Moving-Landmark Assisted Distributed Learning Based Decentralized Cooperative Localization (DL-DCL) with Fault Tolerance

Shubhankar Gupta, Suresh Sundaram
Artificial Intelligence and Robotics Lab (AIRL)
Department of Aerospace Engineering
Indian Institute of Science, Bengaluru, Karnataka, India
shubhankarg@iisc.ac.in, vssuresh@iisc.ac.in

Abstract

This paper considers the problem of cooperative localization of multiple robots under uncertainty, communicating over a partially connected, dynamic communication network and assisted by an agile landmark. Each robot owns an IMU and a relative pose sensing suite, which can get faulty due to system or environmental uncertainty and therefore exhibit large bias in their estimation output. For the robots to localize accurately under sensor failure and system or environmental uncertainty, a novel Distributed Learning based Decentralized Cooperative Localization (DL-DCL) algorithm is proposed that involves real-time learning of an information fusion strategy by each robot for combining pose estimates from its own sensors as well as from those of its neighboring robots and utilizing the moving landmark's pose information as a feedback to the learning process. Convergence analysis shows that the learning process converges exponentially under certain reasonable assumptions. Simulations involving sensor failures inducing around 40-60 times increase in the nominal bias show DL-DCL's estimation performance to be approximately 40% better than the well-known covariance-based estimate fusion methods. For the evaluation of DL-DCL's implementability and fault-tolerance capability in practice, a high-fidelity simulation is carried out in Gazebo with ROS2.

Introduction

In autonomous robotic applications involving mobile robots, localization is considered one of the most fundamental challenges (Huang and Dissanayake 1999); localization is the process of determining where a robot is located with respect to its environment, or in other words, determining an accurate estimate of its location and orientation with respect to a global frame of reference. Localization of a single robot is usually carried out using a filtering method, like Kalman Filter, for estimating the robot pose by fusing information from its proprioceptive sensors (e.g., IMU, INS) and exteroceptive sensors (e.g., RADAR, LiDAR, GPS, camera). In ideal situations, one can achieve success in autonomous robotic mission objectives by simply deploying a single robot. But in practice, system and environmental uncertainty play a critical role by increasing the chances of sensor failure, especially in applications involving adverse conditions like search and rescue (Scherer et al. 2015), disaster relief (Gregory et al. 2016), convoy protection (Spry, Girard, and Hedrick 2005), traffic monitoring (Khan et al. 2020), etc. Therefore, in such uncertain scenarios, having multiple robots carry out the mission increases the chances of success by increasing the reliability of the overall multi-robot system (MRS) (Mohiuddin et al. 2020).

A key challenge in the localization of a MRS is to determine an appropriate estimate fusion strategy each robot should employ so that they collaboratively share their sensor information and improve their pose estimates under system and environmental uncertainty. The literature on Decentralized Cooperative Localization (DCL) of a MRS mainly utilizes covariance-based methods like Kalman Fusion (KF) (Maybeck 1982), (Uhlmann 2003), Covariance Intersection (CI) (Matzka and Altendorfer 2009), (Julier and Uhlmann 2017), and Covariance Union (CU) (Matzka and Altendorfer 2009), (Reece and Roberts 2010), as information fusion strategy that robots use to improve their estimation accuracy. (Carrillo-Arce et al. 2013) proposes an approximate decentralized multi-robot cooperative localization algorithm with reduced processing and communication costs, using CI to maintain consistency while handling asynchronous communication constraints. (Assa and Janabi-Sharifi 2015) proposes a nonlinear KF-based sensor fusion framework that adaptively compensates for system noise variations and iteratively deals with the fast system dynamics. CI is explicitly used in the communication update of the multi-robot localization algorithm proposed in (Chang, Chen, and Mehta 2021) for ensuring estimation consistency and resilience. Similarly, (Pires et al. 2021) use CI in their cooperative localization and mapping algorithm for robotic swarms. In (Wang et al. 2021), a fully decentralized multi-robot cooperative localization algorithm based on CU is proposed, where CU handles spurious sensor data in the fusion process ensuring consistency of the fused estimates. The above-mentioned covariance-based fusion methods usually involve assumptions regarding consistency and correlation among the estimates being fused – KF requires the estimates to be uncorrelated, CI requires the estimates to be consistent, and CU requires at least one of the estimates to be consistent. Further, covariance-based methods require covariance information of the estimates being fused as input. In applications involving adverse conditions, temporary or permanent sensor

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

failures can lead to dynamic and/or large bias or drift in the estimates being fused. In such situations, covariance-based methods may not perform satisfactorily or may even fail. Thus, there is a need for DCL algorithms that do not involve any assumptions regarding consistency and correlation, do not require any covariance information of the estimates being fused, and can handle large dynamic bias or drift in the estimates. In this regard, this paper presents a novel Distributed Learning-based Decentralized Cooperative Localization (DL-DCL) framework in which robots, assisted by a moving landmark, utilize the DL-DCL algorithm that satisfies the above-mentioned requirements.

In the DL-DCL framework, robots collaborate over a dynamic communication network sharing sensor information, and get assistance from a stationary or an agile landmark (e.g., mothership) in learning an estimate fusion strategy that improves their pose estimates. The learning process in DL-DCL is inspired by the exponentially weighted online learning forecaster (Cesa-Bianchi and Lugosi 2006). Each robot owns an Inertial Measurement Unit (IMU) and a Relative Pose Sensing Suite (RPSS; e.g., range-bearing sensors, camera, LiDAR, RADAR, etc). Each robot observes its neighboring robots and the moving landmark via its RPSS, and shares its estimates and sensor information with its neighboring robots. With this shared information, DL-DCL utilizes an estimation loss feedback which is responsible for learning weights for a two-layered exponentially weighted multi-estimate fusion process. This way, DL-DCL quickly adapts to the uncertainty in the system and/or the environment.

In this paper, convergence analysis of the learned weights shows that they converge exponentially under reasonable assumptions. DL-DCL's performance is evaluated in a simulation with an adverse setting where temporary and permanent faults occur in the robots' sensors, and it is compared against the well-known estimate fusion methods – Kalman Fusion, Covariance Intersection, and Covariance Union. Further, a scalability study is performed as well to check how scalable DL-DCL is compared to the three well-known fusion methods as the no. of robots increases. DL-DCL outperforms these three covariance-based methods in the simulation studies by a substantial margin. For evaluating its Sim2Real aspect, DL-DCL is also simulated in Gazebo with ROS2.

