Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 56-2 (2023) 2127–2133

Distributed Optimal Control Framework for High-Speed Convoys: Theory and Hardware Results

Namya Bagree * **Charles Noren** ** **Damanpreet Singh** *
Matthew Travers ** **Bhaskar Vundurthy** **

* *Department of Mechanical Engineering, Carnegie Mellon University, USA (e-mail: {nbagree, damanprs}@andrew.cmu.edu).*
** *The Robotics Institute, Carnegie Mellon University, USA (e-mail: {cnoren, mtravers, pvundurt}@andrew.cmu.edu).*

Abstract: Practical deployments of coordinated fleets of mobile robots in different environments have revealed the benefits of maintaining small distances between robots, especially as they move at higher speeds. However, this is counter-intuitive in that as speed increases, reducing the amount of space between robots also reduces the time available to the robots to respond to sudden motion variations in surrounding robots. However, in certain examples, the benefits in performance due to traveling at closer distances can outweigh the potential instability issues, for instance, autonomous trucks on highways that optimize energy by vehicle "drafting" or smaller robots in cluttered environments that need to maintain close, line of sight communication, etc. To achieve this kind of closely coordinated fleet behavior, this work introduces a model predictive optimal control framework that directly takes non-linear dynamics of the vehicles in the fleet into account while planning motions for each robot. The robots are able to follow each other closely at high speeds by proactively making predictions and reactively biasing their responses based on state information from the adjacent robots. This control framework is naturally decentralized and, as such, is able to apply to an arbitrary number of robots without any additional computational burden. We show that our approach is able to achieve lower inter-robot distances at higher speeds compared to existing controllers. We demonstrate the success of our approach through simulated and hardware results on mobile ground robots.

Keywords: Mobile robots, Autonomous robotic systems, Decentralized control, Nonlinear predictive control, Field robotics, Convoy, Platooning, Multi-agent systems

1. INTRODUCTION

Fleets of mobile robots have shown practical benefits by moving closer to each other at high speeds to optimize system resource utilization and improve process efficiencies. An example of this is a group of trucks that reduce fuel consumption through the reduction of aerodynamic drag by following at lower inter-vehicle distances (Nahavandi et al., 2022; Turri et al., 2017). However, reducing inter-vehicle distances reduces the amount of time available to respond to sudden unexpected behaviors or motions of the other robots in the fleet. This can lead to instabilities in the behavior of the overall system. To address these problems, our work investigates non-linear control techniques for autonomous robots that can maintain low inter-robot distance over various environments at high speeds.

While the advantages of a fleet of robots exist in many domains (Nahavandi et al., 2022), small mobile robots can also see performance improvements by driving fast at close distances in unstructured regions. A practical challenge in search and rescue missions is maintaining consistent communication between the robots, especially in cluttered environments where the loss of direct line of sight can hinder inter-robot communications (Scherer et al., 2022). This is another example of a scenario where actively maintaining low inter-robot separation distances is essential to the performance of the team. To this end, in this work, we deploy such a small-scale system in both simulations as well as hardware.

Fig. 1. Robot convoy performing a search and rescue task

The primary focus of this work is to present a non-linear model based controller that addresses many issues with the current state-of-the-art methods for controlling fleets of mobile robots operating in cluttered environments. Our controller takes information from all of the adjacent

10.1016/j.ifacol.2023.10.1116

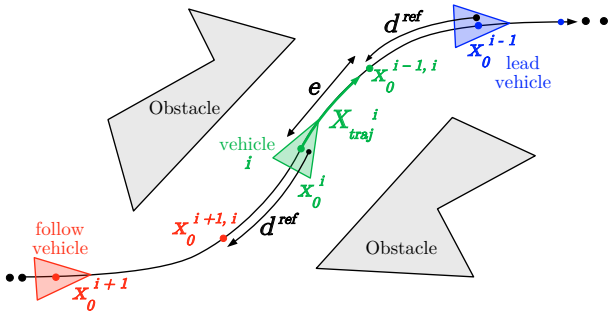


Fig. 2. Schematic for the convoy control problem. Lead and follow vehicles are defined with respect to vehicle i. Error e is the Euclidean distance between a vehicle's desired position (d^{ref} behind the lead vehicle) and its current location.

robots and computes both the feedforward as well as the feedback components that help it proactively plan as well as reactively adjust behavior to compensate for any unexpected behaviors across the fleet.

In addition, we designed the control framework to operate in a decentralized fashion. This allows us to extend our framework to an arbitrary number of robots with a constant computational requirement for each robot. We demonstrate the overall efficacy of our approach via experiments in simulation as well as hardware, for a fleet of wheeled ground vehicles.

