Cooperative Drift Mitigation for UAV Swarms in GNSS-Denied Environments

Matei-Alexandru Pânzariu[1]
Supervisor(s): Dr. Arash Asadi[1]**, Florian Kosterhon**[1]
[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

June 25, 2026

Name of the student: Matei-Alexandru Pânzariu
Final project course: CSE3000 Research Project
Thesis committee: Dr. Arash Asadi, Florian Kosterhon, Dr. Georgios Iosifidis

***Abstract*—Unmanned Aerial Vehicles (UAVs) operating in GNSS-denied environments typically rely on Inertial Measurement Units (IMUs) for position estimation. However, this approach is susceptible to error accumulation, commonly known as inertial drift. Standard industry solutions mitigate this issue by fusing IMU data with external sensors such as LiDAR or cameras. However, these sensing modalities are not suitable for all environments. An alternative approach is to leverage cooperation within a swarm of drones, enabling agents to exchange information and improve their position estimates collectively. One such method employs a Distributed Graph Optimization (DGO) algorithm to cross-reference spatial uncertainties among UAVs in the swarm. However, existing DGO frameworks are primarily validated using relative swarm cohesion metrics, which provide little insight into the swarm's absolute positioning accuracy.**

To address this limitation, this paper evaluates a basic DGO state estimation model against a basic Dead Reckoning (DR) baseline. A Python-based simulation environment was developed, and four experimental conditions were investigated: varying sensor quality, swarm size, flight duration, and trajectory geometry. The results show that DGO outperforms DR under degraded sensor conditions, whereas DR maintains lower error during short-duration flights when high-quality sensors are available. Crucially, a temporal breakeven point is identified beyond which the unbounded error growth of DR exceeds that of the cooperative DGO framework. This finding demonstrates that while standalone DR offers superior short-term precision, cooperative estimation provides a more stable and sustainable framework for prolonged operations in GNSS-denied environments.

I. Introduction

Unmanned aerial vehicles (UAVs) are increasingly being deployed in complex environments for deliveries, rescue missions or even data collection [7]. While Global Navigation Satellite Systems (GNSS) typically provide absolute positioning, their signals are sometimes degraded, spoofed or entirely unavailable in critical areas [16]. Consequently, modern autonomous UAVs must be capable of navigating strictly using onboard, local sensing modalities [1].

In GNSS-denied environments, drones rely on the Inertial Measurement Unit (IMU) to estimate their trajectory through inertial navigation [17]. However, IMUs are inherently susceptible to high-frequency noise and bias instability.

To mitigate this drift, standard industry approaches fuse IMU data with a suite of exteroceptive sensors [22]. These frameworks commonly employ Visual Odometry (VO) via cameras, spatial mapping using LiDAR or Ultra-Wideband (UWB) to continuously correct inertial estimates [4]. However, the reliability of these standalone fusion techniques degrades significantly in challenging operational scenarios. Vision and LiDAR systems frequently fail in visually obscured or featureless conditions—such as heavy smoke or dust [23].

Alternatively, Artificial Intelligence (AI) techniques can be employed to model and compensate for non-linear sensor noise, theoretically mitigating inertial drift [5] [8]. However, these same studies note that learning-based methods present significant practical shortcomings for resource-constrained UAVs, including prohibitive computational overhead and a severe vulnerability to predictive degradation in unfamiliar environments.

A promising solution to mitigate navigation drift is cooperative estimation. By deploying a swarm of interconnected UAVs into a jammed environment, individual agents can exchange relative measurements and state data to collectively estimate the group's position. Through continuous information sharing, the swarm can cross-reference localization uncertainties and heavily constrain the growth of individual positional errors, typically utilizing either centralized or distributed estimation architectures [3].

Existing literature frequently points to cooperative estimation as a solution for navigational drift [2] [3] [12] [20]. To demonstrate these advantages, this paper utilizes a simulation framework to answer a comparative question: To what degree does drone cooperation mitigate inertial drift when compared to a standalone Inertial Navigation System?

To answer this question, we investigate the difference between standalone and cooperative approaches and explore the impact of swarm size on the quality of the estimation. Furthermore, this paper also explores the impact that IMU quality has on drift mitigation.

We developed a simulation environment to evaluate the absolute positioning accuracy of a modified Distributed Graph Optimization (DGO) algorithm, building upon the framework proposed by Xiong and You [20]. While the original study primarily validated the algorithm's efficiency in maintaining relative swarm cohesion, our work isolates the state-estimation module to rigorously test its efficiency for global trajectory tracking as individual sensor noise accumulates over time.

The remainder of this paper is structured as follows: Section II reviews existing mitigation techniques, Section III formalizes the proposed architecture, Section IV introduces the simulation setup, Section V analyzes the implications of the results, Section VI addresses responsible research practices, and Section VII summarizes the findings.

II. Background and related work

This section surveys current works on mitigating inertial drift when absolute global positioning is unavailable.

A. Position estimation for single agent approaches

When operating inside GNSS-denied environments, UAVs need to operate entirely on their on-board sensors, to estimate their position without the use of external signals or infrastructure [1]. This section presents some of these approaches, from basic Dead Reckoning, to traditional sensor fusion algorithm and learning based methods.

1) Basic Dead Reckoning

Basic Dead Reckoning (DR) serves as a fundamental trajectory estimation methodology for UAVs [17]. To estimate the position, DR utilizes a recursive state-estimation process. It takes the last known absolute position and continuously advances the vehicle's estimated coordinates by applying displacement vectors derived from onboard sensors.

The efficiency of DR in GNSS-denied or degraded environments has been the subject of extensive evaluation. For

instance, Szykula and Furtak [18] investigated DR performance under varying degrees of signal denial, demonstrating that intermittent GNSS availability significantly bounds error growth compared to environments with absolute signal loss. Furthermore, Kissai and Smith [14] mathematically established that relying strictly on simple DR without external feedback loops rapidly yields cumulative positional errors.

