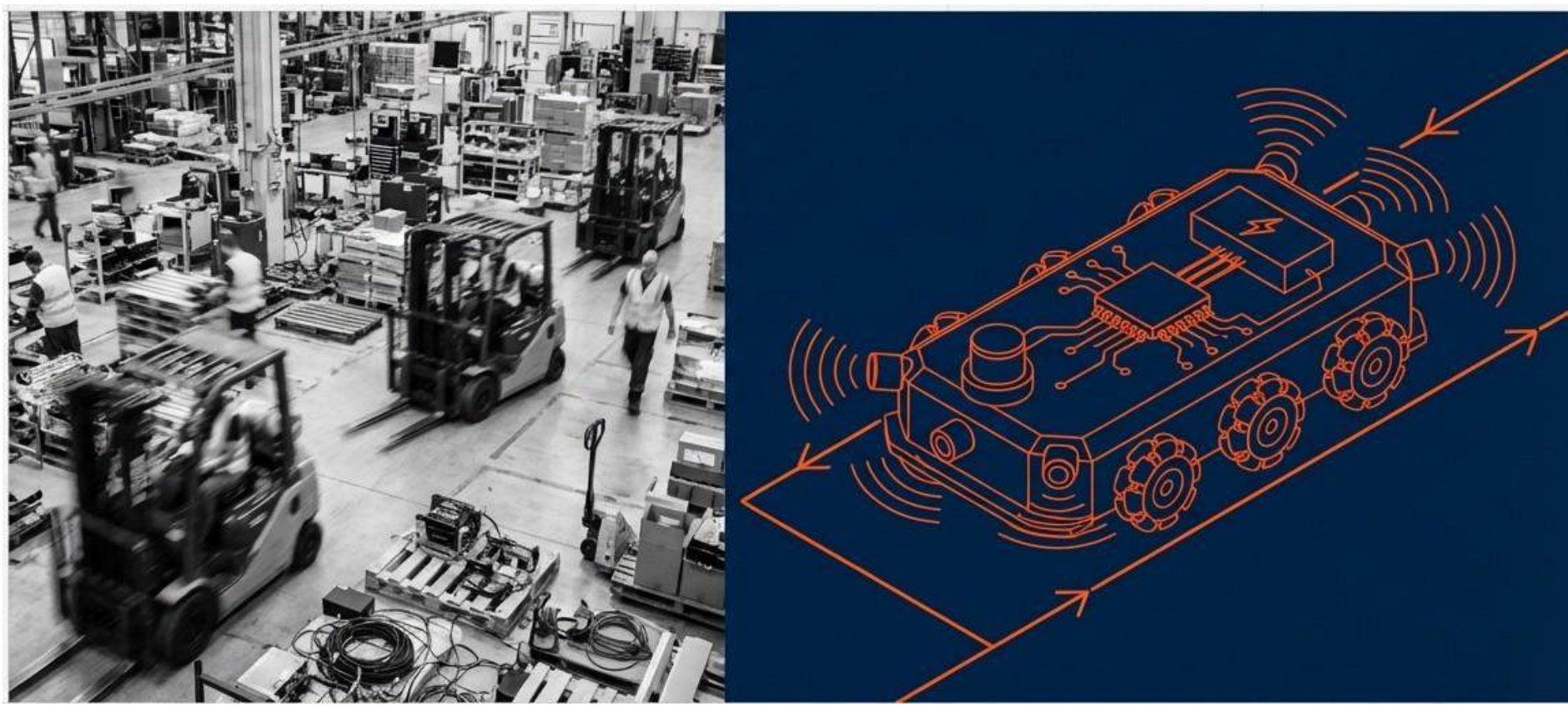
Lightweight Collision Avoidance for AGV Systems

Bridging Theory and Practice: A 6-Layer Modular Architecture for Real-Time Safety

Team #2:

Myroslav Mishchuk
Asiful Huda
Milad Jafari
Leonardo Schiavo



Presentation Roadmap

Scientific summary 🔬

01 • Conceptualization

The 6-Layer modular architecture

02 • Instantiation & Per-component validation

Specific algorithms: HySDG-ESD, DWA, GapNav

03 • End-To-End Validation

Current progress and next steps

Commercialization proposal 🏭

04 • Infrastructure & Deployment

Real-world data collection, hardware integration

05 • Extensive Framework

Isaac Sim simulation with Web UI

TRACK ONE

Scientific summary: Conceptualization

A theoretical 6-Layer model for modular, maintainable collision avoidance systems

The 6-Layer Collision Avoidance Architecture

L6 Translation Layer — Robot commands

L5 Control Layer — Trajectory planning

L4 Decision Layer — Risk assessment

L3 World Model — Sensor fusion

L2 Perception Layer — Detection & tracking

L1 Sensor Layer — Data acquisition

↑ Data flows up

Why This Architecture?

Modularity: Replace any layer without affecting others

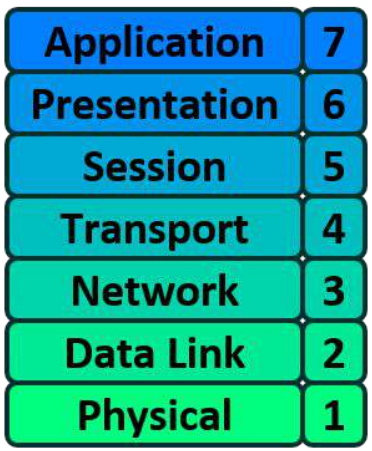
Testability: Isolate and test individual components

Transparency: Clear data flow for debugging

Flexibility: Mix different algorithms per layer

ANALOGY

Like the OSI model for networking, our architecture defines theoretical layers that enable interoperable implementations.



Layers 1–3: From Raw Data to Understanding

L1 • Sensor Layer

Purpose

Raw data acquisition from all sensors

Inputs

- LiDAR point clouds
- Camera images (RGB/Depth)
- Ultrasonic distances

Output

MultiSensorData with timestamps

L2 • Perception Layer

Purpose

Detect, classify, and track obstacles

Key Functions

- DBSCAN/K-means clustering
- Static/Dynamic classification
- Kalman filter tracking

Output

DetectedObject (position, velocity, type)

L3 • World Model

Purpose

Unified environment representation

Key Functions

- 2D Occupancy grid
- Sensor fusion (feature-level)
- World coordinate transform

Output

OccupancyGrid + tracked objects

Layers 4–6: Decision to Execution

L4: Decision Layer

Evaluates risk and selects behaviors based on world state.

- Zone-based risk assessment
- Warning/Slowdown/Emergency zones
- Behavior selection (continue/slow/stop)
- Risk score computation

L5: Control Layer

Generates safe trajectories using DWA algorithm.

- Velocity Obstacle algorithm
- Collision cone
- Selection of collision-free velocity closest to desired velocity
- Goal-directed planning

L6: Translation Layer

Converts commands to robot-specific controls.

- Differential drive kinematics
- Velocity to wheel speeds
- Hardware-specific limits
- Command rate control

Each layer transforms data: World State → Risk/Behavior → Trajectory → Robot Commands

Track 1: Publication and future research

Proposed Publication Title: "The 'OSI Model' for Autonomous Logistics: A 6-Layer Modular Architecture for Explainable and ISO-Compliant AGV Collision Avoidance"

Extension for TUAI Objectives:

- Focus on XAI (Explainability): Extend the architecture to include a "Transparency Module" that logs decision boundaries between layers. For example, if the Decision Layer triggers an emergency stop, the system should be able to backtrack to the specific Perception Layer object ID and Sensor Layer raw data that caused it. This directly addresses DC7’s objective of making "black box" decisions transparent for human operators.

Track 1 Contribution

CONCEPTUAL / THEORETICAL

A Theoretical Framework for Modular Design

Unlike approaches that focus on ready-made solutions, we propose a theoretical 6-Layer architecture—analogous to the OSI model for networking. This abstraction enables researchers and engineers to build collision avoidance systems with interchangeable algorithms at each layer, facilitating experimentation, debugging, and systematic improvements.

OSI Model

Theoretical layers for networking

→

6-Layer Model

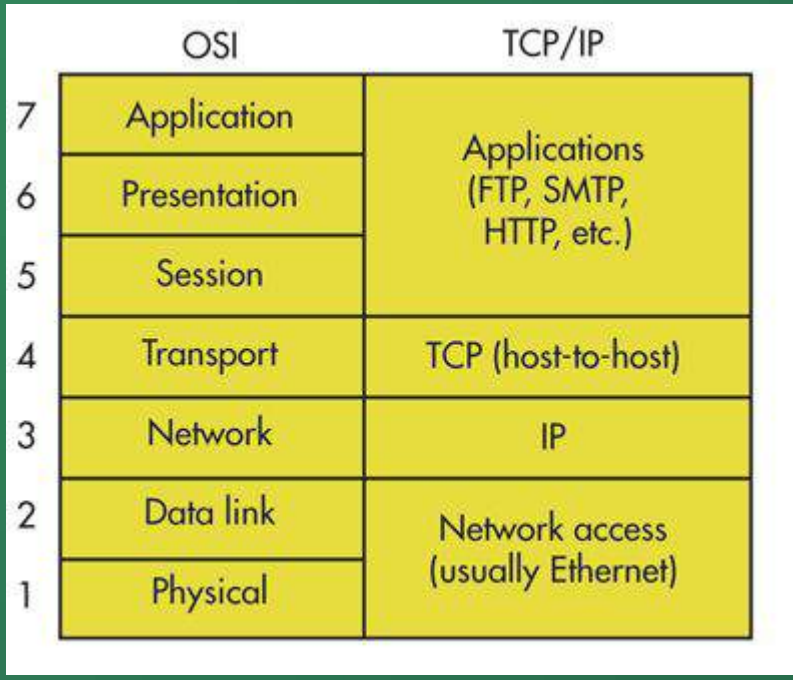
Theoretical layers for collision avoidance

TRACK 2

Scientific summary: Instantiation & Per-Component validation

Concrete Methods for Each Layer

HySDG-ESD Perception • DBSCAN Clustering • GapNav+DWA+APF Navigation

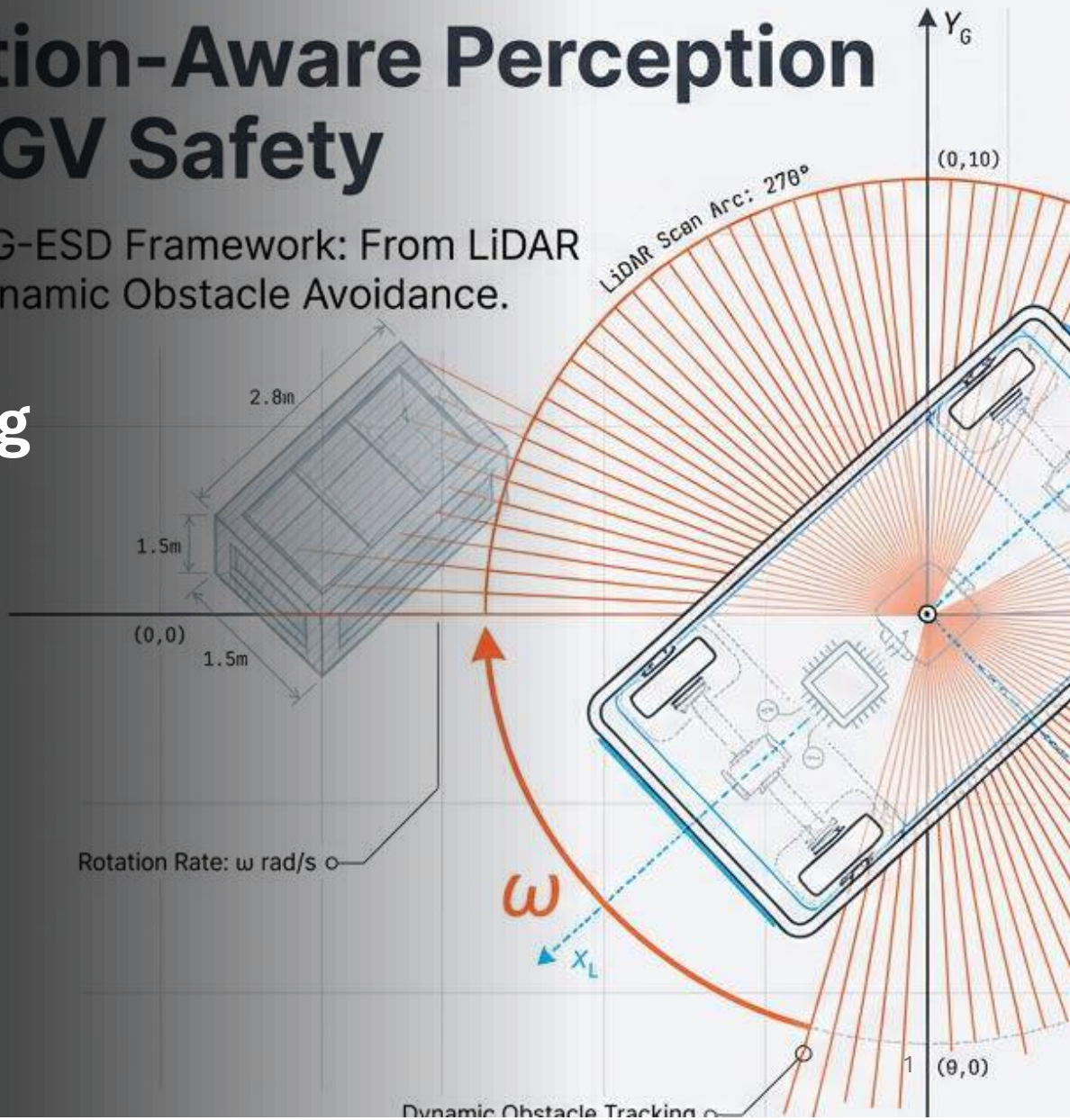


HySDG-ESD: Dynamic Obstacle Detection for Moving AGVs Using LiDAR and Ego-Motion Compensation

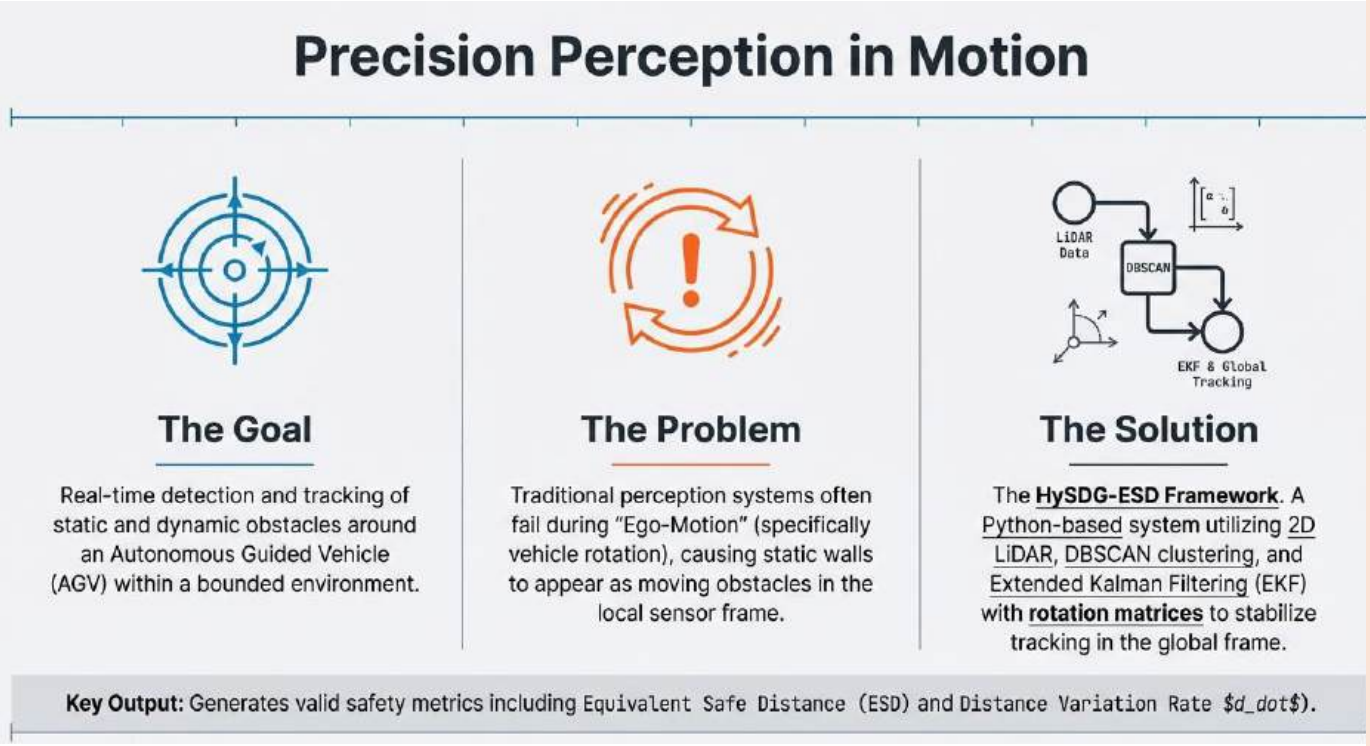
Milad Jafari Barani

Autonomous Guided Vehicle (AGV) Perception System

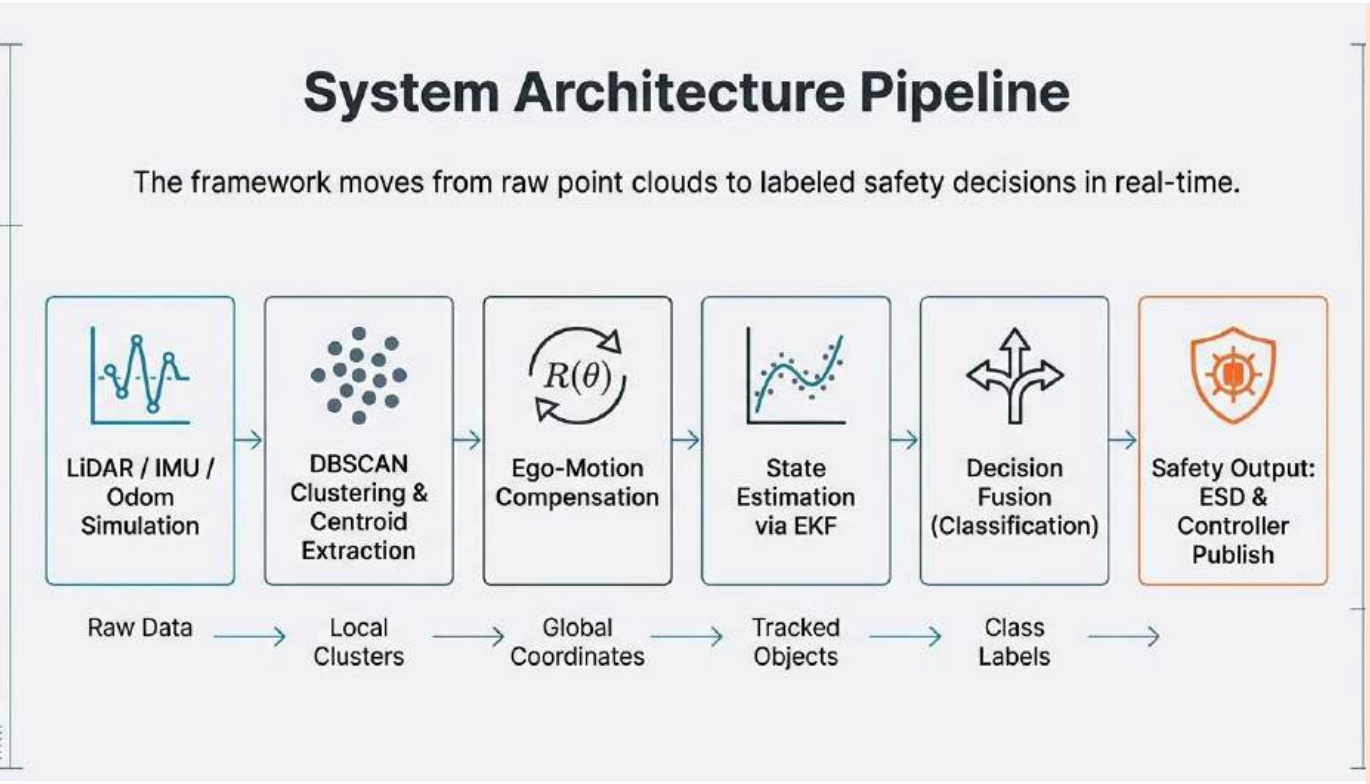
Milad Jafari Barani - Milad.jafare@gmail.com



System Overview



- **Project Objective**
  - Real-time detection and tracking of static and dynamic obstacles around an Autonomous Guided Vehicle (AGV)
  - Robust perception under vehicle rotation using LiDAR measurements
- **System Architecture**
  - LiDAR point parsing and filtering
  - Clustering using DBSCAN
  - Coordinate transformation with rotation-aware model
  - Multi-object tracking with Hungarian data association
  - State estimation using Extended Kalman Filter (Constant Velocity model)
- **System Outputs**
  - Obstacle position and velocity
  - Obstacle state classification: STATIC / DYNAMIC / UNKNOWN
  - Safety indicators: equivalent distance (d_eq), distance rate (d_dot), confidence score



