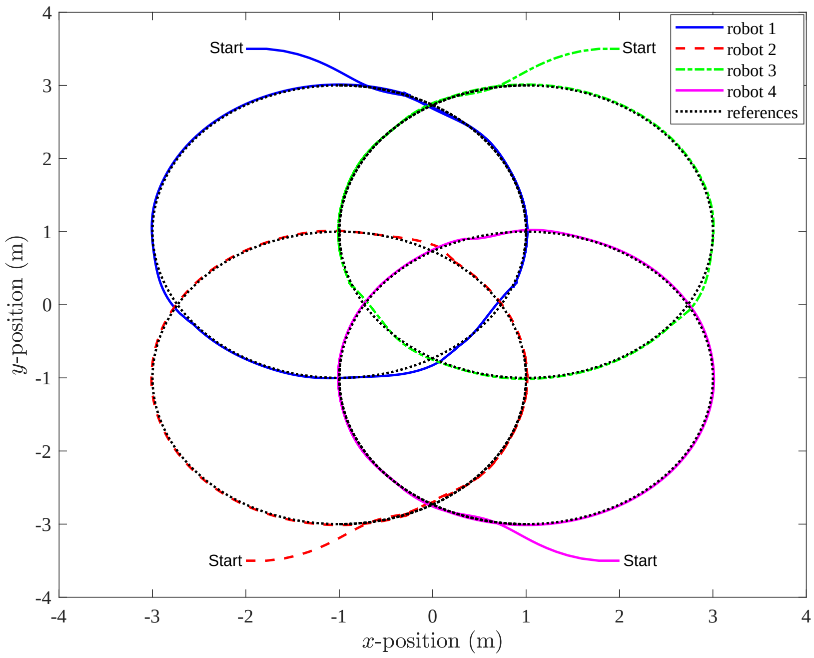

Figure 2: Trajectory Tracking for Each Robot using the Decentralized NMPC.

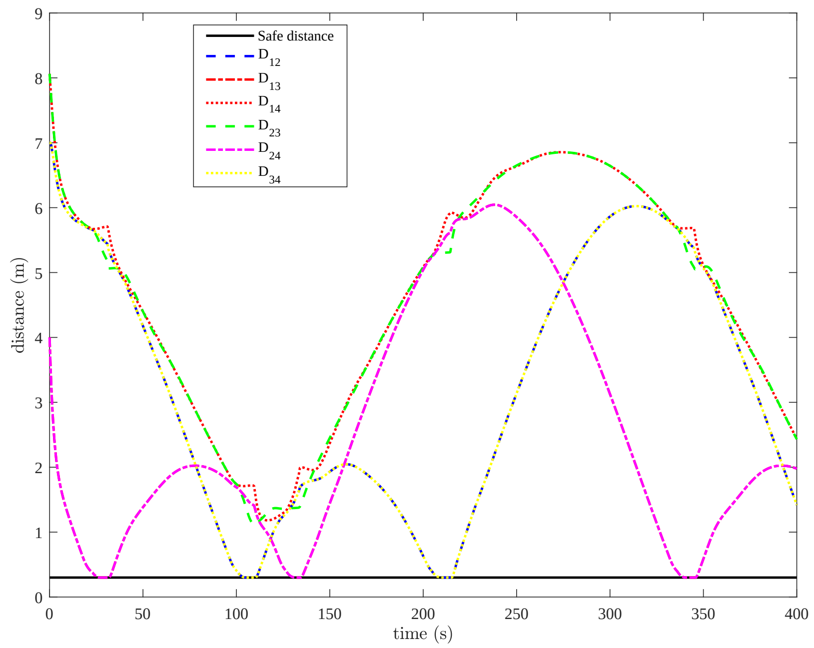

Figure 3: The Distance Between One Robot and Other Robots Based on the Centralized Scheme.

