# Synchronization-Based Cooperative Distributed Model Predictive Control

Julius Beerwerth[1], Maximilian Kloock[2] and Bassam Alrifaee[1]

[1] Department of Aerospace Engineering,
University of the Bundeswehr Munich,
85579 Neubiberg, Germany,
[firstname].[lastname]@unibw.de
[2] Department of Computer Science,
RWTH Aachen University,
52062 Aachen, Germany

**Abstract.** Distributed control algorithms are known to reduce overall computation time compared to centralized control algorithms. However, they can result in inconsistent solutions leading to the violation of safety-critical constraints. Inconsistent solutions can arise when two or more agents compute concurrently while making predictions on each others control actions. To address this issue, we propose an iterative algorithm called Synchronization-Based Cooperative Distributed Model Predictive Control, which we presented in [1]. The algorithm consists of two steps: 1. computing the optimal control inputs for each agent and 2. synchronizing the predicted states across all agents. We demonstrate the efficacy of our algorithm in the control of multiple small-scale vehicles in our Cyber-Physical Mobility Lab.

**Keywords:** Cooperative Control, Distributed Control, Prediction Consistency, Model Predictive Control, Connected and Automated Vehicles

## 1 Introduction

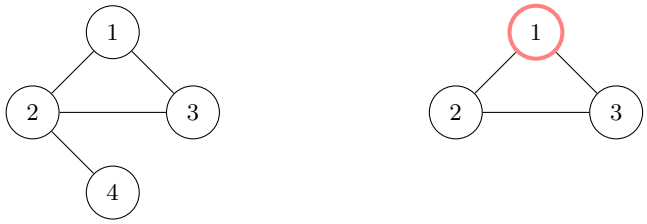
Distributed control algorithms in networked control system (NCS) offer enhanced scalability, flexibility, and fault tolerance compared to Centralized Model Predictive Control (CMPC). Cooperative Distributed Model Predictive Control (CDMPC) is a popular approach, but it faces challenges with prediction inconsistencies due to limited local system knowledge. All agents compute their control inputs based on local subsystem knowledge while only predicting the control inputs of the neighboring agents. If a predicted control input diverts from the computed control input, this is called a prediction inconsistency and can lead to safety-critical failures. In this work, we show how to guarantee prediction consistency using synchronization based on [1] and [2]. Previous work on achieving prediction consistency in CDMPC includes sequential approaches and parallel approaches. In sequential approaches, the agents compute sequentially and achieve prediction consistency by sharing their predicted outputs [3,4,5].

This extended abstract is based on the original paper published in 2024 European Control Conference (ECC). For full details, including complete methodology and experimental results, please refer to the published work, DOI: ITSC57777.2023.10422078

Achieving prediction consistency in parallel approaches is more difficult, but desirable as they are more scalable. The approaches explored in the literature can be clustered into different ideas. In [6], the coupling graph is divided into disjoint sub-graphs that are fully connected and can be analyzed independently. In [7], the agents are given initial decisions and adapt these decisions slightly to reduce their local cost. This approach does not scale well, as it requires computing the initial decisions. The works of [8,9,10] use fuzzy logic to deal with the prediction inconsistencies. The authors of [11,12,13] use coalitional games from game theory to cluster agents in the coupling topology, leading to potentially high node degrees. Instead of distributing the control problem, it is also possible to solve the centralized optimization problem using distributed optimization as shown in [14,15,16,17,18].

## 2 Background

We consider multiple agents in an NCS. The topology of the NCS is modeled as a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ called coupling graph. Here, $\mathcal{V}$ represents the set of nodes, $\mathcal{E}$ the set of edges and $\mathcal{W}$ their corresponding weights. Each node in the coupling graph represents an agent and the edges the coupling relations between them. We define the coupling sub-graph $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i, \mathcal{W}_i)$, consisting of agent $i$, its neighbors and the edges between them. Figure 1 shows an example coupling graph and the coupling sub-graph for agent 1.




(a) Coupling graph $\mathcal{G}$. (b) Coupling sub-graph $\mathcal{G}_1$ for agent 1.