The rest of this paper is organized as follows: section II presents problem formulation and a novel distributed learning framework for decentralized cooperative localization, along with the proposed DL-DCL algorithm. Section III presents a convergence analysis of the weights involved in DL-DCL algorithm. Section IV presents DL-DCL's performance and scalability results via two simulation studies, comparing the three well-known covariance-based fusion methods. Finally, section V concludes this paper.

Distributed Learning based Decentralized Cooperative Localization (DL-DCL)

Problem Formulation

The scenario of Decentralized Cooperative Localization (DCL) of a heterogeneous MRS with a moving landmark

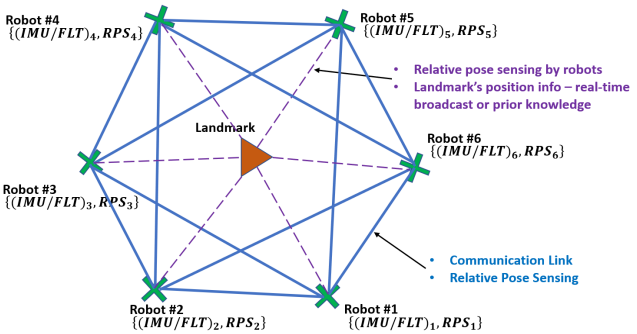

Figure 1: Multi-robot cooperative localization with a moving landmark

(shown in Fig.1) involves robots collaborating over a dynamic interaction (communication and relative pose sensing) network to localize themselves and help localize their neighboring robots in the network with the assistance of a landmark, which can either be moving or stationary. Each robot is installed with an Inertial Measurement Unit (IMU), an Estimation Filter, and a Relative Pose Sensing System (RPSS) for cooperative localization. The landmark can be observed by all the robots by their RPSS. Any object or feature, either moving or stationary, whose accurate pose information is known at all times, either as information known to the robots a priori or via active broadcast by the object itself, can be regarded as a landmark. Heterogeneity is either desirable or unavoidable in multi-robot cooperative localization because of the desired complementarity in sensors/filters installed among the robots or the deterioration of sensor/filter performance in some robots due to failure or environmental uncertainty. One can find such problem scenarios in multi-robot applications involving perimeter defense (Velhal, Sundaram, and Sundararajan 2022), (Shishika and Kumar 2020), convoy protection (Sivakumar and Sujit 2021), (Hentati and Fourati 2021), robotic swarms (Skorobogatov, Barrado, and Salamí 2020), (Mohanty et al. 2020).

The interaction network abstracts the interactions between robots in terms of mutual communication and relative pose sensing, i.e., each robot can sense the relative pose of and can communicate with only its neighbors in the interaction network. The topology of the dynamic interaction network is represented by an underlying bi-directional dynamic graph $G(t)$, where t is the discrete-time variable. The robots share information with their neighbors only once between two successive discrete-time steps. The robots can infer from and observe only their neighboring robots and the landmark, i.e., the robots only have local knowledge of the interaction network and do not know the overall network topology or the total number of robots in the multi-robot system. The neighbour set for the i^{th} robot is defined as: $\Omega_i(t) = \{j : j^{th}$ agent is the neighbour of i^{th} agent at time t, as per $G(t)\}$, where $i = 1, 2, \cdots, N$.

Let N denote the total number of robots in the MRS, and let each robot be represented by its index i, where $i \in [N]$. The robots are equipped with an IMU/Filter suite and a RPSS. Lets denote i^{th} robot's IMU/Filter suite and RPSS as

$(\textit{IMU}/\textit{FLT})_i$ and $\textit{RPS}_i$, respectively, $\forall i \in [N]$. i^{th} robot's IMU/Filter suite $(\textit{IMU}/\textit{FLT})_i$ and RPSS $\textit{RPS}_i$ can be different from j^{th} robot's IMU/Filter suite $(\textit{IMU}/\textit{FLT})_j$ and RPSS $\textit{RPS}_j$, $i \neq j$ and $\forall i, j \in [N]$; the sensors and filters can be of a different class (or type), same class but different parameters, or some of them may undergo failure due to system or environmental uncertainty. This implies that their estimation performance is likely to be different from each other.

Robot Model: Consider the following discrete-time 3-DOF kinematic model for the i^{th} robot, where ΔT is the sampling period (seconds), $\forall i \in [N]$

$$x_{t+1,i} = x_{t,i} + \Delta T \begin{bmatrix} \cos\phi_{t,i} & -\sin\phi_{t,i} \\ \sin\phi_{t,i} & \cos\phi_{t,i} \end{bmatrix} \bar{v}_{t,i} \tag{1a}$$

$$\phi_{t+1,i} = \phi_{t,i} + \Delta T \bar{w}_{t,i} \tag{1b}$$

where $x_{t,i} \in \mathbb{R}^2$ is the i^{th} robot's 2-D position vector (in m), $\bar{v}_{t,i} \in \mathbb{R}^2$ is the i^{th} robot's body-axis velocity vector (m/s), $\phi_{t,i} \in \mathbb{R}$ is the i^{th} robot's heading angle (radians), and $\bar{w}_{t,i} \in \mathbb{R}$ is i^{th} robot's yaw rate (rad/s) at discrete-time t, respectively. Here, the body-axis velocity $\bar{v}_{t,i}$ and yaw rate $\bar{w}_{t,i}$ act as bounded control inputs for the i^{th} robot.

Landmark Model: The landmark model is similar to the robot model. The landmark's position vector $x_{t,A} \in \mathbb{R}^2$ (in m), heading angle $\phi_{t,A} \in \mathbb{R}$ (radians), body-axis velocity $\bar{v}_{t,A} \in \mathbb{R}^2$ (m/s), and yaw rate $\bar{w}_{t,A} \in \mathbb{R}$ (rad/s), respectively, can be represented by replacing i with A in the set of equations (1). Similarly, $\bar{v}_{t,A}$ and $\bar{w}_{t,A}$ act as bounded control inputs for the landmark at time t, which are considered unknown to the robots.

