

Master Thesis Posters

Promotion 2023-2025

IFRoS



ELTE
EÖTVÖS LORÁND
UNIVERSITY

Erasmus Mundus Joint Master in
INTELLIGENT FIELD ROBOTIC SYSTEMS



With the support of the
Erasmus+ Programme
of the European Union

Contents

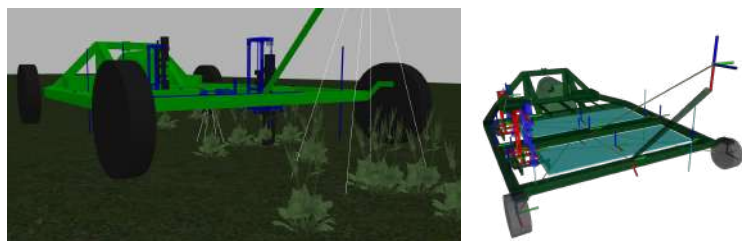
1	TASK ALLOCATION AND STOP OPTIMIZATION FOR AUTONOMOUS WEED- ING David razhiel ceres arroyo	3
2	Robust LiDAR-Inertial Localization with Prior Maps in GNSS-Challenged Envi- ronments Eliyas Kidanemaraim Abraha	4
3	Indirect Visual-Inertial Simultaneous Localization and Mapping for Dense 3D Re- construction Goitom Abrha Leaku	5
4	LiDAR Odometry and Mapping Beyond RTK Accuracy Liviu-Daniel Florescu	6
5	DEEP REINFORCEMENT LEARNING FOR DRONE OBSTACLE AVOIDANCE Thanh Loc Pham	7
6	Deep Reinforcement Learning for Robot Manipulation Vania Katherine Mulia	8
7	Sim2Real Transfer Learning for Object Tracking in Robotic Applications Mir Mohibullah Sazid	9
8	Fire Detection and Ranging forAutonomous Firefighting Drones Mohamed Khaled OthmanJune	10
9	ENHANCING INDOOR MAPPING AND LOCALIZATION IN SPECULAR RICH ENVIRONMENTS USING DEEP LEARNING AND SENSOR FUSION Renatto Tommasi Hernández	11
10	Detection of Invisible Obstacles for Drone Obstacle Avoidance Sawera Yaseen	12
11	Urban Object Detection Using Sensor Fusion for Autonomous Navigation Selin Yavuz	13
12	Enhancing Deep Reinforcement Learning with Curriculum Learning for Robotic Tasks Syma Afsha	14
13	Supervised Learning for Robot Manipulation Tanakrit Lertcompeesin	15
14	Vision-Based Tracking and Following of a Moving Target Using an Unmanned Aerial Vehicle Fatima Yousif Rustamani	16
15	Globally Consistent Mapping in Indoor and Outdoor Environments Using Hybrid LiDAR SLAM Zewdie Habtie Sisay	17
16	Joint Underwater Mapping with Acoustic and Optical Images Precious Philip-Ifabiyi	18
17	Diurnal and Nocturnal Robust Visual-Aided GNSS Navigation on Horticultural Fields Lisa Paul Magoti	19

Abstract

This thesis explores three different algorithms; **graph search**, **optimization-based**, and **market-based** to solve the task allocation problem of assigning detected weeds to the weed-removal mechanisms of the NUGA system. Our results demonstrate that these approaches can **reduce total mission time** by up to **22.8%** in high-density scenarios and **decrease tool idle time** by as much as **94.9%** compared to the baseline allocation method.

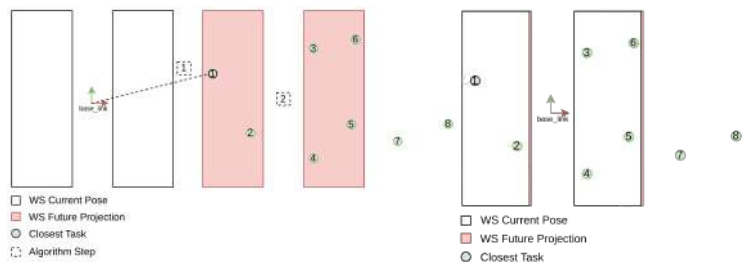
Contributions

- **Simulation Testbed:** Developed a realistic simulation environment to evaluate task allocation algorithms for autonomous weed removal, with performance logging and mission time tracking.
- **Task Allocation Algorithms:** Designed and implemented three novel approaches with distinct paradigms that demonstrate reduction in total mission time and tool idle time.
- **NUGA System Integration:** Integrated the task allocation strategies into the NUGA autonomous robot system for real-time decision-making and execution.

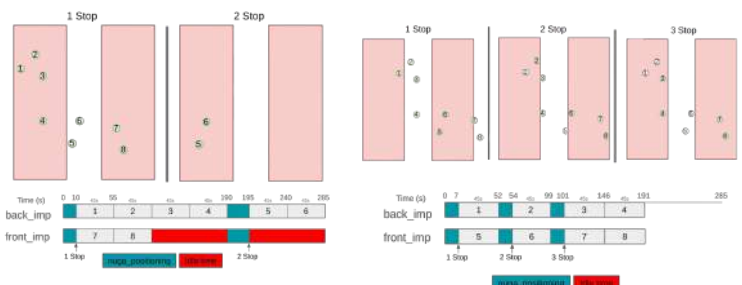


Methodology

- **Heuristic:** 1: Get the position of the closest weed from the current robot position. 2: Project the tools' Workspace (WS) forward (in the future). 3: Allocate weeds to each tool if they fall within its projected WS. 4: Move the robot until the tools' WS are aligned.

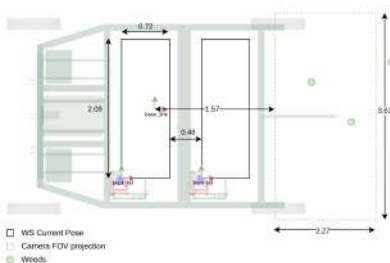


Suboptimality Example

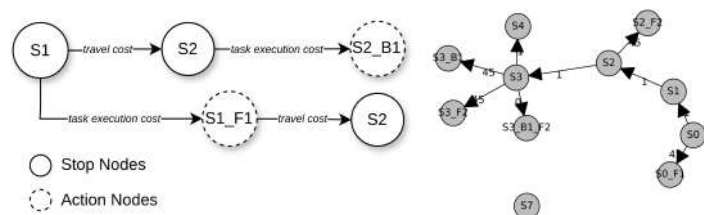


Problem Constraints

- Geometric Constraints
- Movement Constraints
- Task Processing
- Dynamic Environment
- Processing Time




- **GraphSearch:** 1: Compute candidate stops based on the robot's current position and weed detections. 2: Associate reachable weeds with candidate stops. 3: Create a graph representation of the problem using DFS algorithm. 4: Get the shortest path in the graph using Dijkstra's algorithm. 5: Decode solution and return the next optimal stop and TA output.



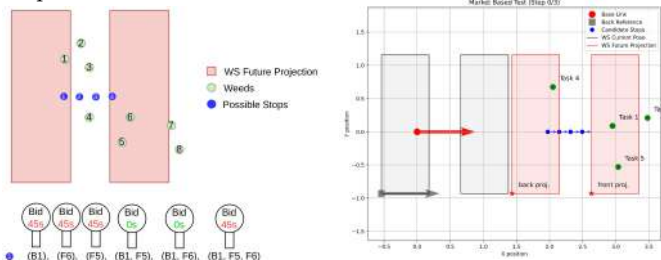
- **Optimization:** 1: Compute candidate stops. 2: Associate reachable weeds with candidate stops. 3: Build the optimization model using decision variables, constraints, and objective function. 4: Call solver to get solution. 5: Decode solution and return the next optimal stop and TA output.

Given $imb_i \geq \tau(f_i - b_i)$ and $imb_i \geq \tau(b_i - f_i)$

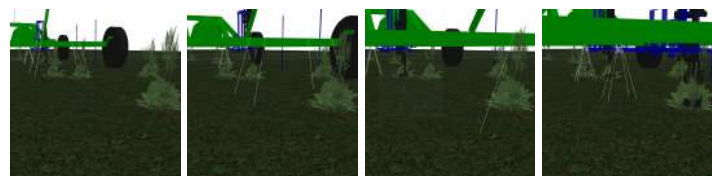
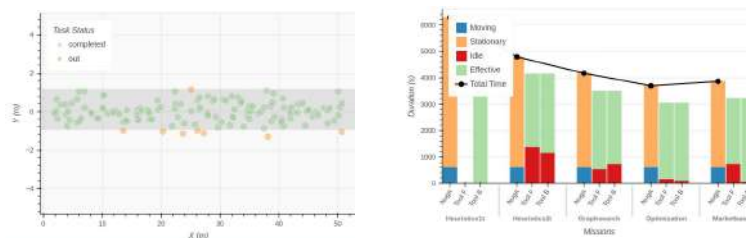
where $f_i = \sum_{j \in F_i} x_{i,j}$ and $b_i = \sum_{j \in B_i} x_{i,j}$  Google OR-Tools

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^S imb_i \quad \text{subject to} \quad \sum_{i=1}^S x_{i,j} = 1 \quad \forall j \in \mathcal{T} \\ & x_{i,j} = 0 \quad \text{if} \quad j \notin \mathcal{V}_i \end{aligned}$$

- **Market-based:** 1: Compute candidate stops. 2: Associate reachable weeds with candidate stops. 3: Create and collect all bids. 4: Select *best bid* (minimum tool idle time and maximum number of removable tasks). 5: Decode solution and return the next optimal stop and TA output.



Results



Conclusion

- **Comparison** of heuristic, graph search, optimization-based, and market-based TA algorithms was conducted using a **custom simulation framework** on the NUGA robotic platform.
- **Optimization-based** approach showed the best scalability and performance, reducing tool idle time by up to 94.9% in high-density weed scenarios.
- **Future research** can explore the impact of adding more tools, analyzing cost-productivity trade-offs, and improving global optimality through enhanced sensing.

Robust LiDAR-Inertial Localization with Prior Maps in GNSS-Challenged Environments

Author :Eliyas Kidanemariam Abraha

Supervisors: Dr.Zoltan Istenes & Dr.Mohammad Aldibaja

Motivation

Autonomous robots require accurate localization in GPS-denied environments like indoors or urban canyons. GNSS-INS systems are prone to failure in these conditions, while real-time SLAM often drift without loop closures.

Map-based localization offers a stable and accurate alternative, but it faces several key challenges:

- **Real-time performance and Scalability:** Handling high-resolution 3D maps and computing scan-to-map registration efficiently.
- **Drift correction:** Fusing local motion estimation with global map constraints while preserving consistency.
- **Dynamic environments:** Removing or mitigating the effect of moving objects during scan matching.
- Localization failures in feature-sparse or unmapped transition zones.

Contribution

This thesis presents a robust and real-time localization framework for GNSS-denied environments by fusing LiDAR-Inertial Odometry (FAST-LIO2) with multithreaded NDT-based map matching using a sliding-window factor graph. It introduces a scalable submap management strategy and integrates dynamic object removal via deep learning, enabling consistent pose estimation even in dynamic, degraded, or feature-sparse areas. The system achieves centimeter- to decimeter-level accuracy across diverse datasets, maintaining low-latency performance suitable for real-world autonomous navigation. Extensive evaluations show that the proposed method not only surpasses standalone odometry and SLAM baselines but also outperforms recent state-of-the-art map-based localization approaches in accuracy, robustness, and scalability.

Methodology

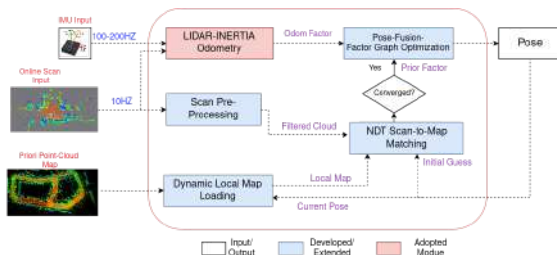
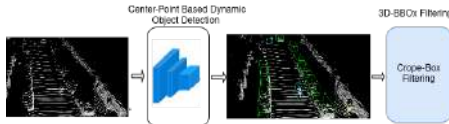


Figure 1: Complete Diagram of The Localization System

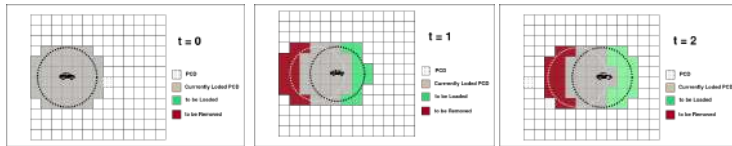
Scan Pre-Processing

Dynamic objects are removed from LiDAR scans using a deep learning-based 3D detector (CenterPoint) to improve scan-to-map alignment.



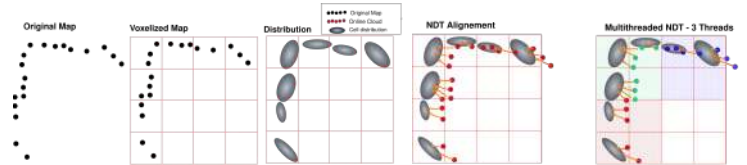
Local Map Loading

loads only relevant submaps based on the robot's estimated position



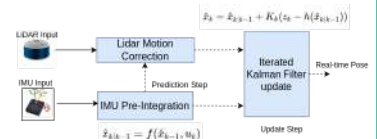
Scan-to-Map Matching

multithreaded implementation of the Normal Distributions Transform (NDT) used to accelerate scan-to-map matching



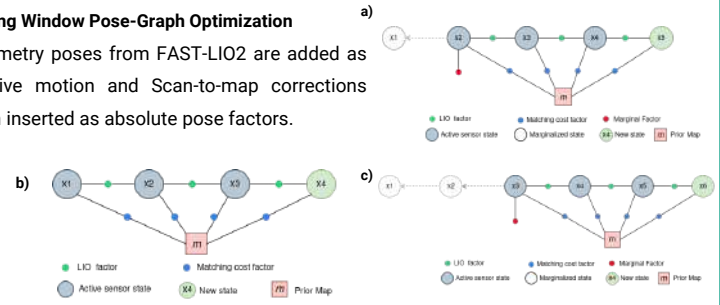
Local Motion Estimation

FAST-LIO2 based LiDAR-Inertial Odometry for real-time local pose estimation.