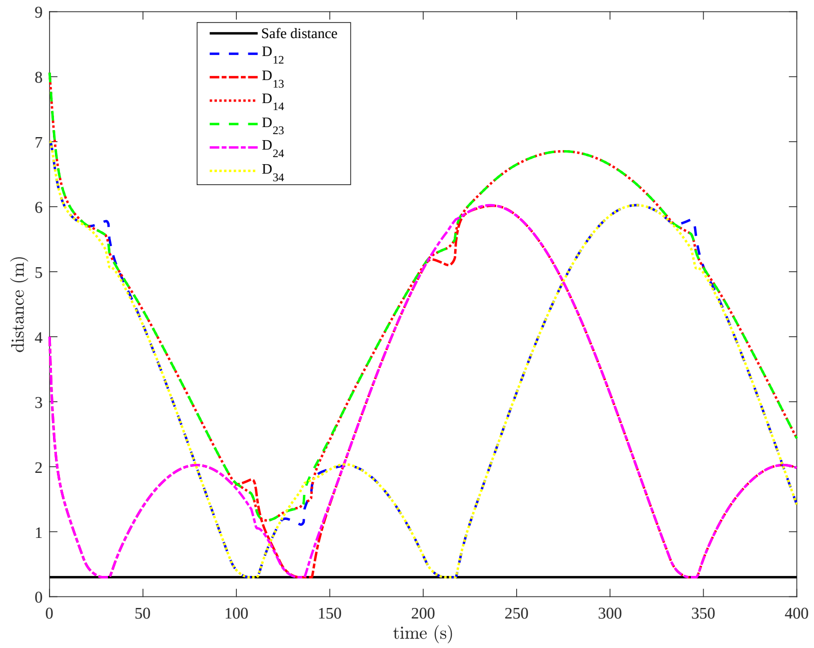

Figure 4: The Distance Between One Robot and Other Robots Based on the Decentralized Scheme.

Based on Figures 3 and 4, it can be seen that the distance among the robots satisfies the safe distance to avoid collision. It can be realized since we defined inequality constraint (5) in our optimization problem based on the NMPC approach at every discrete-time k. The optimal inputs of each robot shown in Figures 5 and 6 also satisfy the lower and upper bounds that we defined previously. This indicates that the solutions of our optimization problem are feasible at every discrete-time k. Figures 5 and 6 also inform that the NMPC tries to keep the inputs to be convergent to 0 in order to minimize costs when the robots move away from each other.

To compare the computational time for the centralized and decentralized schemes, the different groups of prediction horizon and sampling time (N, T_s) are shown in Table 2. This is used to show which method is more effective to control and guide all robots in order that they are on their trajectories.

Table 2: Parameters for different simulations.

Parameters	Scenario 1	Scenario 2
Prediction horizon (N)	5	8
Sampling time (T_s)	1 s	0.5 s

Based on Table 2, the complexity of computational time for both proposed schemes in the case of controlling 4 robots can be investigated. The results of this idea are presented in Table 3.

Table 3: The computational time between the centralized and decentralized schemes.

Scenario	Centralized	Decentralized
1	59.7629 s	39.3990 s
2	244.6630 s	184.7699 s

Based on Table 3, it is clear that the increase of prediction horizon implies the increase of computational time for both centralized and decentralized schemes. The suggestion of this issue is needed to choose an appropriate prediction horizon and sampling time to reduce the complexity time while satisfying a desired performance. Table 3 also gives an information that the decentralized scheme is more effective than the centralized one in terms of computational time since the decentralized scheme provides fast computational time with good performance as shown in the previous simulation in Figures 2 and 4.

5 Conclusion

In this study, two different schemes are proposed, namely, a centralized and decentralized NMPC to control multi-robot systems. These schemes are implemented for the trajectory tracking problem where there are some intersections among the robot paths. Based on our simulation, both schemes give satisfying results as all robots can follow their references without a collision between one robot and another robot. In addition, the distance between one robot and another robot is greater than or equal to the safe distance, so this ensures that all robots can avoid collision between each other. For control inputs, the NMPC also guarantees that the actuator saturation limits are satisfied. In our simulation, it can be concluded that the decentralized scheme is better than the centralized scheme.

