LM-RtA WORKSHOP • FINAL PRESENTATION

Lightweight Collision Avoidance for AGV Systems

A Modular 6-Layer Architecture for Real-Time Safety

Team #2 • Myroslav Mishchuk, Asif Huda, Milad Jafari, Sadat Hossain, Leonardo Schiavo

January 2026

Presentation Roadmap

01 • Infrastructure
Real-world data collection, hardware integration

02 • Conceptualization
The 6-Layer modular architecture

03 • Instantiation
Specific algorithms: HySDG-ESD, DWA, GapNav

04 • Framework
Isaac Sim simulation with Web UI

05 • Validation & Future
Current progress and next steps

KEY DIFFERENTIATOR

We focus on both theoretical foundations and practical implementation

TRACK ONE

Infrastructure

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

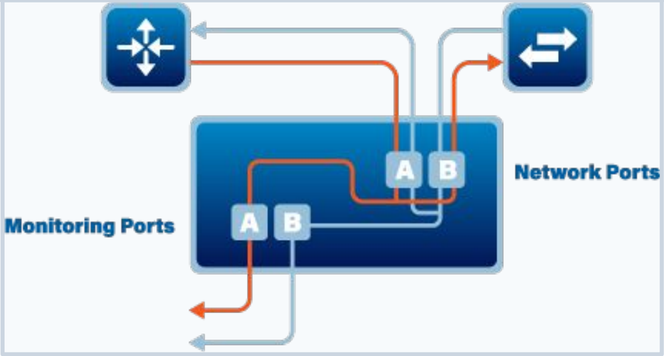
LiDAR Data Collection via Network TAP

✓ **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
0010  00 4c 9e 15 00 00 40 11  5a cc c0 a8 00 0b c0 a8
0020  00 64 27 0f 27 0f 00 38  5a 9c 30 00 00 00 08 00
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

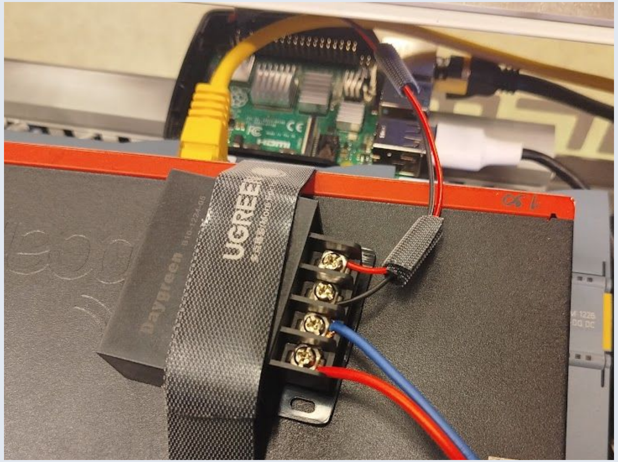
Result: First LiDAR dataset collected in collaboration with Team #1 🥳

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)