Sliding Window Pose-Graph Optimization

Odometry poses from FAST-LIO2 are added as relative motion and Scan-to-map corrections from inserted as absolute pose factors.



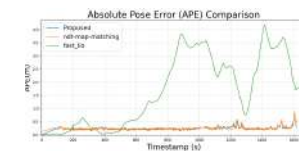
Experimental Results

Comparison with baseline methods: LIO exhibits high drift over time while proposed method achieves both low localization error and high temporal consistency.

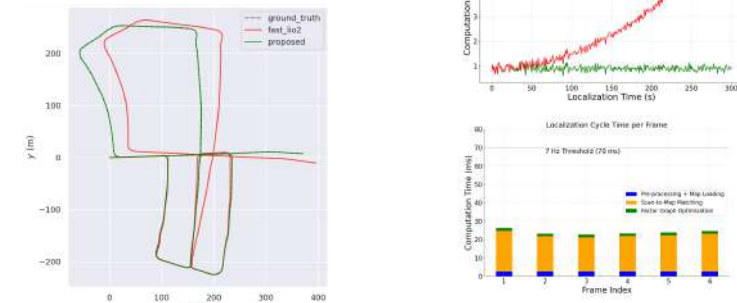
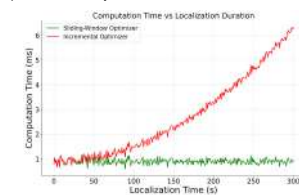
Method	Metric	Sequence 2				Sequence 3			
		Max	RMS	Mean	Std	Max	RMS	Mean	Std
Proposed	APE (m)	0.44	0.11	0.09 ± 0.08	0.02	0.14	0.10 ± 0.09		
	Rot. (deg)	4.30	1.05	0.75 ± 0.70	2.02	1.04	0.75 ± 0.74		
Map Matching	APE (m)	0.25	0.12	0.10 ± 0.06	0.30	0.33	0.11 ± 0.06		
	Rot. (deg)	2.64	1.39	1.39 ± 0.72	3.24	1.65	1.37 ± 0.92		
Fast-LIO2	APE (m)	6.09	3.88	3.56 ± 3.88	4.08	2.16	1.47 ± 1.15		
	Rot. (deg)	5.49	2.68	2.25 ± 1.46	2.75	1.09	0.96 ± 0.52		

Benchmarking on Public Dataset: tested on kitt05

Method	Map Correction	Translation Error (m)	Rotation Error (°)
Proposed (Ours)	✓	0.121 ± 0.077	0.30 ± 0.144
Yongqi Kim et al.	✓	0.15 ± 0.14	0.34 ± 0.40
D. Rozenberszki et al.	✓	~2.5 ± 2.0	
Xiaohu Liu et al.	✓	3.18 ± 5.58	1.27 ± 1.97
FAST-LIO2	✗	9.18 ± 3.58	5.2 ± 1.7



Real-time performance: 23ms (✓ 43Hz real-time) per frame latency



Conclusions and Future Work

Accurate & Drift-Free

Achieves **centimeter-to-sub-decimeter** accuracy by fusing FAST-LIO2 and NDT with a sliding-window factor graph, effectively reducing drift without loop closures.

Real-Time & Scalable

Maintains **<23 ms** latency using multithreaded NDT and dynamic submap loading. Sliding window factor graph optimization remains bounded regardless of trajectory length.

Robust to Challenges

Dynamic object removal improves convergence, and fused graph keeps **localization stable** even when scan matching fails.

Limitations & Future Work

Assumes a known initial pose and a static map. Future directions include global re-localization, adaptive map updating, and integration of camera/radar for increased robustness.

References

- [1] Xu, W. et al. "FAST-LIO2: Lightweight LiDAR-Inertial Odometry", IROS 2022
- [2] Rozenberszki, D. et al., LOL: LiDAR-only Odometry and Localization, ICRA 2020
- [3] Kim, Y. et al. "Stereo Camera Localization in 3D LiDAR Maps", IROS 2018
- [4] Lin, X. et al. Autonomous Vehicle Localization with Prior Visual Point Cloud Map Constraints in GNSS-Challenged Environments. Remote Sensing, 2021

LiDAR Odometry and Mapping Beyond RTK Accuracy

Liviu-Daniel Florescu^{1,2}, Iván Eichhardt¹, Maximilian Fenkart³

¹Eötvös Loránd University, Budapest; ²Universitat de Girona; ³Sodex Innovations GmbH, Austria

Contact: liviu.flrsc@gmail.com

1. Introduction

LiDAR sensors are a very popular modality for outdoor robotic systems. As they provide highly accurate 3D information, they can be used to generate faithful digital twins. In the case of the SDX-Compact (Fig. 1), the location and orientation readings from an RTK-corrected Inertial Navigation System (INS) are used to transform point clouds into the global frame and generate large 3D models. However, relying on a GPS signal of fluctuating accuracy and availability is not optimal, and can lead to incorrect mapping.

We propose a graph-based pose estimation method that uses LiDAR scans to compute displacement and improves map quality over RTK-only localization, thanks to point cloud registration constraints.

2. Dataset

Our approach was developed on a custom dataset collected in a rural environment (Fig. 2). The sensor rig was mounted on a vehicle and driven at up to 40 km/h. With our settings, the INS operates at 100Hz, while one LiDAR scan takes 100ms (10Hz). We perform scan de-skewing/motion compensation using the intermediate INS readings, on point batches, and store a single global pose per scan. The GPS readings have very low standard deviation (Fig. 3), indicating high measurement accuracy.



Figure 2: Example trajectory overlaid on Google Maps satellite image.

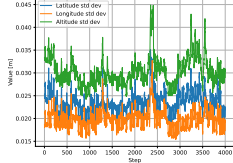


Figure 3: Standard deviation of GPS readings along the trajectory.

3. Methodology

Initial experiments adopted the KISS-ICP [1] architecture, thanks to its simplicity, and we introduced three significant improvements, suitable for our use case:

- Variable motion prediction, using interpolation in the Lie manifold. If t_k is the timestamp of scan k and we define $\alpha_{k+1} = (t_{k+1} - t_k) / (t_k - t_{k-1})$, then

$$\mathbf{T}'_{k+1} = \mathbf{T}_k \exp(\alpha_{k+1} \log(\mathbf{T}_{k-1}^{-1} \mathbf{T}_k)). \quad (1)$$
- Additional GPS constraints: the estimated pose \mathbf{T}_k should be a weighted combination of \mathbf{T}'_k , GPS input $\mathbf{T}_k^{\text{GPS}}$, and registration result $\mathbf{T}_k^{\text{REG}}$.
- Two-stage registration: ICP with an outlier-robust kernel and percentile-based correspondence filtering, on voxelized point clouds, and Generalized ICP [2] on denser point clouds, for improved surface alignment.

However, such an approach does not take into account corrections that should be applied to previous states, given a new GPS reading. To fix this, we use the Factor Graph [3] formulation, with two types of factors:

- Registration — the registration result is used as an odometry factor between consecutive poses and as

skip connections, to enforce scan alignment. This is sensible because the local map is constructed from the last 10 scans. Registration covariance is approximated based on the history of RMSE values.

- GPS — when a GPS reading is available, a unary factor is attached to the corresponding pose node, with the covariance indicated by the sensor.

This is visualized in Fig. 4, while Fig. 5 presents the architecture of our complete solution.

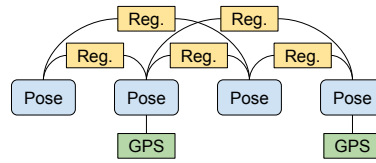


Figure 4: Factor Graph structure. We use the GICP registration result to create odometry factors (yellow), but we also add such factors between non-consecutive poses, as they are involved in the registration process. GPS readings introduce unary factors (green).

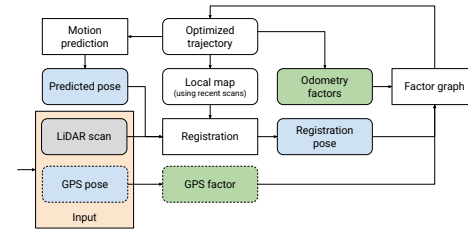


Figure 5: Solution architecture and workflow. We predict the next pose using Lie Algebra interpolation, perform registration on the LiDAR data, then combine this with GPS constraints inside the Factor Graph.

4. Results

First, we assess the odometry capabilities of our method on the custom dataset (Fig. 6, 7) and on KITTI [4] sequences (Table 1, Fig. 8). Next, we evaluate the behaviour of the method in the presence of GPS noise (Fig. 9), and with gaps in the GPS data (Table 2). Finally, we observe the map improvements over GPS-only reconstruction.

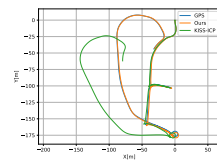


Figure 6: The complete trajectory, spanning approx. 645m. KISS-ICP diverges when a significant time jump occurs.

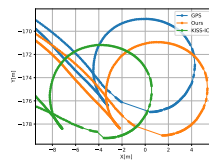


Figure 7: A region with gaps in scan data. Our method can handle irregular time intervals between scans.

Seq.	Length (m)	ATE			Final			Avg. RTE (100m)		
		XY (m)	tra. (m)	rot. (rad)	XY (m)	tra. (m)	rot. (rad)	XY (%)	tra. (%)	rot. (%)
02	5067.02	22.20	50.39	0.13	49.72	99.53	0.22	0.70	1.22	11.99
00	3723.24	7.29	16.28	0.07	10.59	11.58	0.06	0.81	1.23	12.88
08	3222.02	16.73	28.33	0.08	20.90	33.15	0.11	1.04	1.52	24.66
01	2453.26	24.53	190.10	0.21	39.90	291.51	0.30	0.98	1.31	45.70
05	2205.20	4.62	6.92	0.04	9.20	13.79	0.07	0.50	1.02	20.44

Table 1: Odometry evaluation on KITTI sequences. Reporting only the five longest trajectories that have associated ground truth.



Figure 1: SDX-Compact, a versatile sensor rig for accurate outdoor surveying. It includes one Hesai Pandar XT32 LiDAR, one Septentrio INS with RTK, and three ArkCam RGB cameras.

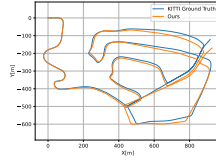


Figure 8: Odometry result on KITTI Sequence 02.

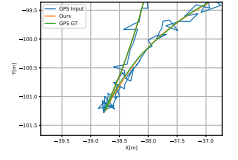


Figure 9: Trajectory result when using noisy GPS input.

GPS Skip Factor	GPS Count	ATE			Final		
		XY (m)	tra. (m)	rot. (rad)	XY (m)	tra. (m)	rot. (rad)
—	1000	0.003	0.026	0.001	0.005	0.040	0.001
5	200	0.009	0.021	0.001	0.008	0.037	0.002
10	100	0.012	0.016	0.002	0.010	0.027	0.002
50	20	0.022	0.025	0.005	0.026	0.028	0.007
100	10	0.027	0.034	0.007	0.050	0.055	0.012
334	3	0.070	0.121	0.013	0.036	0.146	0.019
500	2	0.107	0.445	0.019	0.107	0.741	0.028

Table 2: Evaluation with reduced GPS frequency, on 1000 LiDAR scans. We obtain small trajectory errors even when a fraction of the readings are used.

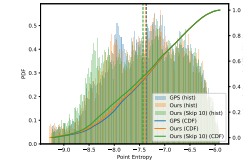


Figure 10: Point entropy evaluation. Our method decreases point entropy, compared to a GPS-only map.

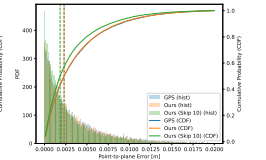


Figure 11: Point-to-plane distance evaluation. Our method improves surface quality, compared to the GPS-only map.

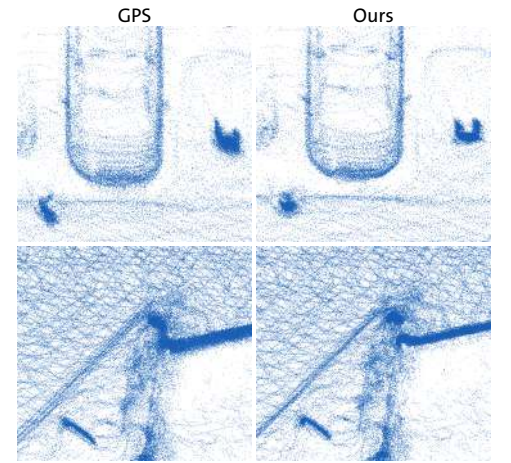


Figure 12: Map comparison. We obtain sharper contours and details, indicating better scan alignment, without damaging localization accuracy.

5. Conclusions

LiDAR data can be used for accurate displacement estimation and accurate 3D mapping, but GPS data is essential to prevent localization drift. Future work could involve an improved scan de-skewing step, a faster implementation in C++, or using the LiDAR intensity values during registration.