Ultimately, while basic DR provides a critical short-term bridging capability, its standalone viability is constrained by sensor drift. This limitation renders simple DR unreliable for extended navigation in strictly GNSS-denied environments. Consequently, modern UAV localization frameworks treat DR not as a standalone solution, but as a state-estimation layer that requires continuous recalibration via exteroceptive modalities, such as Optical Flow [19] or LiDAR [13].

2) Sensor Fusion Techniques

To overcome the inherent vulnerabilities of individual sensing modalities—such as the unbounded drift of basic DR or the limited range of local sensors—modern navigation frameworks rely on sensor fusion. This algorithmic process mathematically blends complementary data streams into a central estimator, reconstructing a vehicle trajectory that is significantly more accurate, reliable, and robust than any single sensor could achieve in isolation. In modern UAV navigation, the Extended Kalman Filter (EKF)—the non-linear counterpart to the foundational Kalman Filter [11]—has become the industry standard for this task.

Instead of blindly propagating sensor inputs through DR, the EKF utilizes a recursive Bayesian framework. It dynamically weights the certainty of a dynamic system model against incoming, noisy sensor measurements to estimate an unobservable state and bound estimation error. In GNSS-denied environments, where absolute positioning is unavailable, this sensor fusion framework is critical. Existing single-agent approaches rely heavily on the EKF to fuse noisy internal data (such as an IMU) with external measurements (such as Visual Odometry or LiDAR).

The advantages of the EKF over basic DR are well documented. While standalone DR inevitably suffers from unbounded error accumulation due to integrated sensor noise, the EKF effectively bounds this uncertainty. For instance, a comparative study by Zhou et al. [24] demonstrated that an EKF-based localization system significantly reduces cumulative trajectory error in GNSS-denied environments compared to unassisted DR.

Despite its widespread adoption in autonomous navigation, the standard EKF has several fundamental limitations. Because UAV motion is inherently nonlinear, the EKF relies on Jacobian matrices to linearize the transition and measurement functions. Consequently, its performance depends strongly on the accuracy of these linear approximations and the underlying model parameters. As discussed in [9] and [10], EKFs are often difficult to implement and tune, and their reliance on complex algebraic derivations makes them particularly prone to implementation errors.

3) AI based estimation techniques

Artificial Intelligence (AI) has emerged as a highly effective strategy for mitigating the complex, non-linear errors that appear in GNSS-denied UAV navigation. Rather than relying solely on rigid mathematical models, recent advancements utilize data-driven approaches to learn the noise characteristics of UAVs directly from sensor measurements.

For instance, Cioffi et al. [5] proposed a hybrid architecture that combines deep learning modules with a traditional EKF. In this setup, a neural network is utilized to learn the complex error dynamics of the inertial sensors, predicting positional displacements to update the filter. This hybrid approach effectively bounds the drift, outperforming standard DR odometry. Building on the concept of adaptive estimation, M. Irfan et al. [8] proposed a deep-learning-based sensor fusion framework. Unlike classic approaches that rely on static parameters, this framework evaluates environmental changes in real-time and dynamically adjusts the fusion weights accordingly. By intelligently prioritizing reliable sensor data, the system can maintain high accuracy even as conditions fluctuate.

Ultimately, the integration of AI and ML represents a significant advancement over traditional, purely mathematical state estimation. However, as acknowledged within these studies, learning-based methods still exhibit significant practical shortcomings [5] [8]. Most notably, because these models rely heavily on their training data, they frequently experience a drop in performance when deployed in unfamiliar environments.

B. Cooperative Position Estimation

When single-agent systems face drift during extended GNSS-denied operations, modern localization frameworks frequently pivot to cooperative estimation [3] [12]. By deploying a swarm of interconnected UAVs, individual agents can exchange relative spatial measurements and state data to collectively estimate the group's position, thereby cross-referencing uncertainties and heavily constraining the growth of individual positional errors.

1) Extended Kalman Filters in Cooperative Scenarios

As mentioned before, EKFs are an industry standard due to their reliability in position estimation. Naturally, scientists have tried to couple them with cooperative approaches.

For instance, Belfadel et al. [2] utilized an EKF to estimate the cooperative position of a drone swarm, relying on data fused from anchor drones stationed outside a jammed area. Similarly, Ellingson and McLain [6] developed an EKF-based relative navigation framework that allows UAVs to share inter-vehicle measurements to limit group drift.

However, as established in the previous section, standard EKF approaches possess foundational mathematical vulnerabilities. Furthermore, these EKF architectures suffer from severe scalability limitations. Unlike lighter distributed methods, cooperative EKF frameworks require the continuous, high-frequency transfer of both state estimates and high-dimensional covariance matrices between agents. Because the EKF must maintain cross-covariances between all agents to

remain consistent, the size of the joint covariance matrix scales quadratically with the swarm size. As the number of drones increases, these dense matrices impose a high communication burden on the members of the swarm [15].

2) Graph Optimization and Distributed Architectures

To overcome the problems introduced by the EKF, Graph Optimization (GO) has emerged as an alternative for multi-robot systems. In a standard GO scheme, the state of each drone is represented as a vertex, and the relative inter-agent measurements are represented as edges. By evaluating these constraints, the swarm's spatial configuration can be determined through mathematical optimization.

These algorithms have been implemented in multiple systems. For example, Hao Xu et al. introduce the Omni-Swarm approach in [21], where a UAV swarm collects omnidirectional visual and UWB distance data. This data is then processed through a graph optimization algorithm that fuses measurements such as local motion, inter-drone distances, and visual detections to minimize the overall error and estimate the position of each drone.

Similarly, Xiong and You [20] propose a lightweight Distributed Graph Optimization (DGO) framework, in which each drone in the swarm solves a localized optimization subproblem in parallel using solely onboard sensors, which measure relative distance, relative angle, and self-state displacement.

However, a limitation of the studies on the DGO algorithm, such as the one proposed by Xiong and You, lies in the evaluation metrics that are used. Xiong and You validate their model [20] by using the relative distance between drones in the swarm as their primary Key Performance Indicator (KPI). While this effectively proves that the control framework maintains swarm cohesion, it fails to quantify how accurately the cooperative algorithm estimates the swarm's absolute global position as individual sensor noise accumulates over time.