2. PROBLEM DEFINITION

Consider the schematic illustrated in Fig. 2 for vehicle i. We assume an increasing index along the trajectory and thus refer to the neighboring vehicles as the lead vehicle and the follow vehicle denoted by $i-1$ and $i+1$, respectively. Given L agents in the fleet, the state and control trajectories for the agent i over a time horizon of length N are given by $X^i = [x_0^i, x_1^i, ..., x_N^i]$ and $U^i = [u_0^i, u_1^i, ..., u_{N-1}^i]$, respectively, where $i \in \mathcal{I}_L = \{1, 2, \cdots, L\}$.

The objective of the convoy problem as we define it in this work is to design a distributed convoy controller that generates a control output u_0^i at every time instant to get a fleet of robots to follow a predefined trajectory. In particular, the desired objectives for each vehicle are to

(1) Maintain a fixed distance d^{ref} along the desired path from its lead vehicle;
(2) Simultaneously travel at desired speeds;
(3) React to disturbances without creating instabilities to the overall behavior of the convoy.

3. LITERATURE REVIEW

To provide context for our control approach, we review relevant prior works that address the convoy control problem defined in Section 2. Despite several use cases for convoys (Nahavandi et al., 2022), we focus our discussion on works for unstructured and cluttered environments.

One of the earliest works (Yazbeck et al., 2014) in convoy control mimics a leader-follower behavior where each vehicle estimates and stores the path of its predecessor as a set of points. The follower then estimates the predecessor's path curvature around a selected target and follows the trajectory. (Nestlinger et al., 2022) extend this work to store position measurements over time and apply a spline-approximation technique to obtain a smooth reference path for the underlying motion controllers. However, these methods force the robot to track the exact positions of its predecessors, restricting system flexibility around obstacles.

Albrecht et al., 2020 provides a framework to switch between exact pose tracking and flexible path search and tracking based on the environment. This allows the trajectory following convoy algorithms to operate in real-world conditions. However, there is no interaction between the outputs of the two tracking methods, reducing the controller performance due to conflicting switch decisions when operating in a cluttered environment.

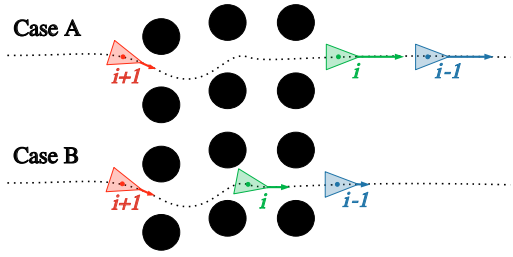
In contrast to (Yazbeck et al., 2014; Nestlinger et al., 2022; Albrecht et al., 2020), there exist prior works of convoy control wherein they specifically provide an integrated obstacle avoidance module to operate a fleet of robots in unstructured environments. For instance, in (Zhao et al., 2017), the authors define a robust solution using an adaptive inter-robot distance control and leader pose (x_0^{i-1}) estimation. However, there are no common velocity-based control parameters and no prediction on future paths for adjacent agents (X^{i-1}, X^{i+1}). This leads to higher angular variation as inter-robot distances reduce, resulting in wavy motions and higher error in tracking at high speeds.

An additional prior work that incorporates specific obstacle avoidance in the control framework is presented by (Shin et al., 2020). Here, the authors approach the convoy problem with a controller that incorporates a passivity-based MPC method that explicitly integrates a traversability map in the planner. However, this framework doesn't take feedback from the following vehicles ($i+1$), which can lead to high inter-robot distances on cluttered terrains. They also rely on continuous communication between the vehicles and a base node to operate.

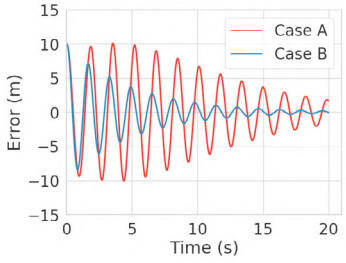
One other work that doesn't apply the "follow-the-leader" framework as discussed earlier, is presented by (Turri et al., 2017). The authors in this work take a centralized approach to designing their controller, focusing primarily on straight-line velocity profiles. The objective of the optimization in their work is to improve fuel efficiency while ensuring desired safe following distances between vehicles. The computational cost increases quadratically as you add new agents and the overall approach needs to be solved on a single computer, creating a single point of failure. This also requires high-rate continuous communications between the robots and the base node for safe convoying.