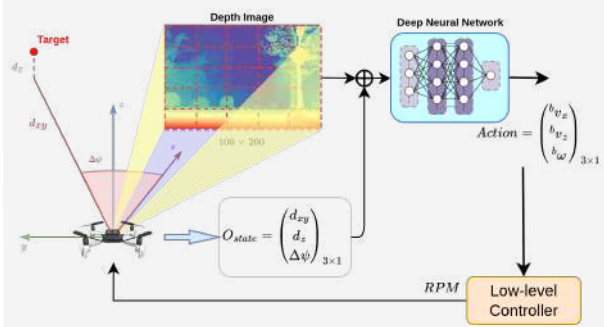
References

- [1] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss. KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way. *IEEE Robotics and Automation Letters (RA-L)*, 8(2):1029–1036, 2023.
- [2] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-ICP. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009.
- [3] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. *Georgia Institute of Technology, Tech. Rep.*, 2(4), 2012.
- [4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237, 2013.

Abstract

This research investigates an **end-to-end deep reinforcement learning (DRL) framework** for drone obstacle avoidance using onboard depth sensing in simulation. The proposed approach proposes a neural architecture that incorporates both a **pre-trained ResNet8-based depth encoder** and two temporal reasoning mechanisms: an **LSTM module for recurrent memory** and a **stacked buffer of recent depth observations** that allows the agent to recover from occlusions and partial observability. The framework is developed in **Gym-PyBullet-Drones** environment with **Stable Baseline 3** library.

Problem Formulation

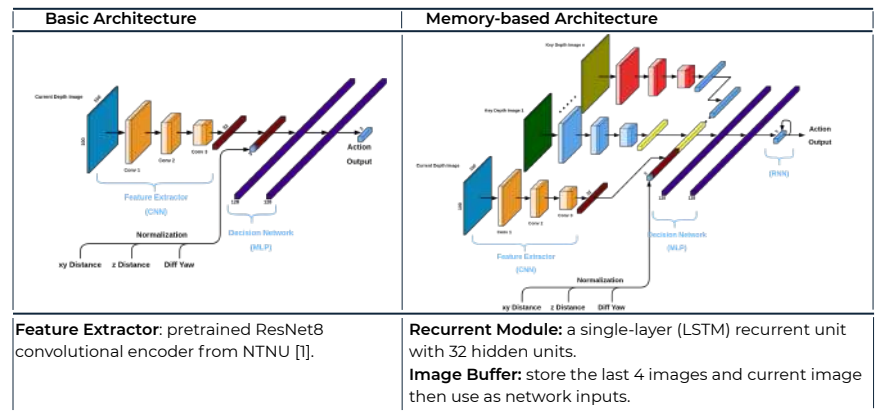


Drone model: Crazyflies 2.1
Low-level Controller: PID

Contributions

- **Unified DRL-based policy:** directly maps raw depth images and goal information to high-level UAV control commands, eliminating the need for explicit mapping, path planning, or trajectory optimization modules.
- **Temporal visual encoding:** a neural architecture that fuses latent features from a sequence of past depth images that maintains spatial awareness over time, particularly critical for avoiding dead-end obstacles and handling occluded threats like overhead collisions.
- **Recurrent memory integration:** an Long short-term memory (LSTM) module, allowing it to learn long-term dependencies and internal representations of the environment.
- **Comprehensive evaluation and comparison:** comparing two DRL algorithms—Proximal Policy Optimization (PPO) and Twin Delayed DDPG (TD3); Basic architecture and Memory-based architecture; and Benchmark our final model against EGO-Planner-v2.

Network Architecture



Feature Extractor: pretrained ResNet8 convolutional encoder from NTNU [1].

Recurrent Module: a single-layer (LSTM) recurrent unit with 32 hidden units.
Image Buffer: store the last 4 images and current image then use as network inputs.

Experiment Setup

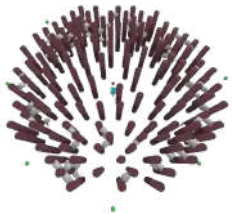


Figure 2: Start and Goal Placement

Starting Point: center of the map
Goal: random around the map
Task: do obstacle avoidance and reach the goal

Reward Function



Figure 3: Reward Function Design

The reward function that integrates both sparse and continuous components, with the weights of these components linearly varied as the curriculum level increases.

Curriculum Learning

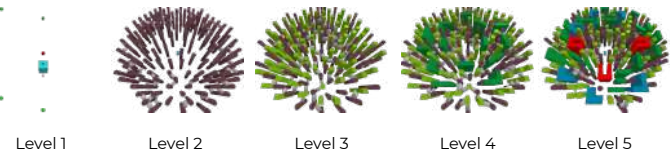
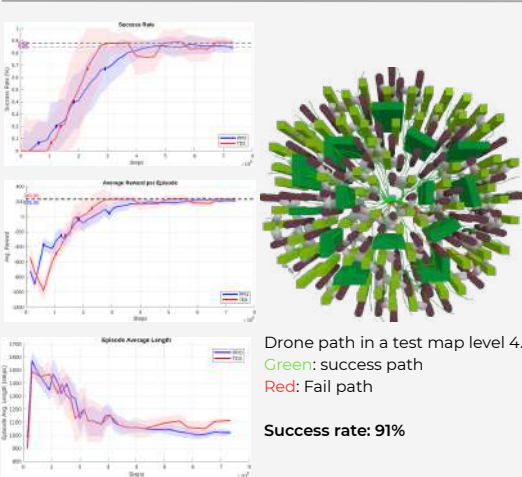


Figure 4: Curriculum Learning

Switch to a higher level map once a certain success rate achieved (90%, 80%, 75%, 70%)
Come back the lower level map if the rate below a given threshold (5%).

PPO - TD3 Comparison



Drone path in a test map level 4.
Green: success path
Red: Fail path

Success rate: 91%

Figure 5: Training and results of the basic architecture with two DRL algorithms.

Memory-based Training

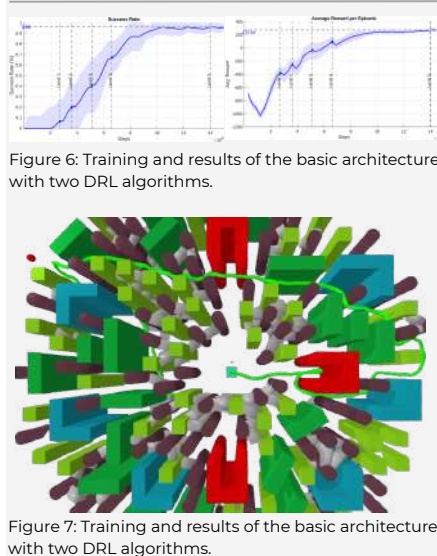
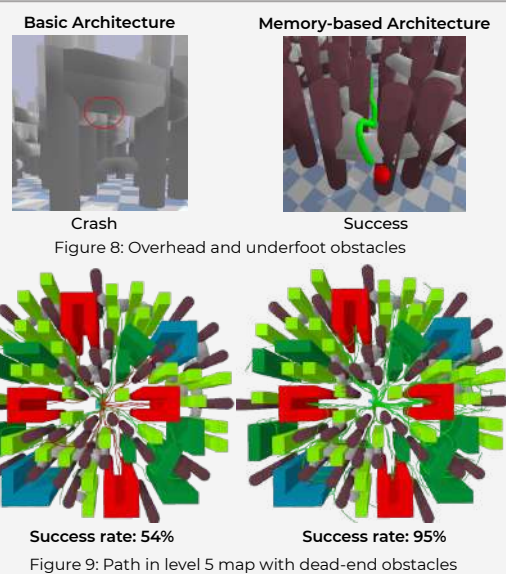


Figure 6: Training and results of the basic architecture with two DRL algorithms.

Figure 7: Training and results of the basic architecture with two DRL algorithms.

Architecture Comparison



Crash

Success

Figure 8: Overhead and underfoot obstacles

Success rate: 54%

Success rate: 95%

Figure 9: Path in level 5 map with dead-end obstacles

Methodology Comparison

Environment	Algorithm	AER	AEL	SR
Training Env	Memory-based DRL	235.62	1125.0	96.667%
	EGO-Planner-v2	210.65	1280.0	91.333%
Testing Env	Memory-based DRL	218.94	1090.0	86.667%
	EGO-Planner-v2	205.86	1350.0	90.0%

Table 1: Comparison between DRL and a conventional method

Conclusion & Future Work

This work shows that end-to-end deep reinforcement learning (DRL) with temporal memory enables UAVs to robustly avoid obstacles in complex environments, outperforming classical planners in simulation. Temporal features, shaped rewards, and curriculum training are critical for generalization and real-world readiness.
Key limitations remain—especially explainability, sim-to-real transfer, and multi-agent scalability. Closing these gaps, with advances in robust adaptation and interpretability, is essential for safe deployment in real-world UAV applications.



Deep Reinforcement Learning for Robot Manipulation

Vania Katherine Mulia

Supervisors: Narcís Palomeras, Tamara Petrović

PROBLEM STATEMENT

This work addresses the problem of enabling a robotic manipulator to **autonomously learn the peg-in-hole insertion task using Deep Reinforcement Learning (DRL)**. This work also investigates the feasibility and challenges of **simulation-to-reality (sim2real) transfer** of a policy trained using DRL in a physics-based simulated environment.

DRL FRAMEWORK

- **Task Description:** peg-in-hole with tolerance ~3 mm, hole depth 60 mm

- **Algorithm:** Soft Actor-Critic (SAC)

- **Observation Space:** end-effector force and torque, position and orientation of the peg and hole, and the joint positions.

$$o = [F_x, F_y, F_z, M_x, M_y, M_z, \mathbf{x}_p^R, \mathbf{x}_h^R, \mathbf{q}]$$

- **Action Space:** end-effector velocity in the end-effector frame

$$a = [v_x, v_y, v_z, \omega_x, \omega_y]$$

- **Reward Function:**

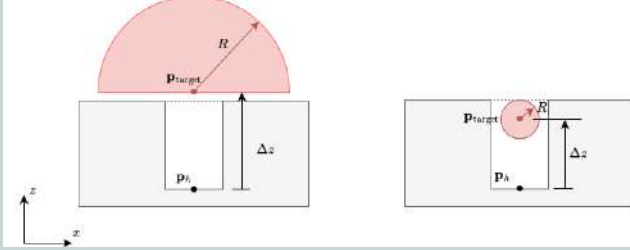
$$r(s, a) = \begin{cases} 100, & \text{if the task is successfully completed,} \\ -20, & \text{if a safety violation occurs,} \\ 1.5 r_{\text{distance}} + r_{\text{force}}, & \text{otherwise.} \end{cases}$$

The term r_{distance} represents a penalty based on the distance between the peg and the hole, while r_{force} represents a penalty for excessive contact forces and torques at the end-effector.

- **Domain Randomization:** randomize hole position based on randomization workspace.

- **Curriculum Implementation:**

- Stages 1-7 aim to guide the agent to move the peg towards the hole with the appropriate trajectory.

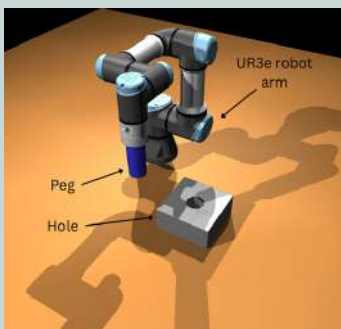


Schematic of Task Definition (left: stages 1-2, right: stages 3-7)

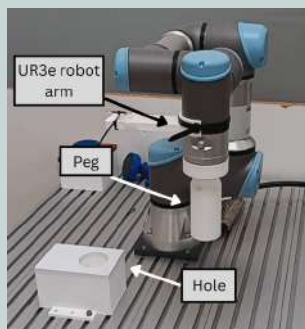
Stage	Vertical Offset from Hole Δz (mm)	Range Threshold R (mm)
1	70	90
2	70	50
3	70	10
4	55	10
5	45	10
6	30	10
7	0	10

- Stage 8 builds on stage 7, with an addition of noise to the observed hole position and force penalization.

ENVIRONMENT SETUP



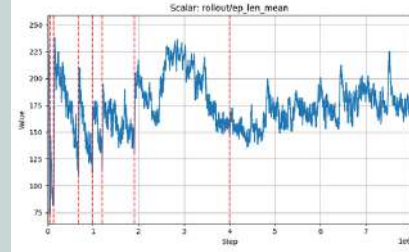
Simulation Setup



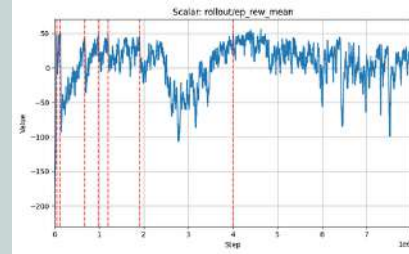
Real-World Setup

SIMULATION RESULTS

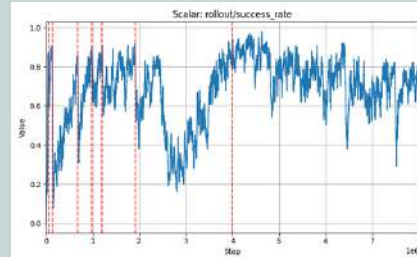
Training metrics



Average Episode Length

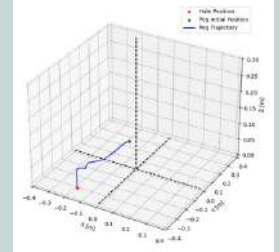


Average Episode Reward

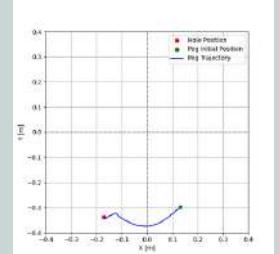


Success Rate

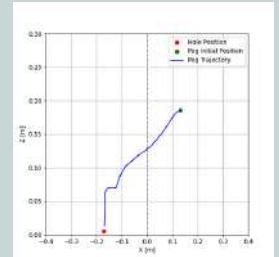
Test Results (Trajectory)



3D Projection



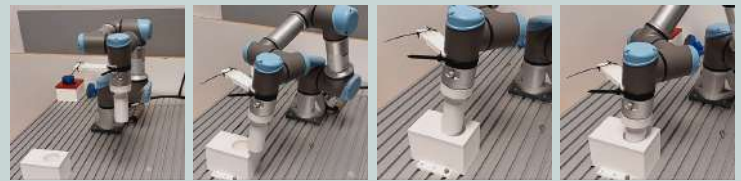
XY Projection



XZ Projection

- Stage 7 policy achieves an **average success rate of 0.808** across five seeds, with an **average episode length of 155.658 timesteps** (corresponding to 3.9 seconds of execution time) and **average episode reward of 32.41**.
- Final policy **reduces the maximum contact force by around 8-12%**, but also **reduces the success rate from ~0.8 to ~0.65**, suggesting partial success in encouraging compliant contact-based search.