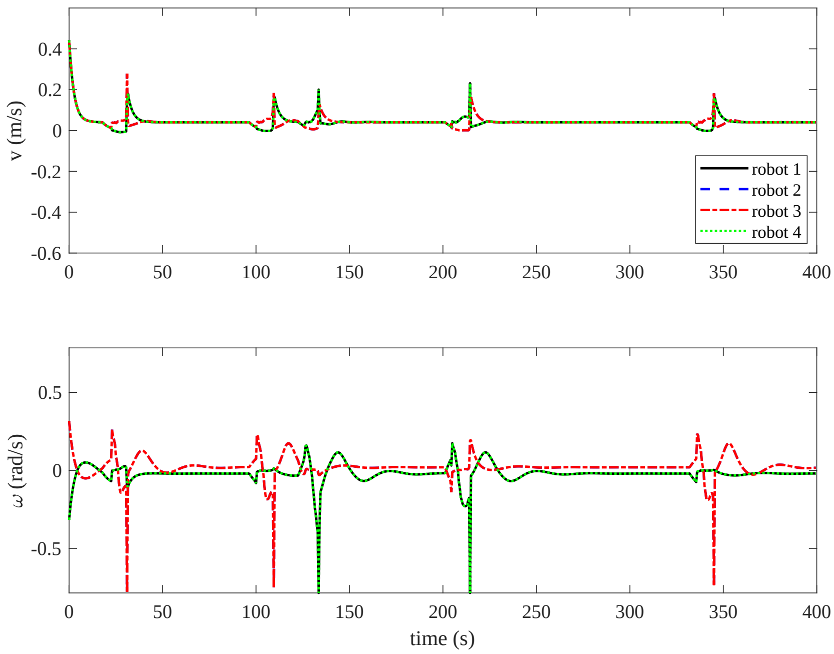

Figure 5: The Control Inputs of Each Robot Based on the Centralized Scheme.

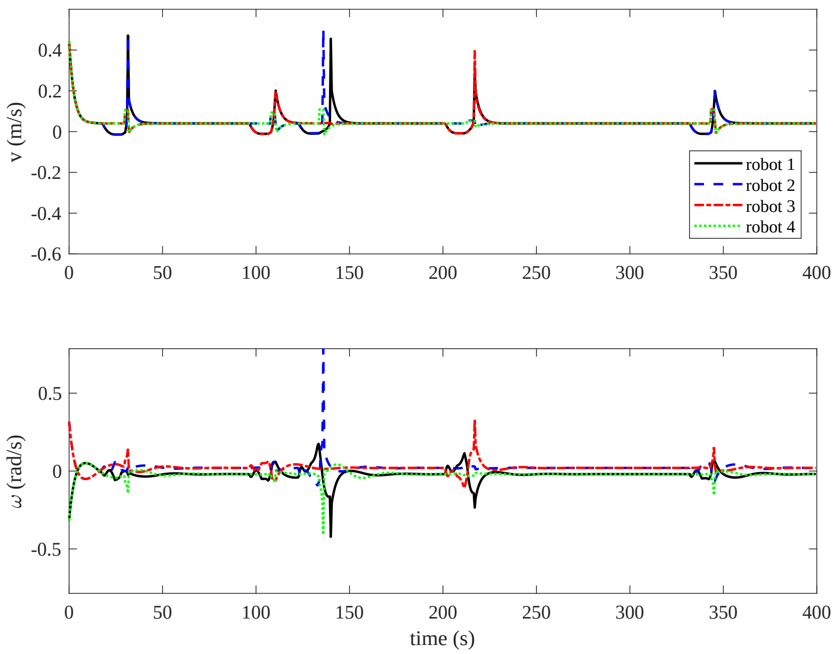

Figure 6: The Control Inputs of Each Robot Based on the Decentralized Scheme.