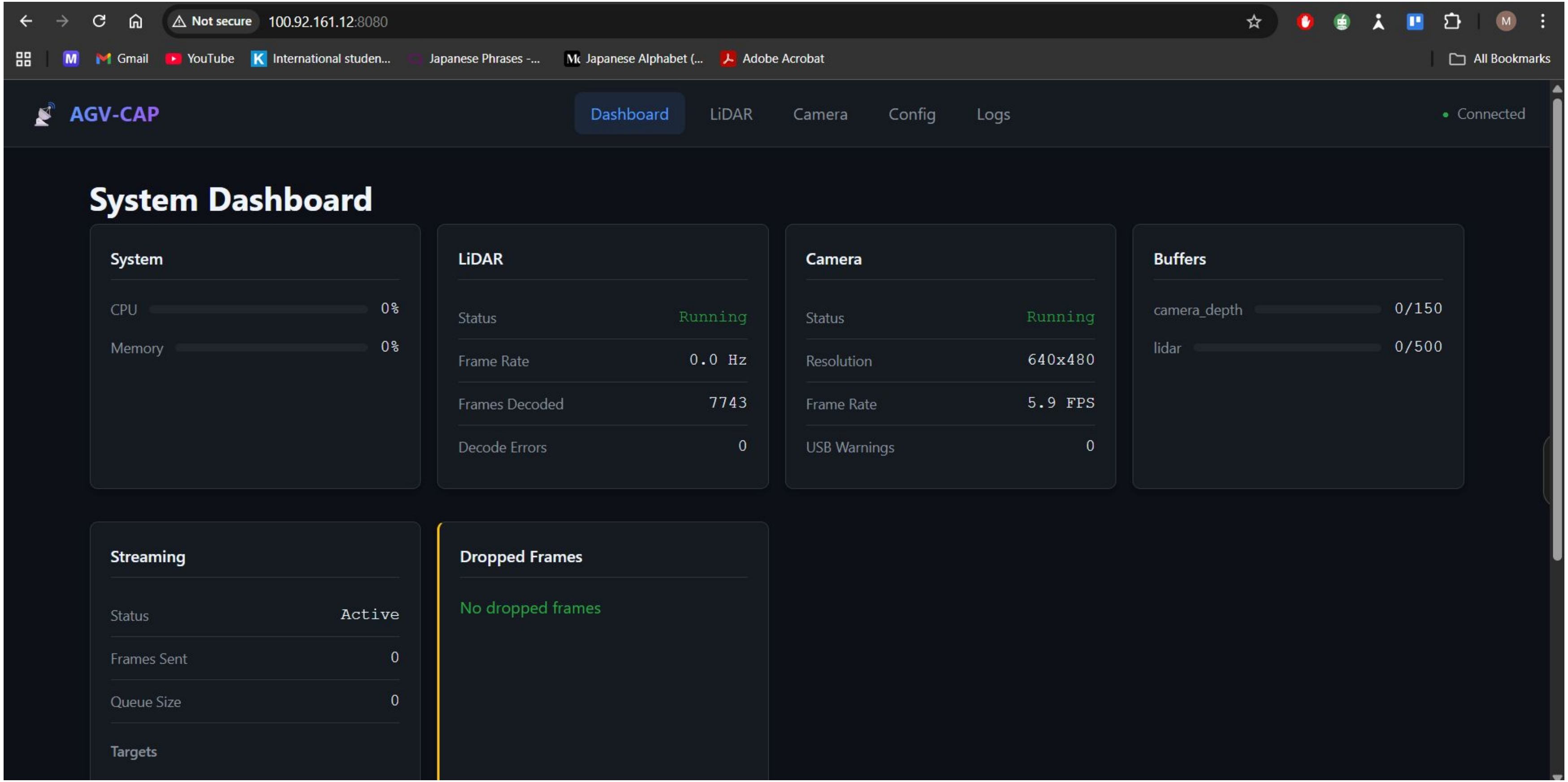


🖥️ Silesian University Server

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

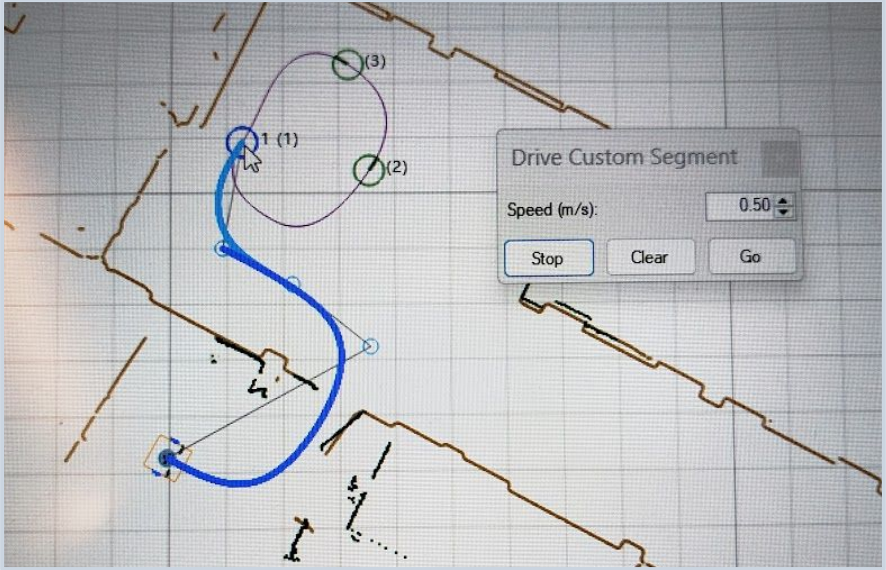
In collaboration with Teams #1 and #3, we investigated deployment options for integrating collision avoidance models directly into AGV operations.

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 TWO

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 • Commands flow down ↓

Why This Architecture?

Modularity: Replace any layer without affecting others

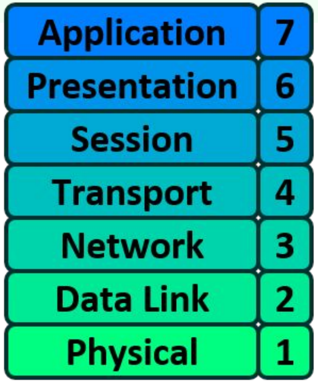
Testability: Isolate and test individual components

Explainability: 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.

- Dynamic Window Approach
- Trajectory scoring
- Velocity smoothing
- 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 2 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

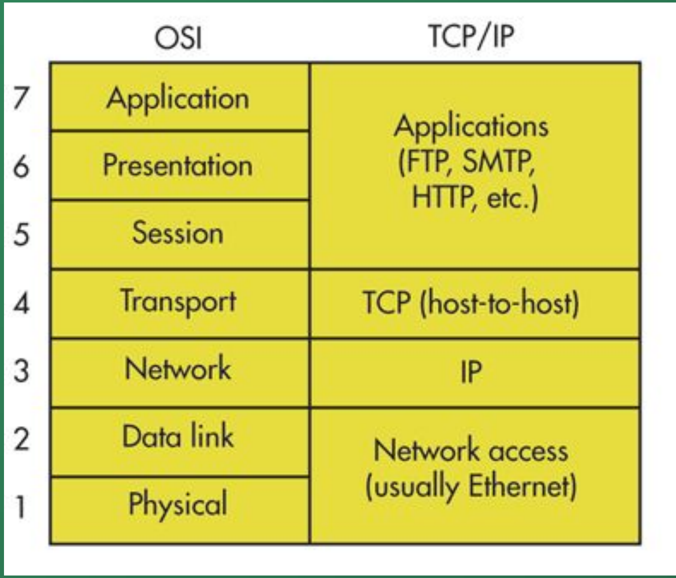
Theoretical layers for collision avoidance