SIM-TO-REAL RESULTS



Sim-to-real gaps found:

- Robot speed needs to be scaled down (~10%)
- Action frequency needs to be reduced (40 Hz in simulation → 10 Hz in real robot)
- The limit of episode length cannot be applied in the real robot.
- Force-torque sensor readings sign mismatch.
- Magnitude of sensed contact forces have different scale.

SUGGESTIONS FOR FUTURE WORKS

- Refining the curriculum to balance between success rate and compliant behavior.
- Exploring hyperparameter tuning strategies.
- Improving the fidelity of simulation to support more robust real-world performance.

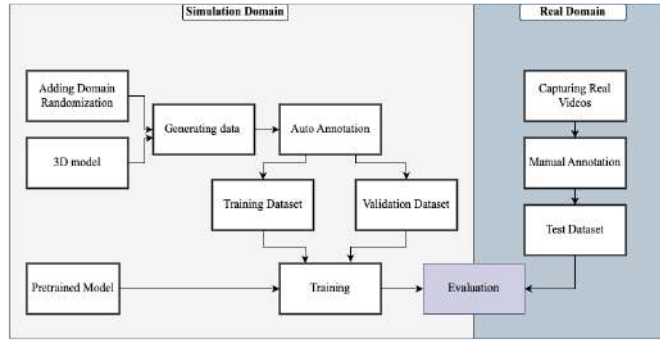
Introduction

- Multi-Object Tracking (MOT):** Detects and tracks multiple objects across frames. Assigns unique IDs and keeps them consistent over time.
- Object tracking:** Key for navigation, manipulation, Human-Robot Interaction and so on.
- Real-world data:** Limited, costly to collect & label.
- Synthetic simulation:** Fast, large-scale annotated datasets.
- Domain randomization:** Vary textures, lighting etc. for robust transfer.

Contribution

- CAD-to-Data Generator:** Converts CAD models into 7K+ labeled frames in minutes, no manual labeling.
- Randomization Study:** First in-depth ablation of photometric & spatial randomizers for sim2real MOT.
- Dual-Pipeline Benchmark:** Comparison of detection vs. segmentation tracking.
- Public MOT Dataset:** Released 2.5K+ real frames with ~23K annotations real world dataset for standardized evaluation.

Sim2Real Overview for MOT



Domain Randomization Techniques



Dataset

Video	Frames	Objects	Annotations	Classes
Video 1	1,426	40	13,959	5
Video 2	1,166	58	8,695	5



Partial Objects Motion Blur Dynamic Lighting

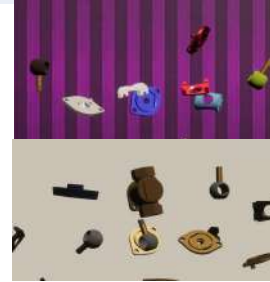
3D CAD Model at Left and Real Object at Right

Synthetic Dataset Variants and Randomization Composition

DATASET	PLACE	ROT.	TEX. BG	LIGHT	OVERLAP	COLOR	HUE
D1	✓	✓	✓	✓	5		
D2	✓	✓	✓	✓	2	✓	✓
D3	✓	✓	✓	✓	2		
D4	✓	✓	✓	✓			
D5	✓	✓			2		
D6	✓	✓	✓		2		
D7		✓	✓	✓			



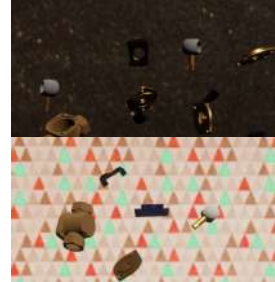
D1 (Top) and D4 (Bottom)



D2 (Top) and D5 (Bottom)



D3



D6 (Top) and D7 (Bottom)

Detection-Based Tracking

Configuration (D)	MOTA ↑	MOTP ↓	IDF1 ↑	Precision ↑	Recall ↑	ID Switches ↓
D1	0.704 ± 0.4	0.319 ± 0.2	0.650 ± 0.5	0.930 ± 0.2	0.764 ± 0.3	29 ± 2
D2	0.823 ± 0.3	0.307 ± 0.2	0.718 ± 0.4	0.939 ± 0.2	0.884 ± 0.1	31 ± 1
D3	0.824 ± 0.2	0.309 ± 0.2	0.698 ± 0.5	0.953 ± 0.1	0.871 ± 0.2	34 ± 2
D4	0.730 ± 0.3	0.310 ± 0.1	0.673 ± 0.6	0.857 ± 0.2	0.888 ± 0.2	82 ± 4
D5	-0.058 ± 0.5	0.325 ± 0.2	0.359 ± 0.6	0.478 ± 0.4	0.540 ± 0.3	76 ± 3
D6	0.734 ± 0.2	0.315 ± 0.1	0.661 ± 0.4	0.877 ± 0.2	0.858 ± 0.2	45 ± 2
D7	0.662 ± 0.3	0.320 ± 0.2	0.601 ± 0.5	0.832 ± 0.3	0.745 ± 0.3	88 ± 3

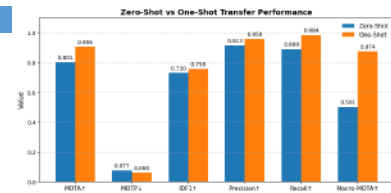
Class	Zero-Shot	One-Shot	Δ (%)
Handle	0.592	0.855	+26.3%
Ball	0.799	0.926	+12.8%
Body	0.596	0.920	+32.4%
Bonnet	0.725	0.952	+22.7%
Seat	0.712	0.912	+19.9%
Macro-MOTA	0.685	0.913	+22.8%



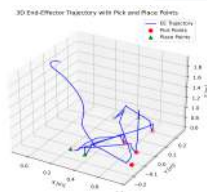
Segmentation-Based Tracking

Configuration(D)	MOTA ↑	MOTP ↓	IDF1 ↑	Precision ↑	Recall ↑	ID Switches ↓
D1	0.821 ± 0.3	0.073 ± 0.4	0.768 ± 0.5	0.909 ± 0.2	0.918 ± 0.3	46
D2	0.788 ± 0.4	0.090 ± 0.3	0.744 ± 0.4	0.862 ± 0.2	0.946 ± 0.4	54
D3	0.659 ± 0.5	0.084 ± 0.2	0.627 ± 0.3	0.949 ± 0.5	0.703 ± 0.2	50
D5	-0.563 ± 0.4	0.253 ± 0.5	0.244 ± 0.4	0.306 ± 0.3	0.427 ± 0.5	171
D6	0.755 ± 0.2	0.101 ± 0.3	0.727 ± 0.2	0.868 ± 0.4	0.896 ± 0.3	41

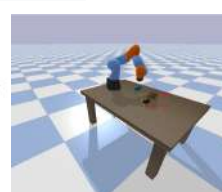
Class	Zero-Shot (ZS)	One-Shot (OS)	Δ (%)
Handle	0.6649	0.8076	+14.27%
Ball	0.9211	0.9723	+5.12%
Body	0.0069	0.7887	+78.18%
Bonnet	0.0907	0.8847	+79.39%
Seat	0.8188	0.9137	+9.49%
Macro-MOTA	0.5010	0.8734	+37.24%



Application



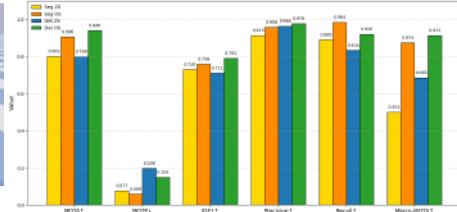
End Effector Trajectory



Pick & Place Operation

Comparison

Detection-Based vs Segmentation-Based Tracking



Conclusion

A fully synthetic pipeline is shown to transfer effectively to real robotic scenes with minimal manual labeling. Using domain randomization, ~7K annotated frames are generated within minutes. Zero-shot tracking achieves 0.799 MOTA on detection-based tracking, improving to 0.940 MOTA with just one real world frame. Ablation studies identify lighting and background textures as key photometric randomizers, and limited two-object occlusion yields a 6–8% MOTA gain. Bounding-box tracking is 1.5× faster and more accurate, while mask-based tracking offers finer contours for pixel-level tasks.



FIRE DETECTION AND RANGING FOR AUTONOMOUS FIREFIGHTING DRONES



Introduction

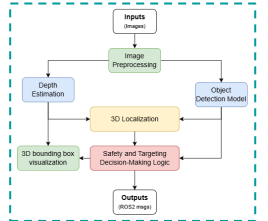
Wildfires and industrial fires pose escalating risks to lives, infrastructure, and the environment. Autonomous aerial systems offer a promising solution by enabling rapid fire detection and response in hazardous or inaccessible areas. This work presents a real-time Fire Detection and Ranging (FDAR) framework designed for deployment on UAVs. The system integrates RGB and thermal imagery, advanced object detection, and depth estimation to accurately identify fire, smoke, and human presence, while ensuring safe and intelligent suppression decisions in dynamic environments.



WIRIS Enterprise Camera for Firefighting UAV Drones

Methodology

The proposed FDAR system is built around a modular architecture designed for real-time operation on autonomous UAVs. It fuses RGB and thermal data, detects critical fire scene elements, estimates spatial depth, and determines safe suppression decisions using a robust decision logic. Each module was extensively evaluated for performance, reliability, and deployability in complex fire scenarios.



- Identifies fire, smoke, and humans using deep learning models
- Trained on RGB, thermal, and RGB-T fused datasets
- Evaluates performance trade-offs across YOLO variants models



- RGB-Thermal stereo calibration and monocular inference
- Monocular depth estimation for accurate 3D
- Enables fire and human distance computation for targeting logic



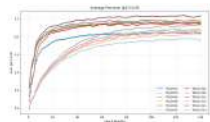
- Classical CV and segmentation refine fire localization
- Safety logic prevents suppression near humans
- Planned integration of human state tracker (conscious/unconscious)



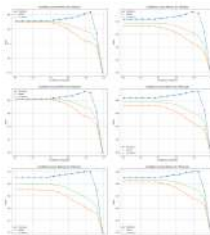
Results & Discussion

Detection Module

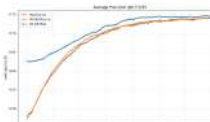
YOLO models from v5 to v12, including transformer-based variants, were evaluated across RGB, thermal, and RGB-T inputs. Lightweight models suit edge deployment, while larger models offer higher accuracy. RGB-T consistently boosted detection performance.



Evaluation of Lightweight Detection Models (Nano and Small)

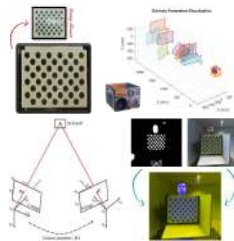


Evaluation of Medium and Transformer-Based Models



RGB-T Fusion

A custom thermal-RGB calibration board enabled accurate image alignment. RGB-T fusion improved detection in low-visibility and cluttered scenes, outperforming single modalities in both precision and recall.



RGB-T Calibration Pipeline

Training convergence of YOLOv12s across different input modalities.



Evaluation of RGB-T Fusion and Single Modalities

Depth Estimation

Monocular depth models were benchmarked due to limitations in RGB-thermal stereo. DepthAnything and ZoeDepth provided robust 3D localization of fire scenes, supporting safe and accurate suppression.



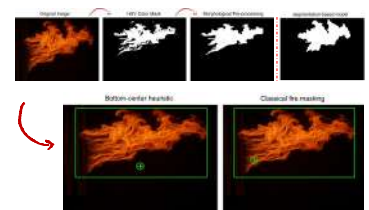
Models understanding for fire scenes



ZoeDepth-based metric depth estimation

Targeting & Safety

Fire source points are estimated using CV techniques on RGB or thermal inputs. A rule-based logic ensures suppression only occurs if no humans are nearby—enhancing operational safety.



RGB - Fire Source Estimation



Safety rules and constraints logic



Conclusion

This work presents a modular Fire Detection and Ranging (FDAR) system for autonomous UAV-based firefighting. Through RGB-thermal (RGB-T) fusion and deep learning models, the system achieves accurate detection, robust 3D localization, and safe fire suppression. RGB-T fusion significantly improves both precision and recall compared to single modalities, validating its effectiveness in complex and low-visibility environments.

More information can be found on our website:

<https://www.saxion.nl/onderzoek/lectoraten/smart-mechatronics-and-robotics>

Future work

Future work will focus on integrating the FDAR system with autonomous UAV navigation, optimizing models for onboard deployment, and enhancing RGB-thermal fusion. Additionally, a state-aware human tracker will be developed to report the number and condition (conscious/unconscious) of individuals. Further testing in diverse environments and the use of temporal models for early fire prediction are also planned.