III. System Design

To address these evaluation shortcomings, the relative positioning estimation framework implemented in this study is based on the DGO architecture proposed by Xiong and You [20], but alters the experimental methodology.

The architecture proposed by the original authors relies on a tightly coupled system: it utilizes DGO to estimate relative spatial positioning, and Decentralized Model Predictive Control (DMPC) to execute physical flight corrections. However, this study intentionally isolates the DGO module to evaluate its standalone state-estimation accuracy.

A. Theoretical Sensor Models

To execute the DGO algorithm, each UAV utilizes three theoretical measurement models representing onboard proprioceptive and exteroceptive sensors.

The relative distance $d_{i,j}(k)$ measured by drone i observing drone j at epoch k is modeled as the true Euclidean distance affected by sensor noise:

$$d_{i,j}(k) = h^d(\boldsymbol{p}_i(k), \boldsymbol{p}_j(k)) + v_d(k) \tag{1}$$

Here, $h^d(\boldsymbol{p}_i(k), \boldsymbol{p}_j(k))$ represents the true geometric distance between the position vectors, and $v_d(k)$ represents the measurement noise associated with distance sensors.

Similarly, the onboard vision system provides bearing measurements representing the relative angle $\theta_{i,j}(k)$ between the drones:

$$\theta_{i,j}(k) = h^\theta(\boldsymbol{p}_i(k), \boldsymbol{p}_j(k)) + v_\theta(k) \tag{2}$$

Where $h^\theta(\boldsymbol{p}_i(k), \boldsymbol{p}_j(k))$ calculates the true geometric bearing, and $v_\theta(k)$ accounts for the angular inaccuracy of the optical sensor.

For self-state estimation, representing high-frequency IMU odometry, the algorithm relies on the step-to-step displacement of each drone. The measured displacement $\Delta p_i(k-1, k)$ between epoch $k-1$ and k is defined as:

$$\Delta \boldsymbol{p}_i(k-1, k) = h^s(\boldsymbol{p}_i(k) - \boldsymbol{p}_i(k-1)) + \boldsymbol{b} + \boldsymbol{v}_s(k) \tag{3}$$

where $h^s(\boldsymbol{p}_i(k) - \boldsymbol{p}_i(k-1))$ is the measurement function of the state estimation, such that the measured displacement is mapped directly to the true displacement vector, $\boldsymbol{b}$ is the bias and $\boldsymbol{v}_s(k)$ is the noise associated with the sensor.

Displacement between epochs is utilized because, while absolute state estimation accumulates unbounded error over time, short-term displacement remains mathematically stable for calculating cost functions.

B. Distributed Graph Optimization (DGO)

The DGO algorithm requires the drones to cooperatively minimize a set of objective functions to estimate their global positions. First, we define $\mathcal{N}_i$ as the neighboring drones of drone i.

The distance cost function penalizes deviations between the measured distance and the theoretical geometric distance calculated by the solver:

$$J_i^d(k) = \sum_{j \in \mathcal{D}_i} \left\| d_{i,j}(k) - \|\boldsymbol{p}_i(k) - \boldsymbol{p}_j^c(k)\|_2 \right\|_{\Sigma_d}^2 \tag{4}$$

In this formulation, $\mathcal{D}_i$ ($\mathcal{D}_i \subseteq \mathcal{N}_i$) denotes the subset of drones observable by drone i, while $\boldsymbol{p}_i(k)$ is the current position vector being evaluated by the optimizer. Furthermore, $\boldsymbol{p}_j^c(k)$ represents the most recently received estimate from neighbor j, corresponding to its state from the previous optimization epoch $(k-1)$. To account for hardware reliability, the residual is weighted by the Mahalanobis norm using the distance sensor variance Σ_d.

The angle cost function minimizes the discrepancy between the camera measurements and the theoretical angle derived from the coordinate estimates:

$$J_i^\theta(k) = \sum_{j \in \Theta_i} \left\| \theta_{i,j}(k) - \angle(\boldsymbol{p}_i(k), \boldsymbol{p}_j^c(k)) \right\|_{\Sigma_\theta}^2 \tag{5}$$

Here, Θ_i ($\Theta_i \subseteq \mathcal{N}_i$) is the set of drones observed by drone i.

Finally, the self-state estimation cost function compares the step-to-step displacement calculated by the optimizer against the drifting odometry measurement:

$$J_i^s(k) = \|\Delta \boldsymbol{p}_i(k-1, k) - (\boldsymbol{p}_i(k) - \boldsymbol{p}_i(k-1))\|_{\Sigma_s}^2 \tag{6}$$

In the final step of the original DGO algorithm, proposed by Xiong and You, the objective function simply sums the individual costs. However, an analysis of this strategy reveals a scaling vulnerability. Because the distance and angle cost functions (Equations 4 and 5) are summations over the sets of observed drones, increasing the swarm size linearly increases the magnitude of these constraints. Consequently, the algorithm prioritizes the swarm cohesion over maintaining the global position of the drones.

To resolve this imbalance, we propose a normalized composite objective function. By scaling the relative measurement costs by the cardinality of their respective observation sets, the algorithm ensures that the inter-agent constraints remain balanced against the self-state estimation, regardless of the swarm density.

Therefore, the final estimated position $\hat{p}_i(k)$ at any given time step can be computed by minimizing the following cost function:

$$\hat{\boldsymbol{p}}_i(k) = \arg\min_{\boldsymbol{p_i}(k)} \left(\frac{1}{|\mathcal{D}_i|} J_i^d(k) + \frac{1}{|\Theta_i|} J_i^\theta(k) + J_i^s(k) \right) \quad (7)$$

C. Algorithm Execution

To execute the position estimation algorithm, the swarm operates in a decentralized structure. At each timestep (epoch k), the drones rely on their onboard sensors to gather relative distance, relative bearing and self-state displacement. The optimizer takes these measurements and computes the spatial coordinates that best satisfy the sensor constraints and minimizes the cost function. Once the optimization is completed, the drones broadcast the computed positions to their neighboring drones, to act as the optimized position estimation ($\boldsymbol{p}_i^c(k)$). The execution flow is outlined in Algorithm 1.