Fig. 1: Example for (a) a coupling graph and (b) the corresponding coupling sub-graph for agent 1.

Consider the distributed control problem in an NCS. At each time step, we aim to compute the optimal control inputs for each agent $i \in \mathcal{V}$ to follow its reference trajectory $\boldsymbol{r}_i$. To solve this distributed control problem we use CDMPC as it allows us to incorporate individual as well as joint objectives and constraints. The set of agents that agent $i$ cooperates with is given by the set $\mathcal{V}_i$ in the coupling sub-graph $\mathcal{G}_i$. We formulate the local CDMPC problem for agent $i$ as

$$
\begin{aligned}
\text{minimize} \quad & \sum_{j\in\mathcal{V}_i} w_{i\to j} \sum_{k=1}^{N_p-1} \ell_j^x\left(\boldsymbol{x}_{i\to j}(k), \boldsymbol{r}_j(k)\right) + \sum_{j\in\mathcal{V}_i} w_{i\to j}\ell_j^f\left(\boldsymbol{x}_{i\to j}(N_p), \boldsymbol{r}_j(N_p)\right) + \\
& \sum_{j\in\mathcal{V}_i} w_{i\to j} \sum_{k=0}^{N_u-1} \ell_j^u\left(\Delta\boldsymbol{u}_{i\to j}(k)\right) + \sum_{(j,q)\in\mathcal{E}_i} \sum_{k=1}^{N_p} \ell_{j\to q}^{\mathrm{c}}(\boldsymbol{x}_{i\to j}(k), \boldsymbol{x}_{i\to q}(k)) \\
\text{subject to} \quad & \forall j \in \mathcal{V}_i, q \in \mathcal{V}_i \cap \mathcal{V}_j \\
& \boldsymbol{x}_{i\to j}(k) = f_j(\boldsymbol{x}_{i\to j}(k), \boldsymbol{u}_{i\to j}(k)), \quad l = 1, \ldots, N_p - 1, \\
& \boldsymbol{x}_{i\to j}(k) \in \mathcal{X}_j, \qquad\qquad l = 1, \ldots, N_p - 1, \\
& \boldsymbol{x}_{i\to j}(N_p) \in \mathcal{X}_j^{\mathrm{f}}, \\
& \boldsymbol{u}_{i\to j}(k) \in \mathcal{U}_j, \qquad\qquad l = 0, \ldots, N_u - 1, \\
& \Delta\boldsymbol{u}_{i\to j}(k) \in \Delta\mathcal{U}_j, \qquad\qquad l = 0, \ldots, N_u - 1, \\
& c_{j\to q}^{\mathrm{c}}(\boldsymbol{x}_{i\to j}(k), \boldsymbol{x}_{i\to q}(k)) \leq 0, \qquad\qquad k = 1, \ldots, N_p,
\end{aligned}
\tag{1}
$$

where $\ell_j^x, \ell_j^{\mathrm{f}}, \ell_j^u$ denote the reference deviation cost, the terminal cost, and the input variation cost for agent $j$, respectively, and $\ell_{j\to q}^{\mathrm{c}}$ denotes the coupling objective between agent $j$ and agent $q$. The variables $\boldsymbol{x}_{i\to j}(k)$, $\boldsymbol{u}_{i\to j}(k)$ and $\Delta\boldsymbol{u}_{i\to j}(k)$ represent the state, the input, and the input variation at time step $k$ of agent $j$ predicted by agent $i$. The parameters $N_p$ and $N_u$ denote the prediction and the control horizon, respectively, and $w_{i\to j}$ represents the weight of the corresponding edge. The function $f_j$ represents the system dynamics and $\mathcal{X}_j, \mathcal{X}_j^{\mathrm{f}}, \mathcal{U}_j$, and $\Delta\mathcal{U}_j$ represent the set of feasible states, the terminal set, the set of feasible inputs, and the set of feasible input variations, respectively. The function $c_{j\to q}^{\mathrm{c}}$ denotes the coupling constraint between agent $j$ and agent $q$.
We outline the CDMPC procedure in Algorithm 1. Each agent computes optimal inputs for itself and predicts the optimal inputs of its neighbors. Due to the limited local system knowledge of each agent, the predictions can be inconsistent between agents.