Acknowledgements

Special thanks to: ir. Benjamin van Manen & Prof. Matko Orsag

Problem Statement

Low-flying drones face a major challenge in detecting “invisible” obstacles such as power lines. Overhead power lines are among the most difficult obstacles for UAVs to detect due to:

Slender Geometry

They have a long, slender geometry and occupy a tiny visual footprint in the image

Low Reflectivity

Power lines have low reflectivity, making them difficult to detect visually.

Variable Appearance

The appearance of power lines varies greatly under different lighting conditions.

Minimal Contrast

Power lines have minimal visual contrast against complex backgrounds like trees or the sky.

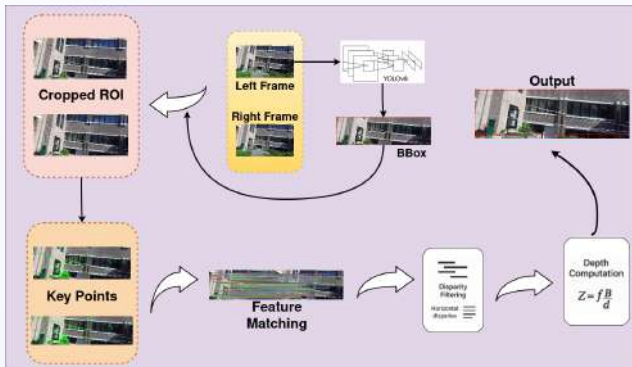
Research Approach

To tackle this challenge this study explores two approaches:

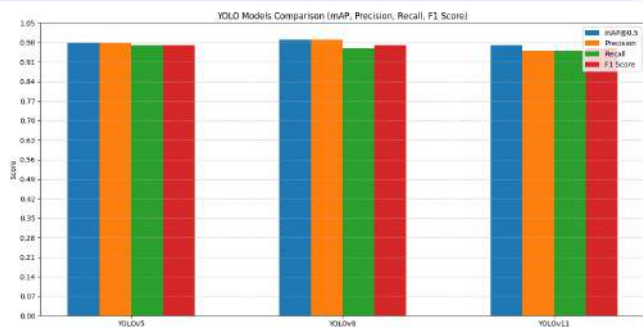
1. The first method involves developing an object detection and ranging pipeline using deep learning based **object detectors** in conjunction with **stereo vision**.
2. The second approach involves segmenting power lines employing **segmentation models**, in conjunction with **monocular vision**.

Power Line Detection & Ranging via Object Detection and Stereo Vision

Three YOLO models, YOLOv5, YOLOv8 and YOLOv11 were trained and evaluated. Among the 3 trained YOLO models, we selected YOLOv8 for deployment due to its superior balance between inference speed and detection accuracy.



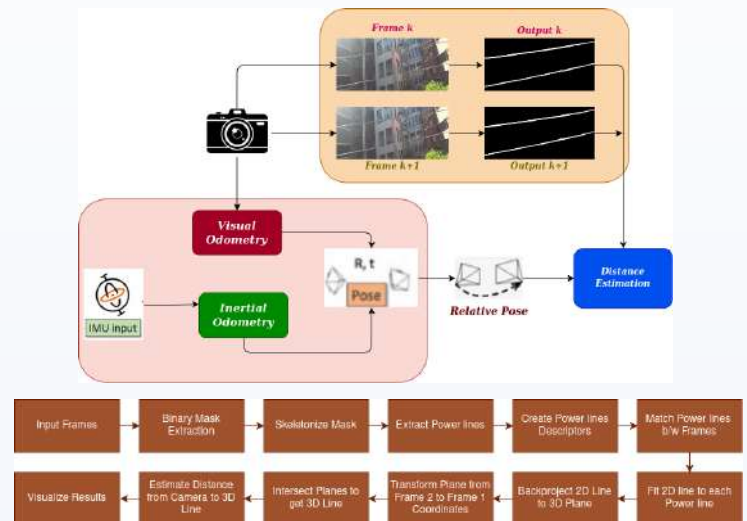
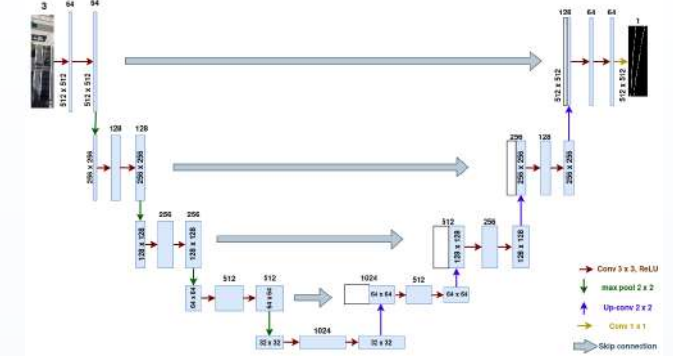
Results



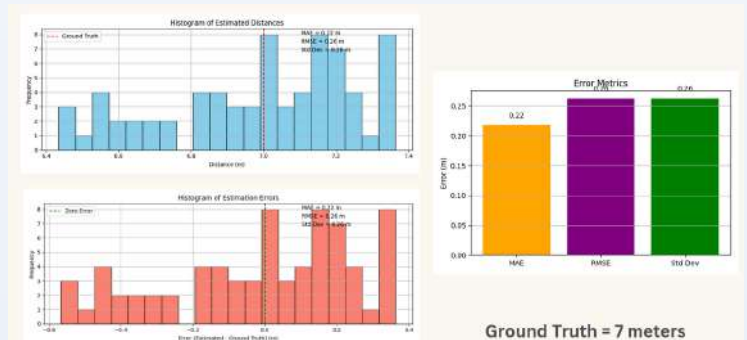
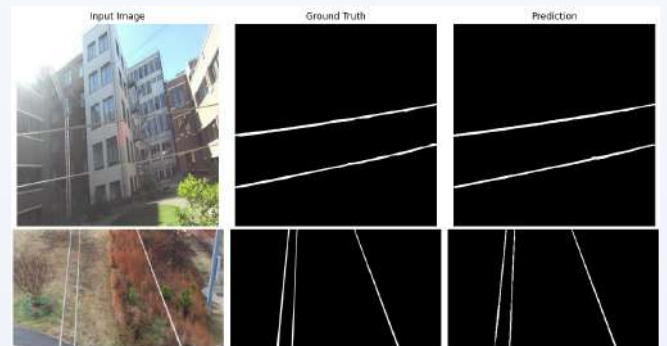
Conclusion

This thesis presented and evaluated two approaches for detecting and ranging thin obstacles like power lines. The segmentation-based monocular vision approach showed promising results, with distance estimates closely matching ground truth. With further refinement, this method holds strong potential for real-world UAV obstacle avoidance applications.

Power Line Detection & Ranging via Segmentation and Monocular Vision



Results



sawera.yaseen1@gmail.com



Scan me to view results!

Urban Object Detection using Sensor Fusion for Autonomous Navigation

Author: Selin Yavuz

Supervisors: Dániel Horváth, Fernando García

[Eötvös Loránd University, Hungary and AMPL Lab at UC3M, Spain]

Introduction

Autonomous vehicles must detect and localize static objects, such as poles, for safe navigation. However, their thin structure, small size and frequent occlusion by trees, vehicles and urban clutter make them particularly difficult to detect.

Contributions

- A 2D detector specialized for pole-like structures, fine-tuned using the YOLOv11[1] architecture.
- A 3D localization pipeline that projects 2D detections onto LiDAR point clouds.
- An approach combining segmentation-derived labels with geometric modeling as a lightweight alternative to supervised 3D inference.

Methodology

A two-stage detection pipeline is presented, combining 2D image-based detection with 3D LiDAR-based localization, without requiring 3D-labeled training data.

Stage 1: 2D Detection

- The pretrained model is fine-tuned on the A2D2 dataset containing pole structures.
- SAHI (Slicing Aided Hyper Inference)[2] is applied to improve recall for small objects.
- The model is applied to the custom dataset, yielding 2D bounding boxes as output.



Stage 2: Projection to 3D

- 2D detections are projected into 3D space.
- Associated LiDAR points are filtered, and RANSAC is applied for outlier rejection.
- After applying geometric constraints, the final output consists of 3D bounding boxes.



Results

2D Detection Results

Metric	Model	Model + SAHI
Precision	0.6333	0.7673
Recall	0.5138	0.8742
F1 Score	0.5673	0.8173
IoU	0.7789	0.7411
mAP@50	0.4509	0.4826
Inference Time (s)	0.0768	1.6146

3D Detection Results

Metric	Value
True Positives (TP)	580
False Positives (FP)	11
False Negatives (FN)	169
Precision	0.981
Recall	0.774
F1 Score	0.866

Visual Results

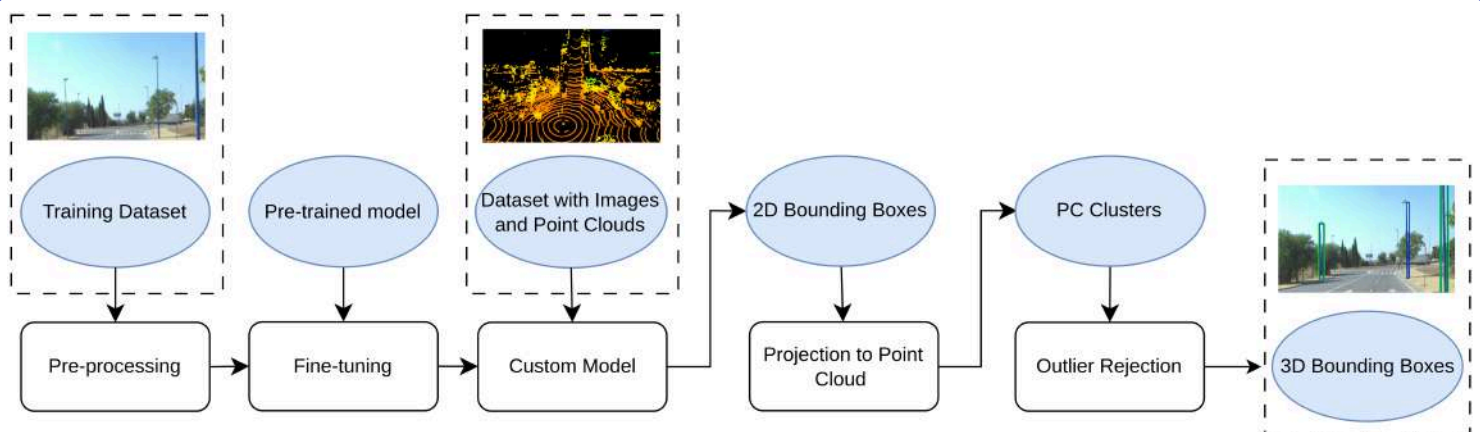


Output Video



References

- [1] YOLOv11 by Ultralytics: <https://github.com/ultralytics/ultralytics>
[2] SAHI: Slicing Aided Hyper Inference: <https://github.com/obss/sahi>



Problem Statement

Challenges in Sparse-Reward, Continuous Action-State Spaces

Inefficient Exploration

Low Sample Efficiency

Training Instability

Contribution

Incremental Graded Curriculum (IGC)

Gradually expands task complexity based on agent performance.

Dual Buffer Mechanism (DB)

Separates positive and negative experiences for balanced replay.

Dual-Mechanism Incremental Curriculum Learning (DICL)

Integrates IGC & DB in a unified pipeline. Optional: HER & PER for advanced replay.

Fixed vs Adaptive Curriculum Evaluation

Comparison of fixed vs. adaptive curricula on robotic benchmarks.

Methodology

Incremental Graded Curriculum (IGC)

Updates the task difficulty range for both the initial state and the goal independently, transitioning from narrowly aligned to widely separated pairs.

Objects and goals are sampled from uniform distributions controlled by dynamically adjusted range factors.

$$o_i = u_{obj} + \delta_i^{(obj)}, \quad \delta_i^{(obj)} \sim \mathcal{U}(-\rho_{obj}, \rho_{obj}), \quad i = 1, \dots, M$$

$$g_j = u_{goal} + \delta_j^{(goal)}, \quad \delta_j^{(goal)} \sim \mathcal{U}(-\rho_{goal}, \rho_{goal}), \quad j = 1, \dots, K$$

$$r(n) = \begin{cases} r_{narrow}, & \text{if } n = 0 \\ r_{inter}, & \text{if } 0 < n < 1 \\ r_{wide}, & \text{if } n = 1 \end{cases}$$

Normalized curriculum progress coefficient

Fig 1: Sampling region expansion in IGC

Dual Buffer Mechanism (DB)

\mathcal{B}_{pos} : Contains transitions from successful episodes that satisfy the success condition $R \geq \vartheta_{pos}$.

\mathcal{B}_{neg} : Contains transitions from failed episodes that satisfy the failure condition $R \leq \vartheta_{neg}$.

Dual-Mechanism Incremental Curriculum Learning (DICL)

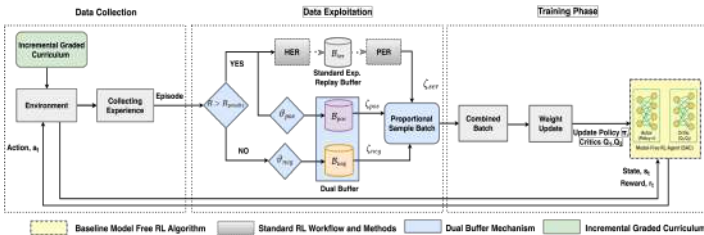


Fig 2: A schematic of the Dual-Mechanism Incremental Curriculum Learning (DICL) framework

Experimental Setup



Fig 3: Experimental setup of Gymnasium Fetch benchmark tasks: FetchPush, FetchPickAndPlace, and FetchSlide.