4. CONVOY BEHAVIOR

Our convoy system is designed to include the best aspects of an explicit "follow-the-leader" behavior and a completely centralized controller. We display a scenario in Fig. 3a to better understand the comparison of a "follow-the-leader" (Case A) behavior against ours (Case B). Agent $i+1$ has been forced to slow down while navigating a cluttered environment. A spring-damper-mass system



(a) Environmental situation where robot $i+1$ is stuck



(b) Fleet response demonstrating the accordion problem

Fig. 3. Comparison of convoys when following robots are neglected in the framework (Case A) vs when they are not (Case B).

was defined for three agents to simulate both behaviors. Each agent is only affected by its lead agent in Case A and both its adjacent agents in Case B. Fig. 3b shows that Case B stabilizes faster and reduces accordion-like effects between the agents. This also ensures lower inter-robot distances through the environment, assisting real-time communication between the agents.

The high-level controller behaviors listed in Section 2 can be expressed in an optimal control framework. The optimal control framework presented in this work expresses the behaviors described in (1) and (2) as terms in the controller objective. The proposed controller's cost structure encodes agent reactivity to disturbances described in behavior (3). We now describe the proposed controller in detail.

4.1 Convoy Controller

For the ith robot in the convoy, the discrete-time optimal control problem framework is posed as:

$$\begin{aligned} \underset{\forall i \in \mathcal{I}_L}{\text{minimize}} \quad & C_{traj}(X^i, U^i) + C_{convoy}(X^i, X^{i-1}, X^{i+1}) \\ \text{subject to} \quad & x^i_{k+1} = f(x^i_k, u^i_k, \Delta t), \forall k = 0, ..., N-1, \\ & x^i_0 = x^i(0), u^i_0 = u^i(0), \forall i \in I_L, \end{aligned} \tag{1}$$

where:

$$\begin{aligned} C_{traj}(X^i, U^i) = & \sum_{k=0}^{N-1} (x^i_k - x^i_{traj,k})^T Q (x^i_k - x^i_{traj,k}) \\ & + u^{(i)T}_k R u^i_k + (x^i_N - x^i_{traj,N})^T Q_f (x^i_N - x^i_{traj,N}), \end{aligned} \tag{2}$$

$$\begin{aligned} C_{convoy}(X^{i-1}, X^i, X^{i+1}) = & \\ & \sum_{k=0}^{N-1} (x^i_k - x^{ref,i-1,i}_k)^T Q_{lead} (x^i_k - x^{ref,i-1,i}_k) \\ & + (x^i_k - x^{ref,i,i+1}_k)^T Q_{follow} (x^i_k - x^{ref,i,i+1}_k), \end{aligned} \tag{3}$$

and where: $x_k \in \mathbb{R}^m$, $u_k \in \mathbb{R}^n$. Additionally, Q, Q_f, Q_{lead}, $Q_{follow} \in \mathbb{R}^{m \times m}$ are symmetric positive definite matrices, and $R \in \mathbb{R}^{n \times n}$ is a positive definite matrix. The state-transition model for the ith robot is defined as $f(x^i_k, u^i_k, \Delta t)$. Superscript ref refers to the reference point defined between the agents with the indices following ref (*i.e.* either $i-1, i$ or $i, i+1$). The computation of this point is described in the following section. As the run-time cost is evaluated over the same time interval, the run-time costs in (2) and (3) may be collapsed into a single quadratic cost expression. This new expression is defined as

$$\begin{aligned} g(x^i_k, u^i_k) = \; & (x^i_k - Q_T^{-1} y_T) Q_T (x^i_k - Q_T^{-1} y_T) \\ & - y_T^T Q_T y_T + Z_T + u^{(i)T}_k R u^i_k, \end{aligned} \tag{4}$$

where:

$$\begin{aligned} Q_T &= Q + Q_{lead} + Q_{follow} \\ y_T &= Q x^i_{traj,k} + Q_{lead} x^{ref,i-1,i}_k + Q_{follow} x^{ref,i,i+1}_k \\ Z_T &= (x^i_{traj,k})^T Q x^i_{traj,k} + (x^{ref,i-1,i}_k)^T Q_{lead} (x^{ref,i-1,i}_k) \\ &\quad + (x^{ref,i,i+1}_k)^T Q_{follow} (x^{ref,i,i+1}_k). \end{aligned}$$

A combination of quadratic expressions may also be applied to the terminal cost, yielding (Q_F, y_F, Z_F), respectively. These parameters may be used to rephrase the terminal cost:

$$\phi(X_N) = (x^i_N - Q_F^{-1} y_F) Q_F (x^i_N - Q_F^{-1} y_F) - y_F^T Q_F y_F + Z_F.$$

Thus the cost function for robot i may be rephrased as:

$$J = C_{traj} + C_{convoy} = \sum_{k=0}^{N-1} \{g(x^i_k, u^i_k)\} + \phi(X_N).$$