TRACK 3

Instantiation

Concrete Methods for Each Layer

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



HySDG-ESD: Dynamic Obstacle Detection Pipeline

1 LiDAR Scan → **2** DBSCAN Clustering → **3** World Transform → **4** Kalman Tracking → **5** Classification

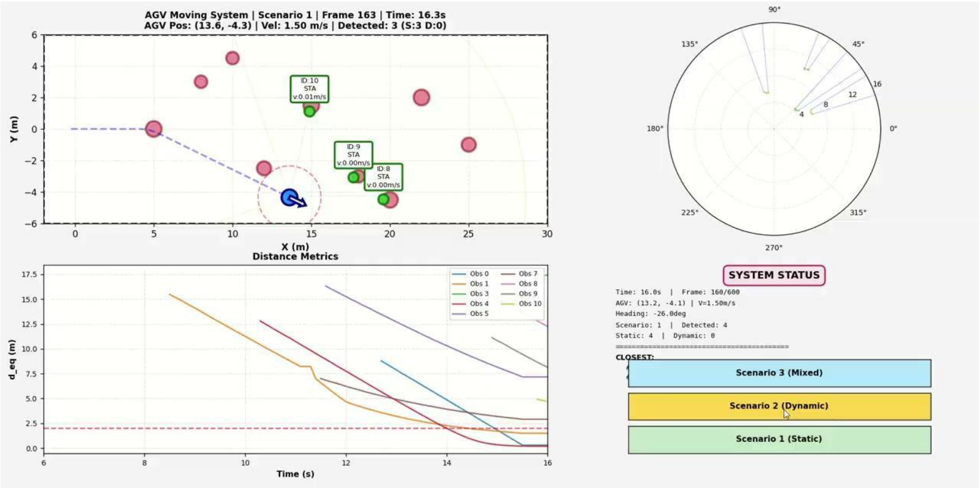
Key Components

DBSCAN: Density-based clustering for obstacle detection

Ego-motion compensation: Rotation matrices for moving AGV

Extended Kalman Filter: Multi-object tracking with Hungarian assignment

HySDG-ESD: Velocity-based static/dynamic classification



Note: Detailed presentation by Milad Jafari separately

World Model Implementation (Layer 3)

2D Occupancy Grid

Probabilistic grid where each cell stores occupancy probability (0.0 = free, 1.0 = occupied).

100×100	**0.2m**	**20×20m**
Grid cells	Resolution	Coverage

Feature-Level Sensor Fusion

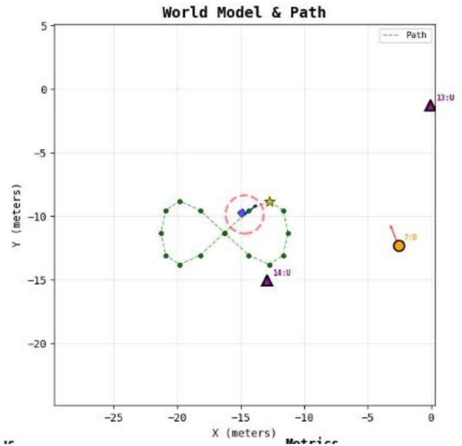
Combines detections from LiDAR, camera, and ultrasonic sensors at feature level.

Coordinate Transformation

Transforms AGV-relative detections to world frame using rotation matrices.

Interpretation Thresholds

FREE	UNKNOWN	OCCUPIED
$P < 0.3$	$0.3 \le P \le 0.7$	$P > 0.7$



Probability Assignment (Direct)

0.9	**0.8**	**0.5**
Static obstacle	Dynamic obstacle	Unknown

Dynamic Decay Formula

```
P_new = P_old × 0.95 + 0.5 × 0.05
```

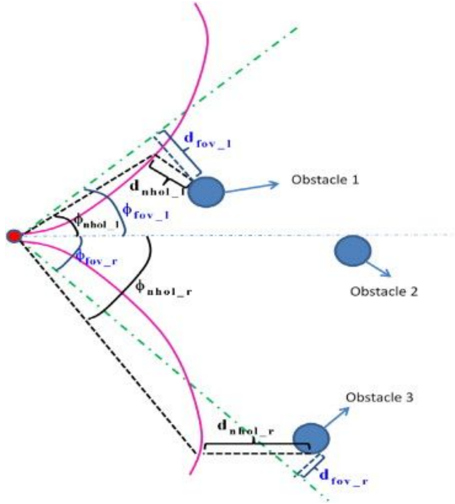
Cells decay toward unknown (0.5) when not reinforced

Navigation Algorithm Comparison

Results: 100 tests, 15 obstacles

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

Key Innovation: Temporary sub-goals via gap detection allow strategic navigation through openings, avoiding local traps.



VFH Issues

Jerky movements, fails in U-shaped traps

DWA Issues

Smooth but shortsighted, computationally heavy

GapNav Advantages

- 40% computation time vs DWA
- 4% fewer steps (more efficient paths)
- Gap navigation avoids traps

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 GapNav+DWA+APF for navigation. Benchmark testing (100 runs, 15 obstacles) demonstrated 88% success rate with 40% faster computation than pure DWA.