Aggregated results across all tasks

Method	Pick-and-Place Success	Mean	StdDev	Median	OG ↓
Baseline+HER	✓	0.44	0.15	0.44	0.56
Baseline+HER+PER	✓	0.50	0.36	0.50	0.50
Baseline+HER+PER	✓	0.06	0.06	0.06	0.94
Baseline+Penalty+HER	✓	0.37	0.15	0.37	0.63
Baseline+Penalty+HER	✓	0.10	0.12	0.10	0.90
Baseline+Penalty+HER	✓	0.06	0.06	0.06	0.94
Baseline+Penalty+HER+PER	✓	0.06	0.06	0.06	0.91
Baseline+Penalty+HER+PER	✓	0.05	0.01	0.05	0.95
Baseline+Penalty+HER+PER	✓	0.10	0.00	0.10	0.90
Baseline+Penalty+HER+PER	✓	0.10	0.12	0.10	0.90
Baseline+Penalty+HER+PER	✓	0.06	0.00	0.06	0.95
Baseline+Penalty+HER+PER	✓	0.10	0.06	0.10	0.90
Baseline+Penalty+HER+PER	✓	0.06	0.00	0.06	0.94
HER+HER	✓	0.76	0.19	0.76	0.24
HER+HER	✓	0.52	0.08	0.52	0.07
HER	✓	0.75	0.03	0.75	0.25
IGC+HER+PER (Fully Adaptive)	✓	1.00	0.00	1.00	0.00
IGC+HER (Fully Adaptive)	✓	1.00	0.00	1.00	0.00
IGC+HER (Fully Adaptive)	✓	0.98	0.01	0.98	0.02
IGC+HER	✓	0.99	0.03	0.99	0.01
IGC+PER	✓	0.10	0.05	0.10	0.90
IGC+PER	✓	0.10	0.06	0.10	0.90
IGC+PER	✓	0.14	0.12	0.14	0.86
DB+HER (Fixed ζ)	✓	1.00	0.00	1.00	0.00
DB+HER (Fixed ζ)	✓	1.00	0.00	1.00	0.00
DB+HER (Fixed ζ)	✓	0.73	0.06	0.73	0.27
DB+HER (Hybrid Adaptive ζ)	✓	0.75	0.05	0.75	0.25
DICL+HER	✓	1.00	0.00	1.00	0.00
DICL+HER	✓	0.97	0.05	0.97	0.03
DICL+HER	✓	0.86	0.12	0.86	0.20

Fig 4: Success rate comparison of proposed methods vs. baselines with 95% confidence intervals

Failure States

To prevent failure states such as dropping the object off the table, a drop penalty is defined as:

$$R_{penalty}(t) = \begin{cases} -100, & \text{if } z_{obj}(t) < z_{thres} \\ 0, & \text{otherwise} \end{cases}$$

FetchPush

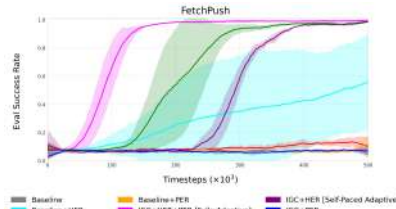


Fig 5: Evaluation of IGC compared to baseline methods on the FetchPush task

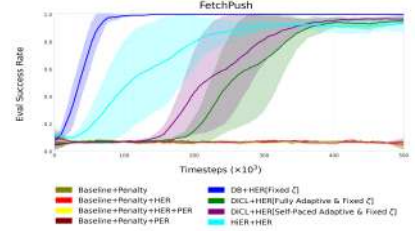


Fig 6: Performance comparison of DB and DICL on the FetchPush task

FetchPickAndPlace

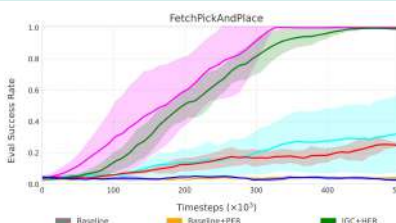


Fig 7: Evaluation of IGC compared to baseline methods on the FetchPickAndPlace task

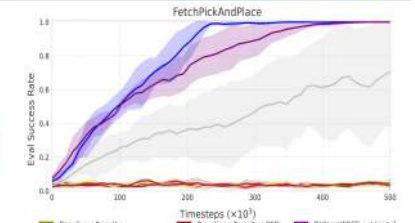


Fig 8: Performance comparison of DB and DICL on the FetchPickAndPlace task

FetchSlide

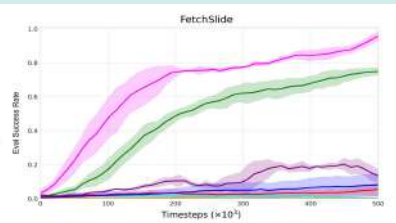


Fig 9: Evaluation of IGC compared to baseline methods on the FetchSlide task

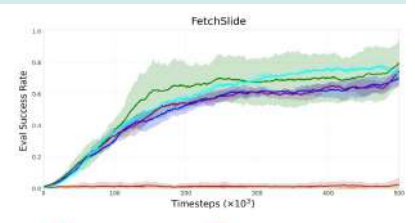
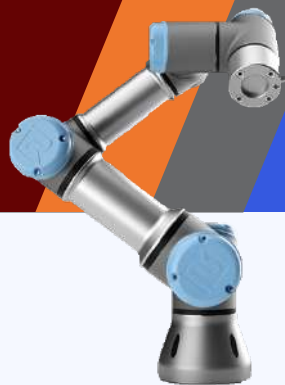


Fig 10: Performance comparison of DB and DICL on the FetchSlide task

Summary

- This research introduces novel curriculum learning strategies designed to boost the performance of deep reinforcement learning agents in sparse-reward continuous control environments.
- IGC dynamically adapts task difficulty based on agent performance, enabling faster and more effective early learning.
- DB separates positive and negative experiences to improve replay efficiency and training stability.
- Combined as DICL, these methods consistently **outperforms baselines** on standard robotic benchmarks.
- Key Results:** faster convergence, improved sample efficiency, and higher success rates across tasks.





SUPERVISED LEARNING FOR MANIPULATION

INTRODUCTION

Robots often struggle with complex visuomotor tasks due to the difficulty of interpreting raw visual input and transferring learned control policies from simulation to the real world. This thesis explores **imitation learning** as a solution, using **expert demonstrations** generated entirely in simulation to train **end-to-end visuomotor policies** without explicit visual programming.

EXPERT DEMONSTRATION

- This research evaluates the feasibility of an imitation learning pipeline through a simplified **long-horizon assembly task**. The robot inserts a rectangular peg into a hole and rotates it 90 degrees—prioritizing sequential execution over fine precision. This task serves as a foundation for more complex operations, such as key-turning.
- We implemented the **oracle controller** using a behavior tree that takes the end-effector and hole poses as input. It determines the current task state, computes a target position, and converts it into an end-effector velocity using a simple feedforward control strategy, which is then sent to the robot.



DOMAIN RANDOMIZATION



- Domain randomization is essential for bridging the gap between simulation and real-world deployment. By leveraging the repeatability of expert demonstrations in simulation, we introduce visual variability to improve the model's robustness.
- We randomize hole and peg poses, arm base height, object colors, camera parameters, lighting conditions, background textures, and add distractor objects.



1. RECOLLECTION

- The learned model is deployed in the simulation with domain randomization, while the expert provides corrective labels. Unlike initial data collection, the robot now follows its own policy. This process adds 400–500 new demonstrations per iteration, gradually improving performance.

DAGGER ALGORITHM

- We combine it with the original dataset to prevent overfitting to recent samples. Although data balancing methods exist, we found simple aggregation to be effective and used it without additional sampling strategies.

2. DATA AGGREGATION

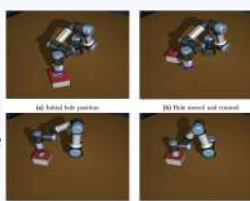
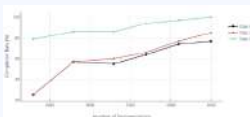
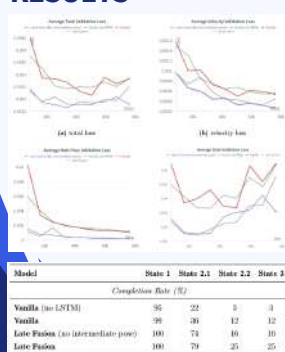
3. RETRAINING

- After aggregation, the model is retrained using the updated dataset, continuing from the previous iteration's weights. We keep the same hyperparameters.

RESULTS

Behavior Cloning Experiment

We evaluated several model variants in a simulation environment with only hole position randomized. The study focuses on models that demonstrate the impact of our architectural modifications: a vanilla model (with/without RNN), a Late Fusion model (with/without intermediate hole pose prediction).



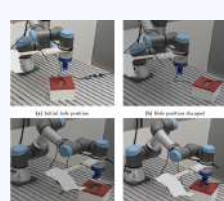
DAGGER Experiment

DAGGER significantly improved task completion across all states, with the most notable gain in the final alignment phase, which reached 100% success.

To evaluate generalization, the hole position and orientation were changed. The model successfully adapted to the new locations, accurately tracking and aligning without any explicit programming—demonstrating strong spatial awareness and flexibility.

Real Robot Experiment

The model trained solely in simulation initially performed poorly on the real robot, often failing to reach the hole due to a gap between simulation and reality. Incorporating 50 real-world demonstrations via DAGGER led to noticeable improvements.



In generalization tests, the robot successfully tracked and aligned with the moving hole, mirroring simulation behavior. However, occasional misalignments during insertion still limited overall success.

- The dataset comprises temporally aligned visual and proprioceptive inputs, along with their corresponding targets.

Inputs:

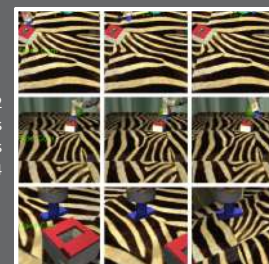
- Images
- End-Effector Pose
- Joint States

Target outputs:

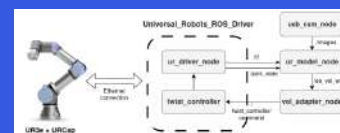
- End-Effector Velocity
- Hole Pose (ground truth).
- Assembly State

- Data is collected from simulation at regular intervals of 0.2 seconds. Each demonstration is yielding 45–60 data points depending on task complexity. The dataset comprises approximately 4,000 demonstrations, gathered over about 14 hours.

DATA COLLECTION



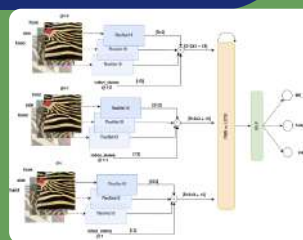
REAL ROBOT DEPLOYMENT



- To deploy our trained policy on the UR3e robot, we created a custom ROS node that receives robot state (joint positions and end-effector pose) and visual input from camera drivers. The model performs inference every 0.2 seconds, and the resulting velocity commands are sent to the driver.
- We applied DAGGER in the real world similarly to simulation, with key differences in data collection. Since the hole's pose is not directly available, expert labels needed to be manually set. Each iteration collects about 50 demonstrations, which are aggregated with prior simulation data for retraining using the same parameters.

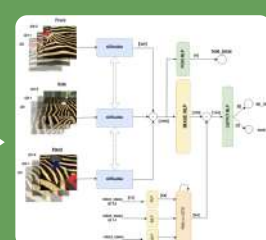


BEHAVIOR CLONING



VANILLA MODEL

- Vision Module:** To better capture temporal correlations in multi-view visual input, we replaced the initial 2D ResNet-18 with a **3D ResNet**, which has shown superior performance in modeling temporal features from image sequences.



LATE FUSION MODEL

- Hole Position as Intermediate Output:** Instead of predicting hole position at the final output, we introduce it as an intermediate target to guide the visual encoder and avoid confusion from end-effector states.
- Temporal Modeling:** We use an **LSTM** to capture sequential patterns in proprioceptive inputs, complementing the temporal encoding provided by the 3D ResNet for visual data.

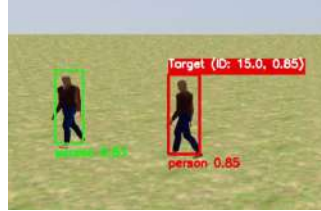


Introduction

- Autonomous UAV tracking using Uni-modal RGB input with state-of-the-art detection and tracking algorithms.
- Applicability: Security systems trigger false alarms (~80%) from motion sensors (e.g., birds, wind). Manual checks waste resources.
- Deployment in: Train yards (intrusion checks), Industrial surveillance.

RESEARCH QUESTIONS:

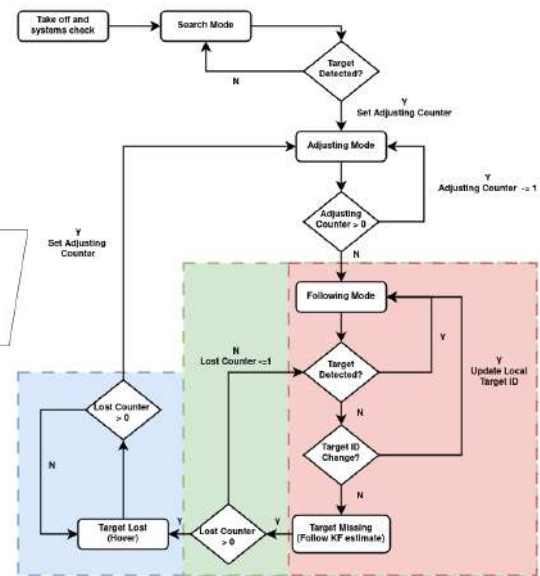
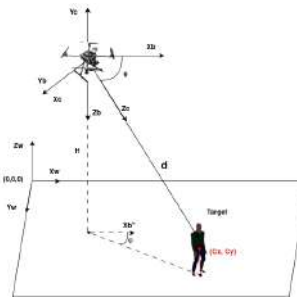
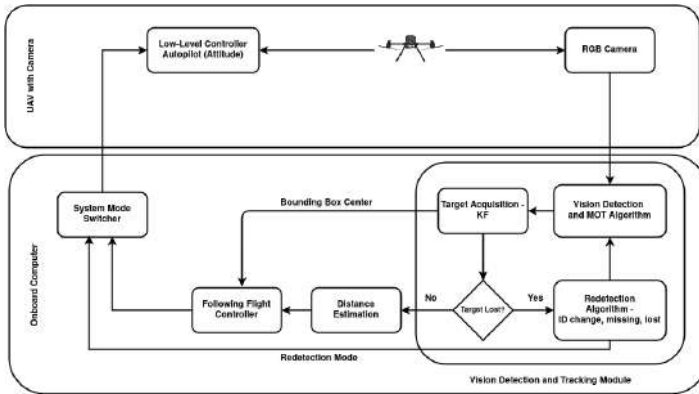
- Can MOT algorithms (BoT-SORT and ByteTrack) paired with YOLOv8 outperform SOT methods for UAV target tracking?
- Can multi-target data improve occlusion recovery?
- Does our flight controller ensure stable following in dynamic scenes?