## 3 The SCDMPC Algorithm

We propose to synchronize the states globally to guarantee prediction consistency. Our approach is inspired by multi-agent consensus [19] and synchronization [20]. To restore consistency, each agent $i$ synchronizes the states it predicted

**Algorithm 1:** CDMPC algorithm for agent $i$

1: **Input:** reference trajectories $\boldsymbol{r}_j$, $\forall j \in \mathcal{V}_i$
2: **Output:** control inputs $\boldsymbol{u}_{i\to j}$ and predicted states $\boldsymbol{x}_{i\to j}$
3: Send and receive states to and from neighboring agents
4: Solve CDMPC problem (1) for $\boldsymbol{u}_{i\to j}$ and $\boldsymbol{x}_{i\to j}$

for its neighbors. Agent $i$ considers the states agent $j$ computed for itself, as well as the states the agents $q$ (that are connected to both agent $i$ and agent $j$) predicted for agent $j$. Specifically agent $i$ computes the weighted average

$$\bar{\boldsymbol{x}}_{i\to j} = \sum_{q\in\mathcal{V}_i\cap\mathcal{V}_j} \frac{1}{w_{q\to j}} \boldsymbol{x}_{q\to j}. \tag{2}$$

Similar to the agent-level planning, agent $i$ only considers a subset of the predictions of other agents that exist for agent $j$. Therefore, prediction consistency may not be satisfied after the first synchronization step. Consequently, we designed the synchronization as an iterative process. At every synchronization step, each agent synchronizes the states and communicates them to its neighbors. The size of these vectors scales linearly with the prediction horizon $N_\mathrm{p}$ and the dimension of the state space, affecting the communication overhead as the number of agents or the complexity of the problem increases. However, as each agent only communicates with its neighbors, the overall communication remains distributed and scalable compared to centralized approaches. Then, each agent checks if the synchronized predictions from its neighbors are consistent with its own predictions. If the predictions are consistent, the synchronization procedure terminates, if not we repeat the procedure until we converged to a consistent solution. Theorem 1 states that the synchronization is guaranteed to converge and consequently terminate if the coupling sub-graph contains a spanning tree, i.e., if at least one agent has a path to all other agents.

**Theorem 1.** *The synchronization converges to a solution if and only if each coupling sub-graph $\mathcal{G}_i$ contains a spanning tree.*

For the proof of Theorem 1, see [1]. We outline the system-level synchronization method in Algorithm 2. Our complete Synchronization-Based Cooperative Distributed Model Predictive Control (SCDMPC) scheme is defined in Algorithm 3. The algorithm loops until the resulting solutions are locally feasible and prediction consistent. Within this loop, each agent first computes a solution for itself and its neighbors using the CDMPC procedure given in Algorithm 1. Then

**Algorithm 2:** Synchronization algorithm for agent $i$

**Input:** inconsistent states $\boldsymbol{x}_{q\to j}, \forall j \in \mathcal{V}_i, q \in \mathcal{V}_i \cap \mathcal{V}_j$
**Output:** consistent states $\bar{\boldsymbol{x}}_j$
Initialize synchronized states $\bar{\boldsymbol{x}}_{q\to j} = \boldsymbol{x}_{q\to j}$
**while** predictions not consistent **do**
    Send states $\bar{\boldsymbol{x}}_{i\to j}$ to agents $j$
    Receive states $\bar{\boldsymbol{x}}_{q\to j}$ from agents $q$
    **for all** $j \in \mathcal{V}_i$ **do**
        $\bar{\boldsymbol{x}}_{i\to j} = \sum_q \frac{1}{w_{q\to j}} \bar{\boldsymbol{x}}_{q\to j}$
    **end for**
**end while**
$\bar{\boldsymbol{x}}_j = \bar{\boldsymbol{x}}_{i\to j}$

the corresponding predictions are synchronized using the synchronization procedure given in Algorithm 2. If the resulting predictions are feasible, the algorithm terminates; if not, the process is repeated using the synchronized predictions as the reference in the next iteration of the loop. If the solution space of the optimization problem is convex, SCDMPC terminates after the first iteration. For a non-convex solution space we state that there exists a coupling topology such that SCDMPC is guaranteed to converge to a feasible solution, if a feasible solution to the corresponding centralized optimization problem exists. For a proof see [1]. Therefore, the convergence and consequently the termination of the algorithm depends on the coupling topology and the corresponding coupling weights.