Translational Control Law: For the i^{th} robot, the translational control law consists of two terms as given below

$$\bar{v}_{t,i} = \bar{v}^R_{t,i} + \Delta\bar{v}_{t,i} \tag{2}$$

where $\bar{v}^R_{t,i}$ is the i^{th} robot's reference command signal responsible for executing the desired maneuver as per the mission, and $\Delta\bar{v}_{t,i}$ is the i^{th} robot's correction control signal responsible for avoiding collisions with other robots. A generic expression for the i^{th} robot's reference command signal $\bar{v}^R_{t,i}$ can be given as

$$\bar{v}^R_{t,i} = f_i(\hat{x}^i_{t,i}, \hat{\phi}^i_{t,i}, x_{t,A}, d_S) \tag{3}$$

where $x_{t,A}$ is the landmark's position vector at time t, and $\hat{x}^i_{t,i}$ is the i^{th} robot's estimate of its 2-D position at time t, $\hat{\phi}^i_{t,i}$ is the i^{th} robot's estimate of its yaw angle at time t, $d_S > 0$ (m) is a parameter indicating the safe distance that each robot should maintain from the landmark, $f_i(\cdot)$ is a nonlinear vector function whose structure defines the i^{th} agent's translational maneuvering requirements.

Further, we assume that each robot is equipped with a collision avoidance system to ensure that the robots do not collide. Considering eq.(2), this behavior is modeled by the correction control signal $\Delta\bar{v}_{t,i}$ for the i^{th} robot by using an *inter-robot collision avoidance* control law.

Heading Control Law for the i^{th} robot: Consider the i^{th} robot's estimated heading direction as $\hat{h}^i_{t,i} = \begin{bmatrix} \cos\hat{\phi}^i_{t,i} & \sin\hat{\phi}^i_{t,i} \end{bmatrix}'$. The i^{th} robot's yaw control law can be given as

$$\bar{w}_{t,i} = g_i(\hat{x}^i_{t,i}, \hat{\phi}^i_{t,i}, x_{t,A}) \tag{4}$$

where $g_i(\cdot)$ is a nonlinear vector function whose structure defines the i^{th} agent's rotational maneuvering requirements.

IMU/Filter Suite Model for the i^{th} robot: Denote $\hat{x}^{\textit{FLT}}_{t,i} \in \mathbb{R}^2$ and $\hat{\phi}^{\textit{FLT}}_{t,i} \in \mathbb{R}$ as the i^{th} robot's IMU/Filter suite's estimate of its 2-D position and yaw angle, respectively, at time t. We use a simplified model for the i^{th} robot's IMU/Filter suite as follows:

$$\hat{x}^{\textit{FLT}}_{t,i} = x_{t,i} + \nu^x_{t,i} \tag{5a}$$

$$\hat{\phi}^{\textit{FLT}}_{t,i} = \phi_{t,i} + \nu^\phi_{t,i} \tag{5b}$$

where the terms $\nu^x_{t,i} \in \mathbb{R}^2$ and $\nu^\phi_{t,i} \in \mathbb{R}$ represent bounded arbitrary noise in the i^{th} robot's IMU/Filter suite's estimate.

Relative Pose Sensing System (RPSS) Model for the i^{th} robot: Denote $\Delta\hat{x}_{t,ij} \in \mathbb{R}^2$ and $\Delta\hat{\phi}_{t,ij} \in \mathbb{R}$ as the i^{th} robot's RPSS's estimate of relative position and relative yaw angle, respectively, of j^{th} robot at time t, $\forall j \in \Omega_i(t) \cup \{A\}$, in the global frame. We use a simplified model for the i^{th} robot's RPSS, $\forall j \in \Omega_i(t) \cup \{A\}$, as follows:

$$\Delta\hat{x}_{t,ij} = x_{t,j} - x_{t,i} + \mu^x_{t,i} \tag{6a}$$

$$\Delta\hat{\phi}_{t,ij} = \phi_{t,j} - \phi_{t,j} + \mu^\phi_{t,i} \tag{6b}$$

where the terms $\mu^x_{t,i} \in \mathbb{R}^2$ and $\mu^\phi_{t,i} \in \mathbb{R}$ represent bounded arbitrary noise in the i^{th} robot's RPSS's relative pose estimation output. Further, for $j = i$, we consider $\Delta\hat{x}_{t,ii} = [0\ 0]^T$ and $\Delta\hat{\phi}_{t,ii} = 0$, since i^{th} robot's RPSS's estimate of its own relative pose would be zero by default.

Given the above description of the MRS, the successful execution of their control commands requires the robots to cooperatively form accurate estimates of their pose and that of their neighboring robots by collaborating over the communication network.

Mathematical Formulation

Define $\Lambda_i(t) := \Omega_i(t) \cup \{i\}$. Since the i^{th} robot has access to its neighbour j^{th} robot's information via communication, where $j \in \Lambda_i(t)$, under the framework of DL-DCL, we denote i^{th} robot's various estimates of landmark's (denoted by A) pose at time t, $\forall i \in [N]$, as follows:

$$\{\hat{x}^{\textit{FLT}}_{t,j} + \Delta\hat{x}_{t,jA}\}, \{\hat{\phi}^{\textit{FLT}}_{t,j} + \Delta\hat{\phi}_{t,jA}\} \tag{7}$$

where $\hat{x}^{\textit{FLT}}_{t,j}$ is the j^{th} robot's IMU/Filter suite's estimate of its 2-D position, $\Delta\hat{x}_{t,jA}$ is the j^{th} robot's RPSS's estimate of the relative position of the landmark A in the global frame, and thus, these two terms when added together, i.e., $\{\hat{x}^{\textit{FLT}}_{t,j} + \Delta\hat{x}_{t,jA}\}$, form an estimate of the landmark A's position in the global frame at time t, $\forall j \in \Lambda_i(t)$. Similarly, $\hat{\phi}^{\textit{FLT}}_{t,j}$ is the j^{th} robot's IMU/Filter suite's estimate of its yaw angle, $\Delta\hat{\phi}_{t,jA}$ is the j^{th} robot's RPSS's estimate of relative yaw angle (orientation) of the landmark A, and

$\{\hat{\phi}^{FLT}_{t,j} + \Delta\hat{\phi}_{t,jA}\}$ forms an estimate of the landmark A's yaw angle (orientation) at time t, $\forall j \in \Lambda_i(t)$.