Algorithm 1 Isolated DGO State-Estimation

1: **Input:** Relative measurements $d_{i,j}(k)$ and $\theta_{i,j}(k)$, State estimation $\Delta \boldsymbol{p}_i(k-1,k)$
2: **Output:** Optimized position estimate $\hat{\boldsymbol{p}}_i(k)$
3: **Initialize** system and starting coordinates
4: **for** each drone $i \in \{1,\dots,N\}$ **in parallel do**
5: **Receive** broadcasted positions $\boldsymbol{p}_j^c(k)$ from neighbors $j \in \mathcal{N}_i$
6: **Calculate** distance cost $J_i^d(k)$ from Equation 4
7: **Calculate** angle cost $J_i^\theta(k)$ from Equation 5
8: **Calculate** self-state estimation cost $J_i^s(k)$ from Equation 6
9: **Optimize:** $\hat{\boldsymbol{p}}_i(k) = \arg\min_{\vec{p}_i(k)} (\frac{1}{|\mathcal{D}_i|} J_i^d(k) + \frac{1}{|\Theta_i|} J_i^\theta(k) + J_i^s(k))$
10: **Broadcast** $\hat{\boldsymbol{p}}_i(k)$ to observable neighborhood $\mathcal{N}_i$ for epoch $k+1$ (this becomes $\boldsymbol{p}_i^c(k+1)$)
11: **end for**

IV. Simulation and Experiments

A. Simulation Setup

To evaluate the efficacy of the proposed cooperative estimation framework, a simulation environment was developed in Python 3.10.12. The experimental setup isolates the DGO module to evaluate its open-loop state-estimation accuracy, independent of physical control loop compensations.

In the original study, the primary evaluation metric was the relative distance between drones. Under this relative metric, the DMPC module did not negatively impact the measured performance, as the KPI only kept track of the swarm formation. Conversely, because this study evaluates absolute trajectory deviation, keeping the DMPC active introduces a critical issue. Physical flight corrections executed by the DMPC—such as steering a drone to maintain formation with a drifting neighbor—would actively alter the absolute trajectory, making it hard to test whether the positioning error was caused by the estimation errors or by physical control errors.

The UAV swarm is programmed to track a predefined 8-shaped trajectory. This specific geometry was selected because its continuous variations provide a robust stress test for uncompensated inertial drift over time. The simulated flight operates at an average velocity of 0.31 m/s.The total flight duration is treated as an independent variable and adjusted per experiment to evaluate both short-term precision and long-term drift accumulation. The trajectory is outlined in Figure 1.

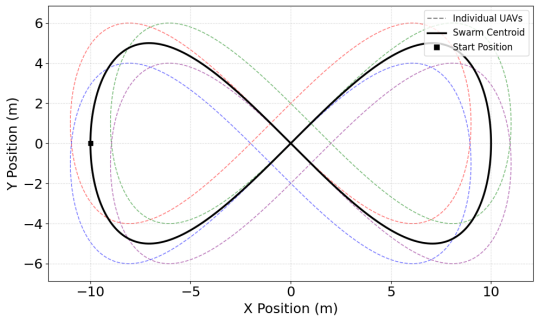

Fig. 1: Predefined Trajectory

B. Validation of implementation

Before evaluating the modified version of the DGO algorithm, it is necessary to validate the underlying simulation environment and the baseline implementation. In the original study, Xiong and You validated their cooperative framework by utilizing the relative distance between drones as the primary KPI to measure swarm cohesion.

To prove the algorithmic integrity of our isolated DGO implementation, a validation experiment was conducted using the original, un-normalized cost functions. The swarm was deployed along the simulated 8-shaped trajectory.

In their original study, Xiong and You [20] demonstrated that the DGO framework maintains swarm cohesion by bounding the relative distance error between agents to approximately

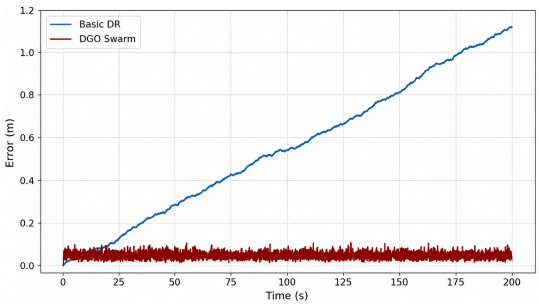

Fig. 2: Original Paper experiment

0.2 meters. As illustrated in Figure 2, our baseline DGO implementation mirrors this performance, successfully bounding the relative distance error to under 0.2 meters throughout flight operations. This comparison confirms that our implementation replicates the capabilities of the original algorithm.

However, while this validation confirms that the internal swarm geometry is accurately preserved, a cohesion-based KPI fails to quantify how accurately the cooperative algorithm estimates the swarm's absolute global position as individual sensor noise accumulates over time. Therefore, to evaluate the true viability of DGO in GNSS-denied environments, the subsequent experiments require a shift in evaluation metrics.

C. Baseline and Evaluation Metrics

To evaluate the effectiveness of the cooperative localization approach, the modified DGO algorithm is compared against a Basic Dead Reckoning (DR) baseline. In the DR baseline, each UAV estimates its position solely from its own displacement measurements, without exchanging information with other agents. To ensure a fair comparison, the baseline uses the same displacement measurements, $\Delta \boldsymbol{p}_i(k-1,k)$, defined in Equation 3, that are used by the DGO framework for self-state estimation. The position estimate is propagated according to Equation 8

$$\hat{\boldsymbol{p}}_{i,DR}(k) = \hat{\boldsymbol{p}}_{i,DR}(k-1) + \Delta \boldsymbol{p}_i(k-1,k) \tag{8}$$

Both approaches are evaluated on the same simulated N-agent swarm. While DGO incorporates inter-agent cooperation, the DR baseline treats each drone independently and disables all communication between agents.