**Algorithm 3:** Synchronization-Based Cooperative Distributed Model Predictive Control for agent $i$

**Input:** reference trajectories $\boldsymbol{r}_j, \forall j \in \mathcal{V}_i$, indices $j_\mathrm{s} \in \mathcal{V}_i, q \in \mathcal{V}_i \cap \mathcal{V}_j$
**Output:** control inputs $\boldsymbol{u}_{i\to j}$ and predicted states $\boldsymbol{x}_{i\to j}$
**while** states are not feasible and prediction consistent **do**
  $(\boldsymbol{u}_{i\to j}, \boldsymbol{x}_{i\to j}) \leftarrow \mathrm{CDMPC}(\boldsymbol{r}_j)$ (Alg. 1)
  Receive predictions $\boldsymbol{x}_{q\to j}$
  **if** predictions inconsistent **then**
    $\boldsymbol{x}_j \leftarrow \mathrm{Synchronization}(\boldsymbol{x}_{q\to j})$ (Alg. 2)
  **end if**
  $\boldsymbol{r}_j \leftarrow \boldsymbol{x}_j$
**end while**

## 4 Evaluation

We evaluate the SCDMPC algorithm in our CPM Lab [21], an open-source, remotely accessible small-scale test-bed using connected and automated vehicles (CAVs) in 1:18 scale. Each vehicle executes the algorithm on a designated computation node, a real-time Ubuntu 18.04 with two 1.6 GHz cores and 16 GB of RAM. Each vehicle communicates with its computation node via WiFi. The computation nodes share the vehicle's information via Ethernet. The algorithm was implemented in MATLAB R2020a. To solve optimization problem (1), we use the optimization toolbox of IBM CPLEX 12.10. For the costs we use the weighted $\ell_2$-norm while omitting the coupling objective. The kinematic bicycle model predicts system dynamics, while coupling constraints enforce safety distances between agents.

We test the algorithm in a formation building scenario. A visualization of the scenario is given in Figure 2. Each vehicle is placed at a random location on the 4 m x 4 m driving area of the test-bed. The driving area itself does not contain obstacles and is therefore convex. The CAVs aim to form a predefined formation, in our case standing next to each other at the top of the driving area. The controller of each vehicle computes the optimal trajectory to arrive at the goal pose as fast as possible while avoiding collisions. We generate a reference

path using Dubins Path algorithm and compute a respective reference trajectory by sampling points on the reference path based on the speed of the vehicle. For more information we refer to [2]. We run the scenario multiple times with random start poses and fixed goal poses using CMPC and SCDMPC and compare the results. The average cumulative path and speed deviations shown in Figure 3 indicate that the deviations are similar for SCDMPC and CMPC. In some instances SCDMPC achieves smaller deviations than CMPC. In those cases SCDMPC had a higher cost for the control input variations or the coupling objective than CMPC, which are not considered here. Furthermore, it can be observed that the average cumulative path deviations increase with the number of CAVs. In Figure 4, the maximum computation time of SCDMPC and CMPC is shown for different numbers of CAVs. It is evident that, for both the maximum computation time increases with the number of CAVs. However, the SCDMPC approach proves to be more scalable as the maximum computation time increases at a slower rate. Note, that the computation time of Algorithm 3 depends on the coupling topology of the agent. The number of neighbors of an agent increases the number of optimization variables and constraints and consequently the computation time needed to solve the optimization problem.