Furthermore, by linearizing the system around x_k, u_k and defining $A_k = \frac{\partial}{\partial x_k} f(x_k, u_k)$ and $B_k = \frac{\partial}{\partial u_k}$, the optimization problem can be interpreted and solved online as an iterative Linear Quadratic Regulator (Li and Todorov, 2004). This yields a control law:

$$\begin{aligned} u^i_k &= -\left[R_k + B_k^T P_{k+1} B_k\right]^{-1} B_k^T P_{k+1} A_k (x^i_k - Q_T^{-1} y_T) \\ &= -K_k (x^i_k - Q_T^{-1} y_T), \end{aligned}$$

with P_k representing the solution to the Riccati Equation and K_k being the optimal control gain matrix. Additional details and proofs can be found in the appendix of the version uploaded to `https://arxiv.org/abs/2211.06287`.

4.2 Controller Implementation and Design Discussion

The first component of the cost function is a quadratic trajectory tracking cost penalizing deviations from a given convoy trajectory: x_{traj}. This cost is defined in (2). In this work, the reference path x_{traj} is provided to the controller as either a pre-defined path or created for each individual robot from observing the motion of other agents in the convoy *e.g.,* "follow-the-leader" style approaches (Nestlinger et al., 2022; Yazbeck et al., 2014).

The convoy cost C_{convoy} penalizes deviance from the convoy structure over the future time horizon. This cost is defined in (3) where $x^{ref,i-1,i}_k, x^{ref,i,i+1}_k$ are the reference positions of the i^{th} given the positions of the $i-1$ and $i+1$ cars, respectively, and Q_{lead}, Q_{follow} are tunable positive semi-definite constant matrices. The state vector includes the x and y coordinates, vehicle orientation ψ and velocity v. The control vector includes acceleration a and steering angle δ.

The computation of x^{ref} is based on the desired inter-robot distance, d^{ref}. This desired inter-robot distance is defined as

$$d^{ref} = \lambda_1 v_t + \lambda_2 (v_i - v_{i-1}) + K,$$

where v_t is the desired target velocity and v_i corresponds to current velocity for agent i. λ_1 and λ_2 are tunable parameters where λ_1 and λ_2 are non-negative values. K is a constant minimum inter-robot distance for safe operation. Only the current velocities of vehicle i and the prior robot in the convoy structure are considered for this work.

As shown in Fig. 2, the reference positions for agents $i-1$ and $i+1$ over horizon $1:N$ are recovered by performing an open-loop forward rollout using the linearized dynamics at the $i-1$ or $i+1$ agent's state. The $i-1$ and $i+1$ agents' current velocity and steering are assumed to be constant over the rollout. The reference positions for (3) are set by moving backward d^{ref} along the convoy trajectory from the predicted $i-1$ agent positions X^{i-1} and moving the same distance ahead of the $i+1$ agent positions X^{i+1}.

The reactivity described in (3) of 2 aims to prevent collisions between agents due to sudden variations in speed. To enable this behavior, the controller computes a weighting factor, w_{convoy}, between the costs (2) and (3) during run-time. This factor is based on the desired convoy spacing and the current Euclidean distance $dist_{i,j}$ between agents i and j. These weights are multiplied to the Q matrices in (3) and the weighting factor is computed as:

$$Q_{lead} = w_{i,i-1} * Q_{lead}$$
$$Q_{follow} = w_{i,i+1} * Q_{follow}$$
$$w_{i,j} = \begin{cases} 1 + \dfrac{w_{far} \times (dist_{i,j} - d^{ref})}{dist_{i,j}} & \text{if } dist_{i,j} \geq d^{ref} \\ 1 + \dfrac{w_{near} \times (d^{ref} - dist_{i,j})}{dist_{i,j}} & \text{otherwise.} \end{cases}$$

The proposed formulation allows the robot to track its predecessor and follower through the predicted x^{ref} terms while also tracking its desired planned path. The proposed additional convoy cost provided in the optimal control formulation may be interpreted as a modification of the local linearization point used in the LQR. This modification of the coordinate transfer from the reference path (x_{traj}) through both the user-defined weightings (Q, Q_{lead}, Q_{follow}) and reference trajectories of the leading (X^{lead}) and following (X^{follow}) robots in the fleet.