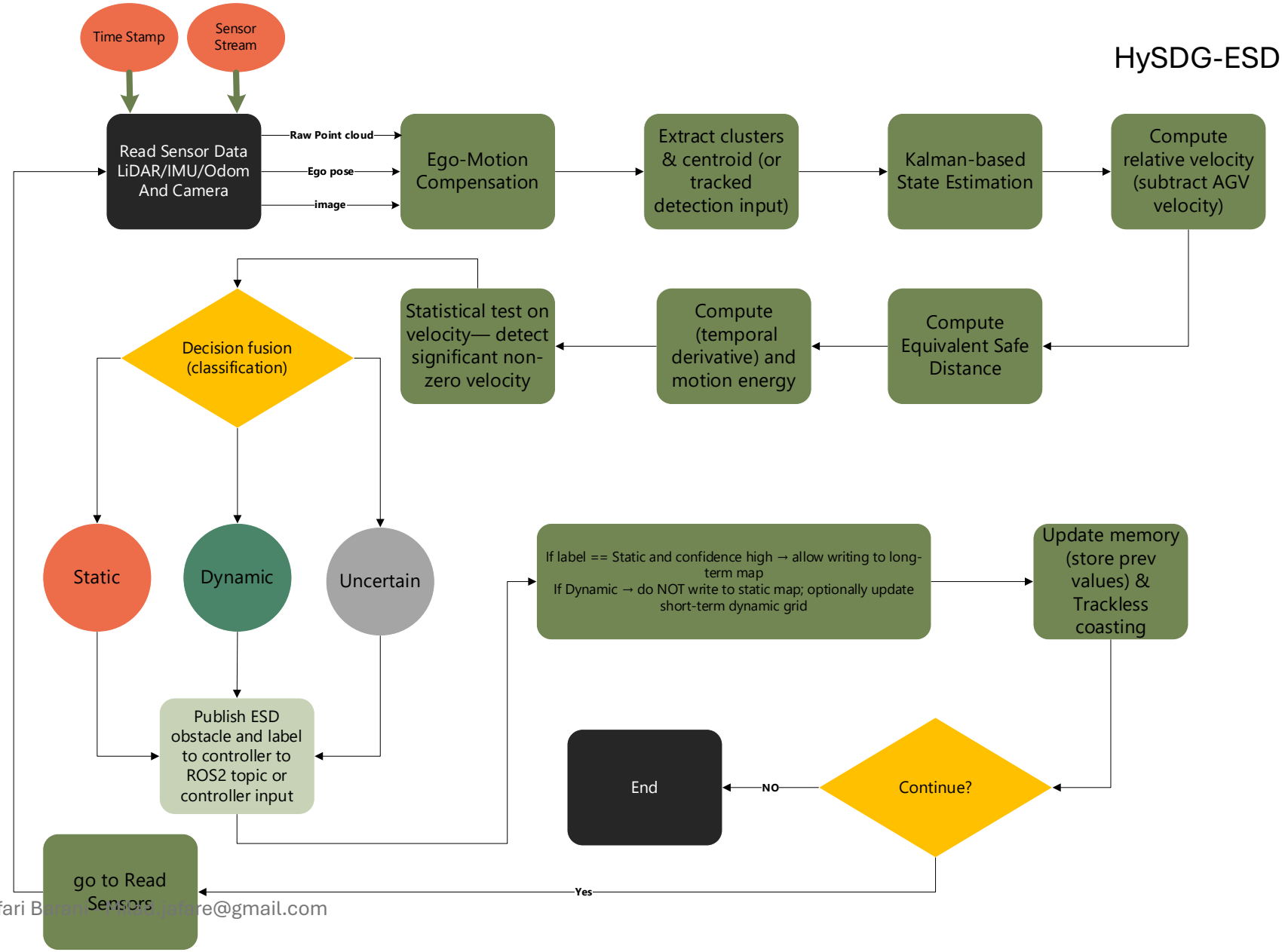
Core Methodology

- **LiDAR Processing**
  - Field-of-view and range filtering
  - Spatial clustering of scan points using DBSCAN
- **Tracking and Estimation**
  - Extended Kalman Filter with Constant Velocity motion model
  - Mahalanobis distance gating for outlier rejection
  - Adaptive velocity damping for stable estimation
- **Rotation-Aware Coordinate Transformation**
  - Transformation from AGV local frame to global world frame
  - Use of 2D rotation matrix based on AGV heading angle
  - Enables correct tracking during vehicle rotation

Classification & Contributions

- **Obstacle Classification Strategy**
  - Velocity-based multi-stage decision logic
  - Temporal voting using velocity and state history
  - Conservative classification to reduce false dynamic detections
- **HySDG-ESD Safety Metrics**
  - Equivalent distance (d_eq) computation
  - Distance variation rate (d_dot) estimation
  - Identification of critical obstacles for collision avoidance
- **Main Contributions**
  - Fully rotation-aware LiDAR perception framework
  - Stable multi-object tracking with adaptive EKF
  - Accurate static vs. dynamic obstacle discrimination
  - Applicable to real-time AGV navigation and safety systems

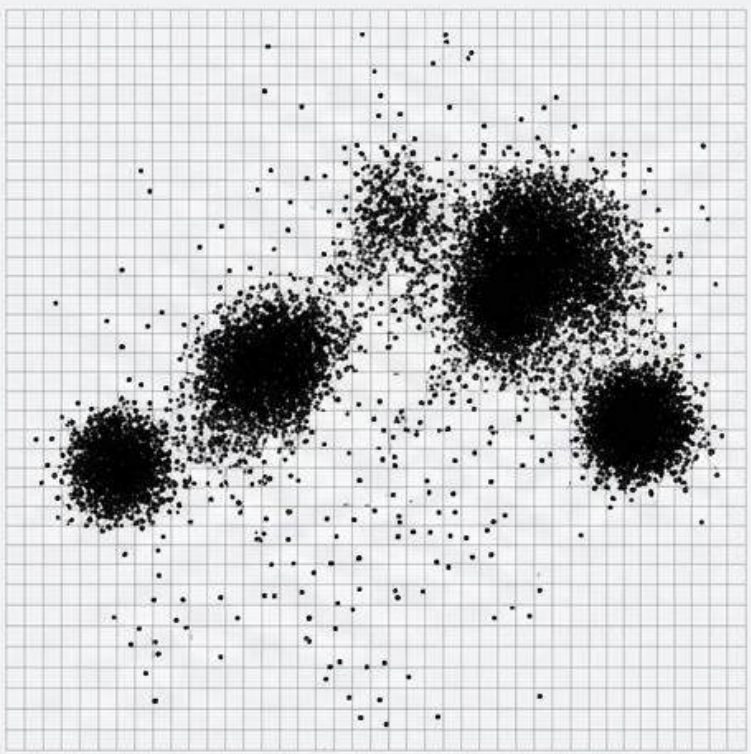
HySDG-ESD Flochart



Step 1: LiDAR Processing & Clustering

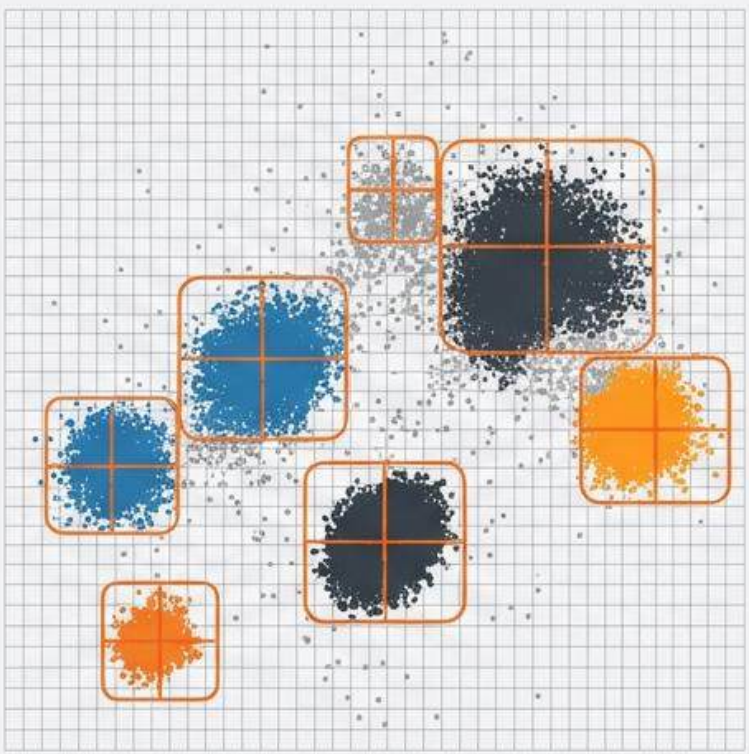
Raw Sensor Input

Simulates 2D LiDAR scans with configurable noise parameters and Field-of-View (FOV) filtering.



DBSCAN Output

Groups adjacent scan points based on density to distinguish objects from background noise.



Data In

Simulates 2D LiDAR scans with configurable noise parameters and Field-of-View (FOV) filtering.

Algorithm

DBSCAN (Density-Based Spatial Clustering of Applications with Noise).

Process

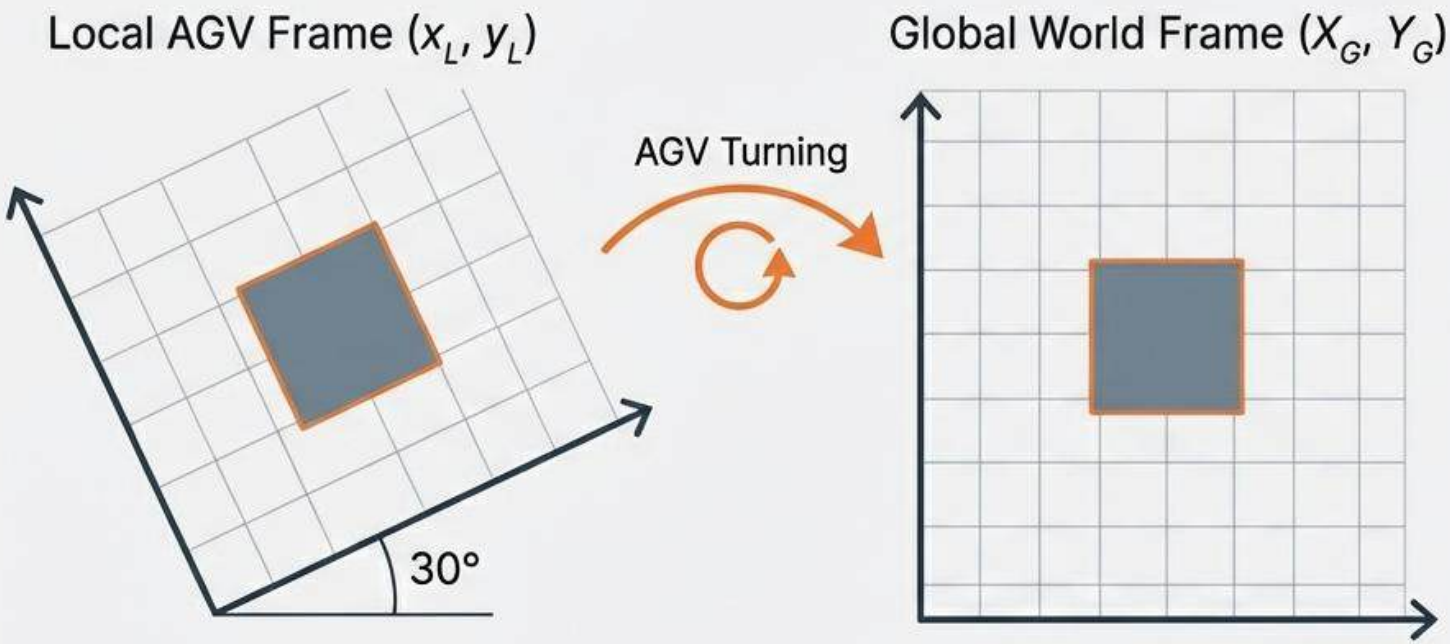
Groups adjacent scan points based on density to distinguish objects from background noise.

Output

Raw clusters are converted into centroids, serving as the input for the tracking system.

Step 2: Ego-Motion Compensation

Solving the Rotation Problem.



The Challenge: When the AGV turns, static objects appear to move relative to the sensor.

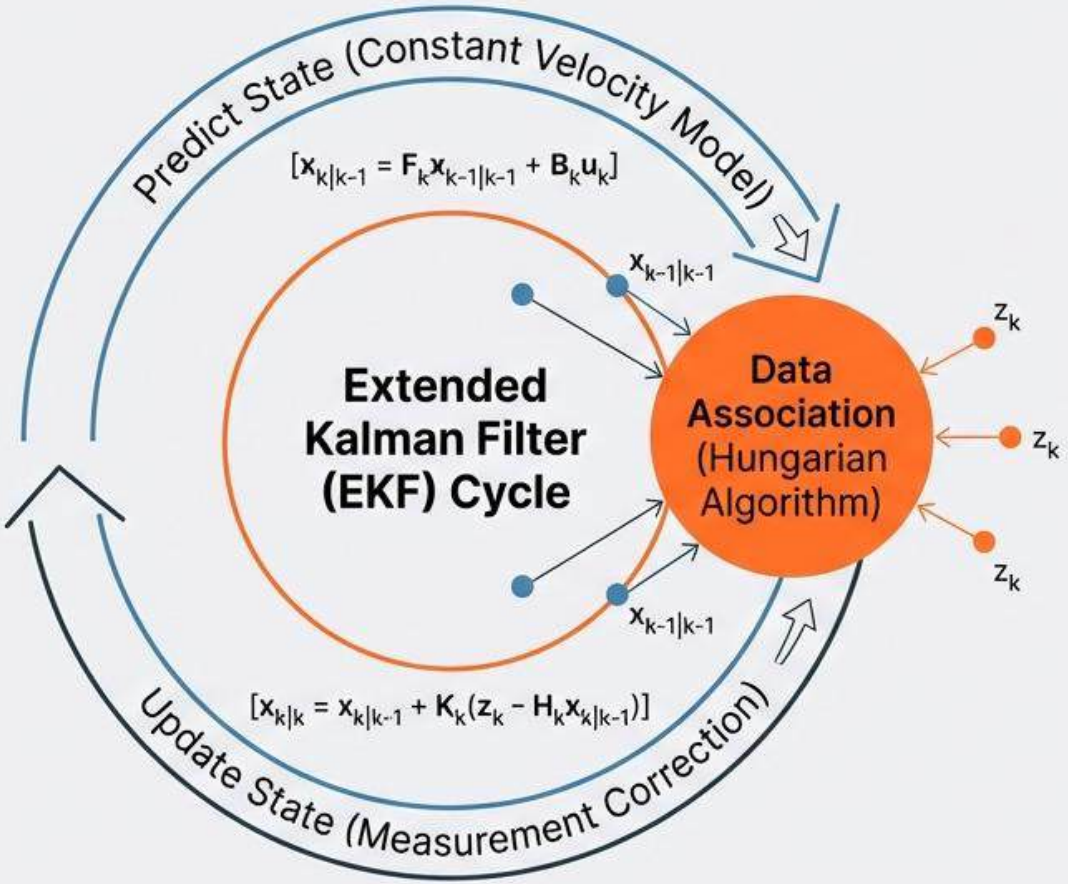
The Fix: Apply 2D Rotation Matrices using the AGV heading angle (θ).

Key Insight: Transforms measurements from Local Frame to Global Frame *before* tracking begins, ensuring static objects remain mathematically static.

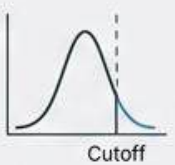
$$\begin{bmatrix} X_G \\ Y_G \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_L \\ y_L \end{bmatrix} + \begin{bmatrix} X_{AGV} \\ Y_{AGV} \end{bmatrix}$$

Milad Jafari Barani - Milad.jafare@gmail.com

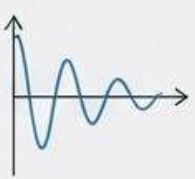
Step 3: Multi-Object Tracking & Estimation



- **Algorithm:** EKF utilizing a Constant Velocity motion model.
- **Association:** Hungarian Algorithm matches new centroids to known tracks.
- **Stability Mechanisms:**
  - ***Mahalanobis Distance Gating***: Statistical outlier rejection to prevent false tracks.

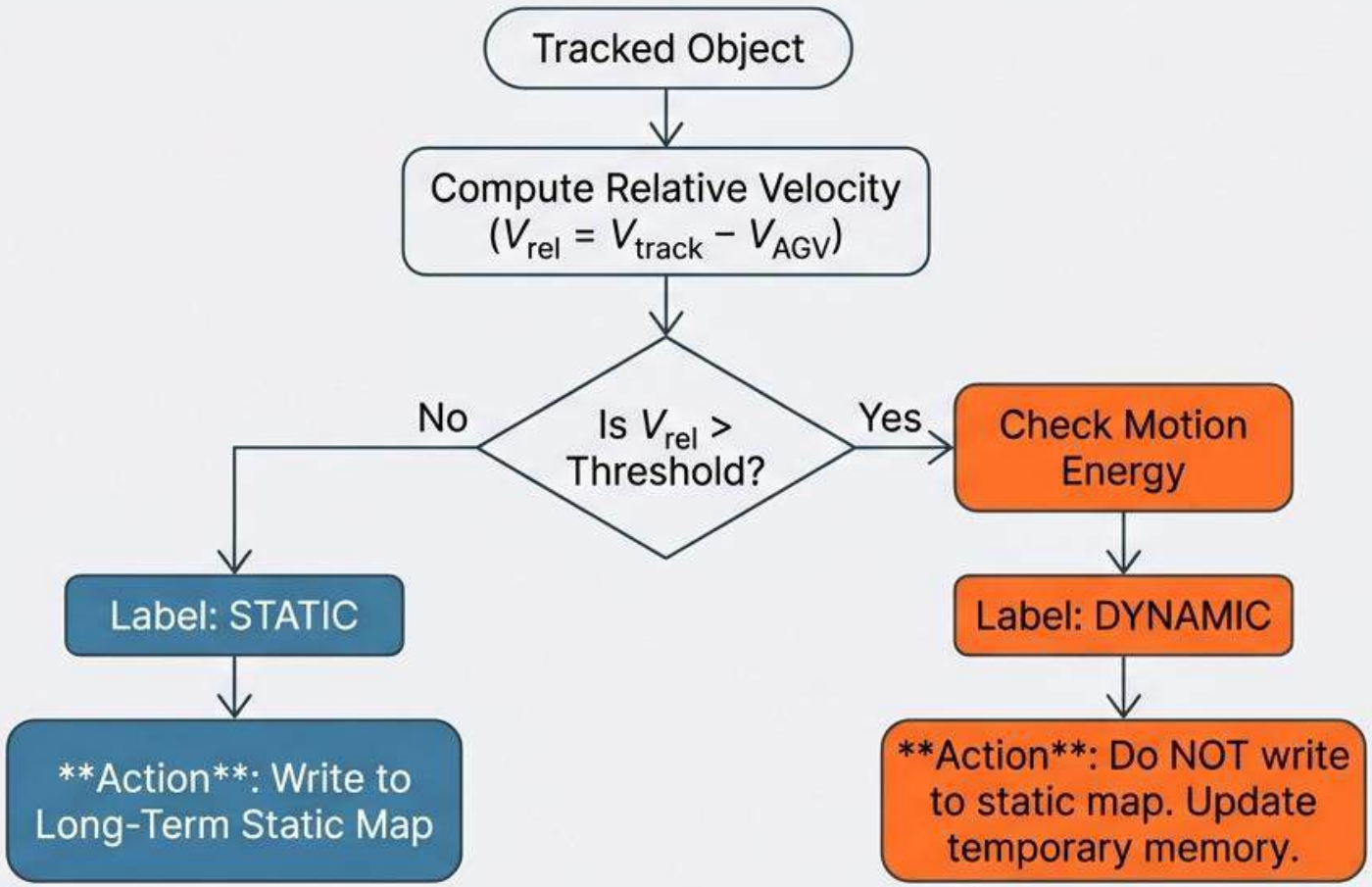


  - ***Adaptive Velocity Damping***: Smooths out jittery measurements for stable estimation.