Performance is quantified using the Absolute Trajectory Error (ATE), defined as the Root Mean Square Error (RMSE) between the estimated and ground-truth trajectories. For drone i, the ATE is given by:

$$ATE_i = \sqrt{\frac{1}{K}\sum_{k=1}^{K} \left|\hat{\boldsymbol{p}}_i(k) - \boldsymbol{p}_i^*(k)\right|^2} \tag{9}$$

where K denotes the total number of time epochs. To obtain a swarm-level performance measure, the ATE values of all drones are averaged. This metric serves as the primary KPI throughout the evaluation.

D. Experiment 1: Impact of sensor quality on Drift Mitigation

The first experiment quantifies how varying degrees of hardware noise impact the drift mitigation capabilities of the DGO framework compared to the standalone DR baseline. To evaluate the mathematical efficiency of this algorithm, we conduct a parametric sensitivity analysis. The system is evaluated across three distinct sensor quality profiles: 'Degraded' (Table I), 'Baseline' (Table II) and 'Ideal' (Table III). The 'Baseline' profile utilizes the sensor variances and drift rate established in the original study by Xiong and You [20]. To test the failure points of the DGO module, these baseline parameters were modified to create the 'Degraded' and 'Ideal' scenarios. By exposing the algorithms to these varying noise profiles, we are able to assess their performance against sensor degradation.

TABLE I: Degraded Sensor Profile

Sensor	**Parameters**	
	Sampling rate	*Error*
Distance sensor	25 Hz	$\Sigma_d = 0.5$ m
Angle sensor	10 Hz	$\Sigma_\theta = 5°$
State estimation	100 Hz	Error = 0.05 m/s $\Sigma_s = 0.09$ m/s

TABLE II: Baseline Sensor Profile

Sensor	**Parameters**	
	Sampling rate	*Error*
Distance sensor	25 Hz	$\Sigma_d = 0.1$ m
Angle sensor	10 Hz	$\Sigma_\theta = 2°$
State estimation	100 Hz	Error = 0.005 m/s $\Sigma_s = 0.05$ m/s

TABLE III: Ideal Sensor Profile

Sensor	**Parameters**	
	Sampling rate	*Error*
Distance sensor	25 Hz	$\Sigma_d = 0.02$ m
Angle sensor	10 Hz	$\Sigma_\theta = 0.5°$
State estimation	100 Hz	Error = 0.001 m/s $\Sigma_s = 0.005$ m/s

The results of Experiment 1, illustrated in Figure 3, demonstrate that the efficacy of the cooperative DGO framework is highly dependent on hardware quality. When operating with highly degraded sensors (Table I), the DGO significantly outperforms basic DR. Conversely, under the 'Baseline' and 'Ideal' profiles, the Absolute Trajectory Error (ATE) detailed in Table IV reveals that Basic DR starts as the more accurate approach, but it is outperformed by the modified DGO algorithm after a certain temporal threshold. This is mathematically consistent, as the DGO actively injects high-frequency relative measurement noise into the state estimate, which temporarily outweighs the slow, linear drift of a high-quality, unassisted IMU.

Additionally, Figure 3 exhibits a distinct, reduction in the DGO error curve. Given that inertial drift is cumulative,

this apparent self-correction is not an algorithmic feature, but rather a geometric artifact caused by the self-intersecting coordinates of the 8-shaped flight path. This phenomenon is isolated and analyzed in Experiment 4.

TABLE IV: Experiment 1: KPI

Sensor Profile	Basic DR ATE(m)	DGO ATE(m)
Degraded	5.75	2.46
Baseline	0.57	0.63
Ideal	0.12	0.14

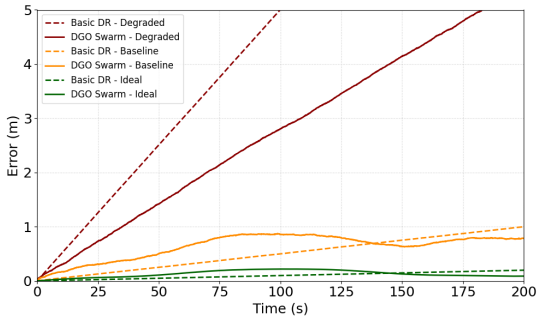

Fig. 3: Algorithmic Performance for Different Sensor Profiles: Accumulation of Positioning Error

E. Experiment 2: Effect of Swarm Size

The second experiment aims to determine how swarm size impacts the cooperative localization accuracy. Utilizing the 'Baseline' quality sensors profile (Table II), the simulation evaluates the absolute positioning error across variable swarm sizes of $N \in \{2, 4, 6, 8\}$ drones. The purpose of this experiment is to test if increasing the swarm size leads to a reduction in trajectory deviation.

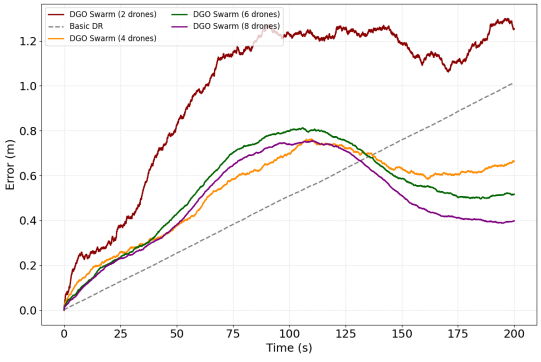

Fig. 4: Algorithmic Performance for Different Swarm Sizes: Accumulation of Positioning Error

As illustrated in Figure 4 and Table V, expanding the swarm size directly improves the global positioning accuracy of the cooperative framework. Although the standalone basic DR baseline exhibits superior initial precision, the Cooperative DGO framework maintains a lower Absolute Trajectory Error (ATE) over extended flight durations. Notably, the data indicates that the most significant performance gain occurs when

TABLE V: Experiment 2: KPI

Swarm Size	Basic DR ATE(m)	DGO ATE(m)
1	0.58	-
2	-	1.07
4	-	0.6
6	-	0.57
8	-	0.47

scaling from two to four agents. While adding more drones continues to reduce the localization error, these subsequent marginal improvements are less pronounced.

F. Experiment 3: Time Horizon experiment

This experiment evaluates the long-term temporal stability of the localization frameworks by isolating flight duration as the primary variable. Utilizing the 'Baseline' quality sensors (Table II), we analyze absolute positioning error over extended trajectories. The objective is to quantify which approach is more sustainable for longer flight durations.