Features

- Target ID Change:** The MOT algorithm assigns a new ID (based on IOU) to the visible target due to motion or misidentification.
- Target Missing:** The target disappears briefly, often due to occlusion or fast motion. A Kalman filter estimates its position.
- Target Lost:** After several frames with no target, the system assumes the target is lost. The UAV hovers in place and waits to redetect the target.

Methodology



FRAMEWORKS:

- Detection: YOLOv8
- Tracking: BoT-SORT & ByteTrack + Kalman filter
- Monocular Distance Estimation: Bounding box height (Pinhole model)
- Control: PX4 flight stack
- Middleware: ROS1 & ROS2

SETUP:

- Simulations: Gazebo with PX4 x500 UAV platform
- Real-world Onboard: x500 UAV equipped with RPi + Camera Module + AI HAT+

Experiments & Results

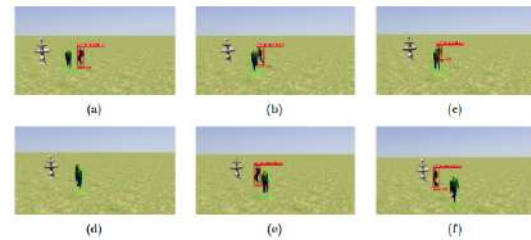


Figure 4.1: Target ID Change redetection experiment with another bystander: system recognizes and adapts to ID changes from the MOT module by updating the locally defined target ID. (a) to (c) Target followed with ID 47. (d) Occlusion due to bystander. (e) Target ID Change overwrite local target ID from 47 to 115. (f) Target followed with new ID

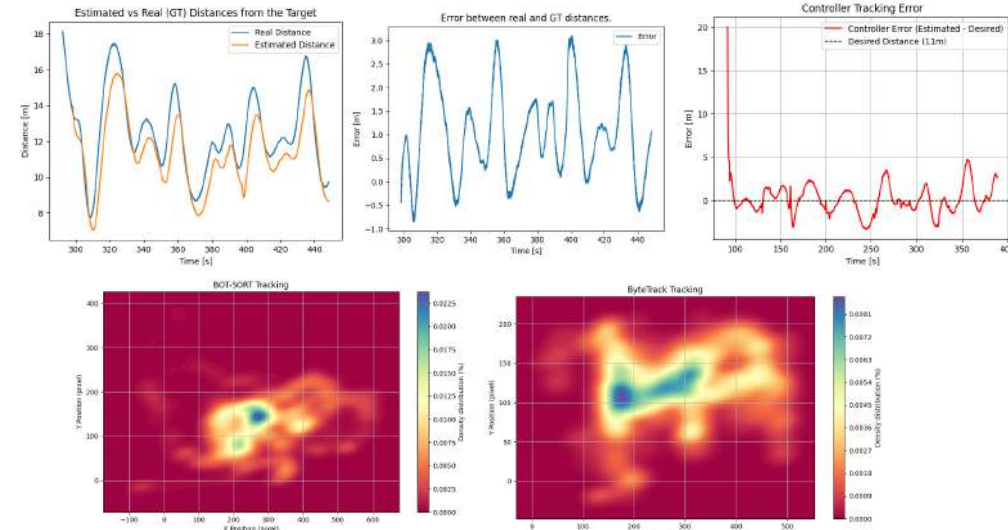
Test No:	Time (s)	Distance UAV (m)	Distance Target (m)	Visual Acc. (%)	BoT-SORT Quantitative Data			
					d_est Error (m)	FPS (Hz)	No. ID Change	Mean Controller Error (m)
1	155.92	80.75	122.94	99.7	1.13	9.03	0	0.34
2	209.81	121.84	177.43	97.89	1.38	10.04	2	0.93
3	183	104.75	165.04	99.43	1.47	9.62	0	0.03

Conclusion

This work presents a UAV-based target tracking system using vision and multi-target data to improve re-identification under occlusions and motion changes. BoT-SORT delivers higher tracking precision with fewer ID switches, while the 3D controller keeps the target centered during dynamic flight.

NEXT STEPS:

- Real-World Testing:** Using prepared hardware (Raspberry Pi 5 + Camera Module + Hailo HAT) in ROS2 for validation.
- Camera Stabilization:** Add a gimbal to decouple camera view from UAV tilt and improve target centering.
- Low-Light Tracking:** Integrate RGB-T (thermal) fusion and depth estimation from RGB to enhance performance.



Globally consistent mapping for indoor and outdoor environments using Hybrid LiDAR SLAM

Zewdie Habtie Sisay

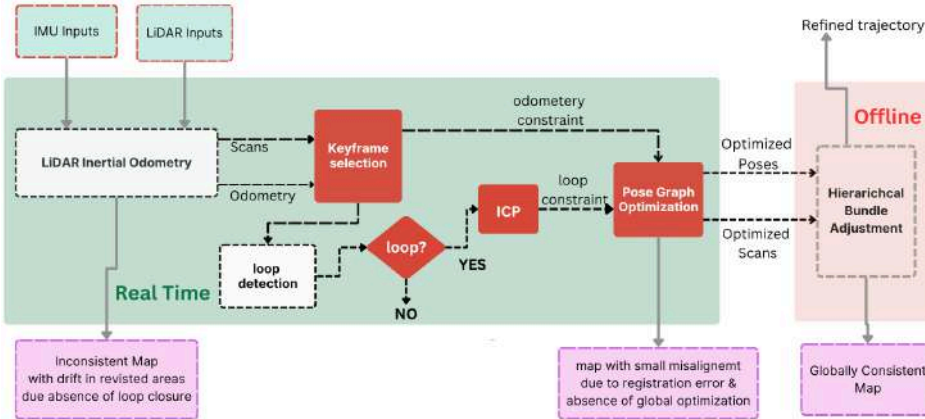
Supervisors: Prof. Zoltan Istenes, Prof. Mohammad Aldibaja

Introduction

An accurate map is an important prerequisite for autonomous robots to navigate, plan paths, localize, avoid obstacles and execute tasks efficiently. Simultaneous Localization and Mapping (SLAM) algorithms using LiDAR point clouds are the most preferred mapping method due to the accuracy of the 3D range measurements of LiDAR sensors.

Problem statement: Due to an inherent noise in sensor measurements, robot motion, and dynamics of the environment, inconsistencies of the map generated with LiDAR SLAM are inevitable. An inconsistent map of the environment causes accumulation of localization errors, failure in path planning, increased risk of collision, and mission failure. Since SLAM is not closed-form solution, generating 100% accurate map of the environment remains a challenge.

Methodology



FAST-LIO2

Prediction

$$\hat{\mathbf{x}}_{t+1} = \hat{\mathbf{x}}_t \oplus (\Delta t f(\hat{\mathbf{x}}_t, \mathbf{u}_t, 0)); \quad \hat{\mathbf{x}}_0 = \hat{\mathbf{x}}_{k-1},$$

$$\hat{\mathbf{P}}_{t+1} = \mathbf{F}_{\mathbf{x}} \hat{\mathbf{P}}_t \mathbf{F}_{\mathbf{x}}^T + \mathbf{F}_{\mathbf{w}} \mathbf{Q}_t \mathbf{F}_{\mathbf{w}}^T; \quad \hat{\mathbf{P}}_0 = \mathbf{P}_{k-1}$$

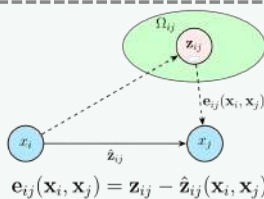
Iterated Update

$$\mathbf{z}_t^* = \mathbf{w}_t^T (G \hat{\mathbf{T}}_k^T \hat{\mathbf{T}}_{k+1}^T \mathbf{p}_t - G \mathbf{q}_t),$$

$$\mathbf{K} = (\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} + \mathbf{P}^{-1})^{-1} \mathbf{H}^T \mathbf{R}^{-1},$$

$$\hat{\mathbf{x}}_{k+1}^* = \hat{\mathbf{x}}_k^* \oplus (-\mathbf{K} \mathbf{z}_k^* - (\mathbf{I} - \mathbf{K} \mathbf{H})(\mathbf{J}^*)^{-1} (\hat{\mathbf{x}}_k^* \oplus \hat{\mathbf{x}}_k))$$

PGO

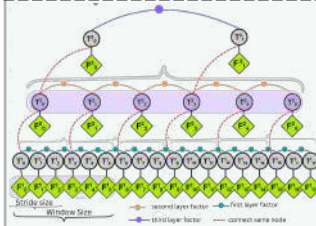


$$\mathbf{e}_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{z}_{ij} - \hat{\mathbf{z}}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$$

$$\mathbf{F}(\mathbf{x}) = \sum_{\langle i, j \rangle \in \mathcal{C}} \mathbf{e}_{ij}^T \Omega_{ij} \mathbf{e}_{ij}$$

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \mathbf{F}(\mathbf{x})$$

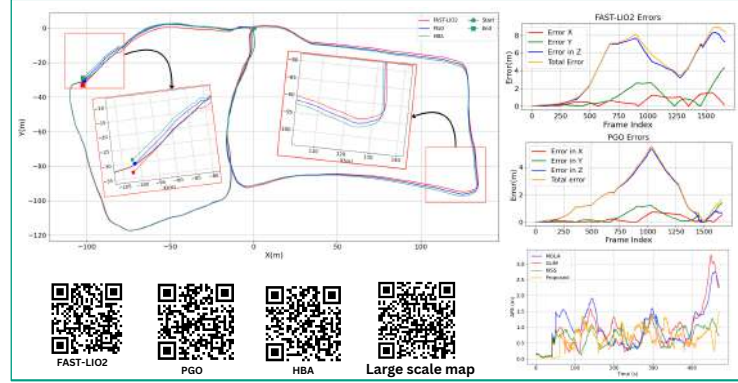
HBA



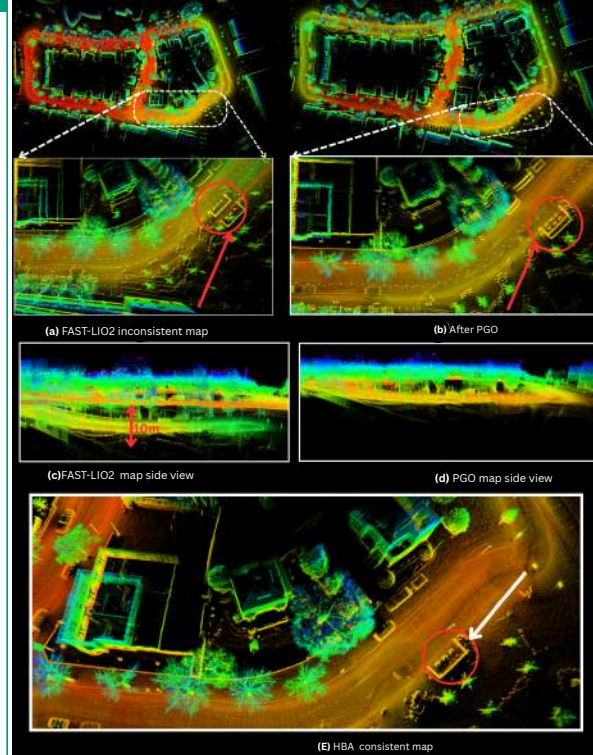
$$\mathbf{T}_j^{i+1} = \mathbf{T}_{s,j}^{i+1} = \prod_{k=1}^j \mathbf{T}_{s,j,k}^{i+1}$$

$$\mathbf{F}_j^{i+1} \triangleq \bigcup_{k=0}^{w-1} (\mathbf{T}_{s,j,s,j+k}^{i+1} \cdot \mathbf{F}_{s,j+k}^i)$$

Trajectory Errors



Map quality



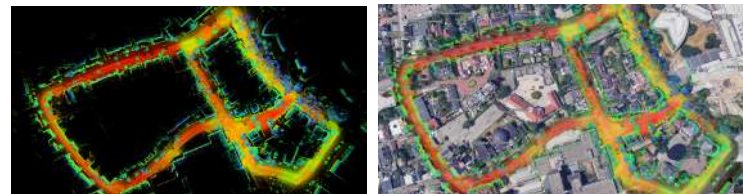
Mean Map Entropy(MME) of the three methods across different datasets

Method	S01	S02	S03	S11	K03	K00
FAST-LIO2	-2.63, 0.11	-1.9, 0.19	-2.46, 0.19	-3.34, 0.10	-1.85, 0.07	-2.40, 0.04
PGO	-2.66, 0.14	-2.50, 0.12	-2.60, 0.30	-3.21, 0.20	-1.91, 0.03	-2.47, 0.08
HBA	-2.71, 0.05	-2.48, 0.084	-2.85, 0.13	-3.20, 0.091	-2.39, 0.05	-2.97, 0.026

S stands for Saxion, K stands for KAIST, A stands for KITTI

Contributions

FAST-LIO2 ensures local consistency but **accumulates drift** over long trajectories, leading to global map errors. Pose Graph Optimization (PGO) with loop closure reduces this drift but still leaves small inconsistencies due to **registration errors**, **insufficient (false) loop closures**, or **a wrong initial guess**. To correct these, Hierarchical Bundle Adjustment (HBA) is applied. Since HBA cannot converge on large inconsistencies, PGO acts as an **interface** between FAST-LIO2 and HBA, minimizing drift to within HBA's convergence range. Our main contribution lies on bridging FAST-LIO2 and HBA incorporating loop closure with PGO.