Milad Jafari Barani - Milad.jafare@gmail.com

Step 4: Classification Logic & Decision Fusion



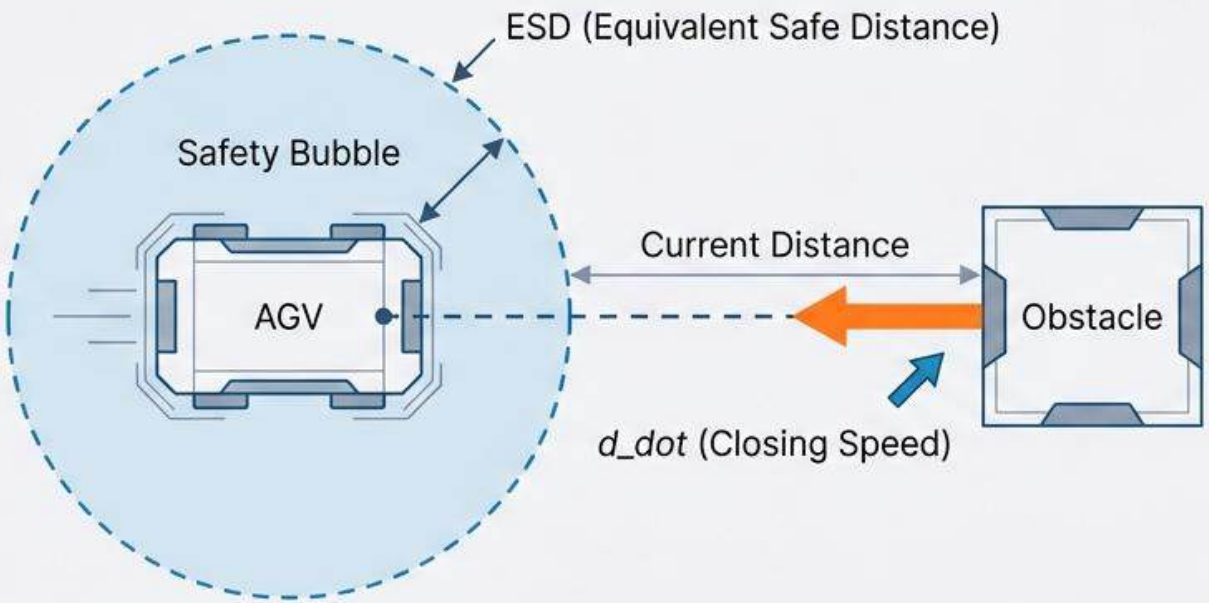
Sidebar Text

Logic: Relative velocity checks + Motion Energy.

Goal: Prevents "ghost obstacles" from polluting the navigation map.

Source Rule: "If `label == Static and confidence high ->` **allow writing to long-term map.**" in `JetBrains Mono`.

Step 5: HySDG-ESD Safety Metrics



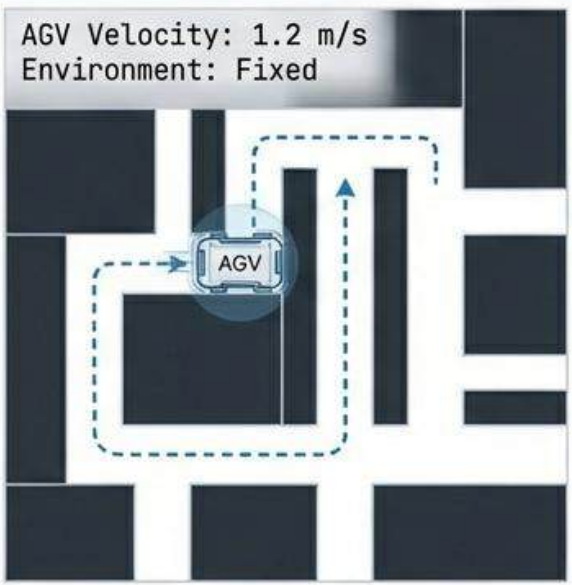
ESD (Equivalent Safe Distance): A dynamic safety buffer computed based on current speed and proximity.

***d_dot* (Distance Variation Rate):** The temporal derivative of distance (how fast is the gap closing?).

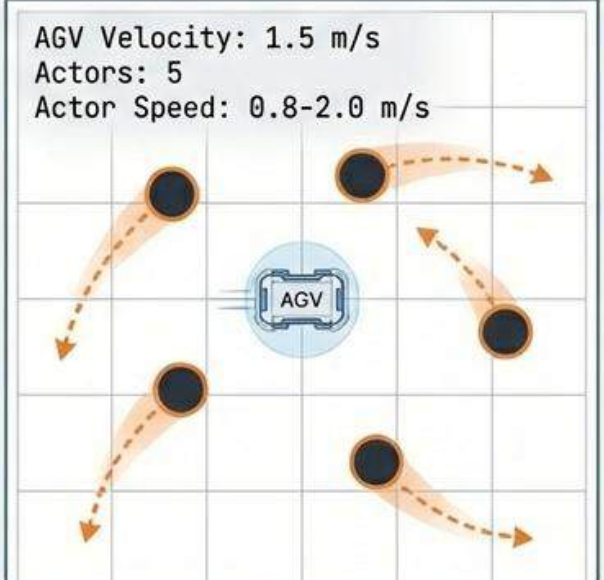
Controller Output: `[Obstacle_ID, Label, Position, ESD_Value]` published to ROS2 topic.

Simulation Scenarios

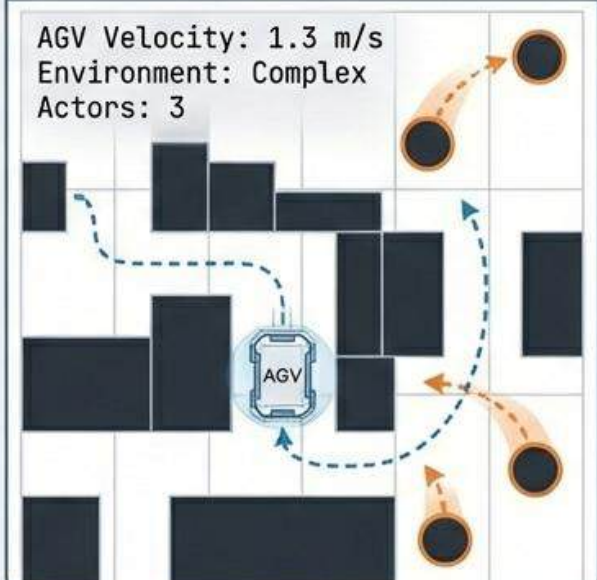
Static Mode



Dynamic Mode



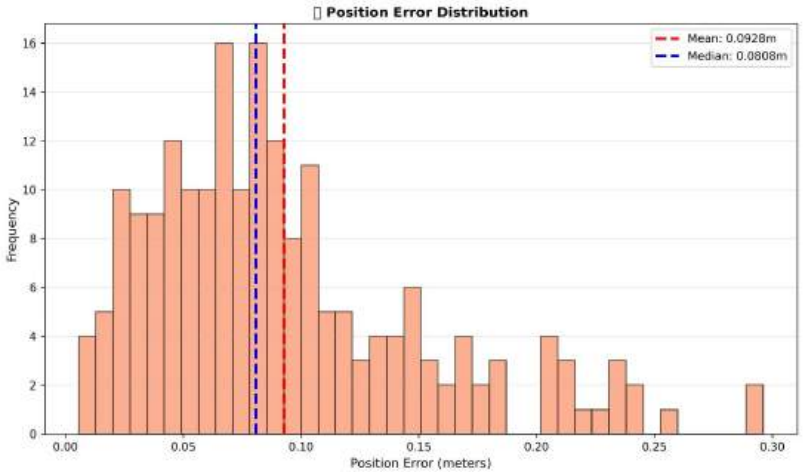
Mixed Mode

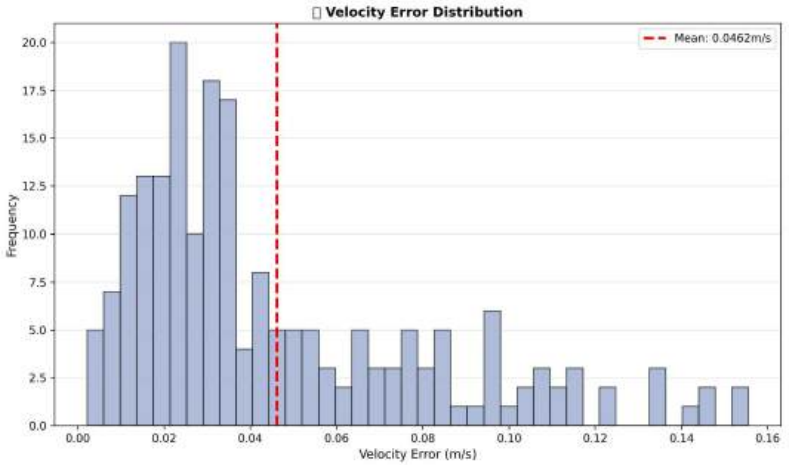


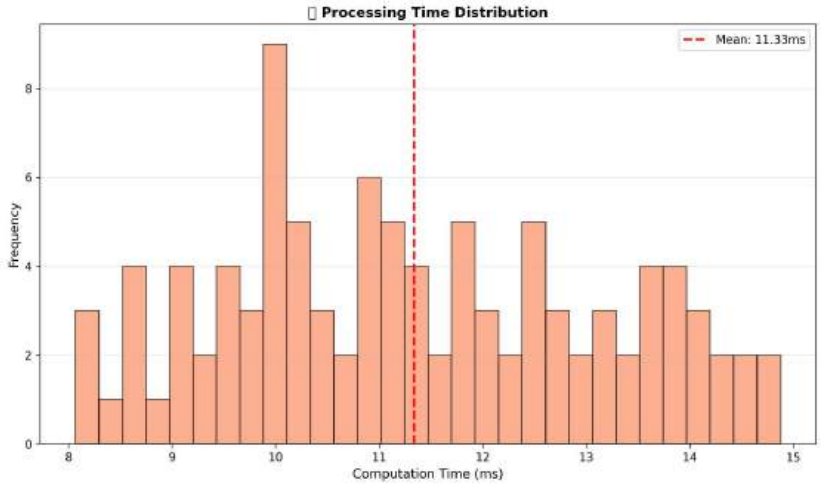
The AGV remains bounded within the environment while calculating real-time kinematics and classification.

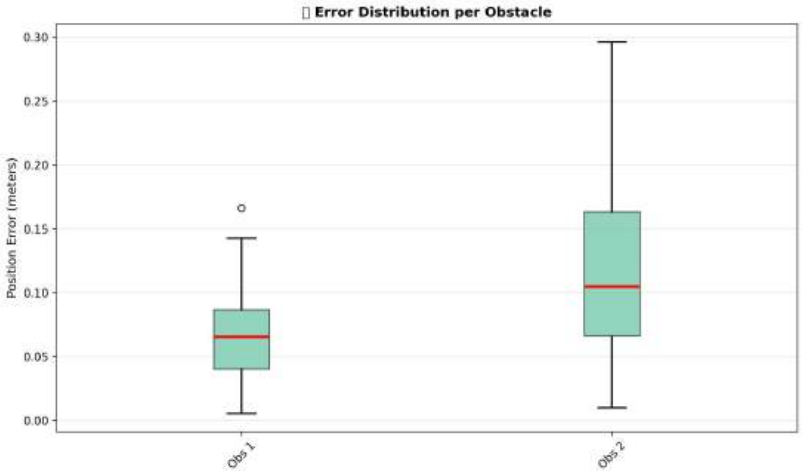
Milad Jafari Barani - Milad.jafare@gmail.com

Obstacle detection: Results










```
PERFORMANCE SUMMARY

Detection:
  Precision: 1.000
  Recall:    1.000
  F1-Score:  1.000

Position (m):
  RMSE:      0.1101
  Mean:      0.0928
  95%ile:    0.2150

Velocity (m/s):
  RMSE:      0.0581
  Mean:      0.0462

Classification:
  Accuracy:  1.000
  S→D:       0
  D→S:       0

Computation:
  Mean:      11.33 ms
  FPS:       90.6
```

Object detection: Publication and future research

Proposed Publication Title: "HySDG-ESD: Rotation-Aware Dynamic Obstacle Detection for Industrial AGVs Using Lightweight Clustering and Ego-Motion Compensation"

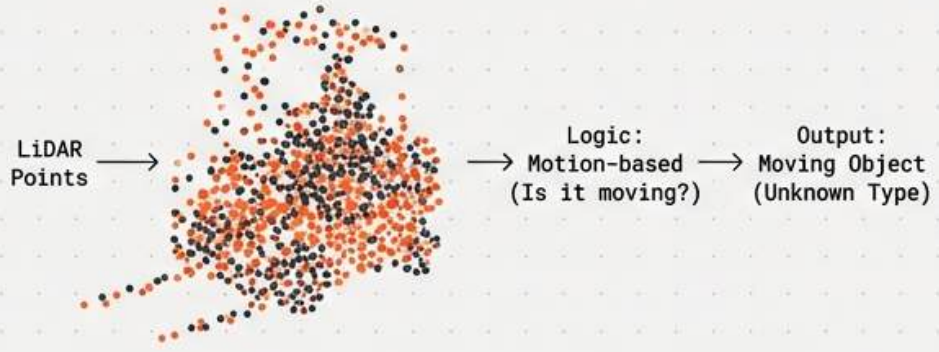
Extension for TUAI Objectives:

- Focus on Time Series Analysis (DC2): Treat the sequence of LiDAR scans and velocity vectors as multivariate time series. Apply anomaly detection techniques (Leonardo's expertise) to identify irregular obstacle behaviors (e.g., a forklift moving erratically).
- Focus on Lightweight Models: Emphasize benchmarking the computational cost of HySDG-ESD against deep learning alternatives (like YOLO) to prove its suitability for edge devices (Raspberry Pi/Jetson).

Future Directions: From Geometric to Semantic

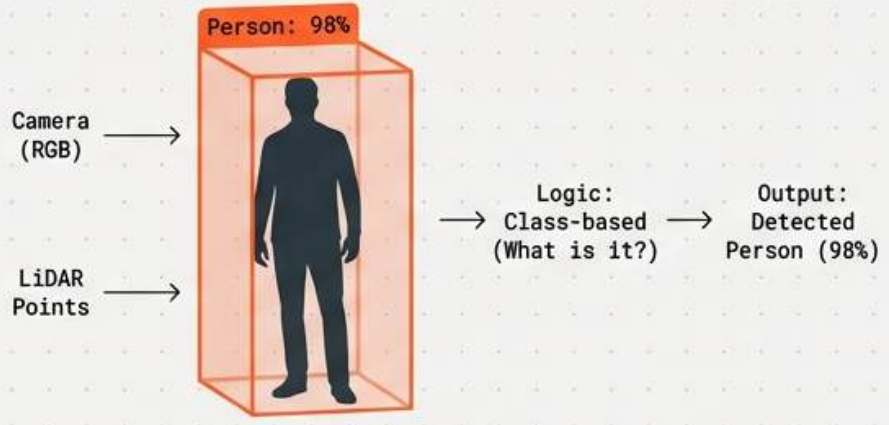
Integrating Deep Learning for context-aware safety.

Current State (Geometric)

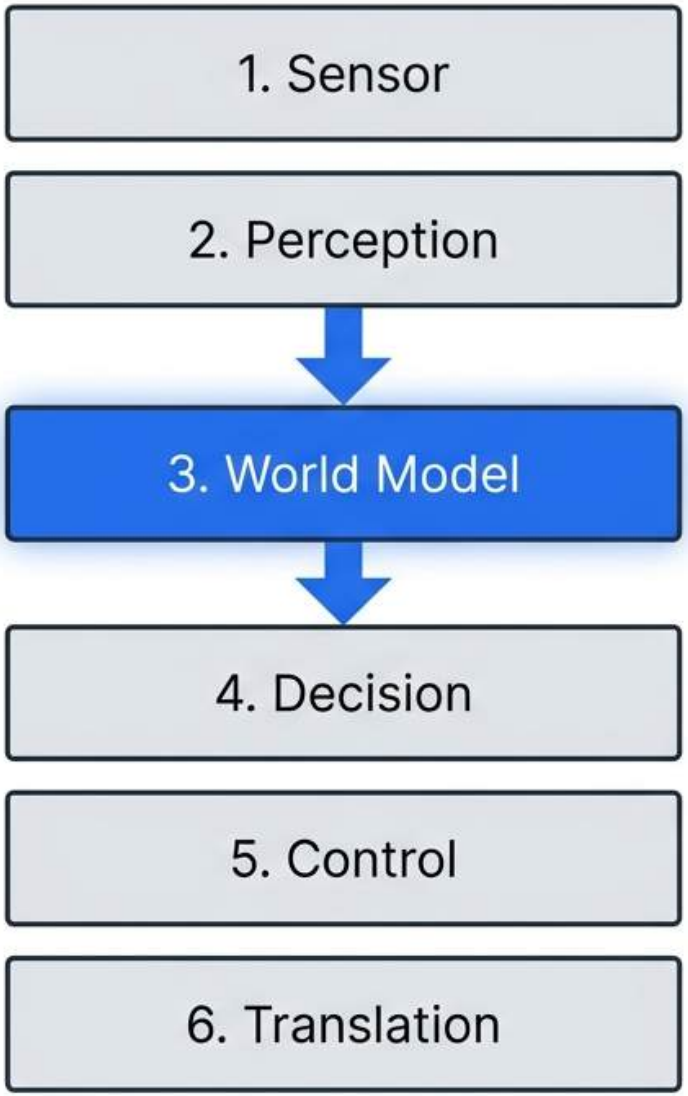


- **Input:** LiDAR Points
- **Logic:** Motion-based (Is it moving?)
- **Limit:** Cannot distinguish Human vs. Forklift if stationary.

Future State (Semantic)



- **Input:** Camera (RGB) + LiDAR
- **Logic:** Class-based (What is it?)
- **Goal:** Apply different safety margins for humans vs. infrastructure.



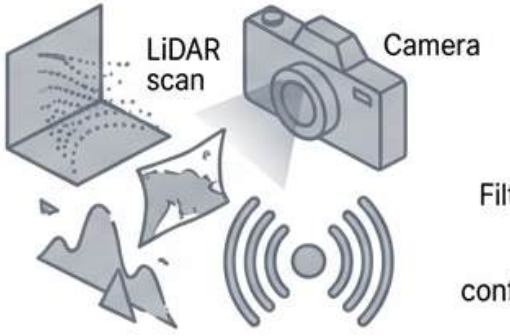
Layer 3: Unified World Model

A Probabilistic Framework for Environment Representation and Multi-Sensor Fusion

Module Responsibility: Environment synthesis and state consistency. Distinct from Detection (L2) and Planning (L4).

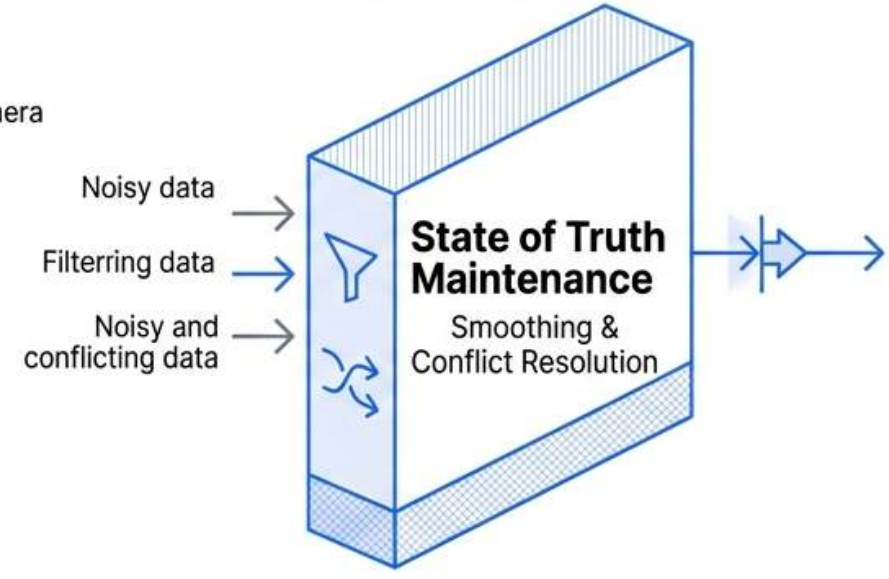
The Bridge Between Perception and Action

Inputs: Raw Chaos

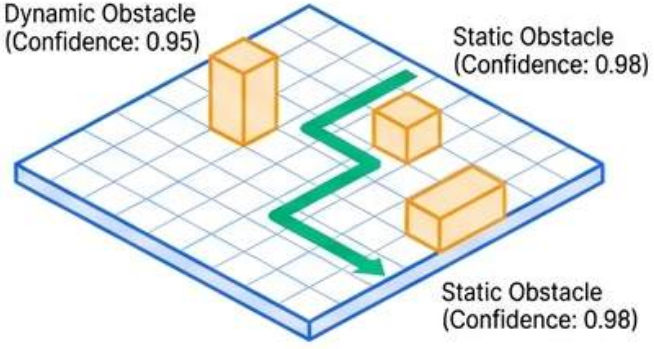


Heterogeneous Sensor Data

Layer 3: Synthesis



Outputs: Ordered Reality



OccupancyGrid + TrackedObjects

The Objective

To transform disparate sensor inputs into a single, reliable 2D representation that downstream Decision layers can trust.