4.3 Local Planner and Trajectory Controller

In the absence of an obstacle free path, Algorithm 1 ensures high-speed following and maintains the convoy formation with the help of an additional velocity scaling term:

$$v_T = (1 + \alpha) v_{tc}$$
$$\alpha = \lambda_3 (d_1 - d^{ref}) - \lambda_4 d_2$$

Where λ's are tunable parameters, v_{tc} is the desired target velocity from the convoy controller, v_T is the modified desired target velocity for the planner, v_r is the robot's velocity, v_l is the leader's velocity, d_1 is the distance between the robot and the leader along the trajectory and d_2 is the distance between the robot and the follower along the trajectory.

Algorithm 1 Convoy controller with obstacle avoidance for robot i

Input: Robot states x_0^{i-1}, x_0^i, x_0^{i+1}
Output: Control sequence U^i

```
while Robots are in convoy, i ∈ I_L do
    Run convoy controller, Section 4.1
    if Output path X^i is obstacle-free then
        return Control sequence U^i
    else
        Define D_LookAhead based on velocity v_t
        Calculate desired velocity v_T and direction θ_T
        Run the local planner Zhang et al., 2020
        Get the modified obstacle-free path X^i
        Send X^i into a path following iLQR controller
        return Control sequence U^i
    end if
end while
```

The desired direction is selected based on the output of the convoy controller. This is done to track the desired controller path to the greatest extent before the robot switches back into the convoy controller mode. The direction θ_T is defined via the look ahead distance $D_{LookAhead}$ along the optimal trajectory generated by the convoy controller.

The local planner takes these inputs and generates a feasible path that avoids obstacles while tracking outputs from the convoy controller. This work is based on the planner defined in (Zhang et al., 2020). The selected path is then sent to an iLQR trajectory following controller to generate the control sequence.

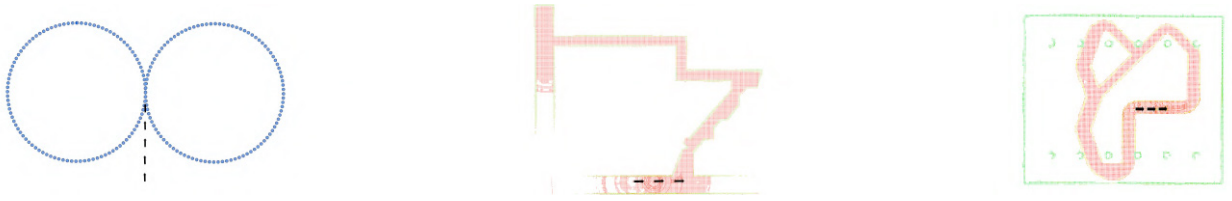
5. SIMULATION RESULTS

The performance of the presented controller ("Convoy Controller") is characterized in a variety of simulated environments shown in Fig. 4. The underlying simulator is Gazebo (Koenig and Howard, 2004). As discussed in Section 3, there are many different approaches to enforcing convoy structure at the control and planning level. To develop a comparison between the presented methodology and existing literature, a baseline "Base Controller" is developed by combining the local planning and distance variation behaviors from (Zhao et al., 2017) with additional modifications from (Bayuwindra et al., 2020; Albrecht et al., 2020; Mohamed-Ahmed et al., 2019). This combination creates a decentralized controller, similar to our proposed method, and outperforms the individual performance of each work separately with respect to minimizing inter-agent distances without breaking convoy formation.

Two error metrics are used to compare the results between both controller performances. The first metric is as follows:

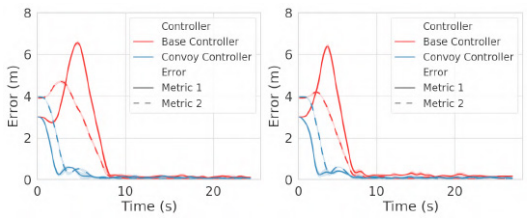
$$e_{m_1} = \begin{cases} abs((d_{i-1,i+1}/2) - d_{i-1,i}) & i \in (1, L) \\ abs((d_{L-2,L}/2) - d_{L-1,L}) & i = L \end{cases}$$

This metric aims to understand how well a robot is able to maintain a position midway between its adjacent robots. Variation in this metric can help understand accordion-like behaviors that negatively affect fleet performance with continuous acceleration and braking requirements. However, this error metric can maintain a low value with all robots maintaining an equally large following distance,



(a) High speed ∞ loop (b) Tunnel environment (c) Race track

Fig. 4. Illustrations of a few simulation environments



(a) Straight Line (b) Low Curvature

Fig. 5. Simulation results

which is not desirable. To that end, we define a second error metric as follows:

$$e_{m_2} = \text{abs}(d_{i-1,i} - d_{desired})$$

This metric measures distance from the desired gap between robots and, along with the first metric, can provide a thorough understanding of the convoy performance.