Similarly, by replacing A with i in the above expressions, we denote i^{th} robot's various estimates of its own pose at time t, $\forall i \in [N]$, as follows:

$$\{\hat{x}^{FLT}_{t,j} + \Delta\hat{x}_{t,ji}\}, \{\hat{\phi}^{FLT}_{t,j} + \Delta\hat{\phi}_{t,ji}\} \tag{8}$$

where $\hat{x}^{FLT}_{t,j}$ is the j^{th} robot's IMU/Filter suite's estimate of its 2-D position, $\Delta\hat{x}_{t,ji}$ is the j^{th} robot's RPSS's estimate of relative position of the i^{th} robot in global frame, and $\{\hat{x}^{FLT}_{t,j} + \Delta\hat{x}_{t,ji}\}$ forms an estimate of the i^{th} robot's position in global frame at time t, $\forall j \in \Lambda_i(t)$. Similarly, $\hat{\phi}^{FLT}_{t,j}$ is the j^{th} robot's IMU/Filter suite's estimate of its yaw angle, $\Delta\hat{\phi}_{t,ji}$ is the j^{th} robot's RPSS's estimate of relative yaw angle (orientation) of the i^{th} robot, and $\{\hat{\phi}^{FLT}_{t,j} + \Delta\hat{\phi}_{t,ji}\}$ forms an estimate of the i^{th} robot's yaw angle (orientation) at time t, $\forall j \in \Lambda_i(t)$.

DL-DCL Algorithm

The proposed DL-DCL algorithm is a distributed online learning algorithm designed for the purpose of real-time fault-tolerant decentralized cooperative localization. Its real-time learning ability allows the robots to form accurate estimates under sensor failure, limited communication, and environmental uncertainty.

The DL-DCL algorithm runs in three phases: a projection phase, a learning phase, and an estimation phase. In the algorithm, the learning and estimation for position and orientation happen separately; first orientation, and then, position learning and estimation are performed. For compactness, we will describe the algorithm by considering the measurement of interest as a generic vector y, which can either be the position x or orientation ϕ as per mention. For example, consider the i^{th} robot's measurement of interest denoted as $\hat{y}^{FLT}_{t,i}$, which would mean it's IMU/Filter's position estimate $\hat{x}^{FLT}_{t,i}$ if $y \equiv x$, or heading estimate $\hat{\phi}^{FLT}_{t,i}$ if $y \equiv \phi$.

Projection Phase: involves each robot predicting the current time estimates from previous time estimates using a suitable model that captures robot dynamics. The final output estimates at time $t-1$ by the DL-DCL are denoted as $\hat{x}^{i}_{t-1,i}$ and $\hat{\phi}^{i}_{t-1,i}$ for i^{th} robot's position and heading angle, respectively. Projected estimates $\hat{x}^{PRJ}_{t,i}$ and $\hat{\phi}^{PRJ}_{t,i}$ can be obtained from the previous time DL-DCL final estimates as follows:

$$[\hat{x}^{PRJ}_{t,i}, \hat{\phi}^{PRJ}_{t,i}] = \hat{F}_i(\hat{x}^{i}_{t-1,i}, \hat{\phi}^{i}_{t-1,i}, \bar{v}^{cmd}_{t-1,i}, \bar{w}^{cmd}_{t-1,i}) \tag{9}$$

where $\hat{F}_i$ is a suitable mathematical model capturing i^{th} robot dynamics, $\bar{v}^{cmd}_{t-1,i}$ and $\bar{w}^{cmd}_{t-1,i}$ are i^{th} robot's commanded body-axis velocity and yaw-rate at time $t-1$, respectively. In this paper, we consider $\hat{F}_i$ to be the same as equations (1). This implies

$$\hat{x}^{PRJ}_{t,i} = \hat{x}^{i}_{t-1,i} + \Delta T \begin{bmatrix} \cos\hat{\phi}^{i}_{t-1,i} & -\sin\hat{\phi}^{i}_{t-1,i} \\ \sin\hat{\phi}^{i}_{t-1,i} & \cos\hat{\phi}^{i}_{t-1,i} \end{bmatrix} \bar{v}^{cmd}_{t-1,i} \tag{10a}$$

$$\hat{\phi}^{PRJ}_{t,i} = \hat{\phi}^{i}_{t-1,i} + \Delta T \bar{w}^{cmd}_{t-1,i} \tag{10b}$$

Learning Phase: For each robot, learning occurs by comparing the landmark's pose estimates (in equation 7) with each other, with respect to closeness to the true landmark pose ($\{x_{t,A}, \phi_{t,A}\}$) which is either known by the robots a priori, or via active broadcast by the landmark itself or some other means. The robots learn the weights for fusing the estimates in equation 7, which would lead to an accurate estimation of the landmark's pose and, therefore, can be used to form an accurate estimate of their own pose as well. The learning phase consists of 2 *learning layers*. The 1^{st} layer involves a weighted fusion of all the estimates given by equation 7, including their projected estimates counterpart as given by equations 10. The 2^{nd} layer involves a weighted fusion of estimates from the 1^{st} layer. The weights involved in the weighted fusion process of each layer are updated based on exponential weighted averaging.

Consider a generic measurement of interest denoted as y, which can either be position vector x or heading angle ϕ as per mention. The 1*st learning layer* for i^{th} robot can be described as follows:

$$\tilde{y}^{i}_{t,A} = \sum_{\forall j \in \Lambda_i(t)} \tilde{w}^{y}_{ij}(t-1)(\hat{y}^{FLT}_{t,j} + \Delta\hat{y}_{t,jA}) \tag{11}$$

$$\bar{y}^{i}_{t,A} = \sum_{\forall j \in \Lambda_i(t)} \bar{w}^{y}_{ij}(t-1)(\hat{y}^{PRJ}_{t,j} + \Delta\hat{y}_{t,jA}) \tag{12}$$

where $\tilde{y}^{i}_{t,A}$ denotes the landmark's pose estimate formed by the i^{th} robot by fusing the filter estimates of all the j^{th} robots using equation 7, $\forall j \in \Lambda_i(t)$. Similarly, $\bar{y}^{i}_{t,A}$ denotes the landmark's pose estimate formed by the i^{th} robot by fusing projected estimates of all the j^{th} robots using equation 7 (replacing *FLT* with *PRJ*), $\forall j \in \Lambda_i(t)$. The estimates $\tilde{y}^{i}_{t,A}$ or $\bar{y}^{i}_{t,A}$ can intuitively be understood as the landmark's pose estimate formed by the i^{th} robot by fusing the IMU/Filter or projected estimate information, respectively, and the Relative Pose Sensing information of all the j^{th} robots such that $\forall j \in \Lambda_i(t)$, where $\Lambda_i(t) := \Omega_i(t) \cup \{i\}$, and $\Omega_i(t)$ is the neighbor set of the i^{th} robot at time t.