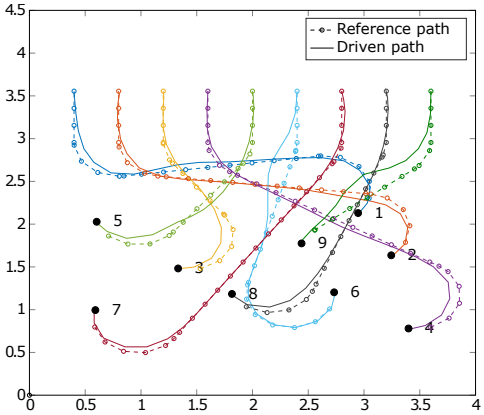
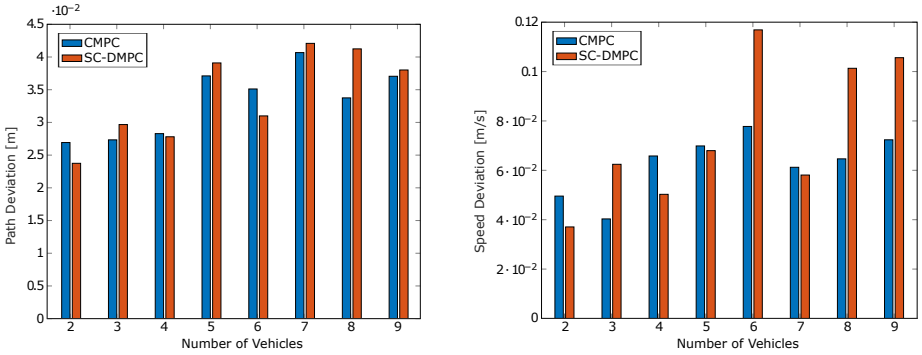



Fig. 2: Visualization of the formation building scenario.




(a) Path deviation.

(b) Speed deviation.

Fig. 3: The average cumulative path and speed deviations of CMPC and SCDMPC for different numbers of CAVs.

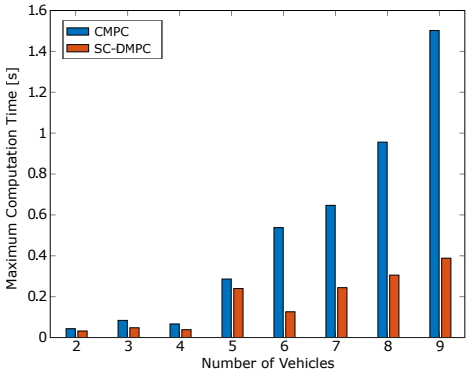



Fig. 4: The maximum computation time of CMPC and SCDMPC for different numbers of CAVs.

## 5 Conclusion

This work presented a Synchronization-Based Cooperative Distributed Model Predictive Control approach. We show that through synchronization, prediction consistency can be guaranteed. In our experiments in the CPM Lab we demonstrate the applicability of the SCDMPC algorithm for planning trajectories of CAVs. The results make it evident that the SCDMPC achieves control performance close to that of CMPC while showcasing better scalability in terms of computation time. In future research, we will investigate the effect of communication delays as well as learning-based approaches to enhance the performance and safety of our approach.

## References

1. Kloock, M., Alrifaee, B.: Coordinated Cooperative Distributed Decision-Making Using Synchronization of Local Plans. In: IEEE Transactions on Intelligent Vehicles, vol. 8, no. 2, pp. 1292-1306 (2023).
2. Kloock, M., Alrifaee, B.: Cooperative Pose Control of Non-Holonomic Vehicles Using Synchronization. In: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), pp. 93-99 (2023).
3. Trodden, P., Richards, A.: Cooperative distributed MPC of linear systems with coupled constraints. In: Automatica, vol. 49, Issue 2, pp. 479-487 (2013).
4. Müller, M. A, Reble, M., Allgöwer, F.: Cooperative control of dynamically decoupled systems via distributed model predictive control. In: International Journal of Robust and Nonlinear Control, 22, pp. 1376-1397 (2012).
5. Blasi, S., Kögel, M., Findeisen, R.: Distributed Model Predictive Control Using Cooperative Contract Options. In: IFAC-PapersOnLine, vol. 51, Issue 20, pp. 448-454 (2018).
6. Trodden, P., Richards, A.: Adaptive cooperation in robust distributed model predictive control. In: 2009 IEEE Control Applications, (CCA) & Intelligent Control, (ISIC), pp. 896-901 (2009).