An initial comparison is done on a straight line and low curvature turn path. Next, an ∞ loop with a 20m radius is simulated at speeds varying between 4 and 8 m/s in order to test controller performance in continuous turns. Finally, runs are performed on tight turns, a tunnel environment and a racetrack. Additional constraints on agent operability and speed are added on these runs to understand fleet performance in edge scenarios.

5.1 Straight line and low curvature runs

The first set of simulations was performed on a group of robots operating on simple paths while starting from scattered initial locations. The aim of this experiment is to understand how quickly various controllers settle and how the error values vary during that period. As can be seen in Fig. 5, our convoy controller is able to settle faster while maintaining lower values on both error metrics. This displays faster convoy structure formation through our framework, as each robot has a cost associated with all of its adjacent agents.

5.2 High speed ∞ loop

We run robots in an ∞ loop at various speeds to understand system capability limits. Fig. 6 provides error metric graphs run at various target speeds between 4 - 8 m/s. It can be seen through the figures that at initial speeds, error metrics for both controllers settle, while our control achieves this faster. However, as speeds increase, the difference in performance between our convoy controller and the base controller increases until a point when the base controller breaks and isn't able to maintain a convoy structure.

5.3 Tight turns

A series of waypoints are sent to the lead robot, the other robots have no information about these points. On running the base controller, the following agents tend to overshoot due to the presence of sharp turns. This behavior is seen even if we tune down the look-ahead value to zero (Bayuwindra et al., 2020), depicting an inherent unwillingness to follow through with such sharp turns at high speeds. The error metric comparison between controllers can be seen in Fig. 7a. Our controller performs better where usual convoying techniques overshoot and has difficulty tracking such tight turns.

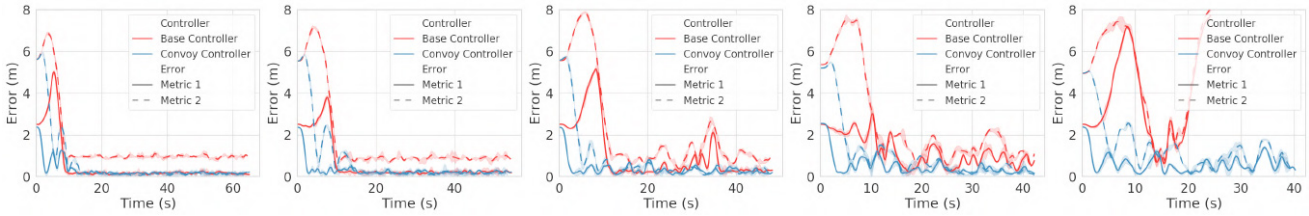
5.4 Tunnel environments

An agent might get stuck during operation, especially in indoor environments at high speeds. To simulate this, we initially stop the first robot in place for 8 seconds and then revert to normal conditions. We simulate this through narrow tunnels. In Fig. 7b, the error metrics are capped in our approach, whereas they continuously rise with the base controller as the last robot isn't able to get back into the convoy. Due to the cost term corresponding to the immediate lead and follower in the convoy controller cost definition, as soon as one of the agents isn't maintaining desired motion, the inter-robot gaps increase and the corresponding cost term shoots up. This causes the remaining agents to slow down and come to a stop until the slowed-down follower agent rejoins, capping the error metric.

5.5 Race track

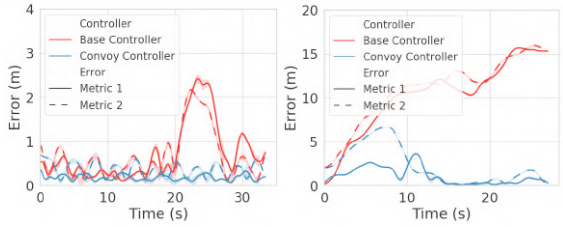
We simulate a constrained outdoor environment at 4 m/s. We also add an additional constraint that one of the robots isn't able to achieve speeds higher than 3 m/s, which would mimic a robot with operational issues or a heterogeneous set of robots. We run this setup on a race track, shown in Fig. 4c. The error metrics are seen in Fig. 7c. The initial performance between the controllers is similar until the point where the leader makes a turn after the straight section on the race track. On the straight section, the base controller notices a continuous drop in performance that it is unable to recover past a turn, and the slow robot can no longer track the lead along the race track. Our system, on the other hand, adapts to this robot and, despite losing a little performance, is able to maintain an upper limit on the error metric and continue on the desired path.

In summary, addressing all requirements from the problem definition in Section 2, our convoy controller achieves lower inter-robot distances, quickly adapts to variations in environments, and achieves faster convergence for error metrics when compared with the base controller.