Define, $\tilde{l}^{i}_{t,A,j} := l(y_{t,A}, (\hat{y}^{FLT}_{t,j} + \Delta\hat{y}_{t,jA}))$, and $\bar{l}^{i}_{t,A,j} := l(y_{t,A}, (\hat{y}^{PRJ}_{t,j} + \Delta\hat{y}_{t,jA}))$. Further, define the following cumulative losses: $\tilde{L}^{i}_{t,A,j} := \sum_{s=1}^{t} \tilde{l}^{i}_{s,A,j}$, and $\bar{L}^{i}_{t,A,j} := \sum_{s=1}^{t} \bar{l}^{i}_{s,A,j}$. The weights involved in the weighted convex sum in equations (11) and (12) are updated as follows:

$$\tilde{w}^{y}_{ij}(t) = \frac{\exp\left(-\eta_w \tilde{L}^{i}_{t,A,j}\right)}{\sum_{\forall j' \in \Lambda_i(t)} \exp\left(-\eta_w \tilde{L}^{i}_{t,A,j'}\right)} \tag{13}$$

and

$$\bar{w}^{y}_{ij}(t) = \frac{\exp\left(-\eta_w \bar{L}^{i}_{t,A,j}\right)}{\sum_{\forall j' \in \Lambda_i(t)} \exp\left(-\eta_w \bar{L}^{i}_{t,A,j'}\right)} \tag{14}$$

An intuitive understanding of the weights involved in the 1^{st} *learning layer* for the i^{th} robot is that they are indicative of which j^{th} robot, among all the neighboring robots including i^{th} robot itself (i.e., $\forall j \in \Lambda_i(t)$), owns the most accurate IMU/Filter or projected estimate information along with accurate relative pose sensing fused information.

The output of the 1^{st} *learning layer* acts as the input for the 2^{nd}*th learning layer*, which can be described as follows:

$$\hat{y}^i_{t,A} = \gamma^y_i(t-1)\tilde{y}^i_{t,A} + (1-\gamma^y_i)\bar{y}^i_{t,A} \tag{15}$$

where $\hat{y}^i_{t,A}$ is the landmark's pose estimate as a final output from the learning phase of the i^{th} robot, which can also be called as the DL-DCL estimate of landmark's pose formed by the i^{th} robot. $\hat{y}^i_{t,A}$ can be seen as the fusion between the estimates based on IMU/Filter information (*FLT*) and projected estimate information (*PRJ*).

Define, $\tilde{l}^i_{t,A} := l(y_{t,A}, \tilde{y}^i_{t,A})$, $\bar{l}^i_{t,A} := l(y_{t,A}, \bar{y}^i_{t,A})$, and $\hat{l}_{t,A} := l(y_{t,A}, \hat{y}_{t,A})$. Further, define the following cumulative losses: $\tilde{L}^i_{t,A} := \sum_{s=1}^{t} \tilde{l}^i_{s,A}$, $\bar{L}^i_{t,A} := \sum_{s=1}^{t} \bar{l}^i_{s,A}$, $\hat{L}_{t,A} := \sum_{s=1}^{t} \hat{l}_{s,A}$. The weights involved in the weighted convex sum in equations (15) are updated as follows:

$$\gamma^y_i(t) = \frac{\exp\left(-\eta_\gamma \tilde{L}^i_{t,A}\right)}{\exp\left(-\eta_\gamma \tilde{L}^i_{t,A}\right) + \exp\left(-\eta_\gamma \bar{L}^i_{t,A}\right)} \tag{16}$$

The weights in the 2^{nd} *learning layer* indicate if the IMU/-Filter estimate by the i^{th} robot is more accurate than the projected estimate by the i^{th} robot, or not.

Estimation Phase: The i^{th} robot in the estimation phase utilizes the weights learned in the learning phase to form a DL-DCL estimate of its own pose, i.e. $\hat{y}^i_{t,i}$. Replacing A by i in equations (11), (12), (15), and using the updated weights, compute $\tilde{y}^i_{t,i}$, $\bar{y}^i_{t,i}$, and $\hat{y}^i_{t,i}$, respectively.

The DL-DCL algorithm is summarized as Algorithm 1.

Algorithm 1: DL-DCL algorithm for the i^{th} agent, $\forall i \in [N]$

Parameters: $T, T_o \geq 1$ (integers); $\eta_w, \eta_\gamma > 0$

Intialization: $\hat{x}^{PRJ}_{0,i} = \hat{x}^{FLT}_{0,i}$, $\hat{\phi}^{PRJ}_{0,i} = \hat{\phi}^{FLT}_{0,i}$;

Initialize all the cumulative losses for $t = 0$ involved in weight update equations (13), (14), and (16) to zero, for y as both x and ϕ, respectively;

Start at $t = 1$.

while $t \leq T$ **do**
  for $y = \{\phi, x\}$ **do**
    Projection Phase: compute $\hat{y}^{PRJ}_{t,i}$ from equations (10)
    Periodic Reset: re-initialize previous time cumulative losses to zero after every T_o discrete time steps
    Learning Phase:
    compute previous time weights as per equations (13), (14), and (16), using previous time cumulative losses with current time local graph connectivity
    compute the landmark's pose estimates given by equations (11), (12), and (15)
    compute current time cumulative losses, and then compute current time weights as per equations (13), (14), and (16), using current time cumulative losses with current time local graph connectivity
    Estimation Phase: with the current time weights, replace A by i in equations (11), (12), and (15) to compute $\hat{y}^i_{t,i}$
    $t = t + 1$
  end for
end while

Convergence Analysis of Weights

Consider $0 \leq \tilde{w}^y_{ij}(0) \leq 1$, where $i = 1, 2, \cdots, N$, and $\forall j \in \Lambda_i(0)$, such that $\sum_{\forall j \in \Lambda_i(0)} \tilde{w}^y_{ij}(0) = 1$. For $t = 1, 2, \cdots, \bar{T}$, from equation (13), $\tilde{w}^y_{ij}(t) = \frac{\exp\left(-\eta_w \tilde{L}^i_{t,A,j}\right)}{\sum_{\forall j' \in \Lambda_i(t)} \exp\left(-\eta_w \tilde{L}^i_{t,A,j'}\right)}$, $\forall j \in \Lambda_i(t)$.

Further, define $j'_*(t) := \arg\min_{j' \in \Lambda_i(t)} \tilde{L}^i_{t,A,j'}$, i.e., $j'_*(t)$ is the index of the robot which incurs the least cumulative loss among all other robots in the index set $\Lambda_i(t) = \Omega_i(t) \cup \{i\}$ at time t, where $\Omega_i(t)$ is the neighbors' index set of the i^{th} robot at time t.