L2 Perception
HySDG-ESD

L3 World Model
Occupancy Grid

L5 Control
GapNav+DWA+APF

TRACK 4

Framework

Isaac Sim Simulation Environment

Synthetic Data Collection • Path Planning • Web UI Control

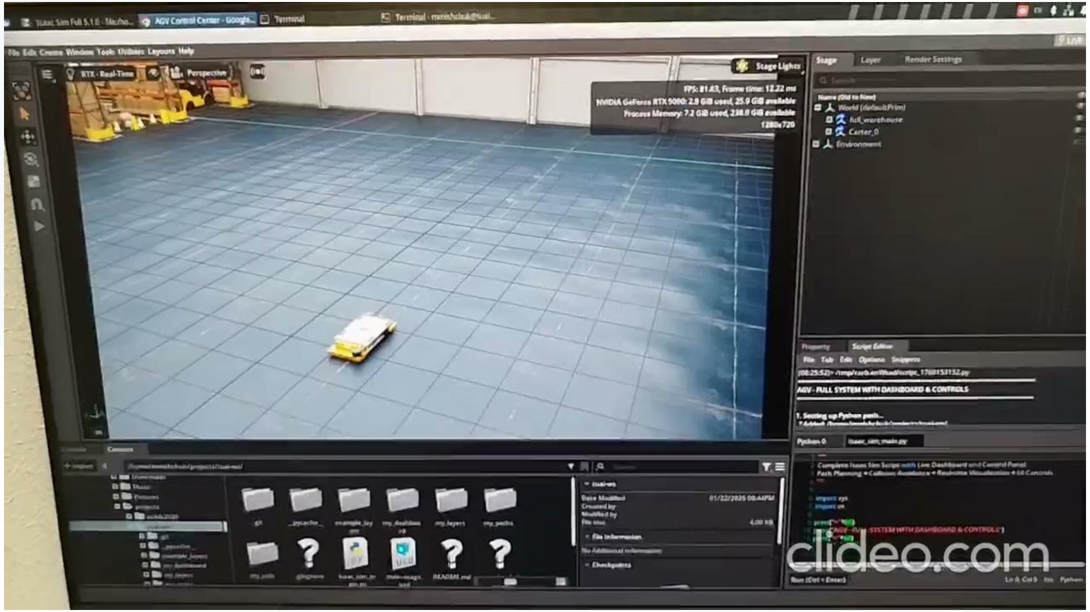
Isaac Sim Simulation Framework

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 4 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

TRACK 5

Validation

Integration Testing & Future Work

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

Track 5 Contribution

IN PROGRESS / FOUNDATIONAL

Groundwork for Systematic Validation

While full stack integration is still in progress, we have completed individual layer implementations, standalone algorithm benchmarks, and framework setup. The foundation is in place for systematic end-to-end validation. Future work will focus on integrating all layers in Isaac Sim, defining quantitative success metrics, and running comprehensive test scenarios.

✓ COMPLETED
Individual layer tests

✓ COMPLETED
Algorithm benchmarks

→ NEXT
Full integration

Validation Status & Next Steps

✓ Completed

- All 6 layer implementations
- Algorithm benchmarks (100 tests)
- Isaac Sim framework with sensor capture
- Web UI for control and monitoring
- Individual layer testing

⏳ In Progress

- Full stack integration in Isaac Sim
- End-to-end collision avoidance demo
- Quantitative validation metrics

Future Work Timeline

January 2026

Complete Isaac Sim integration

February 2026

Validation suite development

Future

Hardware deployment on real AGV

Full integration pending due to technical challenges — foundation is complete and ready for next phase

Summary: Key Contributions

Track 1: Infrastructure

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

Contribution: Practical/Supportive

Track 2: Conceptualization

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

Contribution: Conceptual/Theoretical

Track 3: Instantiation

HySDG-ESD perception + GapNav+DWA+APF navigation achieving 88% 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