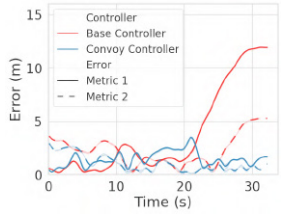


(a) 4 m/s (b) 5 m/s (c) 6 m/s (d) 7 m/s (e) 8 m/s

Fig. 6. Speed Variations in an ∞ loop



(a) Tight Turns (b) Tunnel



(c) Race Track

Fig. 7. Simulation results - edge cases

(a) Column room (b) Long corridor

Fig. 8. Test environments

6. HARDWARE IMPLEMENTATION

6.1 System Overview

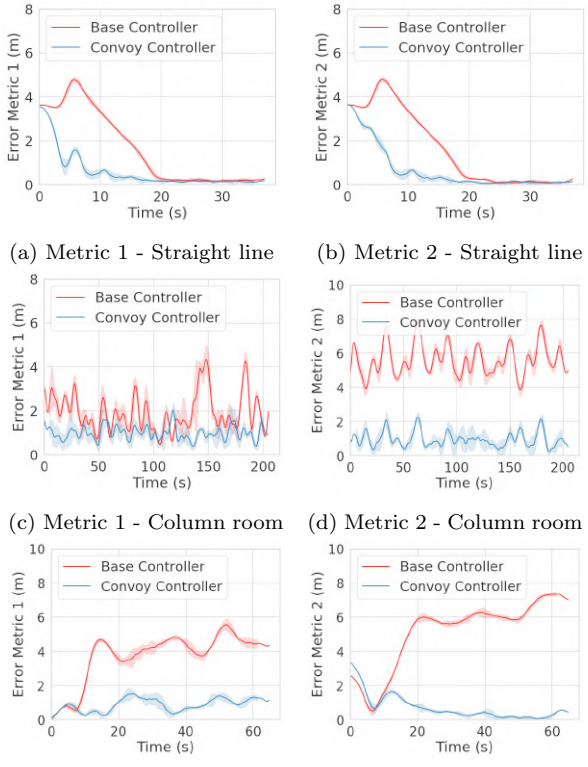
Each robot in our setup (Fig. 1) consists of Traxxas remote-controlled trucks fitted with a communication node, LiDAR and IMU sensors, a Jetson AGX Xavier and a motor controller.

For our perception framework, we make use of Super Odometry Zhao et al., 2021, which is an IMU-centric pipeline that provides estimates of each agent's odometry. To ensure neighboring robots don't create drifts in odometry in low LiDAR feature regions such as narrow corridors, the LiDAR data was filtered using LiDAR masks in the directions of the forward and rear robots. This ensures robust perception performance in all environments.

A base station is used to provide input commands to the convoy leader, either via a joystick or through a waypoint-sharing interface. All systems are run over ROS and use the DDS protocol for real-time inter-robot communication.

Environment	Average Error Metric 1 (m)		Average Error Metric 2 (m)	
	Base	Ours	Base	Ours
Straight Line	1.35	0.31	1.08	0.44
Sine Curve	1.12	0.30	0.80	0.47
Figure of 8	1.69	0.32	2.82	0.73
Tight Turn	0.72	0.18	0.65	0.37
Tunnel	10.20	1.08	10.94	2.24
Race Track	3.77	1.45	2.29	1.02
Column Room	1.91	0.97	5.61	0.95
Long Corridor	3.78	0.89	5.14	0.68

Table 1. Overall results comparison



(a) Metric 1 - Straight line (b) Metric 2 - Straight line

(c) Metric 1 - Column room (d) Metric 2 - Column room

(e) Metric 1 - Long corridor (f) Metric 2 - Long corridor

Fig. 9. Test results

6.2 Results

Straight Line: As discussed in Section 5.1, we test similar conditions on the three ground robots. It can be seen that the error metrics in Fig. 9a and 9b that the metrics vary similarly to those in Fig. 5a and 5b where our convoy controller is able to achieve steady state values faster than the existing base controller. This also re-validates simulation results on a real system.

Column room: Such an environment (Fig. 8a) is able to test convoy agility due to the continuous obstacle avoidance requirements via low-curvature maneuvers while maintaining a stable convoy structure during high-speed maneuvers. The room has dimensions of 35 m x 20 m. The operator commands a general direction of motion to the lead vehicle and all three robots make use of the convoy controller to follow closely while achieving speeds of 4 m/s. The robots make continuous turns to not hit walls, columns and other obstacles. The robots run in various directions around this room to cover a total distance of over 600 m. The error metrics between our and the base controller are shown in Fig. 9c and 9d. Despite having to continuously accelerate, decelerate and turn in such an environment, the convoy controller is able to maintain average gaps of just over 5 m while operating at 4 m/s compared to 9 m following distances through the base controller. We also run the same path with a single robot and notice an increase in average speed by under 5%, displaying the fact that our framework doesn't reduce overall robot capabilities while increasing agents.