Assumption 1: Both $\lim_{t\to\infty} j'_*(t)$ and $\lim_{t\to\infty} \Lambda_i(t)$ exist uniquely.

Remark: Assumption 1 implies that at $t \to \infty$, the interaction network configuration ($\Lambda_i(t)$) becomes fixed, and for every robot, there is a unique robot (either itself or its neighbor given by the index $j'_*(t)$) that incurs the least cumulative loss out of all the robots in the set $\lim_{t\to\infty} \Lambda_i(t)$.

Theorem 1. *Under assumption 1, DL-DCL algorithm's weights $\tilde{w}^y_{ij}(t)$ satisfy the following:*

$$\lim_{t\to\infty} \tilde{w}^y_{ij}(t) = 0, \quad \forall j \in \lim_{t\to\infty} \Lambda_i(t) \setminus \{j'_*(t)\} \tag{17}$$

and

$$\lim_{t\to\infty} \tilde{w}^y_{ij'_*(t)}(t) = 1 \tag{18}$$

where $j'_*(t)$ is the index of the neighbor of the i^{th} robot whose estimate incurs the least cumulative loss at time t, i.e., $j'_*(t) = \arg\min_{j' \in \Lambda_i(t)} \tilde{L}^i_{t,A,j'}$, $\forall i \in [N]$.

Proof. provided in the supplementary document[1] (hint: find suitable upper and lower bounds on the weights and show that they converge) □

Similar to convergence analysis of weights $\tilde{w}^y_{ij}(t)$, one can also derive the convergence results for the weights $\bar{w}^y_{ij}(t)$ and $\gamma^y_i(t)$.

Performance Evaluation

For evaluating the performance of the proposed DL-DCL algorithm, we consider a simulation scenario involving $N = 6$ robots executing a perimeter monitoring task around a mothership (moving landmark). Such a mothership scenario can be practical in swarm applications where due to cost constraints, all the robots cannot have high-end sensor equipment, but only a few can. In such scenarios, those few robots with high-end sensor equipment can act as virtual moving

[1] supplementary doc. sharepoint link: http://surl.li/fgxsq

landmarks for the other robots with cheap sensor equipment. The scenario involves a random robot-robot interaction network with a link drop probability of 0.5, whose underlying base graph topology is that of the ring network. The mothership broadcasts its true position and orientation to all the robots and is observable to all the robots (via their RPSS) at all times. The limited field of view of RPSS has been simulated by the degree (no. of links a node is connected to) of a node (robot) in the interaction network (an interaction network link implies both a communication link and an observation link with the neighboring robot).

Results are averaged over 50 simulation runs. Each simulation run is carried out for a horizon of $T = 1400$ discrete time steps, with a sampling period of $\Delta T = 0.1$ second. The scenario considered is adverse, where three robots are chosen randomly in each simulation run so that their IMUs fail successively at times 0.0, 23.4, and 46.7 seconds. The RPSS of at least one and at most three robots fail, also chosen randomly after approximately every $23.4 \sim 46.7$ seconds in each simulation run. The noise $\nu^x_{t,i}$ and $\nu^\phi_{t,i}$ in the IMUs (equations (5)) that are functioning is assumed to be Gaussian with a mean of $0.05m$ and 0.5 deg., respectively, with a covariance of $0.1 \times (0.05)^2 m^2$ and $0.1 \times (0.05)^2 rad.^2$, respectively. For the IMUs that fail, the noise terms are still Gaussian but with a large bias (mean) of $3m$ and 30 deg., respectively, with a covariance of $6 \times (0.05)^2 m^2$ and $6 \times (0.05)^2 rad.^2$, respectively. Similarly, the noise $\mu^x_{t,i}$ and $\mu^\phi_{t,i}$ in the RPSS (equations (6)) that are functioning is assumed to be Gaussian with a mean of $0.05m$ and 0.5 deg., respectively, with a covariance of $0.1 \times (0.05)^2 m^2$ and $0.1 \times (0.05)^2 rad.^2$, respectively. For the RPSS that fail, the noise terms are still Gaussian but with a large bias (mean) of $2m$ and 20 deg., respectively, with a covariance of $4 \times (0.05)^2 m^2$ and $4 \times (0.05)^2 rad.^2$, respectively. The loss functions are defined to be $l_x(p_1, p_2) = \min(||p_1 - p_2||/15, 1)$ and $l_\theta(q_1, q_2) = \min(|wrapToPi(q_1 - q_2)|/(15\pi/180), 1)$ for 2-D position and heading angle, respectively. Define $\hat{l}_{t,i} := l(y_{t,i}, \hat{y}_{t,i})$, and the cumulative loss: $\hat{L}_{t,i} := \sum_{s=1}^{t} \hat{l}_{s,i}$, where $y \in \{x, \phi\}$ as per mention.

For the above-described adverse setup, DL-DCL is compared against three well-known decentralized estimate fusion methods (Kalman Fusion, Covariance Intersection, Covariance Fusion) and two methods involving no communication among the robots, described as follows:

No Comm., IMU only: involves no communication among the robots; the robots just rely on their respective IMU for their pose estimation, i.e., $\hat{x}^i_{t,i} = \hat{x}^{FLT}_{t,i}$ and $\hat{\phi}^i_{t,i} = \hat{\phi}^{FLT}_{t,i}$, from equations (5).

No Comm., RPSS only: involves no communication among the robots; the robots rely on their RPSS and the landmark's pose information for their pose estimation, i.e., $\hat{x}^i_{t,i} = x_{t,A} - \Delta\hat{x}_{t,iA}$ and $\hat{\phi}^i_{t,i} = \phi_{t,A} - \Delta\hat{\phi}_{t,iA}$, from Eq.(6).

Kalman Fusion ((Maybeck 1982),(Uhlmann 2003)): each robot takes the Kalman fusion of all its pose estimates given by its own sensors and that of its neighbors (from equation (8)); assumes that the estimates being fused are uncorrelated and their associated zero-mean Gaussian noises' covariance are known.