This is shown in the computational complexity that gives the fast computational time as the prediction horizon increases. For the future research, we will include disturbance and consider some obstacles in the more complex cases such as unmanned aerial vehicles (UAVs) and autonomous surface vehicles (ASVs).

Acknowledgment

The authors would like to thank DRPM RISTEKDIKTI (No. 3/E1/KP.PTNBH/2021) for funding this research by Doctoral Dissertation Research (Penelitian Disertasi Doktor) scheme with contract No. 851/PKS/ITS/2021.

References

[1] J. Alonso-Mora, P. Beardsley and R. Siegwart. Cooperative collision avoidance for nonholonomic robots. *IEEE Transactions on Robotics* **34** (2) (2018) 404–420.

[2] M. Mendili and F. Bouani. Predictive control of mobile robot using kinematic and dynamic models. *Journal of Control Science and Engineering* **2017** (2017).

[3] F. G. Rossomando, C. Soria and R. Carelli. Sliding mode neuro adaptive control in trajectory tracking for mobile robots. *Journal of Intelligent & Robotic Systems* **74** (3-4) (2014) 931–944.

[4] A. Bechar and C. Vigneault. Agricultural robots for field operations. Part 2: Operations and systems. *Biosystems Engineering* **153** (2017) 110–128.

[5] L. Haiming, L. Weidong, Z. Mei and C. An. Algorithm of Path Planning Based on Time Window for Multiple Mobile Robots in Warehousing System. In: *2019 Chinese Control Conference (CCC)* 2019, 2193–2199.

[6] M. Tavakoli, J. Carriere and A. Torabi. Robotics For COVID-19: How Can Robots Help Health Care in the Fight Against Coronavirus, 03, 2020.

[7] G. Farias, G. Garcia, G. Montenegro, E. Fabregas, S. Dormido-Canto and S. Dormido. Reinforcement Learning for Position Control Problem of a Mobile Robot. *IEEE Access* **8** (2020) 152941–152951.

[8] F. Ouendi, S. Houti and M. Tadjine. Comparison of Quadrotor Performance Using Forwarding and PID-Backstepping Control. *Nonlinear Dynamics and System Theory* **19** (3) (2019) 427–443.

[9] T. Herlambang, S. Subchan, H. Nurhadi and D. Adzkiya. Motion Control Design of UNUSAITS AUV Using Sliding PID. *Nonlinear Dynamics and System Theory* **20** (1) (2020) 51–60.

[10] G. He, J. Li and P. Cui. Decoupling control design for the module suspension control system in maglev train. *Mathematical Problems in Engineering* **2015** (2015).

[11] J. A. Paulson, E. A. Buehler, R. D. Braatz and A. Mesbah. Stochastic model predictive control with joint chance constraints. *International Journal of Control* **93** (1) (2020) 126–139.

[12] I. B. Nascimento, A. Ferramosca, L. C. Piment and G. V. Raffo. NMPC Strategy for a Quadrotor UAV in a 3D Unknown Environment. In: *2019 19th International Conference on Advanced Robotics (ICAR)* 2019, 179–184.

[13] S. S. Mansouri, G. Nikolakopoulos and T. Gustafsson. Distributed model predictive control for unmanned aerial vehicles. In: *2015 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)* 2015, 152–161.

[14] S. Kim, H. Oh and A. Tsourdos. Nonlinear model predictive coordinated standoff tracking of a moving ground vehicle. *Journal of Guidance, Control and Dynamics* **36** (2) (2013) 557–566.

[15] J. Shin and H. J. Kim. Nonlinear model predictive formation flight. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* **39** (5) (2009) 1116–1125.

[16] R. Siegwart, I. R. Nourbakhsh and D. Scaramuzza. *Introduction to Autonomous Mobile Robots*. MIT press, 2011.

[17] L. Grüne and J. Pannek. *Nonlinear Model Predictive Control*. Springer, 2017.

[18] A. Franco and V. Santos. Short-term path planning with multiple moving obstacle avoidance based on adaptive MPC. In: *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)* 2019, 1–7.