The Function

Layer 3 does not detect (L2) and does not decide (L4). It synthesizes. It smooths sensor jitter and resolves conflicting data over time.

Data Interfaces

Input: PerceptionOutput (Position, Velocity).

Output: OccupancyGrid + TrackedObjects.

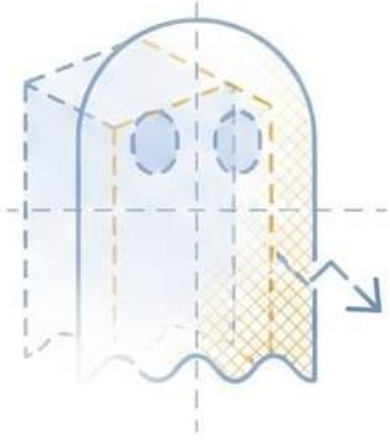
The Challenge: Unifying Heterogeneity and Uncertainty

Temporal Misalignment

Sensors operate at different frequencies (LiDAR 10-30Hz, Cameras, Ultrasonics). Direct timestamp fusion is prone to latency errors, necessitating a state-based update approach.

Data Conflicts

Handling discrepancies where one sensor detects an object (e.g., Camera) while another (e.g., LiDAR) reports empty space. Who is right?



Ephemeral Data ('Ghosts')

Sensor noise can create temporary "ghost obstacles." Without memory, these artifacts cause the AGV to stop unnecessarily. The system must decay old data to filter noise.

Spatiotemporal Representation: The Occupancy Grid

1. Grid Specification

Dimensions: 100x100 (20x20m area).
Resolution: 0.2m per cell.
Values: 0.0 (Free) to 1.0 (Occupied).

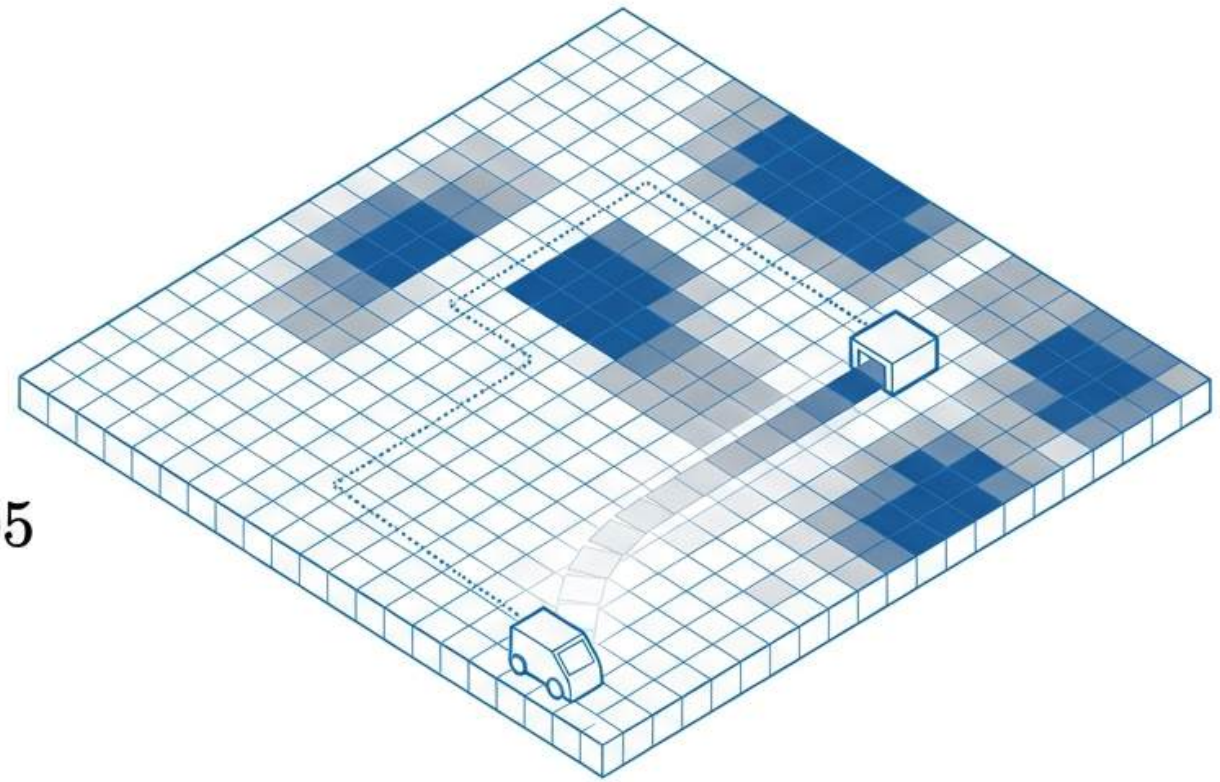
2. Dynamic Decay Algorithm

To prevent ghost obstacles, cells decay toward uncertainty (0.5) if not reinforced.

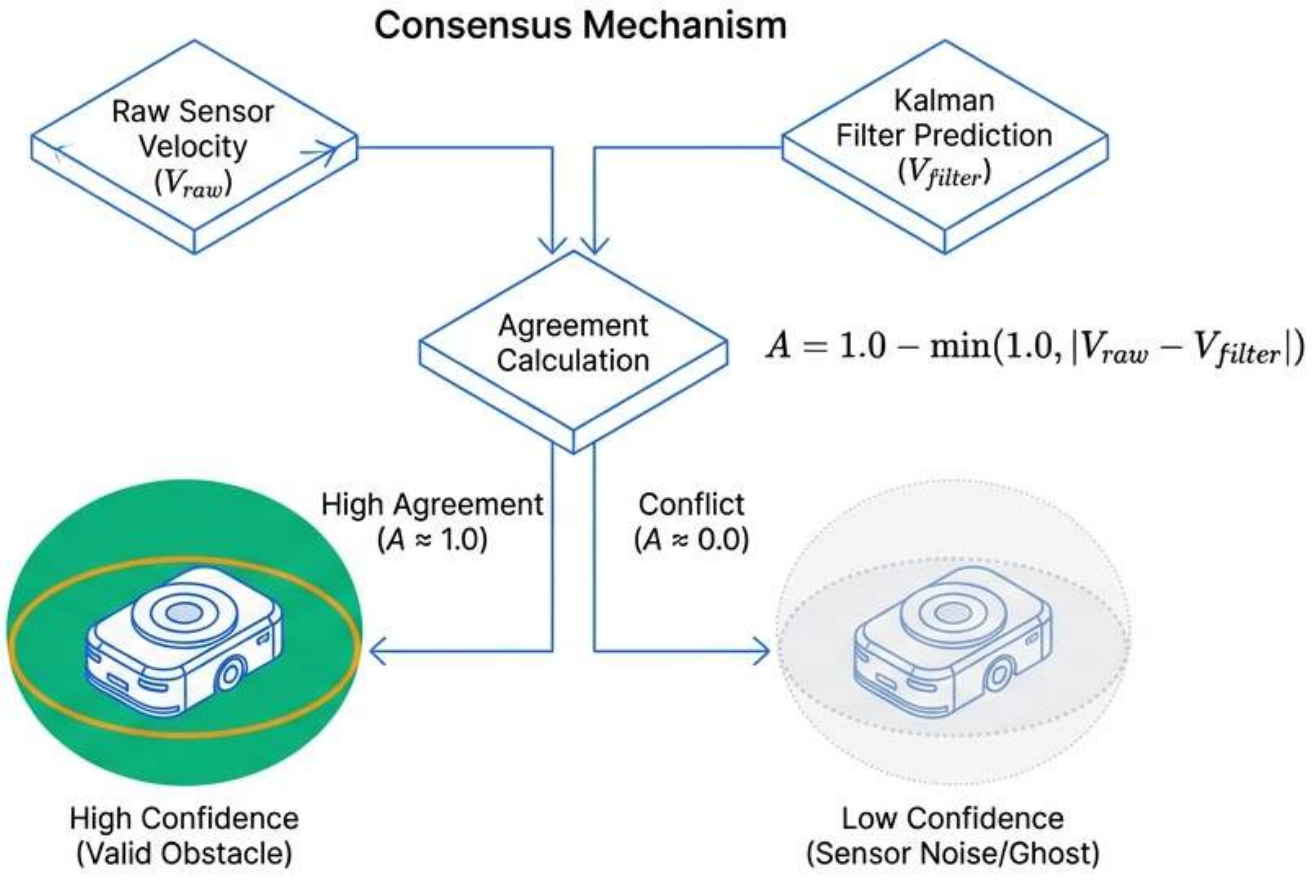
$$P_{new} = P_{old} \times 0.95 + 0.5 \times 0.05$$

3. Interpretation Thresholds

Free: $P < 0.3$
Unknown: $0.3 \leq P \leq 0.7$
Occupied: $P > 0.7$



Handling Uncertainty via Feature-Level Consensus



The Logic

We fuse features, not raw data. By comparing the raw observation against physics-based predictions (EKF), we calculate a "Trust Score".
Only validated features populate the map, preventing erratic braking.

World model: Publication and future research

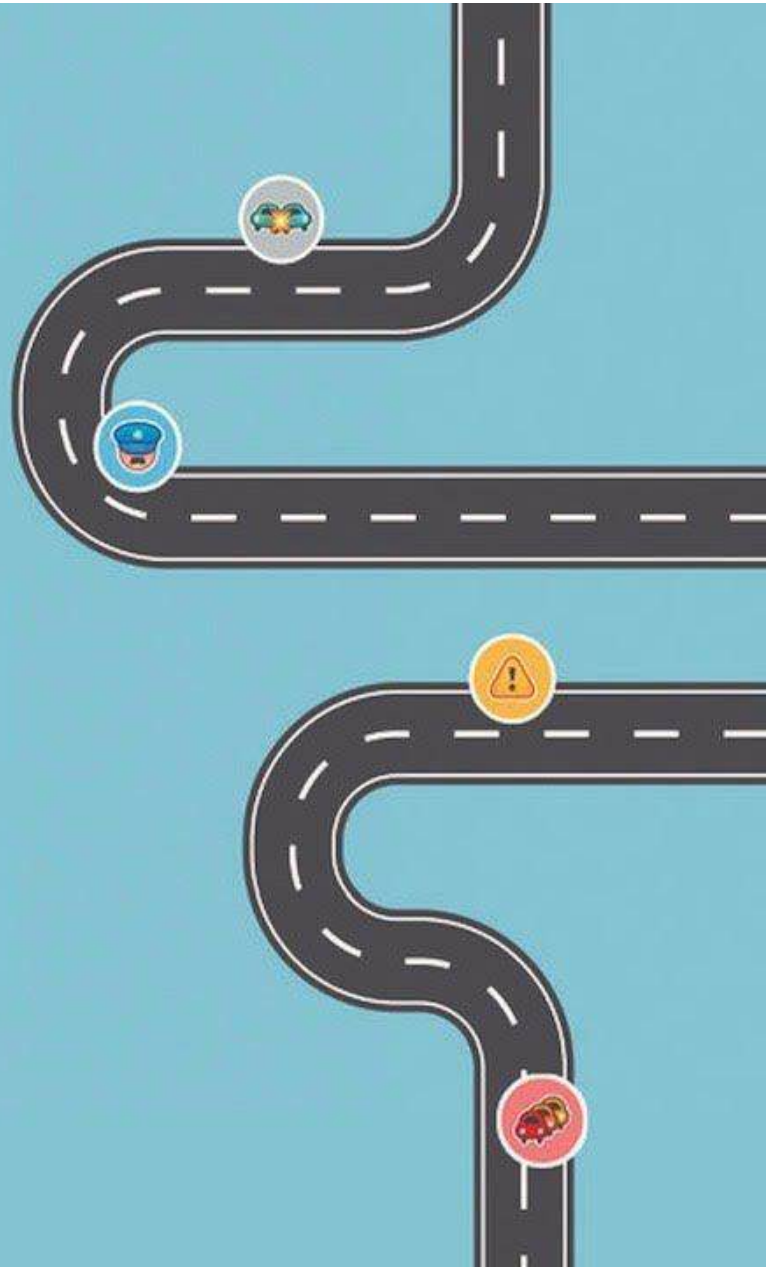
Proposed Publication Title: "Algorithmic Consensus in Occupancy Grids: A Probabilistic Feature-Level Fusion Approach for Mitigating Sensor Noise in Industrial Environments"

Extension for TUAI Objectives:

- Focus on Federated Learning (DC9): Extend the world model to be shared across a fleet of AGVs. Instead of a centralized server processing the map, use Federated Learning to allow AGVs to update a shared semantic map (e.g., identifying a temporary construction zone) without sharing raw sensor data, preserving bandwidth and privacy

REROUTING

LIVING IN A NEW DIRECTION



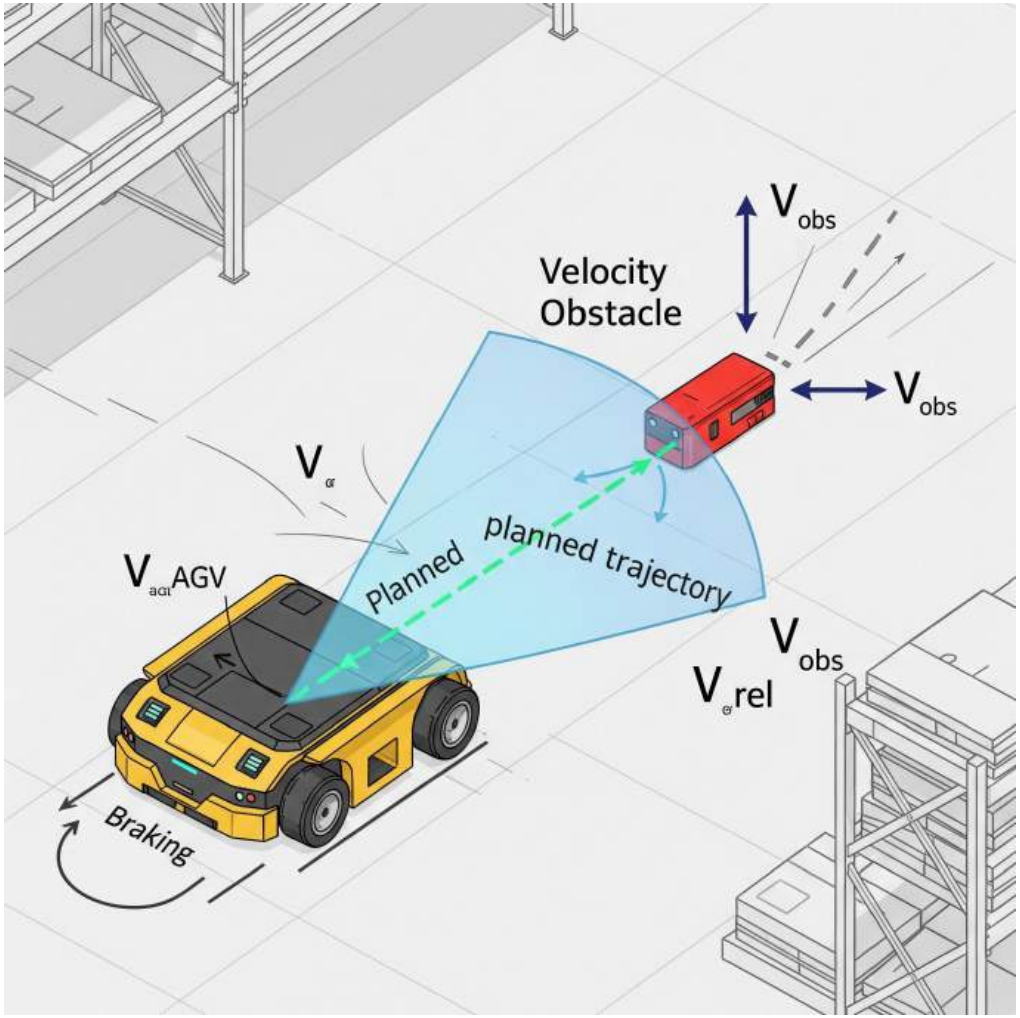
Velocity Obstacle: Why?

Velocity Obstacle (VO) algorithm chosen for L5.

- It enables effective collision prediction based on **TTC (Time-To-Collision)**.
- It handles the differentiation between **static and dynamic obstacles**.
- It implements **emergency logic** such as stopping and reversing.

Empirical Comparison

- Other methods showed **stuck/drifting** issues and needed heavy tuning.
- With tuned hyperparameters, VO was the most stable.
- It avoided obstacles well and went **straight to the goal**.



Results: 100 tests, 15 obstacles

Algorithm	Success	Steps	Time
GapNav	88%	492	6.55s
DWA	88%	511	10.89s
VFH	85%	810	7.52s

Navigation: Publication and future research

Proposed Publication: Comparison study on rerouting algorithms using the developed environment.

Extension for TUAI Objectives:

- Focus on Energy Efficiency (Green Manufacturing - DC7): Extend the navigation cost function to include battery consumption. Instead of just "shortest path" or "safest path," optimize for "energy-efficient smoothness," minimizing sharp accelerations and braking, which drains battery and increases wear.

Track 3 Contribution

PRACTICAL / ALGORITHMIC

Concrete Algorithms for Each Layer

We instantiated the theoretical model with specific, tested algorithms: HySDG-ESD for perception (with DBSCAN clustering and Kalman tracking), occupancy grids for world modeling, and Velocity Obstacle algorithm for navigation.

L2 Perception
HySDG-ESD

L3 World Model
Occupancy Grid

L5 Control
Velocity Obstacle

TRACK 3

End-To-End Validation

Integration Testing & Future Work

Full Stack Integration • End-to-End Testing • Hardware Deployment

Track 3: Developed Open-source software

README MIT license

HySDG-ESD AGV Simulator

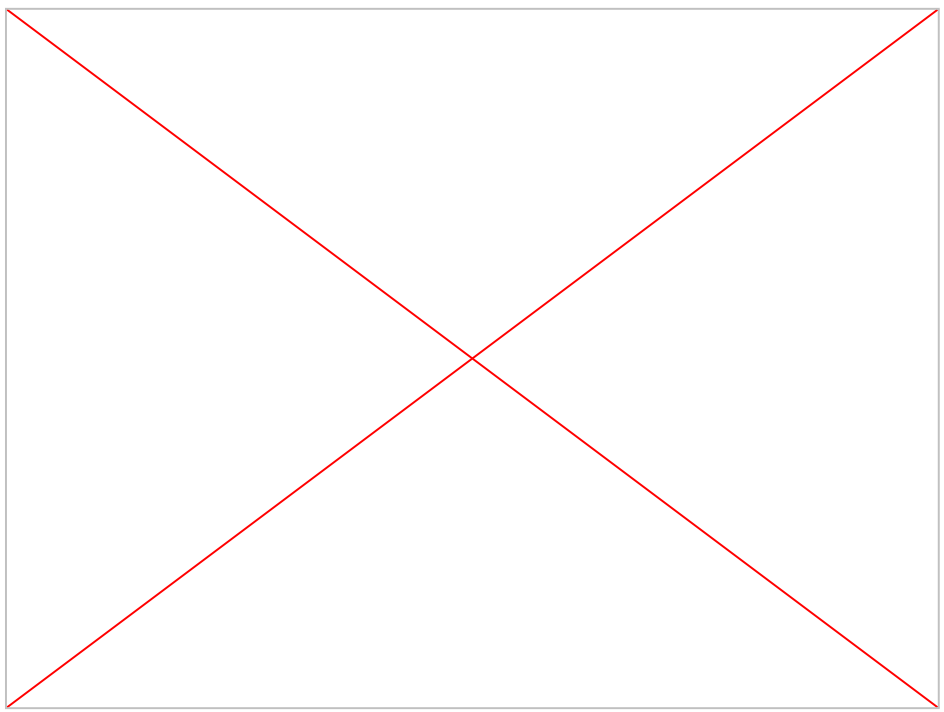
A multi-layer AGV (Autonomous Guided Vehicle) navigation simulator with LiDAR-based obstacle detection, tracking, classification, and autonomous decision making.