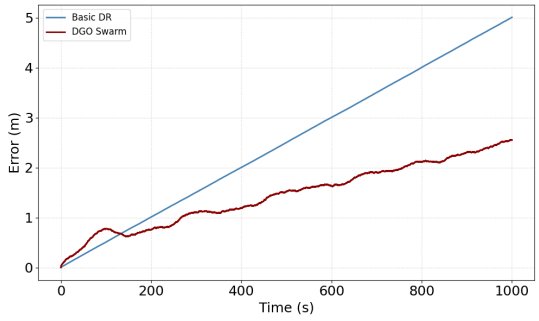

Fig. 5: Algorithmic Performance over Long Time Horizons: Accumulation of Positioning Error

TABLE VI: Experiment 3: KPI

Sensor Profile	Basic DR ATE(m)	DGO ATE(m)
Baseline	2.89	1.58

The results presented in Table VI show that the DGO algorithm achieves a lower overall ATE of 1.58 m, compared to 2.89 m for basic DR. Figure 5 provides further insight into this difference. Although basic DR exhibits a lower absolute positioning error during the initial stages of the flight, its error grows at a nearly constant rate, reaching approximately 5.0 m after 1000 seconds. In contrast, the DGO algorithm demonstrates superior long-term stability.

This result is caused by the characteristics of each individual algorithm. Basic DR relies exclusively on the IMU, which provides a lower initial noise floor. This allows the algorithm to perform better on short-term flights. However, because the error in basic DR accumulates over time and the algorithm lacks a bounding mechanism, the estimation ends up degrading over time.

On the other hand, the DGO algorithm forces the swarm to incorporate external measurements. In the short term, adding

these terms to the optimization leads to a worse system precision. However, for longer time horizons, the cross-referencing of the DGO algorithm leads to a better performance.

G. Experiment 4: Trajectory Geometry

To address the anomaly of the decreasing error observed in the previous experiments, this evaluation introduces a simple, straight-line flight path. Utilizing the average quality sensor profile (Table II), we analyze the positioning error over this new trajectory to pinpoint the exact cause of the previously observed error reductions.

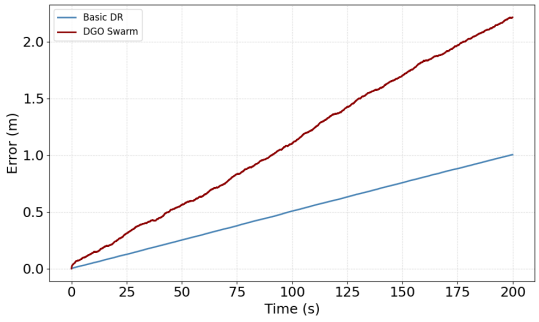

Fig. 6: Algorithmic Performance on a Linear Trajectory: Accumulation of Positioning Error

The results of the linear flight, illustrated in Figure 6, reveal a contrast to the 8-shaped trajectory. When constrained to a straight path, the basic DR baseline exhibits a slow, linear drift, accumulating to approximately 1.0 meters of error. Conversely, the DGO swarm experiences a severe degradation in localization accuracy, reaching over 2.0 meters of error within the same time frame. The previously observed increase and decrease in the error graph are absent, confirming that the decrease was a geometric artifact, created by the 8-shaped trajectory.

In a strictly linear flight, the relative angle and distance between drones remains almost entirely constant. Because of this, the algorithm lacks the diverse spatial constraints necessary to correct the global trajectory. Instead of correcting the drift, the DGO algorithm degrades the estimate, by injecting distance and angular sensor noise into the state optimization, without the geometric diversity required to balance it.

V. Discussion

The experimental results highlight a functional trade-off in GNSS-denied environments. As observed in Experiment 3, the standalone basic DR baseline maintains superior trajectory estimation during the initial flight phases. This is consistent with the sensor model, as the IMU data provides a low initial noise floor. However, because basic DR lacks an external bounding mechanism, this precision degrades into an unbounded, linear drift, ultimately reaching an absolute positioning error of 5.0 meters at the 1000-second mark. Conversely, the DGO Swarm model introduces a higher initial baseline error, due to the inherent measurement variance of relative distance and bearing sensors. By concluding the extended trajectory with a reduced error of approximately 2.6 meters, it surpasses the standalone method over longer operational horizons.

Furthermore, the specific temporal breakeven point observed at approximately 170 seconds exposes a gap in existing literature. While prior work, such as the study by Xiong and You [20], validates DGO primarily through the metric of relative swarm cohesion, our evaluation reveals a critical operational nuance: cooperative estimation imposes a definitive short-term accuracy penalty compared to standalone DR. However, for prolonged missions, the DGO algorithm performs better.

In addition to these temporal constraints, the results of the linear trajectory evaluation (Experiment 4) establish a strict geometric boundary for cooperative estimation. The experiment confirmed that the periodic error reductions observed in the 8-shaped trajectory were geometric artifacts. More importantly, it demonstrated that in strictly linear flight, the DGO framework underperforms basic DR. Because linear paths lack the geometric variations necessary to make relative spatial constraints observable, the continuous injection of high-frequency sensor noise actively degrades the estimate faster than stable DR. Thus, the true utility of DGO is fundamentally tied not only to mission duration but also to trajectory complexity.

The results of Experiment 2 show that the modified DGO algorithm scales well as more drones are added, although the benefits gradually decrease. Increasing the swarm size improves the absolute state estimation by providing more spatial information between agents. This confirms that the normalized objective function works as intended, addressing a key issue in the original DGO formulation. By balancing the influence of relative measurements and odometry data, the algorithm prevents inter-agent observations from dominating the optimization process. As a result, the swarm maintains an accurate global trajectory instead of focusing only on its internal formation. While increasing the number of agents from two to four significantly reduces the estimation error, adding more agents leads to smaller improvements, indicating a point where the extra computational cost may no longer justify the performance gain.