Conclusion

The proposed SLAM system significantly improved trajectory accuracy and map consistency across datasets. PGO reduced real-time drift, while HBA enhanced global alignment and consistency. The hybrid approach performed well in indoor and large-scale outdoor environments with frequent loop closures, resulting in a robust and consistent mapping pipeline.



Introduction

Accurate and comprehensive mapping of the seafloor is crucial for numerous applications in marine science, archaeology, and environmental monitoring. Traditional sonar mapping provides broad coverage but often lacks the fine detail and clarity of optical images due to inherent limitations and distortions. Optical mapping offers high-resolution visual information but is limited by water visibility and precise localization. By combining optical and acoustic systems, the advantages of both sensory modalities can be exploited.

This thesis tackles this challenge by developing a robust method to accurately align and integrate sonar and optical data. Our goal is to create precise, consistent, and composite maps that combine the strengths of both sensing modalities, ultimately improving seafloor classification and our understanding of the underwater environment.

Contributions

- **Novel Multimodal Mapping Framework:** We introduce the first systematic approach to spatially align and fuse side-scan sonar and optical imagery from multiple underwater survey sessions.
- **Comprehensive Non-Rigid Alignment:** We jointly optimize vehicle trajectories, 3D landmark positions, session-to-world transformations, and sensor extrinsics for a comprehensive, non-rigid alignment.
- **Two-Level Alignment for Error Correction:** The framework corrects a wide range of errors (from inter-session offsets to intra-session distortions) by employing a two-level alignment (global rigid session transformation combined with local non-rigid trajectory deformations).

Methodology

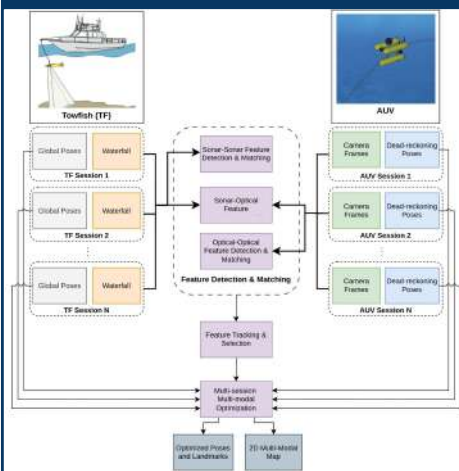


Fig 1: An overview of the proposed system pipeline

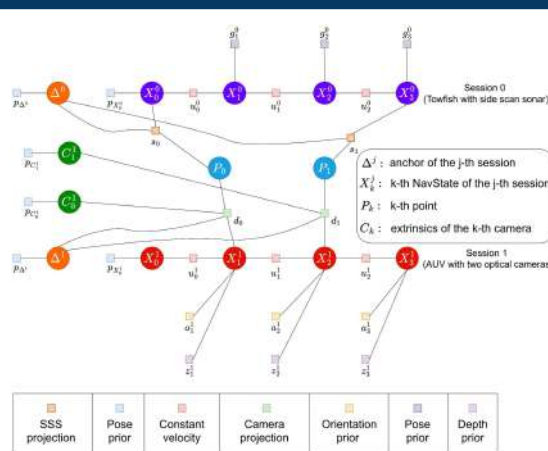


Fig 2: The factor graph representation of the proposed multi-session, multimodal mapping framework

Data Acquisition

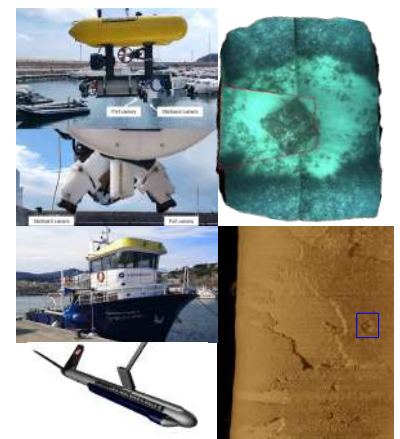


Fig 3: The vehicles used for the experiment and the data collected by them.

Qualitative Evaluation

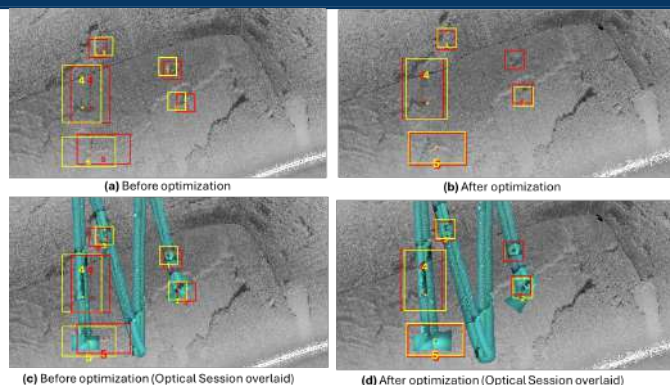


Fig 4: Mosaic of 2 sonar sessions and 1 optical session

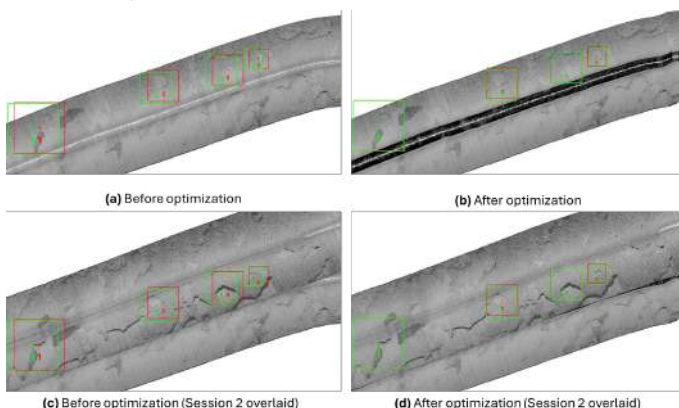


Fig 5: Mosaic of 2 sonar sessions

Quantitative Evaluation

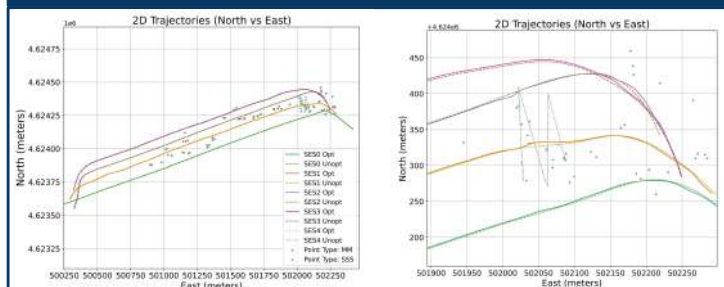


Fig 6: XY Position plots of the vehicles before and after optimization

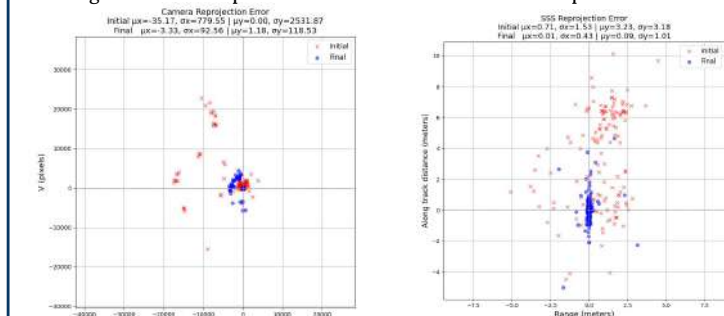


Fig 7: Scatter plots of reprojection errors

Conclusion

Our framework successfully aligns sonar and optical images, demonstrating its effectiveness in creating comprehensive seafloor maps. Future work will focus on automating feature detection and matching across these diverse imaging modalities.

Diurnal and Nocturnal Robust Visual-Aided GNSS Navigation on Horticultural Fields

Author: Lisa Paul Magoti

Supervisors: Zoltán Istenes, Bram Benist and Ricard Padell
[Eötvös Loránd University, Budapest and Agrikola.AI, Barcelona]

Introduction

GNSS alone can lead to navigation errors in agriculture due to mismatches between mapped and actual crop bed positions. Integrating real-time visual perception allows robots to adapt to the true field layout, improving accuracy and protecting crops.



Research Questions

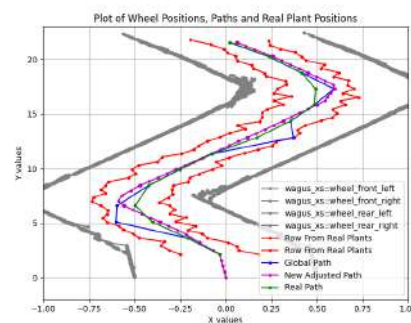
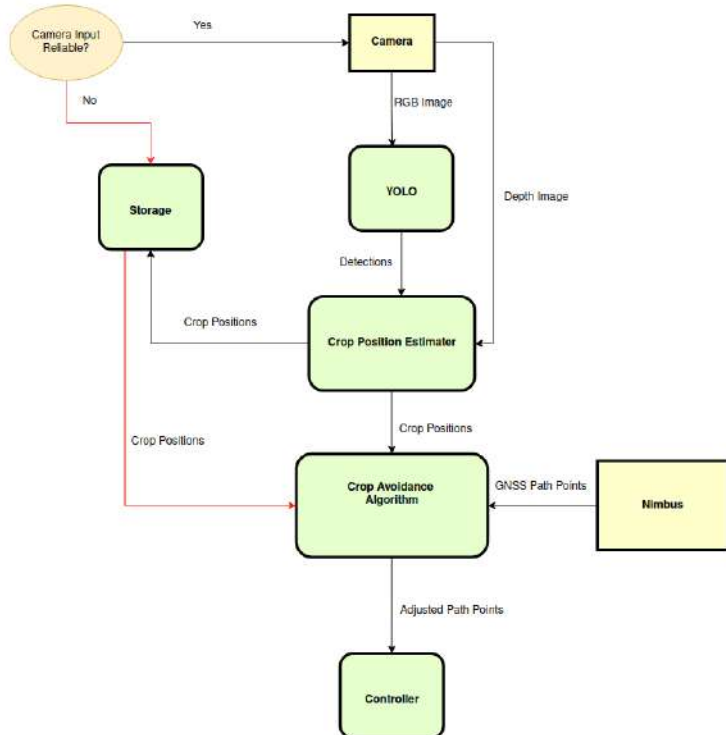
- How can vision-based algorithms combined with GNSS improve the accuracy and safety of autonomous navigation in horticultural beds?
- How can GeoData from autonomous bed discovery be effectively used to enhance subsequent navigation and mission performance?

Results

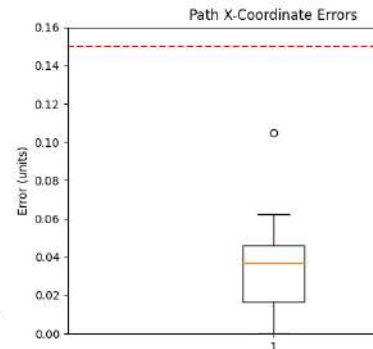
For the same given GNSS path, using the algorithm, the robot is able to avoid plants where as without it, the robot steps on plants

Method

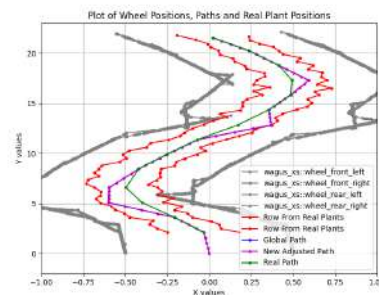
This research leverages neural networks and computer vision to enable crop detection and adaptive path planning. The use of deep learning techniques allows the system to recognize plants, dynamically adjusting the robot's trajectory to maintain precise alignment with the crop beds while not causing any damage to the plants.



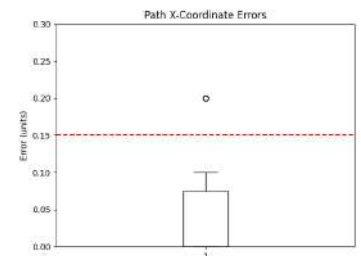
Path adjusted to ensure safety.



All path points are within max limit.



Path is not adjusted.



Not all path points are within max limits.

Plant Detection

YOLOv5 pre-trained model used to detect horticultural plants.

Detections' Processing

Plant detections are stored as obstacles and used to guide path planning.

Global Path

A GNSS global path is provided, which is checked and corrected if necessary to avoid obstacles.

Safe Navigation

Algorithm ensures the robot's wheels move in positions safe from plant collisions.

Conclusion

The method demonstrated that using vision input to compensate for GNSS drift improves both **accuracy** and **safety** by allowing **real-time correction of the navigation path**. Additionally, storing geodata from previous runs reduces **unnecessary computation**, particularly in **static or minimally changing environments**. This approach also **enhances safety**, as it provides fallback input data when **vision sensors are unreliable**, such as during **nighttime or poor weather conditions**.