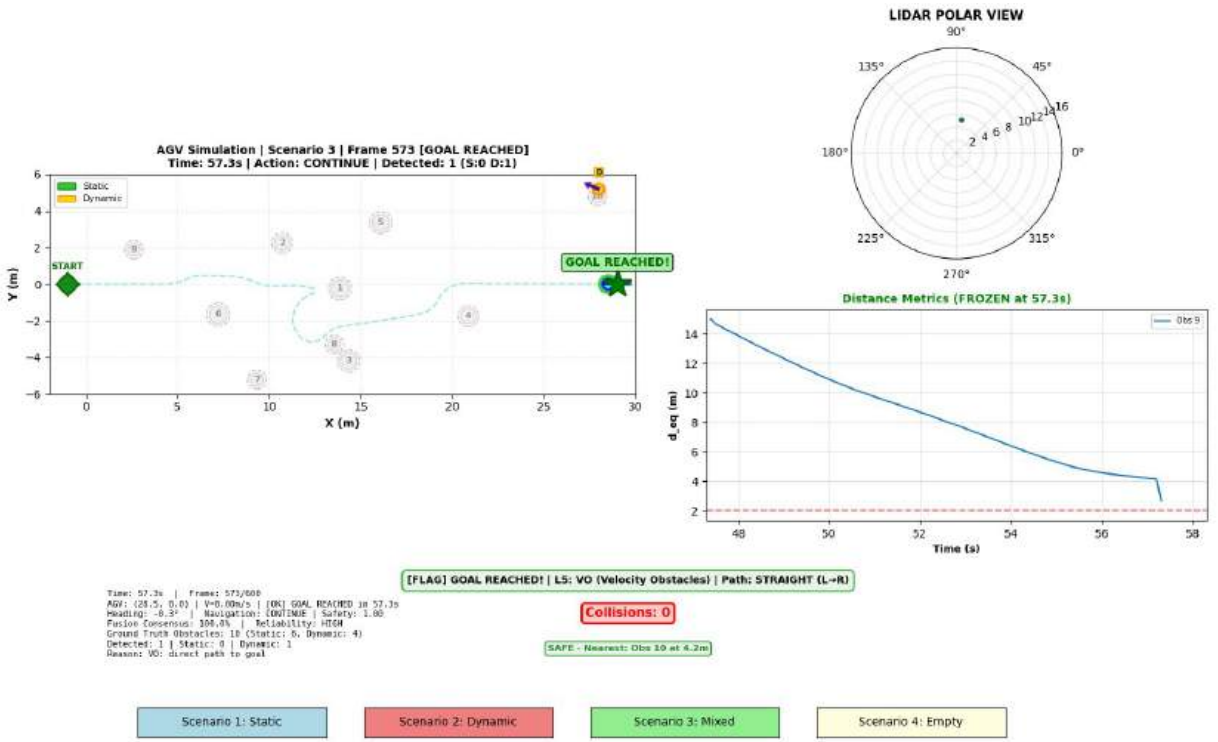
✨ Features

- **Modular Package Architecture**: Clean separation between World Model (`L3_world`), Detection (`L4_detection`), and Decision (`L5_decision`) packages
- **Multiple Navigation Algorithms**:
  - Simple reactive navigation (repulsive field)
  - VFH (Vector Field Histogram)
  - DWA (Dynamic Window Approach)
  - GapNav + APF (Gap Navigation with Artificial Potential Fields)
  - VO (Velocity Obstacles with TTC for dynamic obstacles, static/dynamic-aware, conservative safety logic)
    - Distinguishes static vs dynamic obstacles for safety margins
    - Emergency reverse and stop logic for critical threats
    - Allows closer approach to static obstacles, more conservative for dynamic
    - Ignores obstacles moving away, goes straight to goal if clear
- **Realistic Visualization**: AGV and obstacles are drawn with true physical radii and safety margins
- **Status Feedback**: Colored status box (safe, warning, danger, collision) in UI info panel (no emojis)
- **Unified Configuration**: All VO and navigation parameters are in `L5_decision/config.py` for easy tuning
- **2D LiDAR Simulation**: Configurable noise, range, and field of view
- **DBSCAN Clustering**: Real-time obstacle clustering from point clouds
- **EKF Tracking**: Extended Kalman Filter for multi-object tracking
- **HySDG-ESD Classification**: Dynamic vs static obstacle classification with ego-motion compensation
- **Unified Configuration**: Each package has its own `config.py` for easy tuning
- **Interactive Visualization**: Real-time plots with scenario switching

https://github.com/lrdsch/AGV_L4L5

Track 3: Demo




AGV Simulation | Scenario 3 | Frame 573 [GOAL REACHED]
Time: 57.3s | Action: CONTINUE | Detected: 1 (S:0 D:1)
Static
Dynamic
START
GOAL REACHED!
X (m)
Y (m)
LIDAR POLAR VIEW
Distance Metrics (FROZEN at 57.3s)
d_eq (m)
Time (s)
[FLAG] GOAL REACHED! | L5: VO (Velocity Obstacles) | Path: STRAIGHT (L→R)
Collisions: 0
SAFE - Nearest: Obs 10 at 4.2m
Scenario 1: Static
Scenario 2: Dynamic
Scenario 3: Mixed
Scenario 4: Empty

Executive Summary

98.1%	100%	ZERO	>0.3m
Overall Success Rate	Low-Density Success	Static Collisions	Safety Maintained

Test Configurations

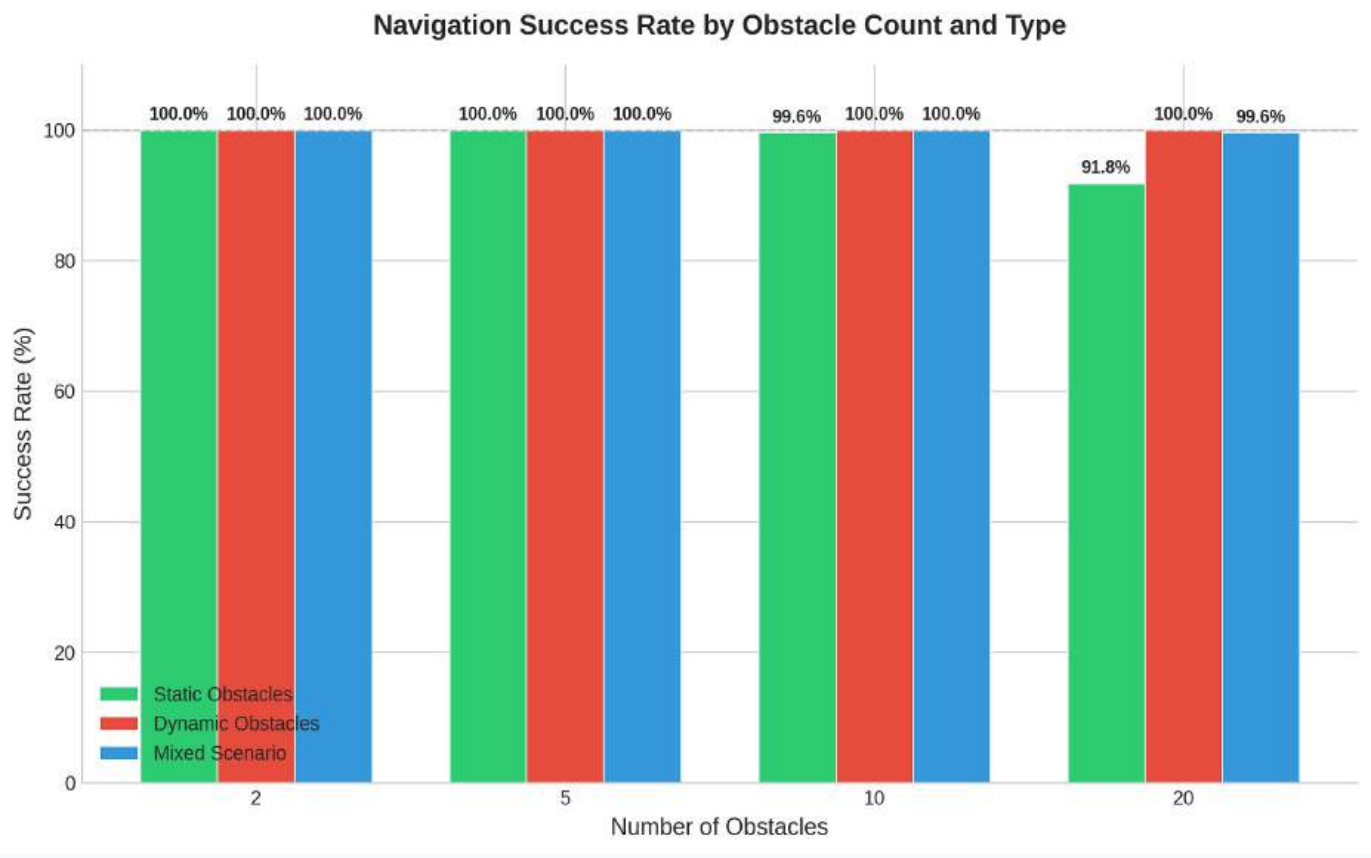
Obstacle Densities: 2, 5, 10, 20 obstacles
Scenario Types: Static, Dynamic, Mixed
Runs per Config: 1,000 Monte Carlo simulations

Velocity Obstacles (VO)

- TTC-based collision prediction
- Static/Dynamic differentiation
- Multi-strategy avoidance
- Emergency stop & reverse logic

Navigation Success Rate

Goal achievement across obstacle densities and scenario types



Key Findings

100% success at 2-5 obstacles

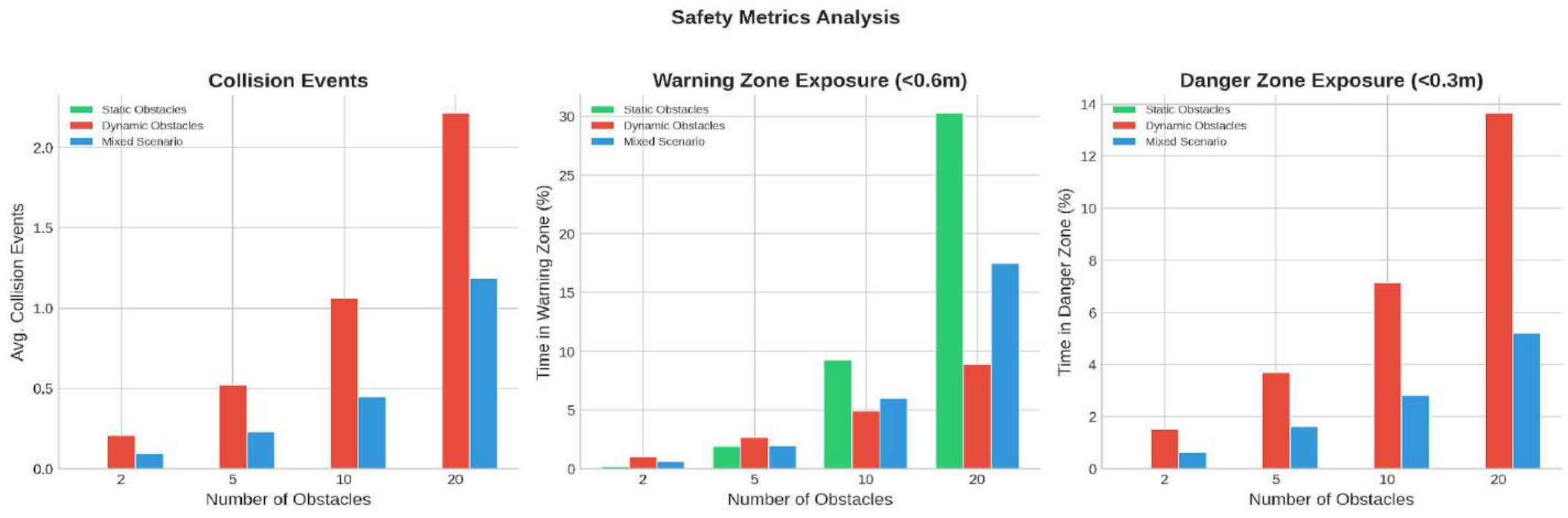
Dynamic: 100% even at 20 obstacles

Static @ 20: 91.8% (narrow passages)

Mixed: 99.6% (best real-world match)

Safety Performance Analysis

Collision events, warning zones, and danger zone exposure

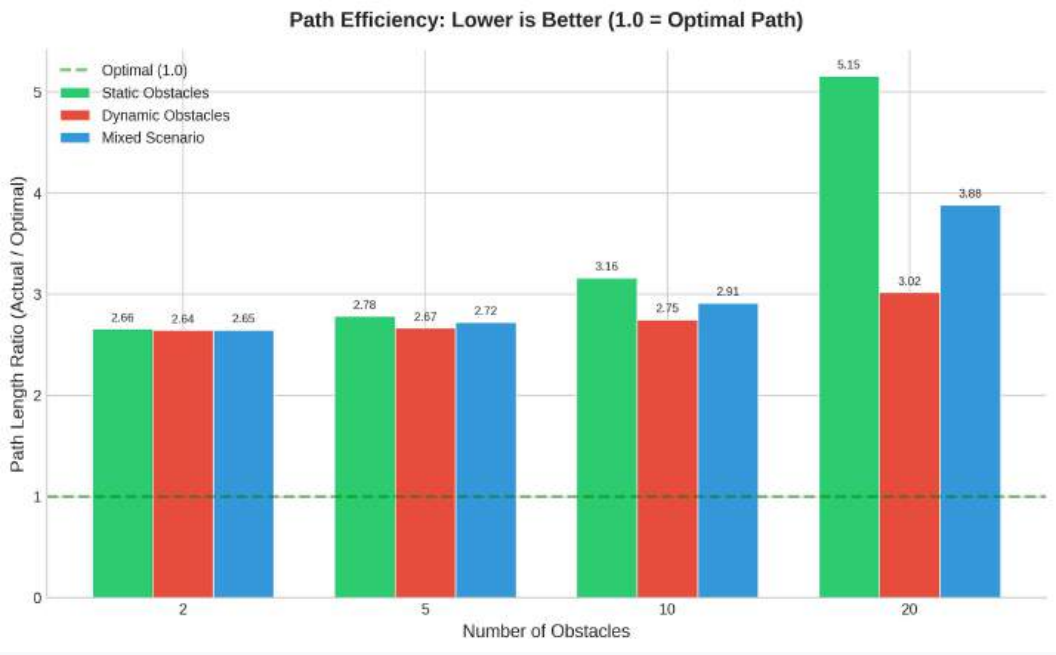


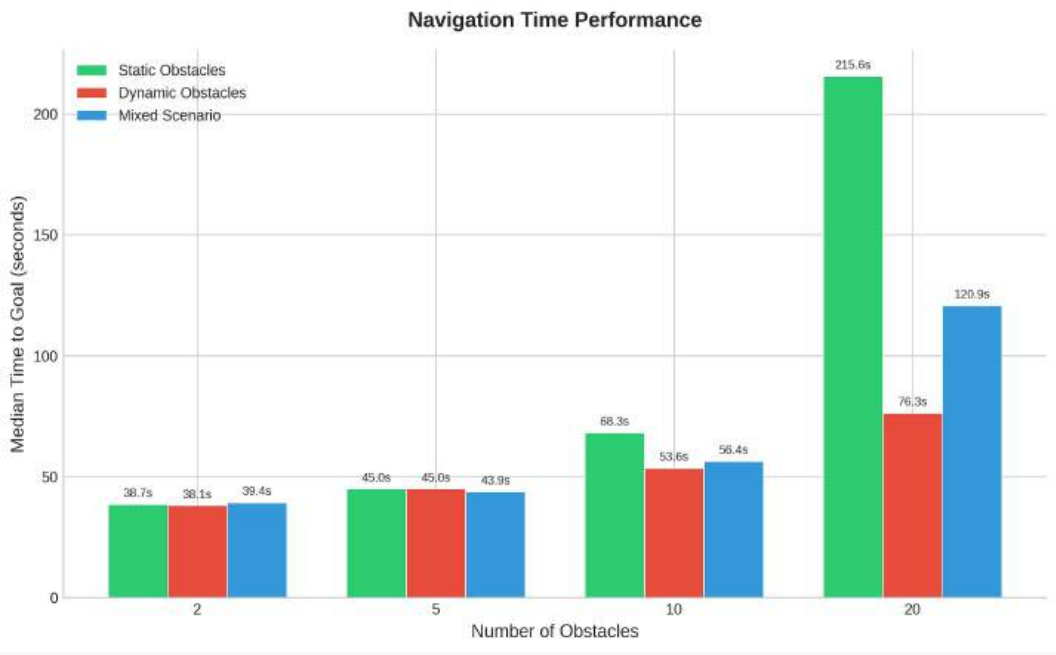
✓ Zero collisions in all static scenarios

! Dynamic collisions scale with density

◉ Danger zone (<0.3m) exposure minimal

Path Efficiency & Navigation Time





Path Efficiency

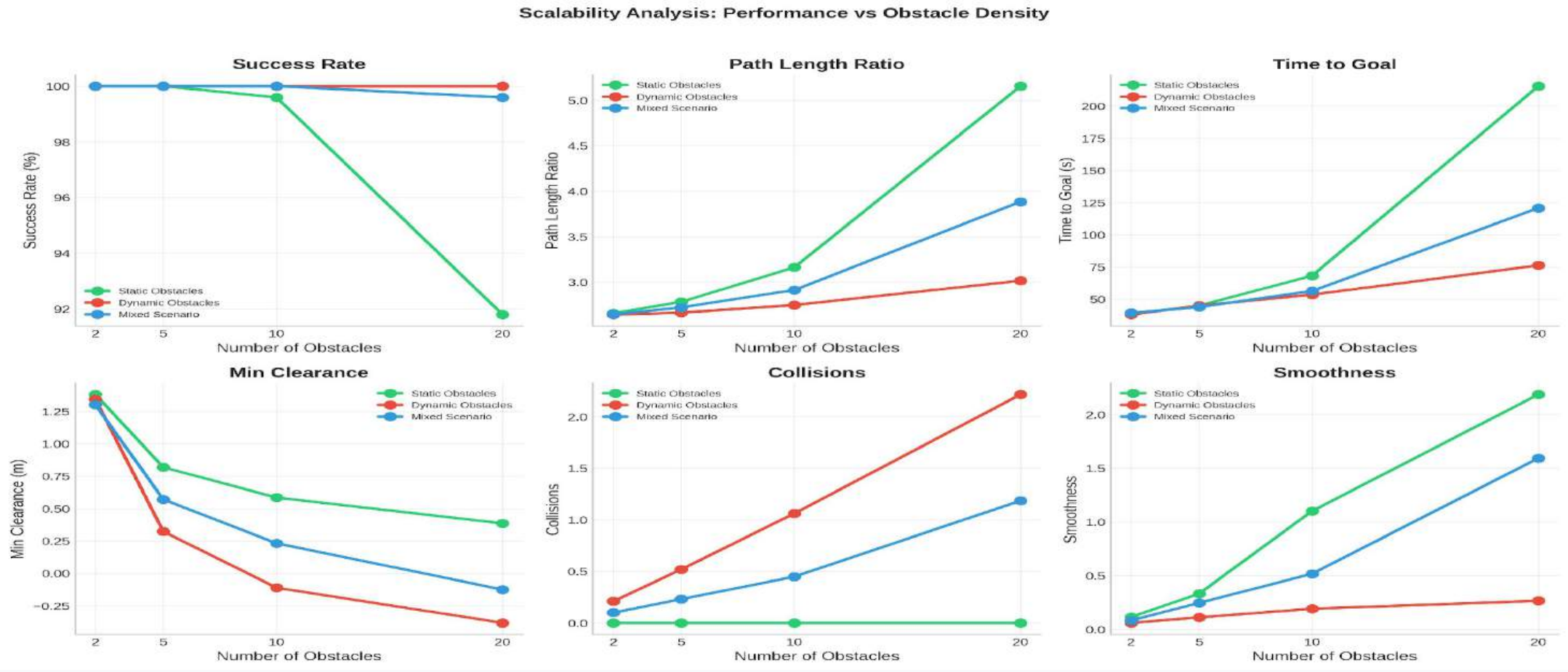
~2.65× optimal at low density
Degrades gracefully to 3-5× at 20 obstacles

Navigation Time

Baseline: ~38s for simple scenarios
Dynamic faster at high density (no blocking)