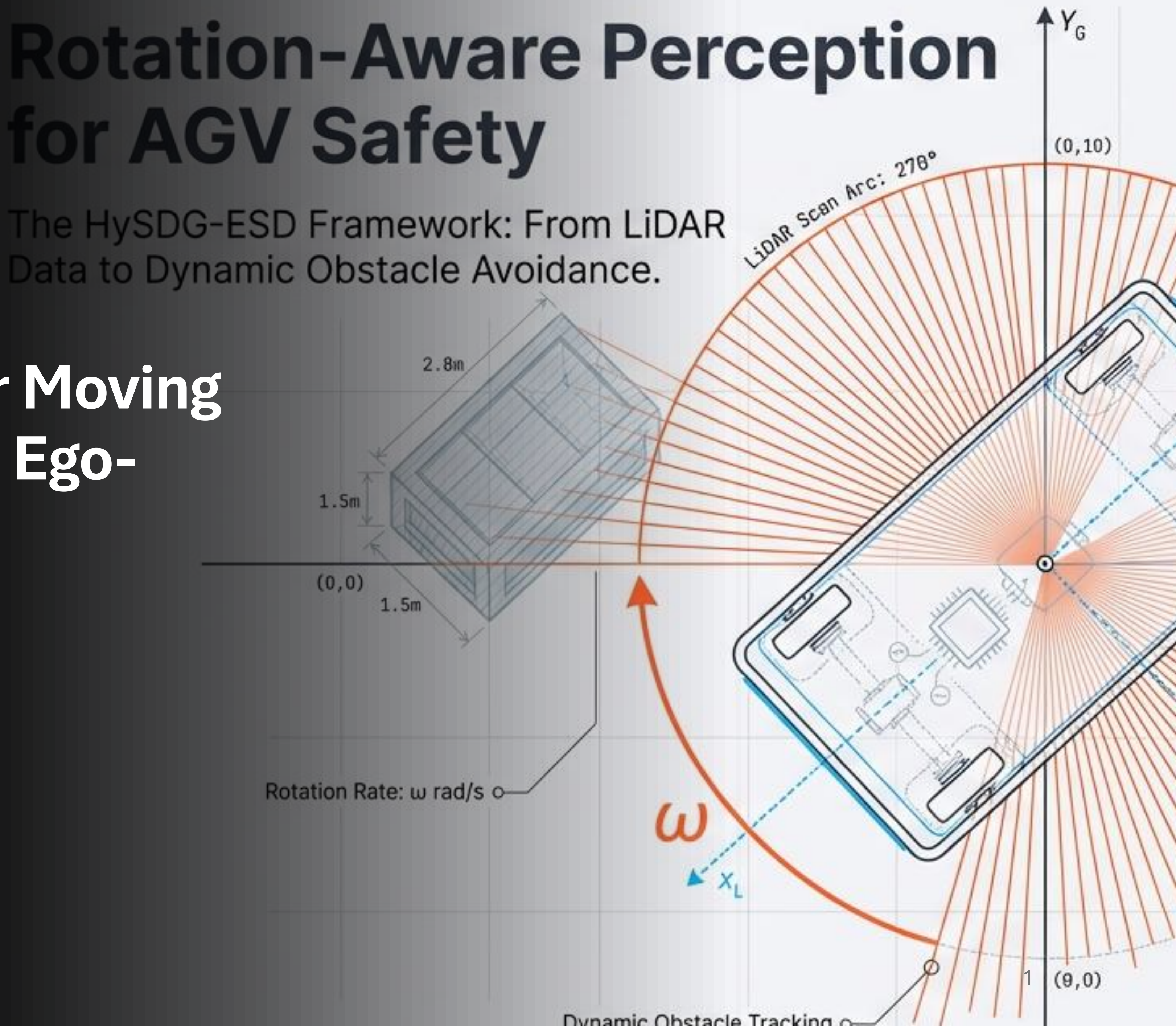
Contribution: Practical/Tooling

Overall: We combined theoretical foundations (OSI-like architecture) with practical implementations (algorithms, simulation, hardware integration) for a comprehensive collision avoidance research platform.

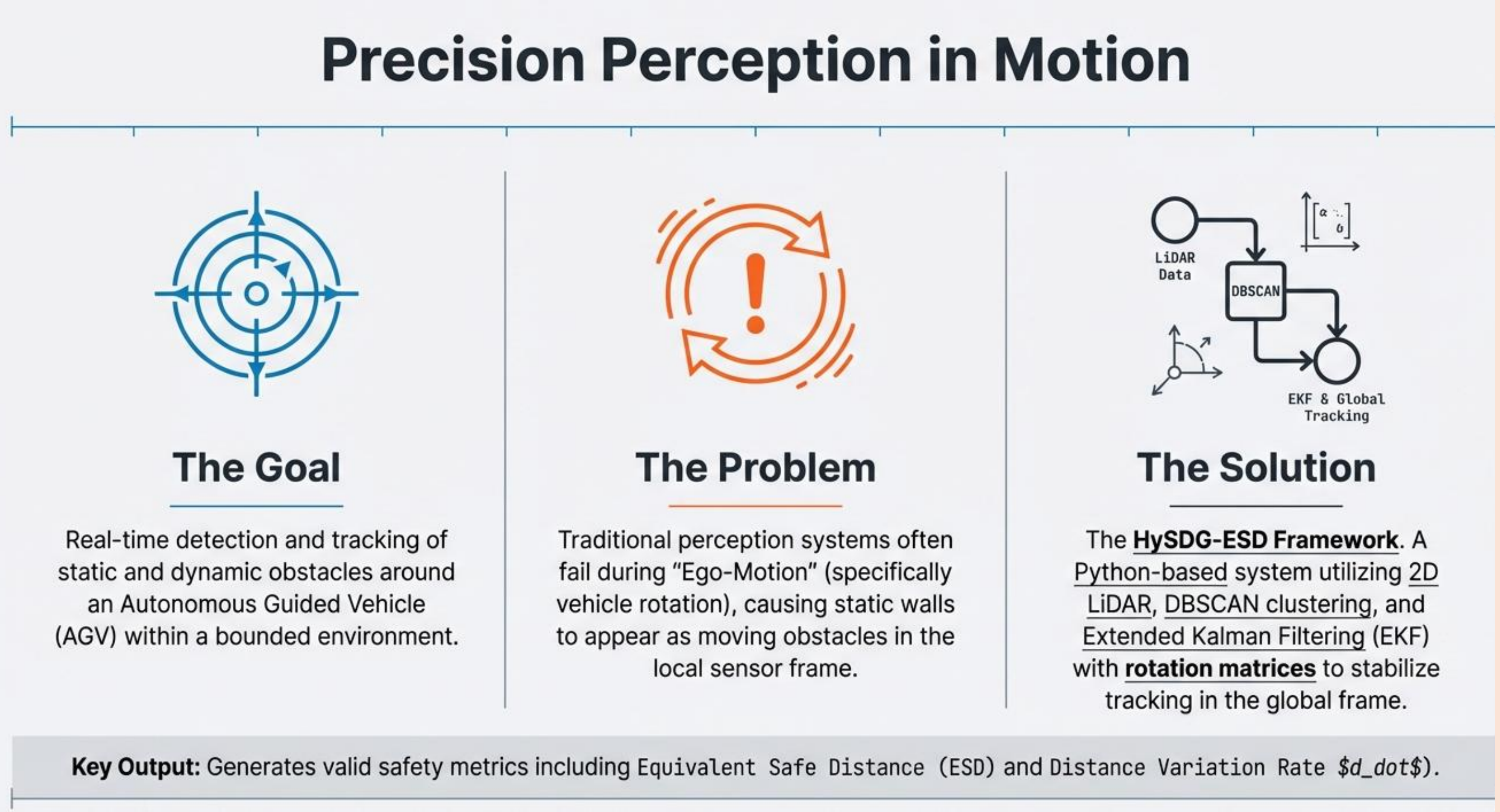
HySDG-ESD: Dynamic Obstacle Detection for Moving AGVs Using LiDAR and Ego-Motion Compensation