Finally, while the isolated DGO framework demonstrates a promising cooperative approach, its operational limitations in real-world deployments must be addressed. This study intentionally isolates the state-estimation loop to strictly quantify algorithmic accuracy. In a fully deployable architecture, integrating a Distributed Model Predictive Control (DMPC) module to execute physical flight corrections would likely introduce computational latency.

VI. Responsible Research

a) Reproducibility and Transparency

To ensure the reproducibility of the results presented in this paper, the experimental framework has been described in detail. The simulation environment was developed using Python 3.10.12. Furthermore, all experimental parameters and sensor models have been explicitly specified in their respective sections, enabling independent verification and replication of the reported findings.

b) Work Integrity

This study is based on the work of Xiong and You [20]. Throughout the paper, clear distinction is made between the original methodology and the modifications introduced in this work, specifically the normalization of the cost functions and the revised performance metric. Furthermore, the assumptions and limitations of the original framework are explicitly acknowledged. In particular, the open-loop nature of the simulation and the omission of the DMPC module are highlighted to avoid overstating the algorithm's applicability to real-world deployment.

c) Use of Artificial Intelligence

Throughout the project, Artificial Intelligence—specifically Large Language Models (LLMs)—was utilized for multiple tasks in the development process. For academic writing, LLMs assisted with text formatting, grammatical refinement, and structural clarity. Additionally, AI served as a tool for brainstorming ideas and connecting related concepts. Crucially, LLMs were not used in the development of the underlying algorithms presented in this paper; all simulated results are the direct output of the mathematical models described. The author comprehensively reviewed all AI-assisted text. Finally, LLMs were employed to assess the overall quality of the manuscript.

d) Impact of the Research

The development of UAVs in GNSS-denied environments is a complex and multifaceted challenge. The primary motivation of this work is to enhance the safety and reliability of autonomous swarms operating in critical applications such as delivery services and search-and-rescue missions in degraded or compromised environments. At the same time, it is acknowledged that algorithms enabling increased autonomy in GNSS-jammed settings may also be adapted for military or surveillance purposes. This dual-use nature is recognized, and the intended focus of this research is on promoting applications that prioritize safety, transparency, and societal benefit.

VII. Conclusion

In conclusion, Cooperative Position Estimation techniques provide a vital mechanism for extending the operational viability of GNSS-denied UAV missions. This study has demonstrated that while basic DR offers short-term precision, the unbounded nature of inertial drift severely limits its long-term reliability. Conversely, the DGO algorithm surpasses standalone DR over extended flight durations.

However, this evaluation also establishes critical operational and architectural boundaries for cooperative localization. Specifically, the framework's efficacy is tied to trajectory complexity; during linear flights, the injection of relative sensor noise degrades DGO performance below that of basic DR. Additionally, the evaluation confirmed that increasing the swarm size successfully improves global positioning accuracy, validating that the normalized cost function effectively balances local swarm cohesion with absolute trajectory estimation.

While this open-loop evaluation provides clear baseline metrics, future work must address the complexities of fully deployed systems. Subsequent research should expand upon this foundation by integrating active DMPC modules into the localization framework. Reintegrating the DMPC will allow researchers to assess how physical trajectory corrections impact the overall absolute positioning error. Furthermore, translating DGO corrections into active formation adjustments requires assessing the computational overhead and potential latency introduced to the system.

The simulation environment must also be expanded to introduce variable UAV kinematics, such as differing flight speeds. Alongside these dynamic kinematics, future studies should investigate whether the initial swarm formation affects the accuracy of the position estimation. Finally, extending the evaluation to analyze how having drones at varying Y-axis coordinates changes the algorithm's performance will be crucial for testing the framework across a wider, more complex spectrum of operational scenarios.

References

[1] Saleh Alghamdi et al. "Autonomous Navigation Systems in GPS-Denied Environments: A Review of Techniques and Applications". In: *2025 11th International Conference on Automation, Robotics, and Applications (ICARA)*. 2025, pp. 290–299. DOI: 10.1109/ICARA64554.2025.10977619.

[2] Djedjiga Belfadel, David Haessig, and Cherif Chibane. "Range-Only EKF-Based Relative Navigation for UAV Swarms in GPS-Denied Zones". In: *IEEE Access* 12 (2024), pp. 154832–154842. DOI: 10.1109/ACCESS.2024.3481409.

[3] Anusna Chakraborty et al. "Cooperative localization: Challenges and future directions". In: *Cooperative localization and navigation*. CRC Press, 2019, pp. 493–519. DOI: 10.1201/9780429507229-24.

[4] Yingxiu Chang et al. "A review of UAV autonomous navigation in GPS-denied environments". In: *Robotics and Autonomous Systems* 170 (2023), p. 104533. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2023.104533. URL: https://www.sciencedirect.com/science/article/pii/S0921889023001720.

[5] Giovanni Cioffi et al. "Learned Inertial Odometry for Autonomous Drone Racing". In: *IEEE Robotics and Automation Letters* 8.5 (2023), pp. 2684–2691. DOI: 10.1109/LRA.2023.3252342.

[6] Gary Ellingson and Tim McLain. "Progress on GPS-Denied, Multi-Vehicle, Fixed-Wing Cooperative Localization". In: *Utah Space Grant Consortium Conference*. Presented May 6, 2019. Logan, Utah, USA: Utah State University, May 2019. URL: https://digitalcommons.usu.edu/spacegrant/2019/Session_three/3/.

[7] Shweta Gupte, Paul Infant Teenu Mohandas, and James M. Conrad. "A survey of quadrotor Unmanned Aerial Vehicles". In: *2012 Proceedings of IEEE Southeastcon*. 2012, pp. 1–6. DOI: 10.1109/SECon.2012.6196930.

[8] Mahammad Irfan et al. “LSAF-LSTM-Based Self-Adaptive Multi-Sensor Fusion for Robust UAV State Estimation in Challenging Environments”. In: *Machines* 13.2 (2025). ISSN: 2075-1702. DOI: 10.3390/machines13020130. URL: https://www.mdpi.com/2075-1702/13/2/130.

[9] S.J. Julier and J.K. Uhlmann. “Unscented filtering and nonlinear estimation”. In: *Proceedings of the IEEE* 92.3 (2004), pp. 401–422. DOI: 10.1109/JPROC.2003.823141.