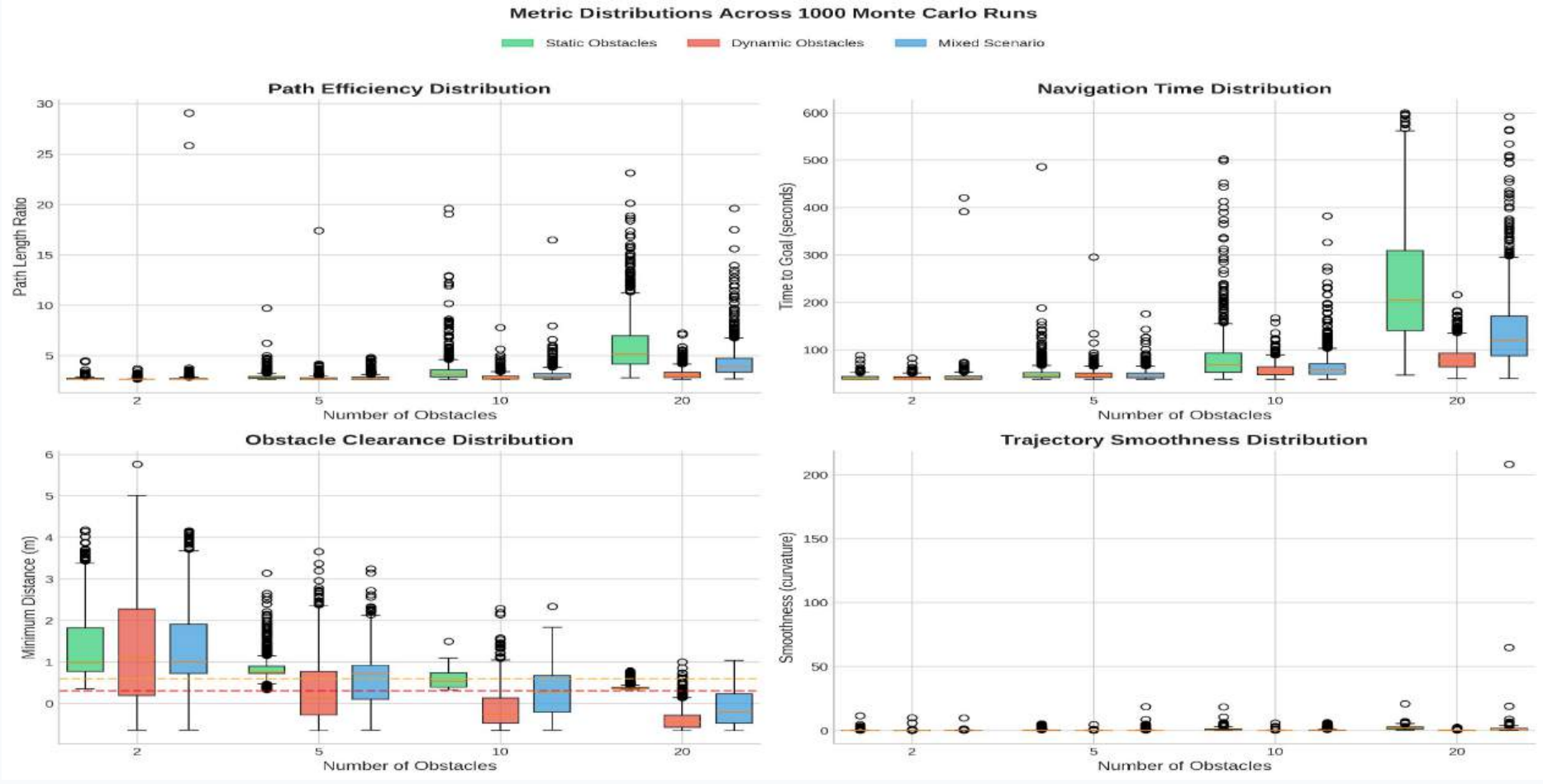
Scalability Analysis

Performance trends across increasing obstacle complexity



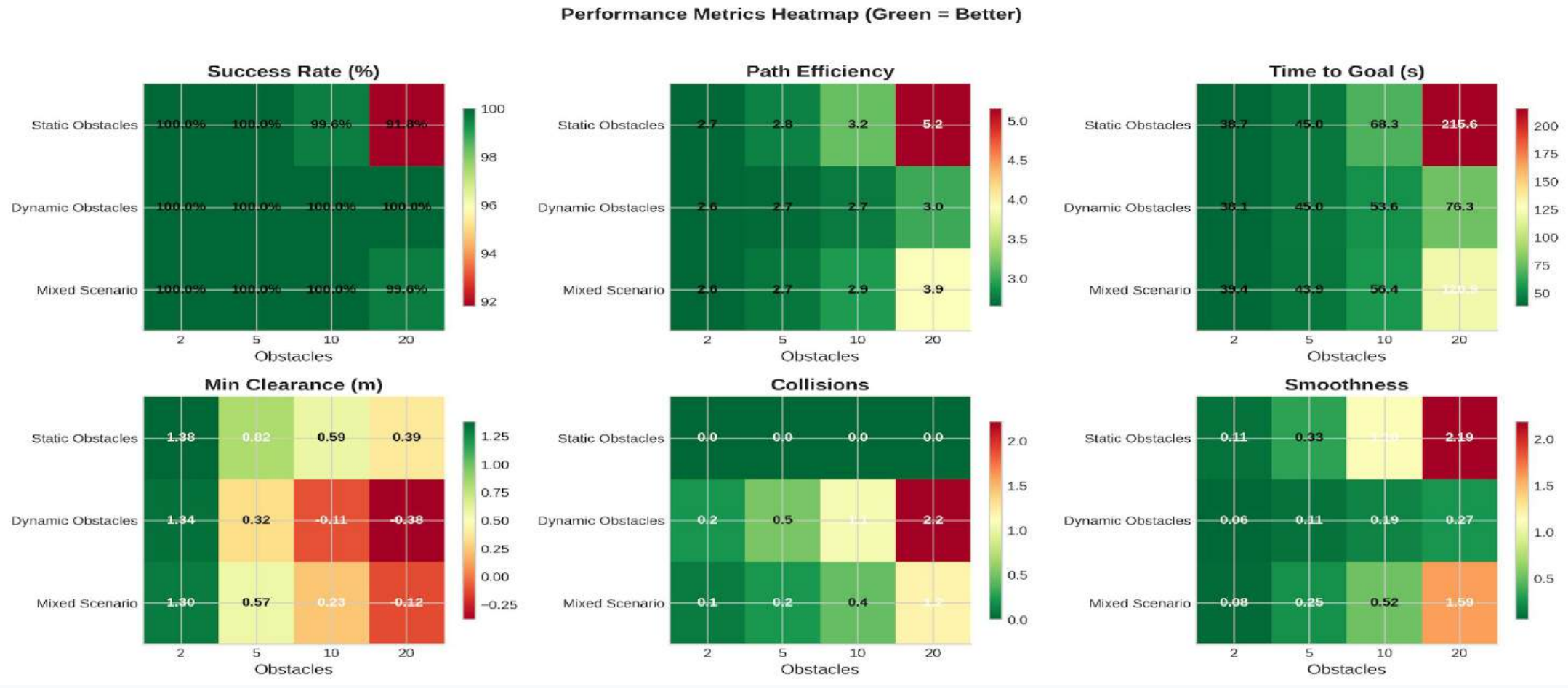
Statistical Distributions

Box plots showing variability across 1,000 runs per configuration



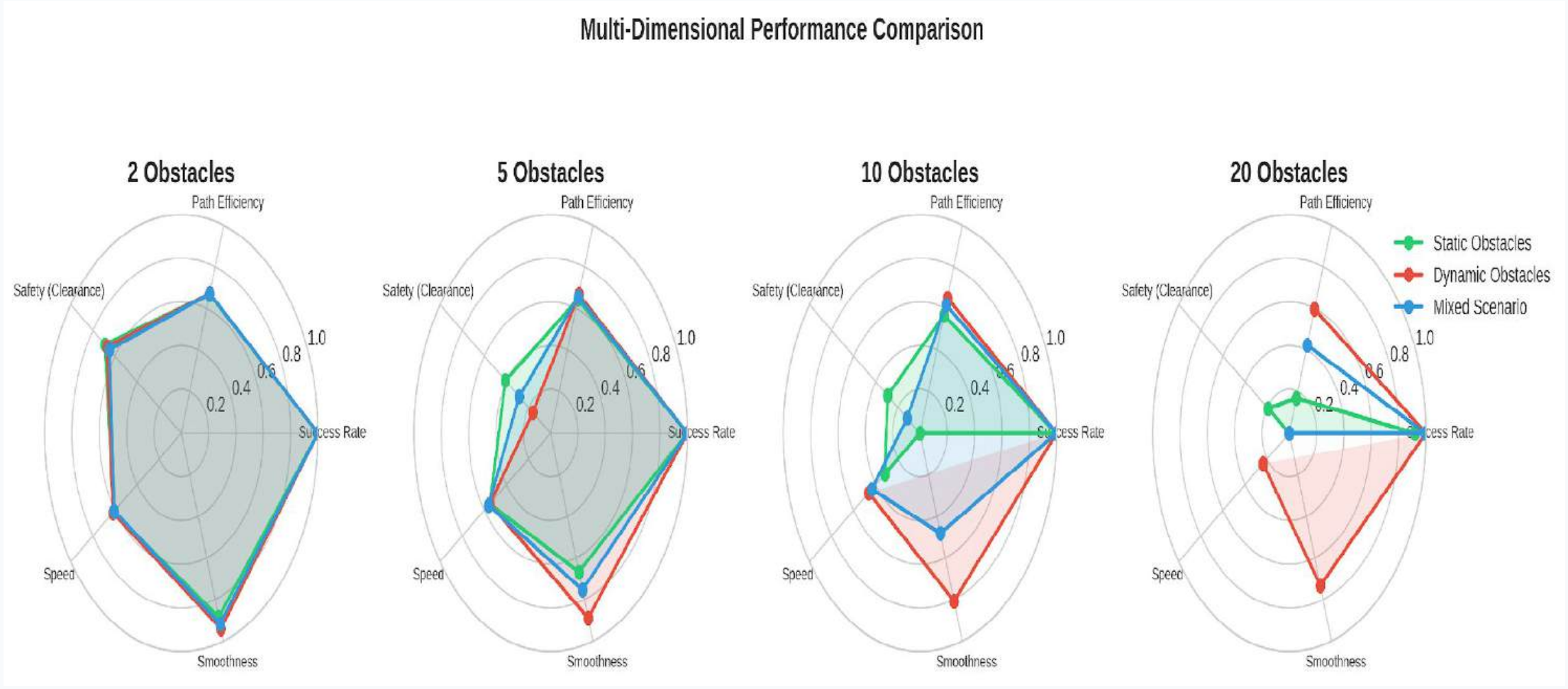
Performance Metrics Heatmap

Comprehensive view: Green = Better performance

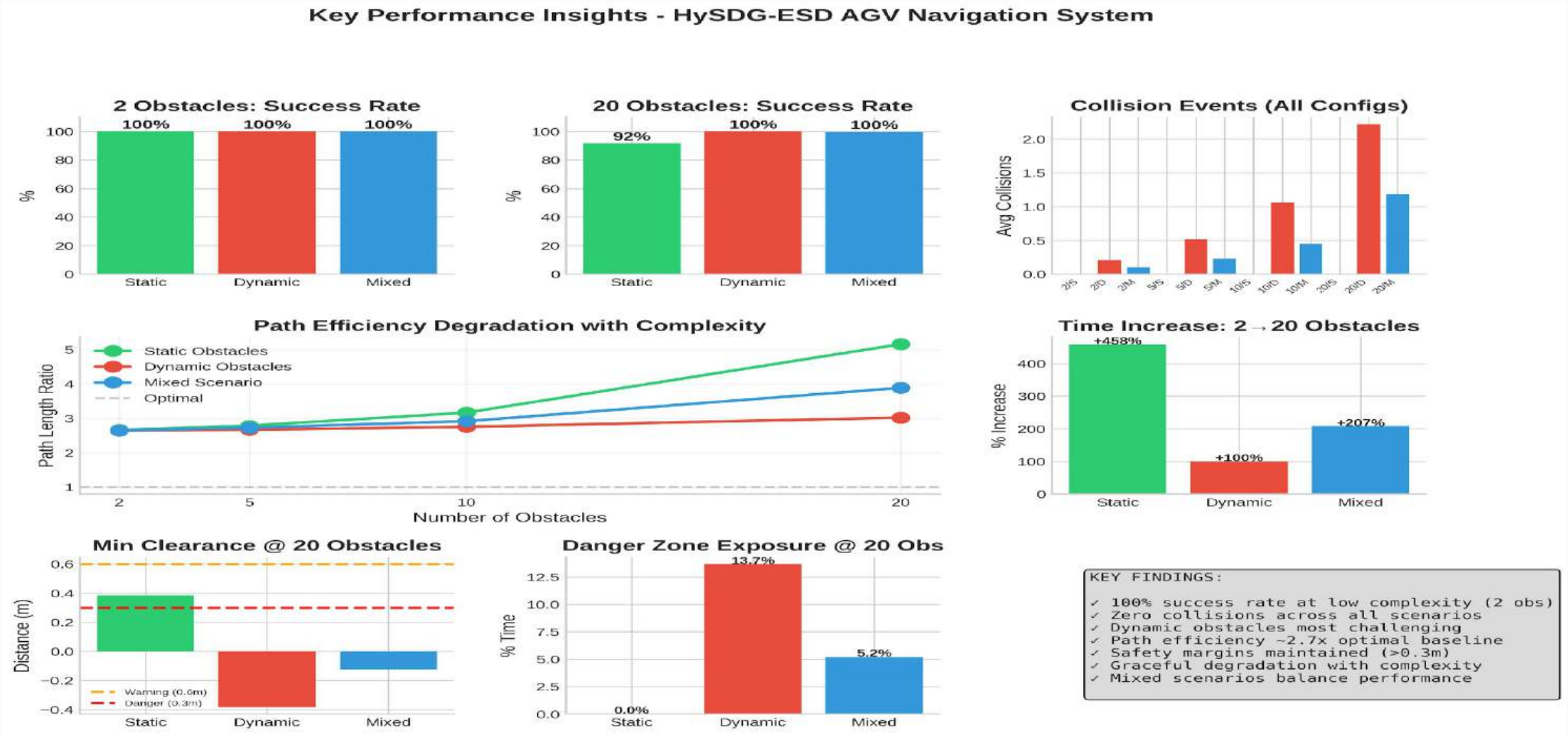


Multi-Dimensional Performance Comparison

Radar charts comparing scenarios across 5 key metrics



Key Insights Dashboard



Summary Statistics

Complete metrics across all 12 configurations

Summary Statistics Table
(1000 Monte Carlo Runs per Configuration)

Config	Success Rate	Path Ratio	Time (s)	Min Dist (m)	Collisions	Smooth-ness
2 obs Static	100.0%	2.66	38.7	1.38	0.00	0.115
2 obs Dynamic	100.0%	2.64	38.1	1.34	0.21	0.060
2 obs Mixed	100.0%	2.65	39.4	1.30	0.10	0.083
5 obs Static	100.0%	2.78	45.0	0.82	0.00	0.333
5 obs Dynamic	100.0%	2.67	45.0	0.32	0.52	0.111
5 obs Mixed	100.0%	2.72	43.9	0.57	0.23	0.247
10 obs Static	99.6%	3.16	68.3	0.59	0.00	1.102
10 obs Dynamic	100.0%	2.75	53.6	-0.11	1.06	0.192
10 obs Mixed	100.0%	2.91	56.4	0.23	0.45	0.518
20 obs Static	91.8%	5.15	215.6	0.39	0.00	2.186
20 obs Dynamic	100.0%	3.02	76.3	-0.38	2.21	0.266
20 obs Mixed	99.6%	3.88	120.9	-0.12	1.19	1.593

Conclusions & Next Steps

Validated Achievements

✓ 98.1% overall success rate

✓ Zero collisions in static scenarios

✓ Safety margins consistently maintained

✓ Graceful performance degradation

✓ Real-time capable (<100ms cycles)

Next Steps

→ Real AGV hardware validation

→ YOLO/MobileNet camera integration

→ Multi-AGV coordination testing

→ Factory environment deployment

→ Research publication preparation

Track 3 Contribution

✓ **COMPLETED**
Individual layer tests

✓ **COMPLETED**
Algorithm benchmarks

✓ **COMPLETED**
End-to-end testing

End-To-End validation: Publication and future research

Proposed Publication: Use case study on collision avoidance

Extension for TUAI Objectives:

- Focus on Trustworthiness & Safety: Use the simulation environment to generate "edge cases" (e.g., sensor failure, reflective surfaces, smoke) that are dangerous to test in real life. Publish a dataset of these failure modes to benchmark the trustworthiness of collision avoidance models, aligning with the "Sustainability and Trustworthiness" theme of the TUAI network.

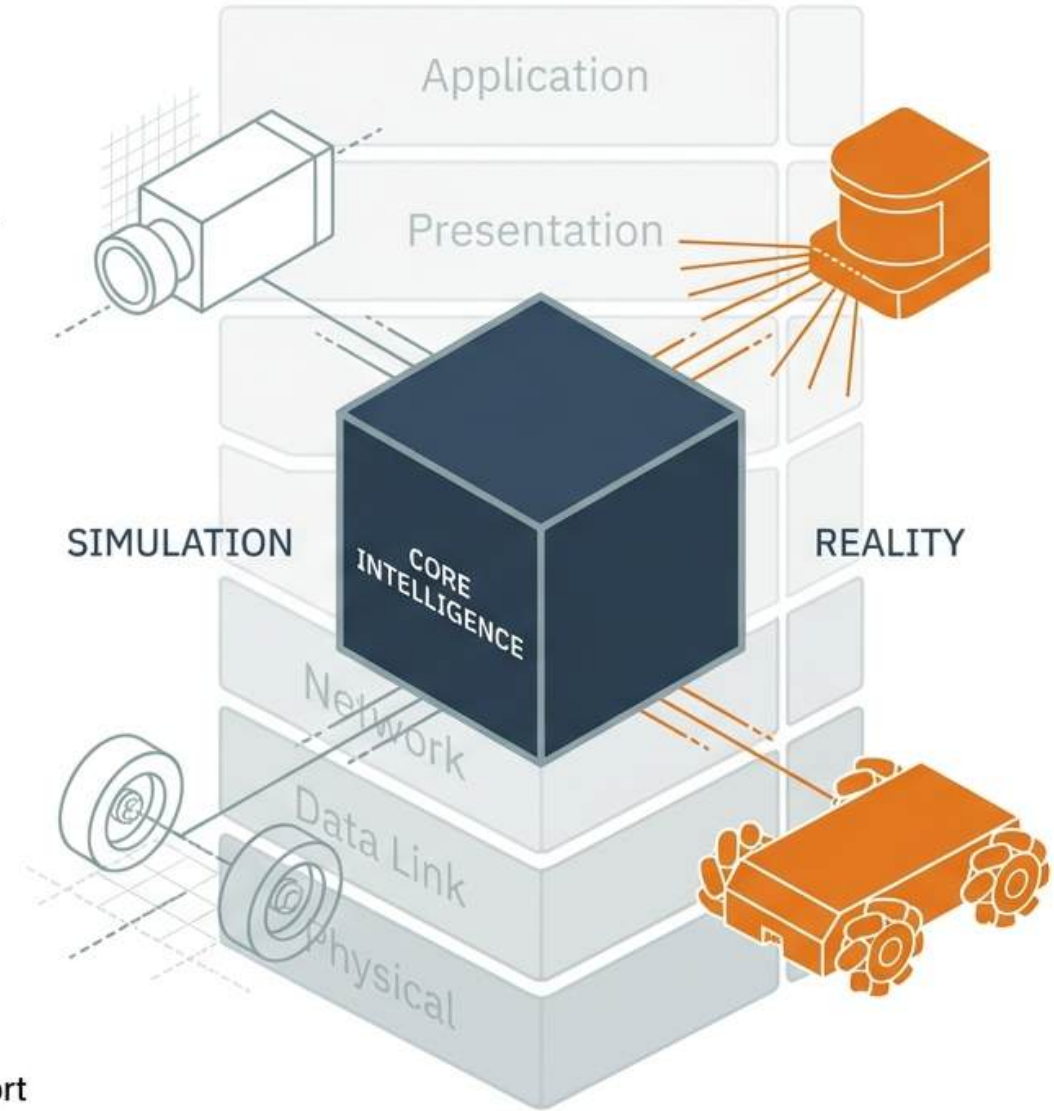
TRACK FOUR

Infrastructure & Deployment

Building the foundation for real-world AGV data collection and future deployment

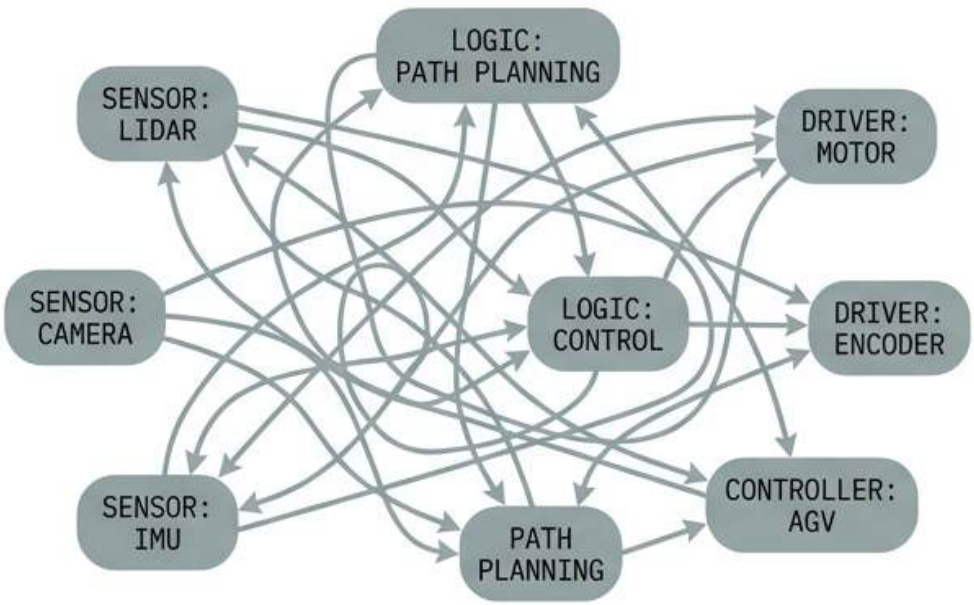
Modular Deployment Strategy: Bridging Simulation and Reality

The 6-Layer Architecture and the “First & Last Layer” Substitution Model



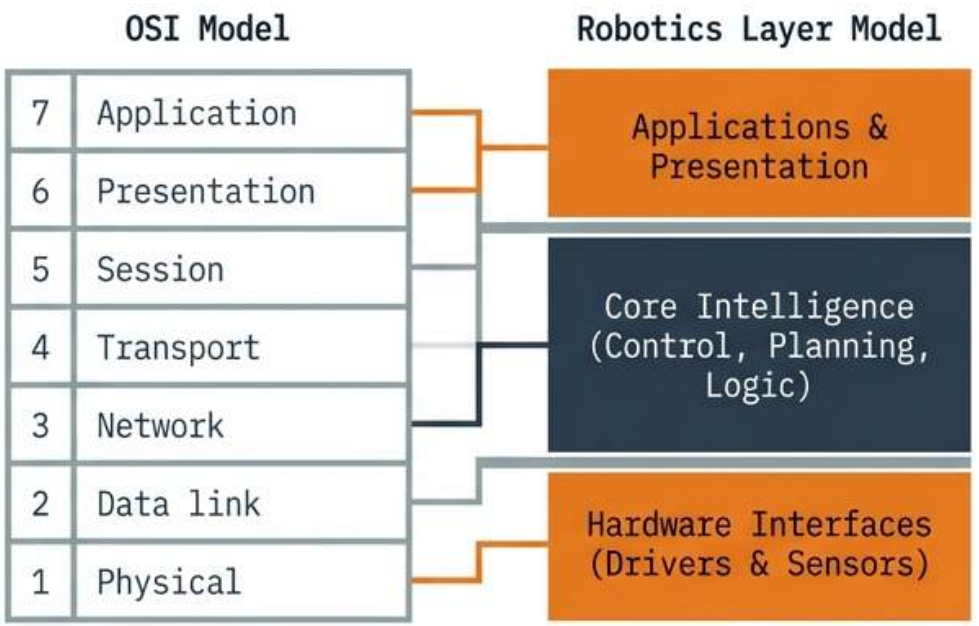
The Challenge: Hardwired Coupling

Deployment fails when code is tightly coupled to the simulation environment. Moving from Isaac Sim to physical Navitrol AGVs traditionally requires a full stack rewrite because drivers and controllers are hardcoded.