Milad Jafari Barani

Autonomous Guided Vehicle (AGV) Perception System

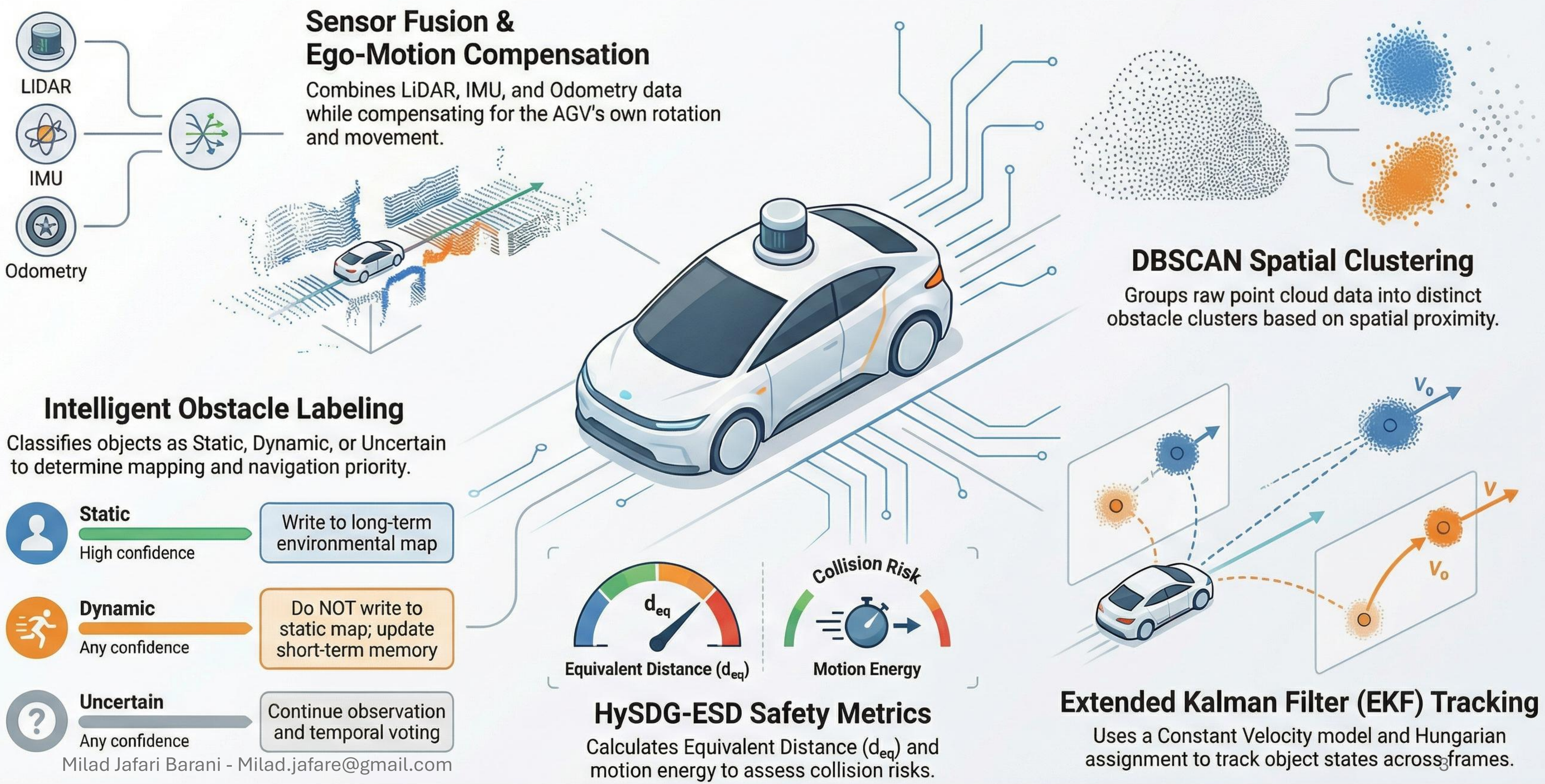


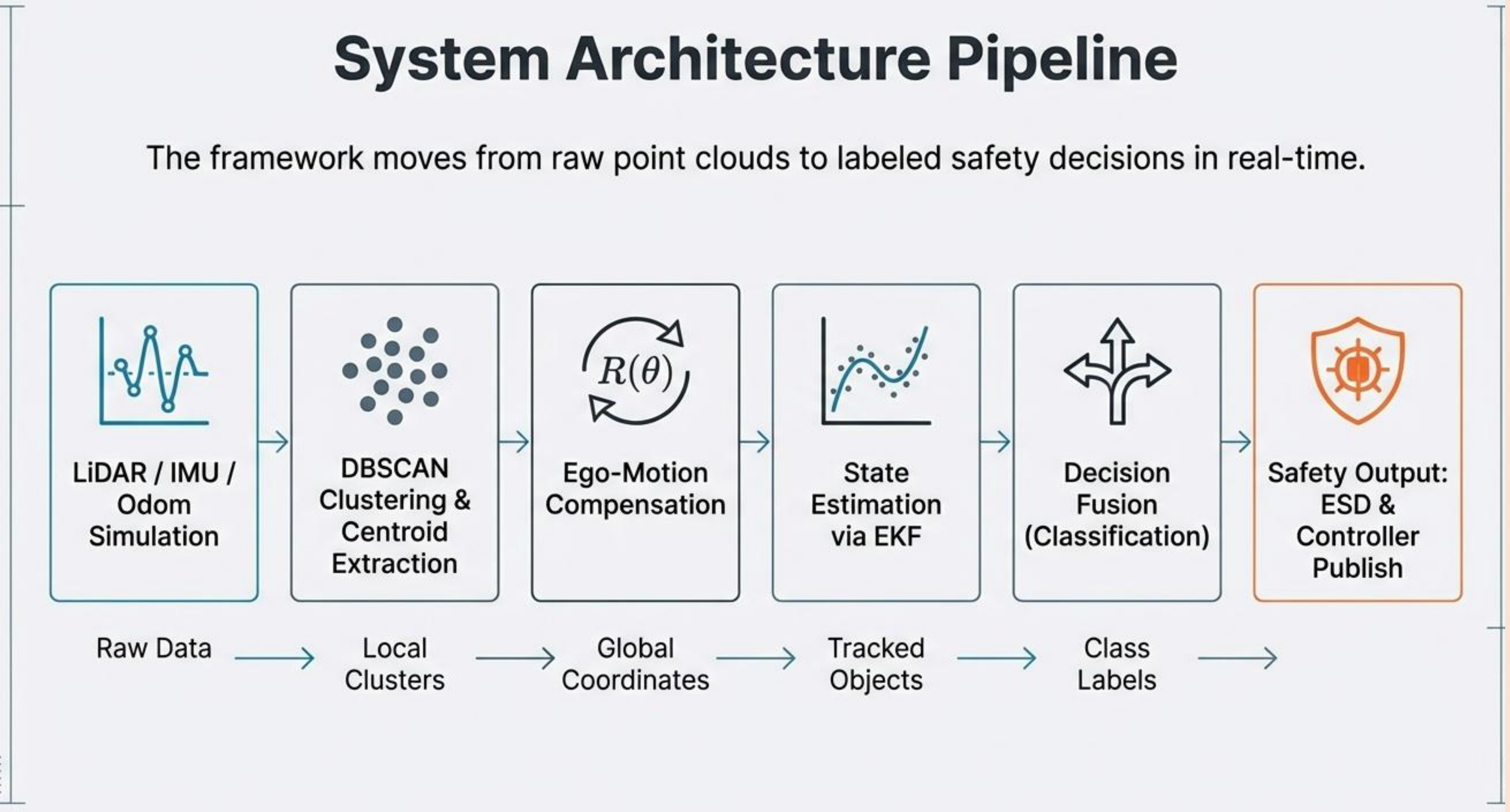
System Overview



- **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