7. Maestre, J.M., Muñoz de la Peña, D., Camacho, E.F., Alamo, T.: Distributed model predictive control based on agent negotiation. In: Journal of Process Control, vol. 21, Issue 5, pp. 685-697 (2011).
8. Sahebjamnia, N., Tavakkoli-Moghaddam, R., Ghorbani, N.: Designing a fuzzy Q-learning multi-agent quality control system for a continuous chemical production line – A case study. In: Computers & Industrial Engineering, vol. 93, pp. 215-226 (2016).
9. Francisco, M., Mezquita, Y.,Revollar, S.,Vega, P., De Paz, J. F.: Multi-agent distributed model predictive control with fuzzy negotiation. In: Expert Systems with Applications, vol. 129, pp. 68-83 (2019).
10. Morales-Rodelo, K., Vega, P., Francisco M., Revollar, S.: Influence of fuzzy layer in distributed control structure applied to four coupled tanks. In: 2019 IEEE 4th Colombian Conference on Automatic Control (CCAC), pp. 1-6 (2019).
11. Maxim, A., Maestre, J. M., Caruntu, C. F., Lazar, C.: Min-Max Coalitional Model Predictive Control Algorithm. In: 2019 22nd International Conference on Control Systems and Computer Science (CSCS), pp. 24-29 (2019).
12. Fele, F., Debada, E., Maestre, J. M., Camacho, E. F.: Coalitional Control for Self-Organizing Agents. In: IEEE Transactions on Automatic Control, vol. 63, no. 9, pp. 2883-2897 (2018).
13. Chanfreut, P., Maestre, J. M., Muros, F. J., Camacho, E. F.: A Coalitional Control Scheme with Topology-Switchings Convexity Guarantees. In: 2019 IEEE 58th Conference on Decision and Control (CDC), pp. 1096-1101 (2019)
14. Falsone, A., Margellos, K., Garatti, S., Prandini, M.: Dual decomposition for multi-agent distributed optimization with coupling constraints. In: Automatica, vol. 84, pp. 149-158 (2017).
15. Hashempour, S., Suratgar, A. A., Afshar, A.: Distributed Nonconvex Optimization for Energy Efficiency in Mobile Ad Hoc Networks. In: IEEE Systems Journal, vol. 15, no. 4, pp. 5683-5693 (2021).
16. Segovia, P., Puig, V., Duviella, E., Etienne, L.: Distributed model predictive control using optimality condition decomposition and community detection. In: Journal of Process Control, vol. 99, pp. 54-68 (2021).
17. Hours, J. -H., Jones, C. N.: A Parametric Nonconvex Decomposition Algorithm for Real-Time and Distributed NMPC. In: IEEE Transactions on Automatic Control, vol. 61, no. 2, pp. 287-302 (2016).
18. Braun, P., Faulwasser, T., Grüne, L., Kellett, C. M., Weller, S. R., Worthmann, K.: Hierarchical distributed ADMM for predictive control with applications in power networks. In: IFAC Journal of Systems and Control, Volume 3, Pages 10-22 (2018)
19. Ren, W., Beard, R. W.: Consensus Seeking in Multiagent Systems Under Dynamically Changing Interaction Topologies. In: IEEE Transactions on Automatic Control, vol. 50, no. 5, pp. 655-661 (2005).
20. Lunze, J.: Networked Control of Multi-agent Systems. Bookmundo, Rotterdam (2019).
21. Kloock, M., Scheffe, P., Maczijewski, J., Kampmann, A., Mokhtarian, A., Kowalewski, S., Alrifaee, B.: Cyber-Physical Mobility Lab: An Open-Source Platform for Networked and Autonomous Vehicles. In: 2021 European Control Conference (ECC), pp. 1937-1944 (2021).