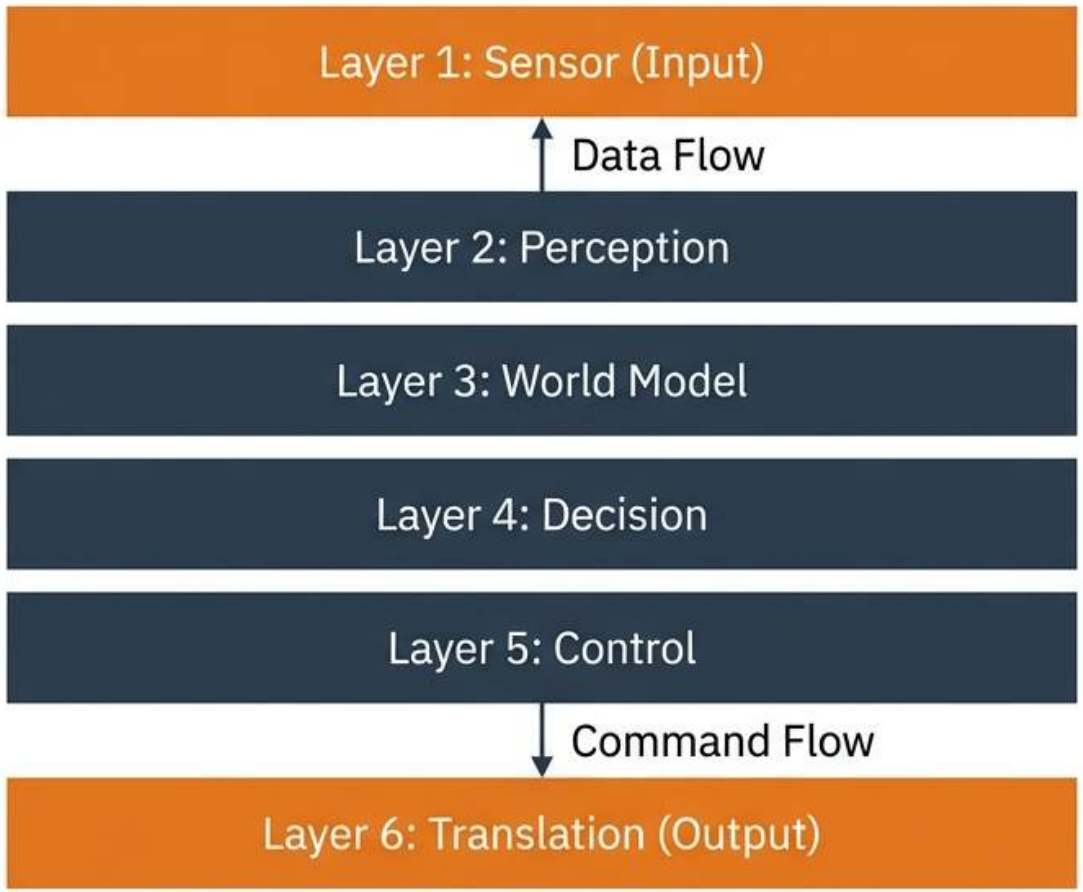


The Solution: Middle-Out Abstraction

We decouple the Core Intelligence (Layers 2-5) from Hardware Interfaces (Layers 1 & 6). Just as TCP/IP functions regardless of the physical medium, our Core Intelligence is agnostic to the source of its data.

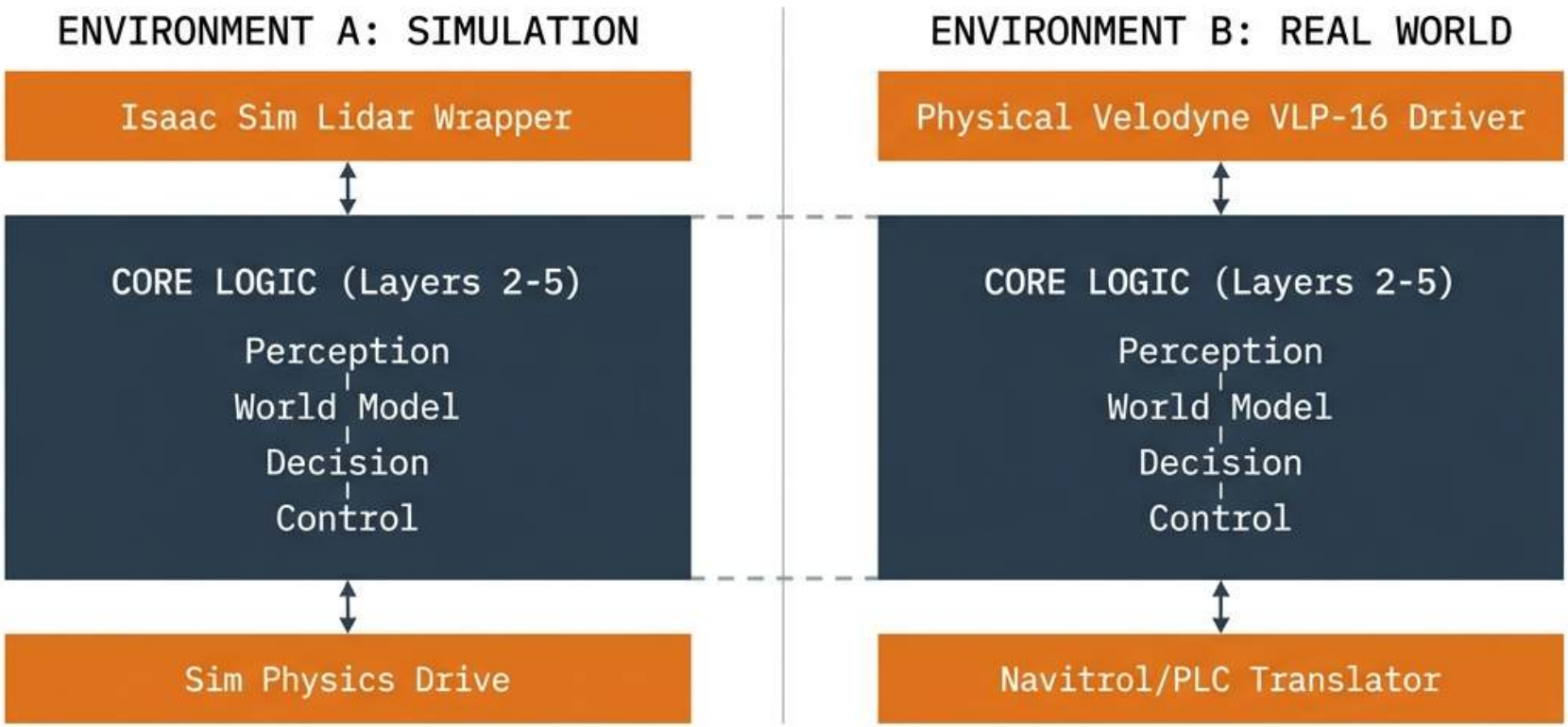


The 6-Layer Architecture



Data travels UP from L1 to L3. Decisions travel DOWN from L4 to L6.

The Substitution Strategy: Same Brain, Different Body

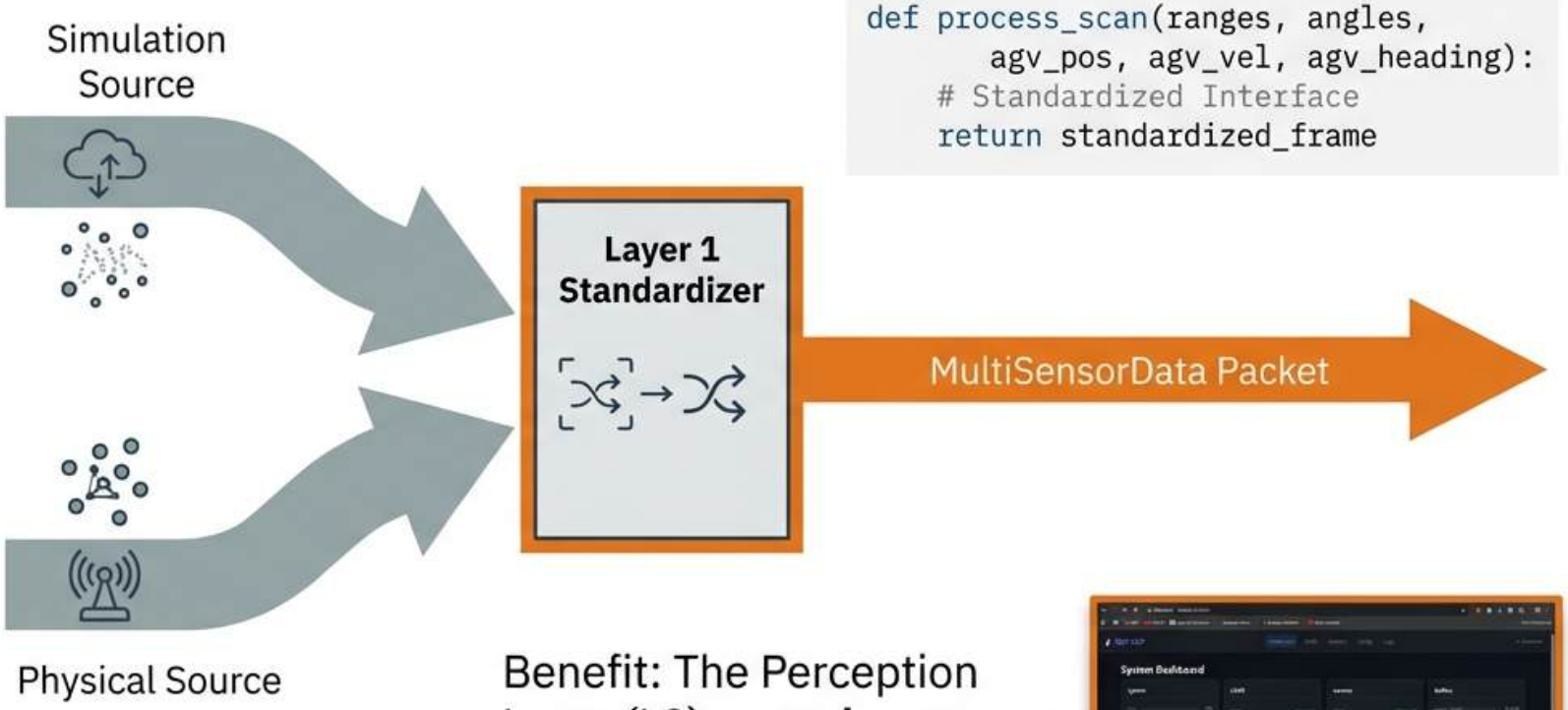


We build the **brain once.** We only swap the **‘Eyes’ (Layer 1) and ‘Hands’ (Layer 6).**

Layer 1 Swap: Input Abstraction

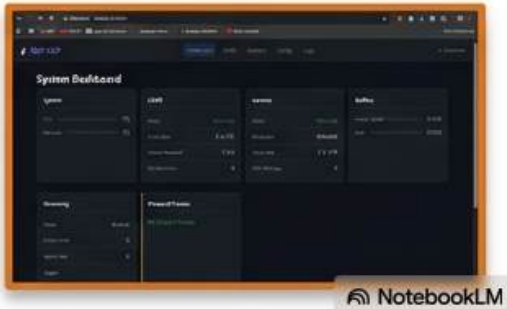
Input Sources

- Simulation Source: Synthetic Point Cloud, Virtual Odometer
- Physical Source: Velodyne VLP-16 Raw Data, AGV Telemetry



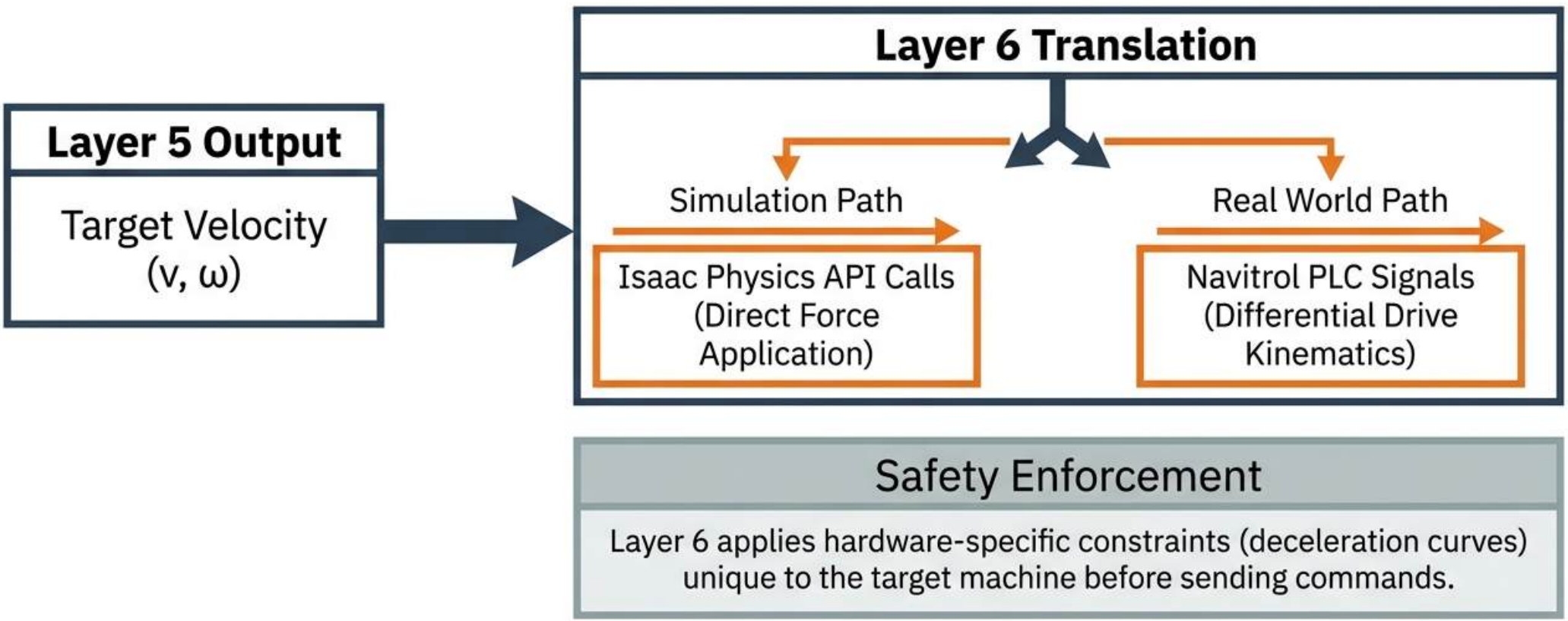

```
def process_scan(ranges, angles,
       agv_pos, agv_vel, agv_heading):
    # Standardized Interface
    return standardized_frame
```

Benefit: The Perception Layer (L2) **never knows—or cares—if the data is real or synthetic.**



Engineering Editorial

Layer 6 Swap: Output Abstraction



Engineering Editorial

The Immutable Core (Layers 2-5)

Preserving Algorithmic Integrity Across Environments

L2 Perception

HySDG-ESD Algorithm

Hybrid Static-Dynamic Graph with Enhanced Spatial Density. Uses DBSCAN + Kalman Filtering.

L3 World Model

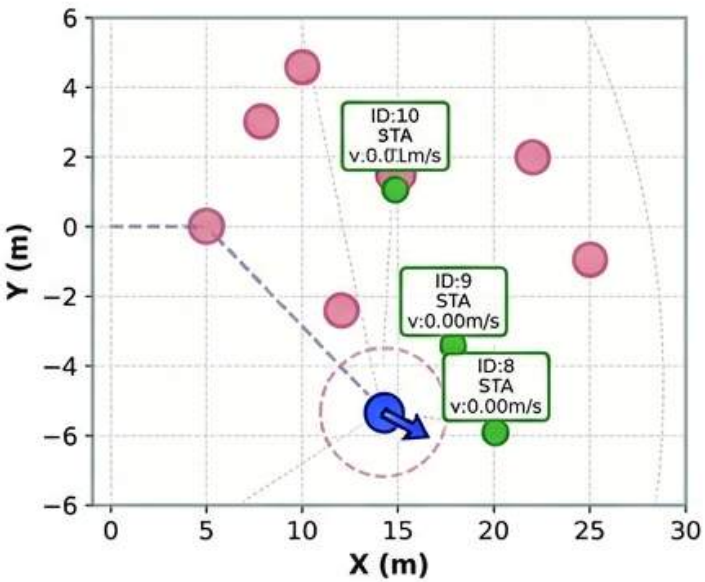
Algorithmic Consensus

Weighted fusion (40% sensor / 60% EKF prediction) to handle sensor jitter.

L5 Control

GapNav + DWA

Hybrid approach using temporary sub-goals to avoid local traps.



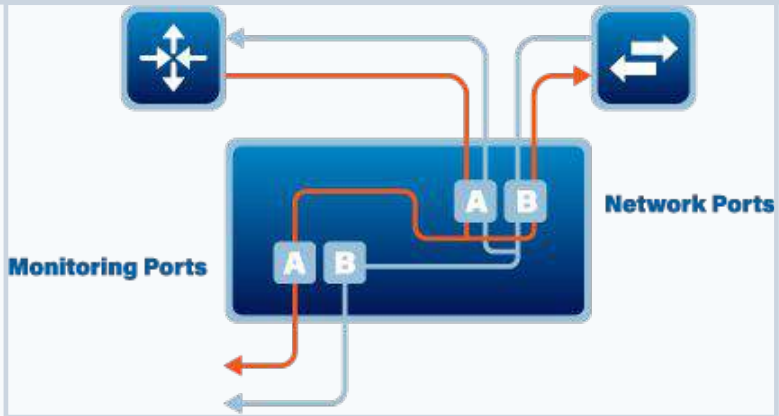
LiDAR Data Collection via Network TAP

✓ **Passive, Non-Invasive Solution**

We installed a Network TAP (Test Access Point) to capture LiDAR UDP packets flowing between the AGV sensor and controller.