Rotation-Aware AGV Perception: The Dynamic Obstacle Detection Pipeline





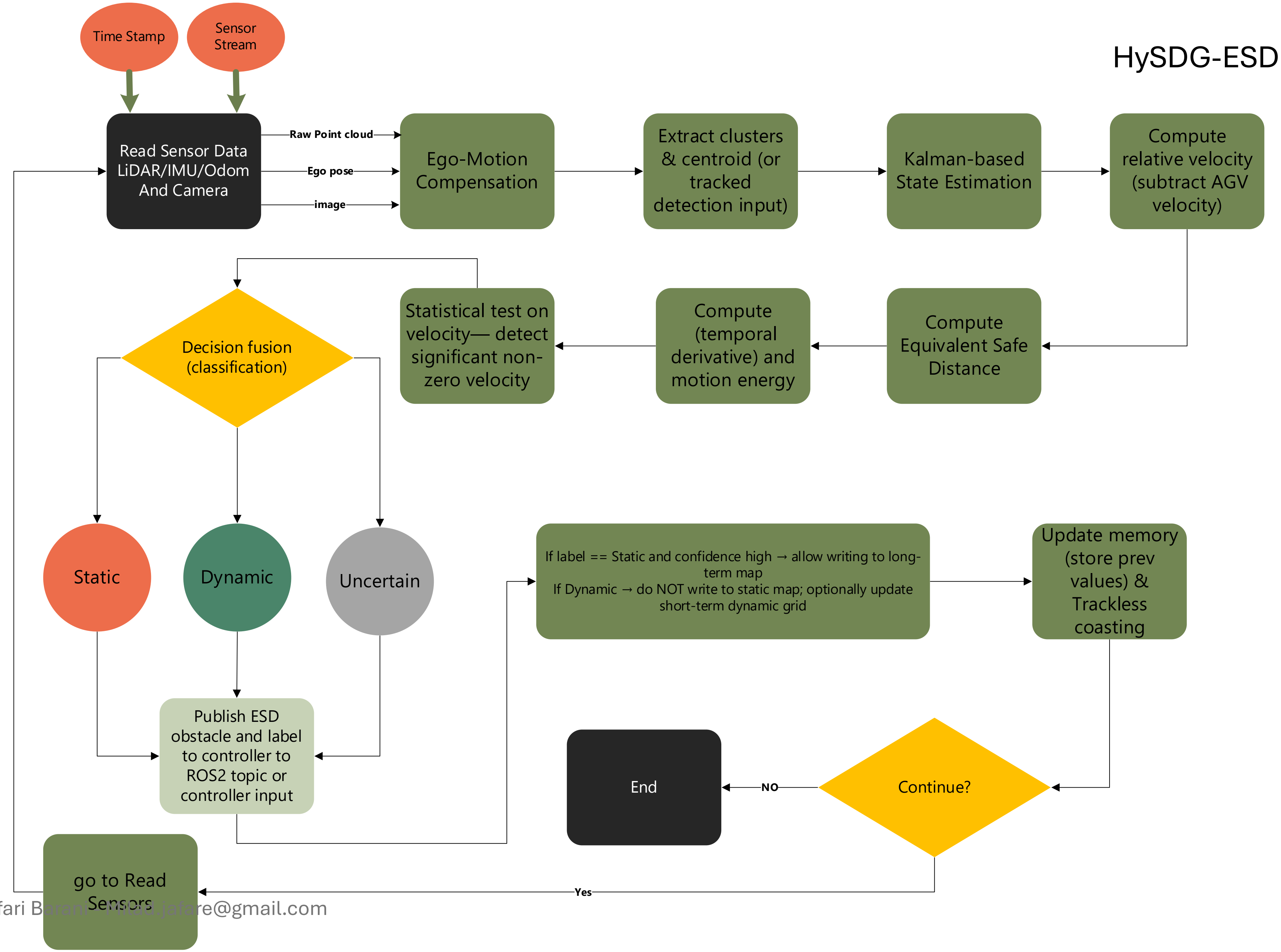
Core Methodology

- **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