[10] Simon J. Julier and Jeffrey K. Uhlmann. “New extension of the Kalman filter to nonlinear systems”. In: *Signal Processing, Sensor Fusion, and Target Recognition VI*. Ed. by Ivan Kadar. Vol. 3068. International Society for Optics and Photonics. SPIE, 1997, pp. 182–193. DOI: 10.1117/12.280797. URL: https://doi.org/10.1117/12.280797.

[11] Rudolf E. Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: *Journal of Basic Engineering* 82.1 (Mar. 1960), pp. 35–45. DOI: 10.1115/1.3662552. URL: https://doi.org/10.1115/1.3662552.

[12] Cagri Kilic, Eduardo Gutierrez, and Jason N. Gross. “Evaluation of the Benefits of Zero Velocity Update in Decentralized Extended Kalman Filter-Based Cooperative Localization Algorithms for GNSS-Denied Multi-Robot Systems”. In: *NAVIGATION: Journal of the Institute of Navigation* 70.4 (2023). ISSN: 0028-1522. DOI: 10.33012/navi.608. eprint: https://navi.ion.org/content/70/4/navi.608.full.pdf. URL: https://navi.ion.org/content/70/4/navi.608.

[13] Hwamog Kim, Donghoon Kim, and Seongjai Kim. “Real-time Geospatial Positioning for UAVs in GPS-Denied Environment Using LiDAR Data”. In: *AIAA Scitech 2020 Forum*. American Institute of Aeronautics and Astronautics, 2020. DOI: 10.2514/6.2020-2194. eprint: https://arc.aiaa.org/doi/pdf/10.2514/6.2020-2194. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2020-2194.

[14] A. Kissai and M. Smith. “UAV Dead Reckoning with and without using INS/GPS Integrated System in GPS denied Polar Regions”. In: *International Journal of Aeronautics and Aerospace Engineering* 1.2 (2019), pp. 58–67. DOI: 10.18689/ijae-1000109.

[15] Eric Nettleton et al. “Decentralised SLAM with Low-Bandwidth Communication for Teams of Vehicles”. In: *Field and Service Robotics: Recent Advances in Reserch and Applications*. Ed. by Shin’ichi Yuta et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 179–188. ISBN: 978-3-540-32854-4. DOI: 10.1007/10991459_18. URL: https://doi.org/10.1007/10991459_18.

[16] Desmond Schmidt et al. “A Survey and Analysis of the GNSS Spoofing Threat and Countermeasures”. In: *ACM Comput. Surv.* 48.4 (May 2016). ISSN: 0360-0300. DOI: 10.1145/2897166. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/2897166.

[17] Artur Shurin and Itzik Klein. “QDR: A Quadrotor Dead Reckoning Framework”. In: *IEEE Access* 8 (2020), pp. 204433–204440. DOI: 10.1109/ACCESS.2020.3037468.

[18] Bartlomiej Szykula and Janusz Furtak. “Dead Reckoning Method for an Unmanned Aerial Vehicle in Conditions of Limited GPS Signal”. In: *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation* 19.4 (2025), pp. 1063–1068. ISSN: 2083-6473. DOI: 10.12716/1001.19.04.01. URL: ./Article_Dead_Reckoning_Method_for_an_Unmanned_Szykula,76,1591.html.

[19] Jakub Walczak et al. “Fusion of Optical Flow and Dead Reckoning Algorithms for UAV Navigation Without GPS”. In: *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation* 19.4 (2025), pp. 1069–1074. ISSN: 2083-6473. DOI: 10.12716/1001.19.04.02. URL: ./Article_Fusion_of_Optical_Flow_and_Dead_Walczak,76,1592.html.

[20] Chengsong Xiong and Zheng You. “UAV Swarm Autonomy through Cooperative Positioning: A Unified Approach with Distributed Graph Optimization and Decentralized MPC”. In: *2025 IEEE International Conference on Mechatronics and Automation (ICMA)*. 2025, pp. 273–279. DOI: 10.1109/ICMA65362.2025.11120831.

[21] Hao Xu et al. “Omni-Swarm: A Decentralized Omnidirectional Visual–Inertial–UWB State Estimation System for Aerial Swarms”. In: *IEEE Transactions on Robotics* 38.6 (2022), pp. 3374–3394. DOI: 10.1109/TRO.2022.3182503.

[22] Xiaowei Xu et al. “RISE-VIO: Robust Initialization and Targeted Pose Robustification for INS-Centric Visual–Inertial Odometry Under Degraded Visual Conditions”. In: *Sensors* 26.8 (2026). ISSN: 1424-8220. DOI: 10.3390/s26082305. URL: https://www.mdpi.com/1424-8220/26/8/2305.

[23] Shuo Zheng et al. “Physically-Based LiDAR Smoke Simulation for Robust 3D Object Detection”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 40. 16. 2026, pp. 13441–13448. DOI: 10.1609/aaai.v40i16.38348.

[24] Qing-Li Zhou et al. “Dead reckoning and Kalman filter design for trajectory tracking of a quadrotor UAV”. In: *Proceedings of 2010 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*. 2010, pp. 119–124. DOI: 10.1109/MESA.2010.5552088.

APPENDIX A
USE OF ARTIFICIAL INTELLIGENCE

As stated in the report, Artificial Intelligence—specifically Large Language Models (LLMs)—was used for multiple tasks in the development process.

Below are examples of prompts that were used for each specific task:

A. Text Formatting, Grammatical Refinement, and Structural Refinement

- "Because standard UAV kinematics involve highly nonlinear rigid-body dynamics, the EKF relies on Jacobian matrices to linearize transition and measurement functions. Rephrase this."
- "Algorithmic Performance For Linear Trajectory: Accumulation of Positioning Error. Polish this title."

B. Brainstorming Ideas

- "How is constraint imbalance solved in mathematical modeling and Artificial Intelligence?"

C. Overall Quality of the Manuscript Assessment

- "Examine my explanation of this experiment as harshly as possible."
- "Give me harsh feedback on this abstract."