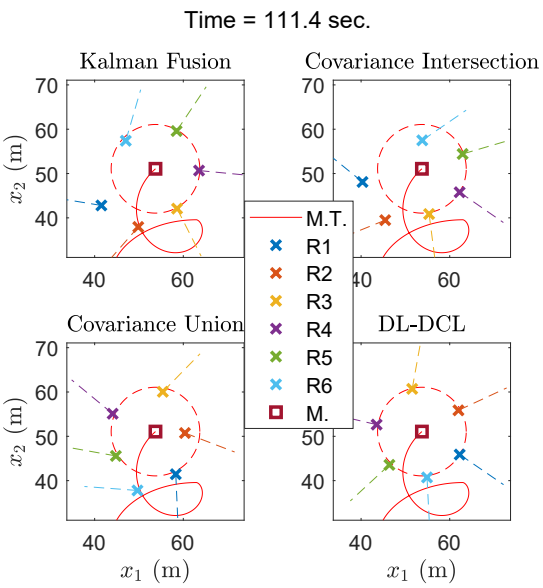

Figure 2: Snapshot of a simulation run (M: Mothership; M.T: Mothership's Trajectory; Robots: Ri, $i = 1, 2, \cdots, 6$)

Covariance Intersection ((Matzka and Altendorfer 2009),(Julier and Uhlmann 2017)): each robot employs the covariance intersection method for the fusion of all its pose estimates given by its own sensors and that of its neighbors (from equation (8)); assumes that the estimates being fused are consistent and their associated zero-mean Gaussian noises' covariance are known, but their cross-correlation is unknown.

Covariance Union ((Matzka and Altendorfer 2009),(Reece and Roberts 2010)): each robot employs the covariance union method for the fusion of all its pose estimates given by its own sensors and that of its neighbors (from equation (8)); assumes that the estimates being fused can be inconsistent, and their cross-correlation is unknown, but their associated zero-mean Gaussian noises' covariance are known.

For the DL-DCL algorithm, the learning parameters are set to the values $\eta_w = 2$ and $\eta_\gamma = 2$ via parameter tuning. DL-DCL periodically resets its cumulative loss variables to zero after every $T_o = 200$ discrete time steps to avoid bias build-up during learning.

Note that DL-DCL does not make assumptions on the noise involved in the pose estimates to be zero mean Gaussian or of any other type in specific. Moreover, DL-DCL does not require noise covariance information as input, unlike the above-described covariance-based fusion methods. Further, for the covariance-based methods, it is assumed that the pose estimates given by equation (8) are formed by adding two uncorrelated estimates: one provided by the IMU of a robot (j^{th} robot's pose estimate) and the other given by its RPSS (estimate of the displacement between j^{th} robot and i^{th} robot, w.r.t. j^{th} robot). A snapshot of the above-described simulation for the considered three well-known covariance-based fusion methods and DL-DCL is shown in

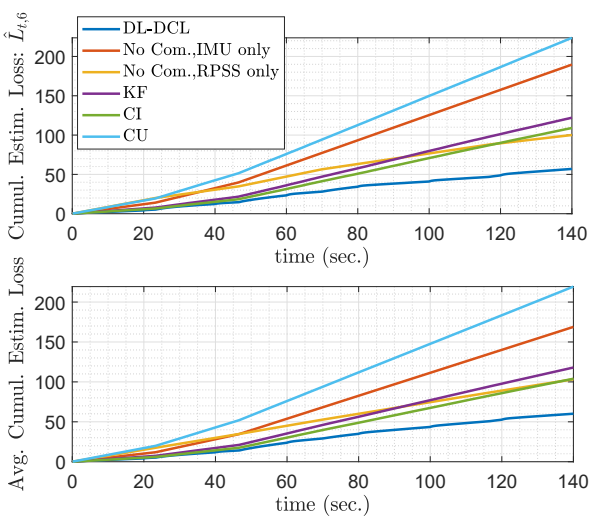

Figure 3: Individual and Average cumulative estimation loss over time

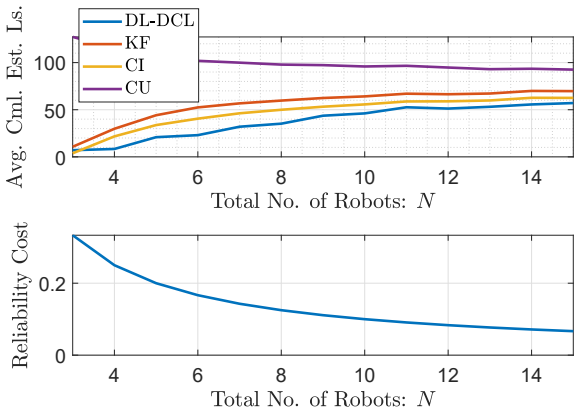

Figure 4: The avg. cumulative estimation loss (just after 60 sec.) and reliability cost versus the number of robots

Fig. 2. Further description regarding the simulation setup is given in the supplementary document[1].

The results for the above-described simulation setup with an adverse setting are shown in Fig. 3. Note that cumulative loss for i^{th} robot is given as $\hat{L}_{t,i} = \sum_{s=1}^{t} \hat{l}_{s,i}$. In Fig. 3, the first plot shows how the cumulative estimation loss incurred by the 6^{th} robot's estimate of its position, averaged over 50 simulation runs, grows with time for DL-DCL and the three well-known fusion methods considered for comparison; DL-DCL clearly performs substantially better than the covariance-based methods considered – approximately 41% improvement compared to the best-performing covariance-based method, CI. Similar behavior can be seen for other robots as well – plots are provided in the supplementary document[1]. The second plot in Fig. 3 shows how the Average total cumulative estimation loss of all six robots ($\frac{1}{6}\sum_{i=1}^{6} \hat{L}_{t,i}$) grows with time, averaged over 50 simulation runs. DL-DCL outperforms the other well-known covariance-based methods – average total cumulative loss incurred is much lesser compared to the best-performing covariance-based method, CI – 40% lesser. In the simulation without failures (zero bias in the estimates and small covariance), DL-DCL, CI, and KF incur approximately zero estimation loss.

Further, a scalability study is carried out in which, starting with three robots – one fully functioning with good sensor equipment, one with below-average sensor equipment, and one with faulty sensor equipment – we keep on adding more robots in a ring network topology by randomly choosing between the one with good sensor equipment or the one with below-average sensor equipment. The results (Fig. 4) clearly show that DL-DCL performs better than the three well-known fusion methods in terms of scalability, incurring lesser avg. cumulative estimation loss ($\frac{1}{N}\sum_{i=1}^{N} \hat{L}_{t,i}$) than other methods even as N increases. Moreover, at larger N ($N \geq 11$), the avg. cumul. estim. loss starts to plateau nearly, whereas the reliability cost decreases. Thus, DL-DCL allows an MRS to ensure high reliability by having more no. of robots while ensuring that the estimation performance does not degrade as N gets large.