Long corridors: In narrow long corridors with doorways (Fig. 8b), we illustrate high-speed convoy performance while adapting to turns and doorways. These environments feature scattered debris and uneven ground surfaces to validate the robustness of our control system in an attempt to mimic search and rescue operations. The performance difference can be seen in Fig. 9e and 9f. The average inter-robot distances are just under 4 m at high speeds, which is significantly lower than the lowest gaps of 9-15 m seen at comparable speeds in other research papers (Zhao et al., 2017; Shin et al., 2020) and the base controller. The deviations in the metrics only take place when the robots make a turn into another corridor, move over rough grounds, or navigate through doorways. The performance drop of a convoy against a single robot is less than 2%, indicating almost similar time to achieve the goal. The robots were able to navigate over 3 km in such environments without running into each other or leaving a robot behind, displaying a robust and safe system.

7. CONCLUSIONS

We have designed an optimal decentralized control system to run on a multi-robot convoy. Our design incorporates future predictions on the adjacent robots and solves a cost minimization incorporating robot states and controls, allowing the agents to operate closer to each other at high speeds. This, in turn, enables robots to operate well in environments that require continuous sharp turns and variations in speeds. The framework operates in a decentralized manner, resulting in no additional computational requirements while increasing the number of robots being deployed. We have been able to show, through simulation and hardware experimentation, the improvement in performance against the current state-of-the-art methods and have been able to cut down following distances against state-of-the-art by half. The system can potentially be improved in the future by incorporating environment-dependent control optimizations and integrating hybrid communication control methods with increased data transfer to reduce state prediction computational loads.

REFERENCES

Albrecht, A., Heide, N.F., Frese, C., and Zube, A. (2020). Generic convoying functionality for autonomous vehicles in unstructured outdoor environments. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, 1949–1955.

Bayuwindra, A., Ploeg, J., Lefeber, E., and Nijmeijer, H. (2020). Combined longitudinal and lateral control of car-like vehicle platooning with extended look-ahead. *IEEE Transactions on Control Systems Technology*, 28(3), 790–803.

Koenig, N. and Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, 2149–2154 vol.3.

Li, W. and Todorov, E. (2004). Iterative linear quadratic regulator design for nonlinear biological systems. In *International Conference on Informatics in Control, Automation and Robotics*, 222–229.

Mohamed-Ahmed, M.M., Naamane, A., and M'Sirdi, N.K. (2019). Path tracking for the convoy of autonomous vehicles based on a non-linear predictive control. In *The 12TH International Conference on Integrated Modeling and Analysis in Applied Control and Automation*. Lisbonne, Portugal.

Nahavandi, S., Mohamed, S., Hossain, I., Nahavandi, D., Salaken, S.M., Rokonuzzaman, M., Ayoub, R., and Smith, R. (2022). Autonomous convoying: A survey on current research and development. *IEEE Access*, 10, 13663–13683.

Nestlinger, G., Rumetshofer, J., and Solmaz, S. (2022). Leader-based trajectory following in unstructured environments;from concept to real-world implementation. *Electronics*, 11(12).

Scherer, S. et al. (2022). Resilient and modular subterranean exploration with a team of roving and flying robots. *Field Robotics Journal*, 678–734.

Shin, J., Kwak, D., and Kim, J. (2020). Autonomous platooning of multiple ground vehicles in rough terrain. *Journal of Field Robotics*, 38.

Turri, V., Besselink, B., and Johansson, K.H. (2017). Cooperative look-ahead control for fuel-efficient and safe heavy-duty vehicle platooning. *IEEE Transactions on Control Systems Technology*, 25(1), 12–28.

Yazbeck, J., Scheuer, A., and Charpillet, F. (2014). Decentralized near-to-near approach for vehicle platooning based on memorization and heuristic search. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 631–638.

Zhang, J., Hu, C., Chadha, R., and Singh, S. (2020). Falco: Fast likelihood-based collision avoidance with extension to human-guided navigation. *Journal of Field Robotics*, 37.

Zhao, S., Zhang, H., Wang, P., Nogueira, L., and Scherer, S. (2021). Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 8729–8736.

Zhao, X., Yao, W., Li, N., and Wang, Y. (2017). Design of leader's path following system for multi-vehicle autonomous convoy. In *2017 IEEE International Conference on Unmanned Systems (ICUS)*, 132–138.