Benefits:

- Zero interference with existing AGV operations
- Complete data capture of all LiDAR packets
- Flexible deployment for future experiments



TAP Network Architecture

TAP infrastructure on AGV

LiDAR Data Collection via Network TAP

No.	Time	Source	Destination	Protocol	Length	Info
3	0.000000	192.168.0.11	192.168.0.100	UDP	1324	9999 → 9999 Len=1282
4	0.034983	192.168.0.11	192.168.0.100	UDP	90	9999 → 9999 Len=48
5	0.034983	192.168.0.11	192.168.0.100	UDP	1502	9999 → 9999 Len=1460
6	0.034983	192.168.0.11	192.168.0.100	UDP	1324	9999 → 9999 Len=1282
7	0.069908	192.168.0.11	192.168.0.100	UDP	90	9999 → 9999 Len=48
8	0.071930	192.168.0.11	192.168.0.100	UDP	1502	9999 → 9999 Len=1460
9	0.072665	192.168.0.11	192.168.0.100	UDP	1324	9999 → 9999 Len=1282
10	0.109829	192.168.0.11	192.168.0.100	UDP	90	9999 → 9999 Len=48
11	0.111697	192.168.0.11	192.168.0.100	UDP	1502	9999 → 9999 Len=1460
12	0.112546	192.168.0.11	192.168.0.100	UDP	1324	9999 → 9999 Len=1282
13	0.149605	192.168.0.11	192.168.0.100	UDP	90	9999 → 9999 Len=48
14	0.151562	192.168.0.11	192.168.0.100	UDP	1502	9999 → 9999 Len=1460
15	0.152300	192.168.0.11	192.168.0.100	UDP	1324	9999 → 9999 Len=1282
16	0.189573	192.168.0.11	192.168.0.100	UDP	90	9999 → 9999 Len=48
17	0.191465	192.168.0.11	192.168.0.100	UDP	1502	9999 → 9999 Len=1460
18	0.192137	192.168.0.11	192.168.0.100	UDP	1324	9999 → 9999 Len=1282
19	0.229593	192.168.0.11	192.168.0.100	UDP	90	9999 → 9999 Len=48
20	0.231487	192.168.0.11	192.168.0.100	UDP	1502	9999 → 9999 Len=1460
21	0.231868	192.168.0.11	192.168.0.100	UDP	1324	9999 → 9999 Len=1282

```
▸ Frame 1: Packet, 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface \Device\NPF_{F41
▸ Ethernet II, Src: Leuzeelectro_c0:44:2e (00:15:7b:c0:44:2e), Dst: AdvantechTec_33:98:b7 (cc:82:7f:33:98:
▸ Internet Protocol Version 4, Src: 192.168.0.11, Dst: 192.168.0.100
▸ User Datagram Protocol, Src Port: 9999, Dst Port: 9999
▸ Data (48 bytes)
```

```
0000  cc 82 7f 33 98 b7 00 15  7b c0 44 2e 08 00 45 00   ···3···· {·D.··E·
0010  00 4c 9e 15 00 00 40 11  5a cc c0 a8 00 0b c0 a8   ·L····@· Z·······
0020  00 64 27 0f 27 0f 00 38  5a 9c 30 00 00 00 08 00   ·d'·'··8 Z·0·····
0030  af fe 04 15 02 c8 01 00  00 00 89 8c 00 00 01 01   ········ ········
0040  02 80 40 00 18 80 89 8c  00 00 e8 12 00 f0 00 00   ··@····· ········
0050  00 00 00 00 8b 0a 02 00  00 00                     ········ ··
```

"Sniffed" LIDAR data

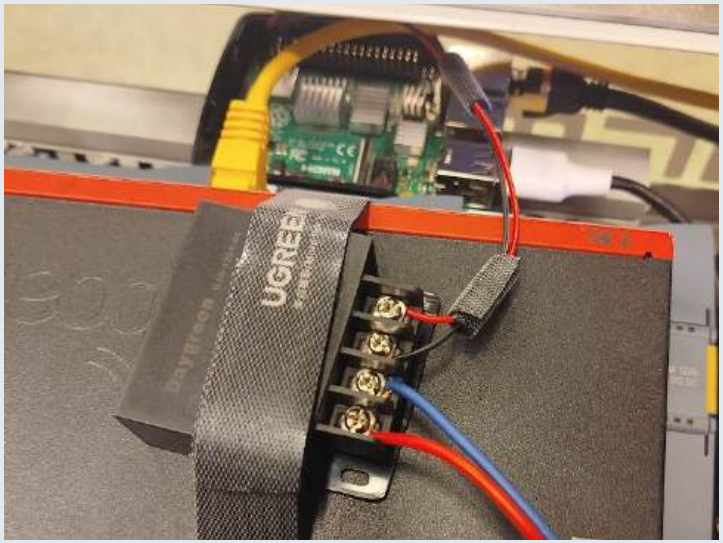
Result: First LiDAR dataset collected in collaboration with Team #1 🥳

Hardware Integration & Computing Infrastructure

🔧 AGV-Mounted Raspberry Pi

Installed directly on the AGV with **DC-DC** converter for power stability.

- Custom data transfer software
- Automatic upload to shared server
- Camera data collection (in progress)

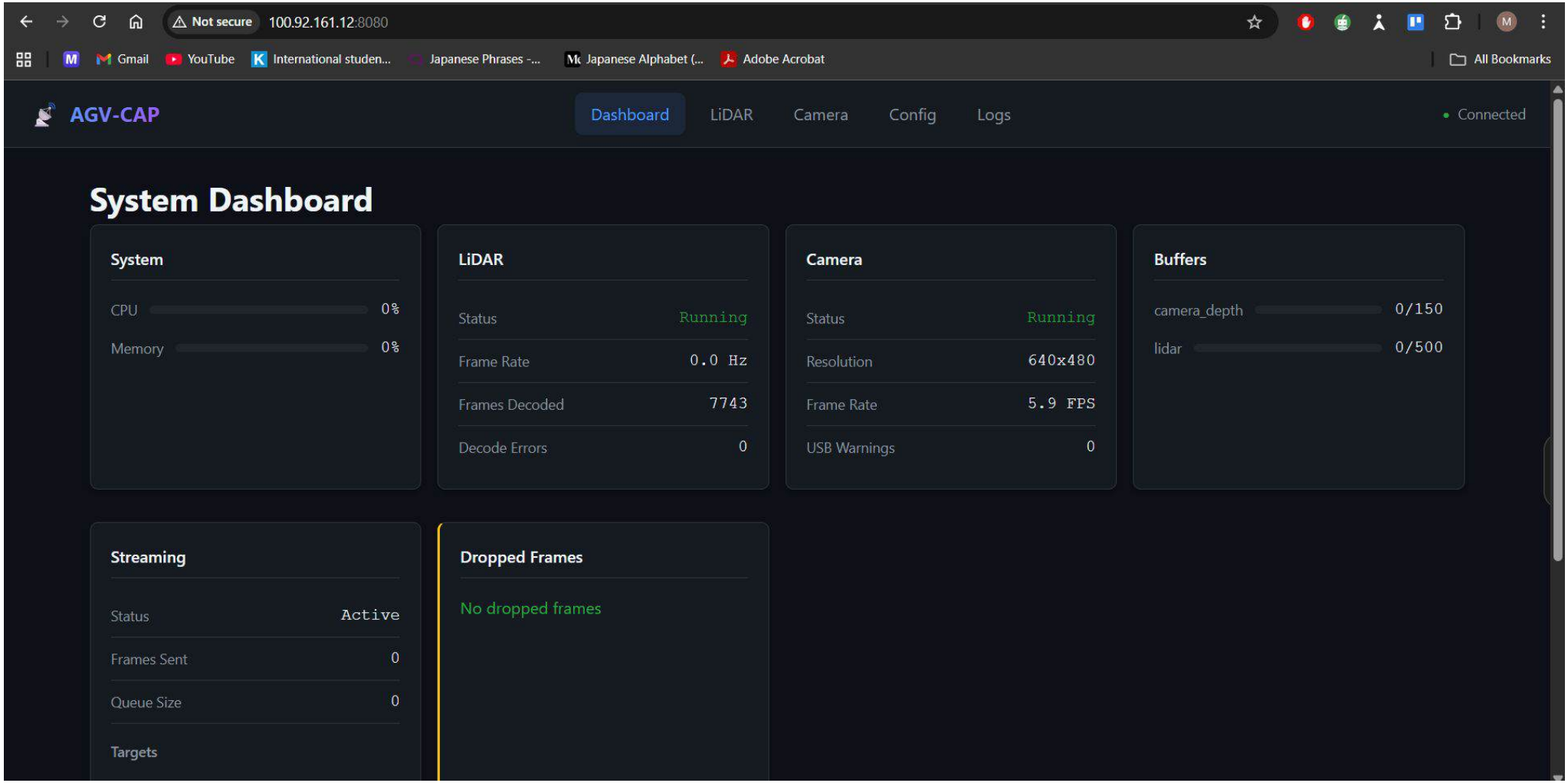


🖥️ Silesian University Server

High-performance computing station configured and shared with all team members.

- Data collection endpoint
- ML training & simulation workloads
- Tailscale tunelling for secure remote access

Hardware Integration & Computing Infrastructure



Deployment Path Investigation

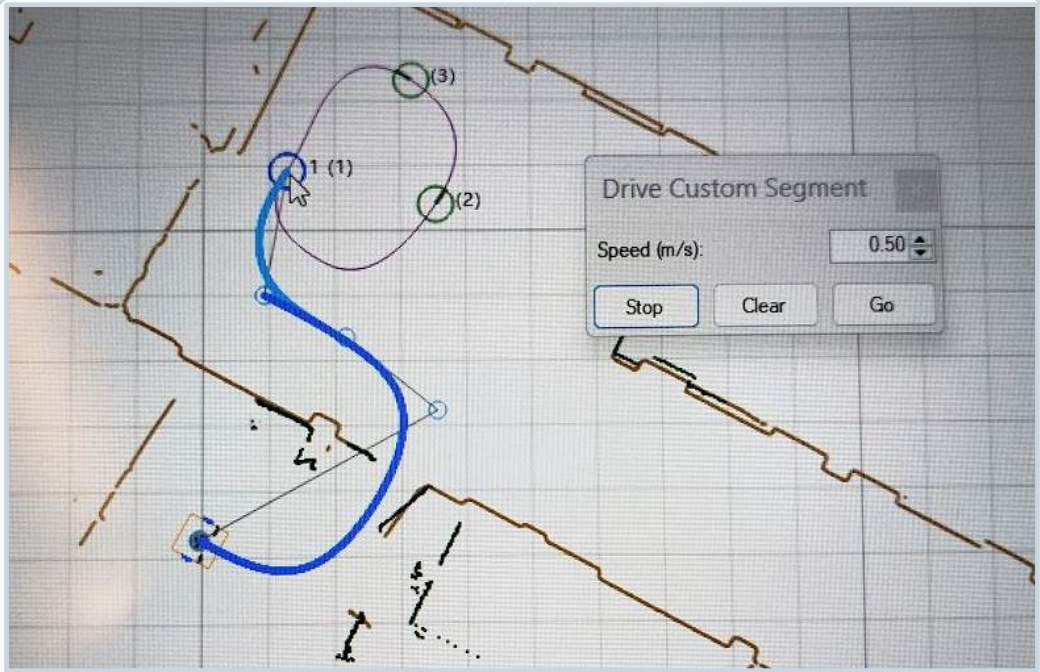
In collaboration with Teams #1 and #3, we investigated deployment options fo
integrating collision avoidance models directly into AGV operations.

> **Discovery: Navitrol Monitor "Custom Route"**
> This existing feature could potentially be leveraged to inject collision-avoidance waypoints without manual control.

Potential Benefits:

- Simpler integration than manual control override
- Works within existing AGV software ecosystem
- Reduced risk of system conflicts

Status: Conceptual investigation complete. Practical validation is future work.



Navitrol Monitor Interface

Track 1 Contribution

PRACTICAL / SUPPORTIVE

Enabling Future Research Infrastructure

We established a complete real-world data collection pipeline that operates passively without interfering with AGV operations. This infrastructure—including the Network TAP, Raspberry Pi integration, and shared server—provides a foundation for future experiments, data gathering, and eventual model deployment on physical AGVs.

TAP
Passive LiDAR capture

RPi
On-AGV data transfer

Server
Shared compute resource

TRACK 5

Extensive Framework

Isaac Sim Simulation Environment

Synthetic Data Collection • Path Planning • Web UI Control

Isaac Sim Simulation Framework: Developed Open-source software

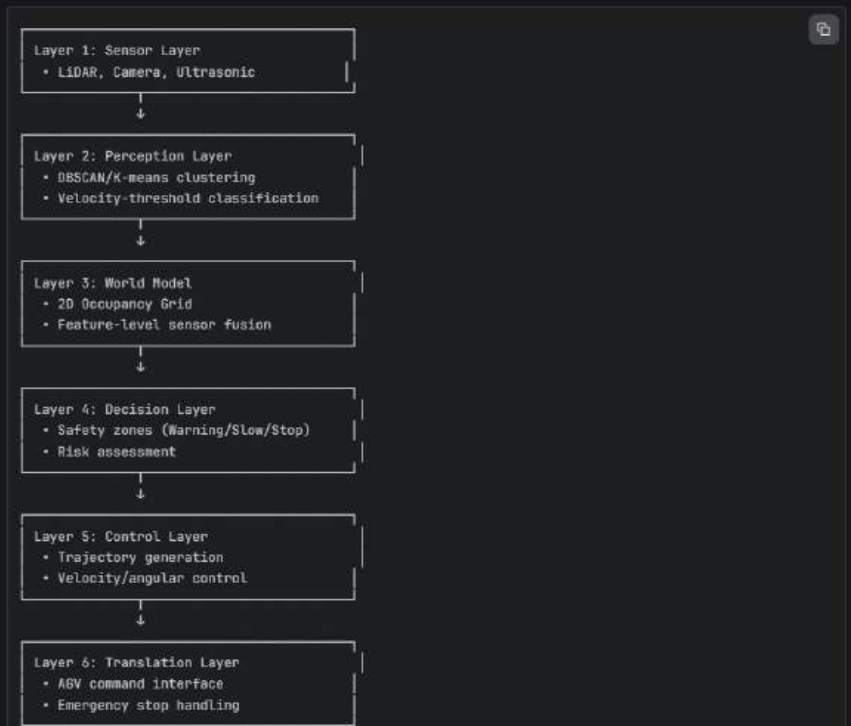
README.md

AGV Collision Avoidance System - Isaac Sim Implementation

Overview

This is a complete implementation of your 6-layer modular architecture for AGV collision avoidance in NVIDIA Isaac Sim. The system uses Phase 1 initial methods as discussed in your project meetings.

Architecture



https://gitlab.com/myroslavm/tuai-ws

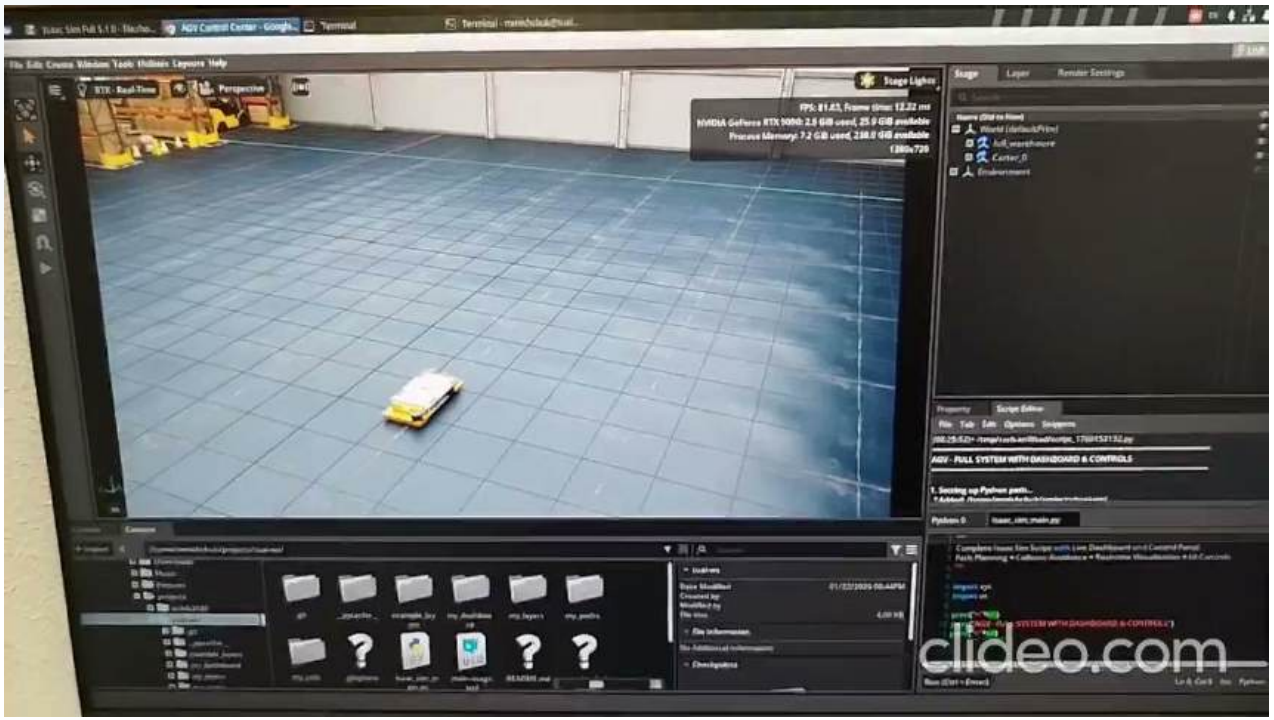
Isaac Sim Simulation Framework

Core Features

- Carter_0 AGV model with differential drive
- LiDAR and camera sensor simulation
- 5 predefined + custom path creation
- Dynamic obstacle spawning
- Route modes: Loop, Once, PingPong

Web UI Dashboard

- Real-time AGV status monitoring
- Path creation and visualization
- Sensor data capture controls
- Interactive obstacle management



Purpose: Enable synthetic data collection and method validation before hardware deployment

Track 5 Contribution

PRACTICAL / TOOLING

A Simulation Platform for Research

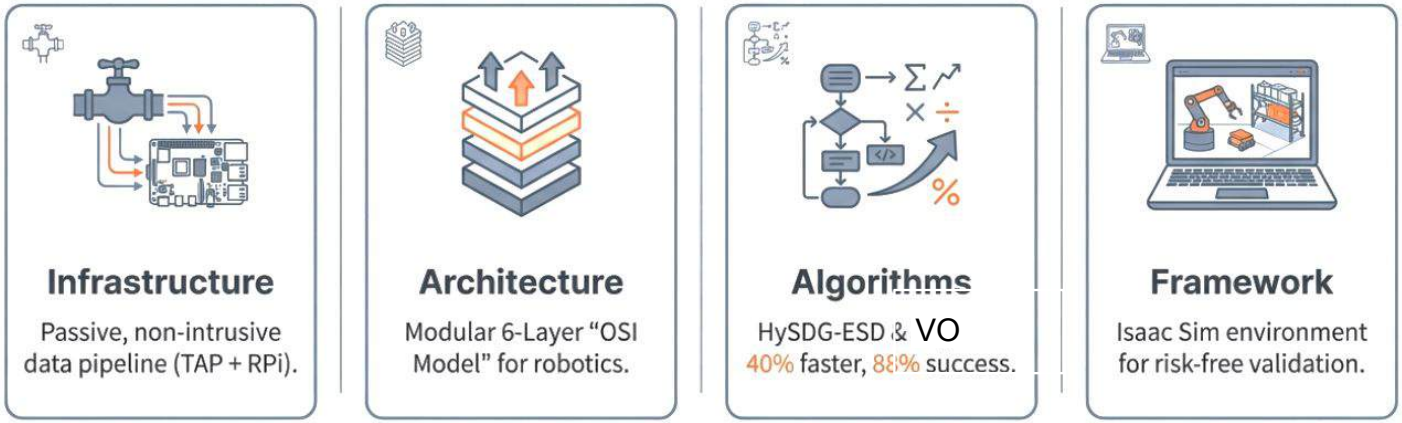
We developed a comprehensive Isaac Sim framework with Web UI that enables synthetic data collection, algorithm validation, and experimentation before hardware deployment. The framework supports custom path creation, obstacle spawning, and real-time sensor data capture at 20Hz—providing a safe, repeatable environment for testing collision avoidance strategies.

20Hz
Real-time updates

Web UI
Intuitive control

LiDAR + Cam
Sensor simulation

Summary: Key Contributions



Track 1: Conceptualization

6-Layer theoretical architecture enabling modular, testable, and explainable collision avoidance systems

Contribution: Conceptual/Theoretical

Track 3: Infrastructure

Real-world data collection pipeline: Network TAP for LiDAR, Raspberry Pi integration, Silesian server infrastructure

Contribution: Practical/Supportive

Track 2: Instantiation

HySDG-ESD perception + Velocity Obstacle Algorithm achieving 100% success, 40% faster computation

Contribution: Practical/Algorithmic

Track 4: Framework

Isaac Sim environment with Web UI at 20Hz real-time performance for synthetic data and validation

Contribution: Practical/Tooling

Thank You!

Questions & Discussion

Team #2 — LM-RtA Workshop

Myroslav Mishchuk • Asif Huda • Milad Jafari • Sadat Hossain • Leonardo Schiavo