The average time taken by one iteration of the DL-DCL algorithm (MATLAB code) is approximately 1.2 milliseconds, whereas that of Kalman Fusion, Covariance Intersection, and Covariance Union is 0.1, 0.31 and 7.4 milliseconds, respectively. The appropriate vectorization of the DL-DCL code can further reduce the average time a DL-DCL iteration takes. For the evaluation of its sim2real aspect, DL-DCL is also simulated in Gazebo with ROS2 (video and GitHub links in the supplementary document[1]). Note that DL-DCL can work for any robot model (holonomic/non-holonomic).

Conclusion

This paper presents a novel Distributed Learning-based Decentralized Cooperative Localization (DL-DCL) approach where robots communicating over a dynamic, partially connected network are assisted by an agile landmark in learning a multi-estimate fusion strategy. DL-DCL's online learning process uses an estimation loss feedback, allowing the algorithm to be adaptive and fault-tolerant. Further, the inclusion of the projection information ensures that information eventually reaches the non-neighboring robots in the communication network. Convergence analysis of the weights involved in DL-DCL shows that the weights converge at an exponential rate under reasonable assumptions. Moreover, the DL-DCL algorithm involves analytic expressions, which makes it computationally inexpensive and easy to implement. Simulation results show that in adverse conditions (sensor failures inducing a 40-60 times increase in the bias), DL-DCL outperforms the well-known covariance-based methods (KF, CI, CU) in terms of estimation performance (40% better) and passes the scalability test. Sim2Real aspects of DL-DCL's fault-tolerance and practical implementability are also evaluated in Gazebo using ROS2. Further, the DL-DCL framework can be extended to the cases of 3D localization and partially observable landmark.

Acknowledgments

The authors would like to thank Nokia Centre for Excellence in Networked Robotics, IISc, Bangalore, and Nokia CSR funds for their support.

References

Assa, A.; and Janabi-Sharifi, F. 2015. A Kalman filter-based framework for enhanced sensor fusion. *IEEE Sensors Journal*, 15(6): 3281–3292.

Carrillo-Arce, L. C.; Nerurkar, E. D.; Gordillo, J. L.; and Roumeliotis, S. I. 2013. Decentralized multi-robot cooperative localization using covariance intersection. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1412–1417. IEEE.

Cesa-Bianchi, N.; and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge university press.

Chang, T.-K.; Chen, K.; and Mehta, A. 2021. Resilient and consistent multirobot cooperative localization with covariance intersection. *IEEE Transactions on Robotics*.

Gregory, J.; Fink, J.; Stump, E.; Twigg, J.; Rogers, J.; Baran, D.; Fung, N.; and Young, S. 2016. Application of multi-robot systems to disaster-relief scenarios with limited communication. In *Field and Service Robotics*, 639–653. Springer.

Hentati, A. I.; and Fourati, L. C. 2021. A convoy of ground mobile vehicles protection using cooperative uavs-based system. In *2021 International Symposium on Networks, Computers and Communications (ISNCC)*, 1–6. IEEE.

Huang, S.; and Dissanayake, G. 1999. Robot localization: An introduction. *Wiley Encyclopedia of Electrical and Electronics Engineering*, 1–10.

Julier, S.; and Uhlmann, J. K. 2017. General decentralized data fusion with covariance intersection. In *Handbook of multisensor data fusion*, 339–364. CRC Press.

Khan, N. A.; Jhanjhi, N.; Brohi, S. N.; Usmani, R. S. A.; and Nayyar, A. 2020. Smart traffic monitoring system using unmanned aerial vehicles (UAVs). *Computer Communications*, 157: 434–443.

Matzka, S.; and Altendorfer, R. 2009. A comparison of track-to-track fusion algorithms for automotive sensor fusion. In *Multisensor Fusion and Integration for Intelligent Systems*, 69–81. Springer.

Maybeck, P. S. 1982. *Stochastic models, estimation, and control*. Academic press.

Mohanty, N.; Gadde, M. S.; Sundaram, S.; Sundararajan, N.; and Sujit, P. 2020. Context-Aware Deep Q-Network for Decentralized Cooperative Reconnaissance by a Robotic Swarm. *arXiv preprint arXiv:2001.11710*.

Mohiuddin, A.; Tarek, T.; Zweiri, Y.; and Gan, D. 2020. A survey of single and multi-UAV aerial manipulation. *Unmanned Systems*, 8(02): 119–147.

Pires, A. G.; Rezeck, P. A.; Chaves, R. A.; Macharet, D. G.; and Chaimowicz, L. 2021. Cooperative Localization and Mapping with Robotic Swarms. *Journal of Intelligent & Robotic Systems*, 102(2): 1–23.

Reece, S.; and Roberts, S. 2010. Generalised covariance union: A unified approach to hypothesis merging in tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 46(1): 207–221.

Scherer, J.; Yahyanejad, S.; Hayat, S.; Yanmaz, E.; Andre, T.; Khan, A.; Vukadinovic, V.; Bettstetter, C.; Hellwagner, H.; and Rinner, B. 2015. An autonomous multi-UAV system for search and rescue. In *Proceedings of the First Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*, 33–38.

Shishika, D.; and Kumar, V. 2020. A review of multi agent perimeter defense games. In *International Conference on Decision and Game Theory for Security*, 472–485. Springer.

Sivakumar, V.; and Sujit, P. 2021. MPC-based Multi-UAV Path Planning for Convoy Protection in 3D. In *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 1554–1559. IEEE.

Skorobogatov, G.; Barrado, C.; and Salamí, E. 2020. Multiple UAV systems: A survey. *Unmanned Systems*, 8(02): 149–169.

Spry, S. C.; Girard, A. R.; and Hedrick, J. K. 2005. Convoy protection using multiple unmanned aerial vehicles: organization and coordination. In *Proceedings of the 2005, American Control Conference, 2005.*, 3524–3529. IEEE.

Uhlmann, J. K. 2003. Covariance consistency methods for fault-tolerant distributed data fusion. *Information Fusion*, 4(3): 201–215.

Velhal, S.; Sundaram, S.; and Sundararajan, N. 2022. A Decentralized Multirobot Spatiotemporal Multitask Assignment Approach for Perimeter Defense. *IEEE Transactions on Robotics*.

Wang, X.; Sun, S.; Li, T.; and Liu, Y. 2021. Fault tolerant multi-robot cooperative localization based on covariance union. *IEEE Robotics and Automation Letters*, 6(4): 7799–7806.