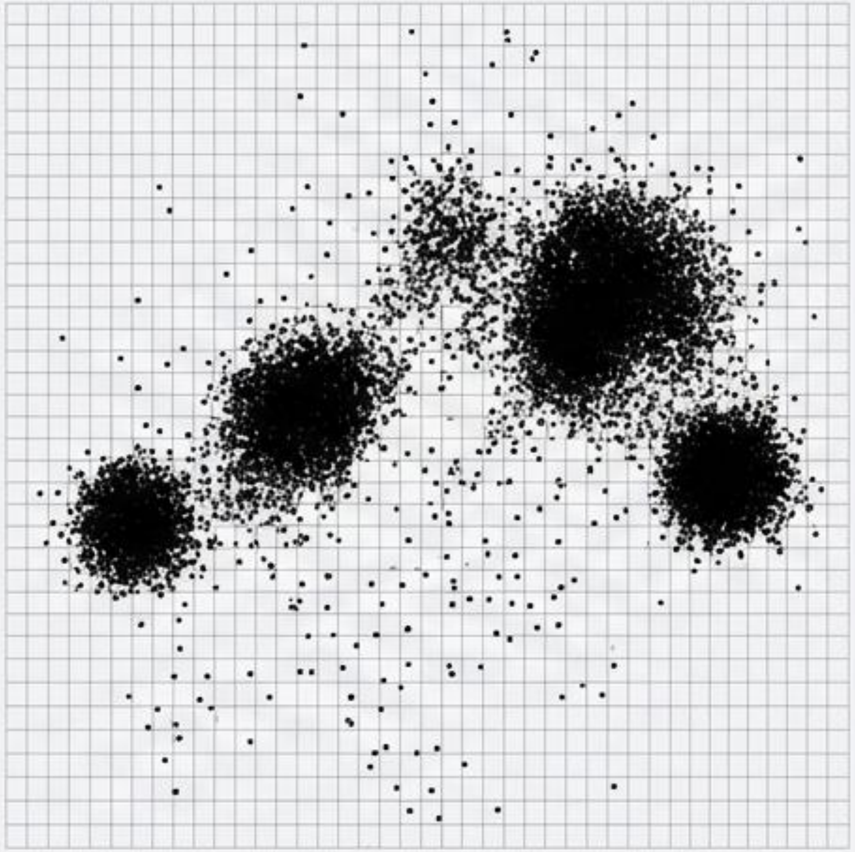
HySDG-ESD Flochart



Step 1: LiDAR Processing & Clustering

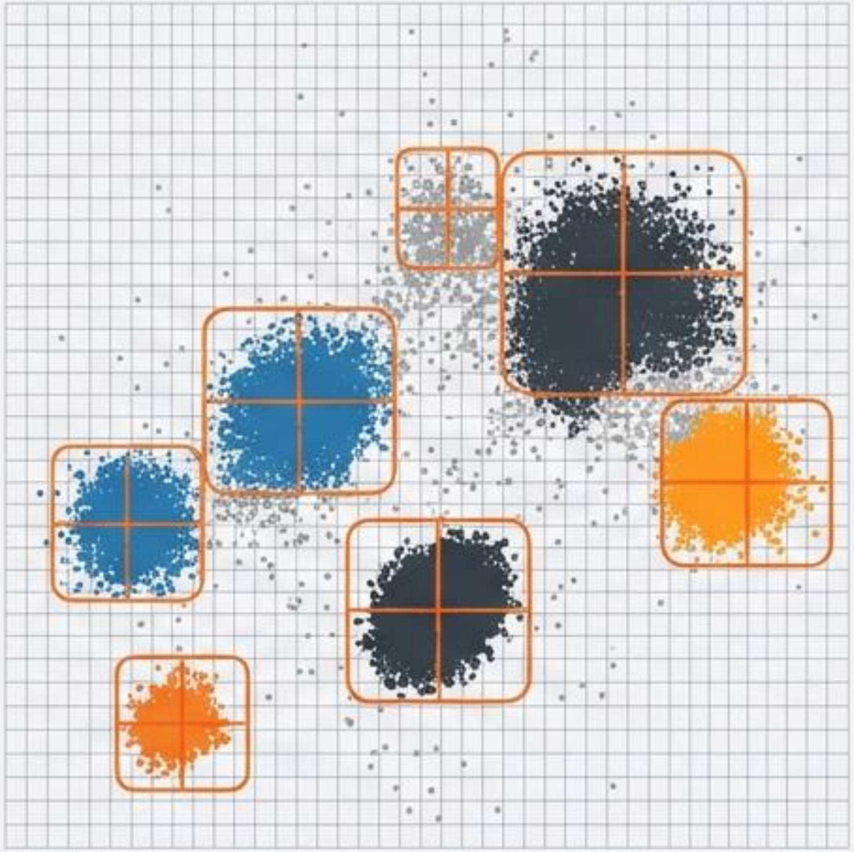
Raw Sensor Input

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.

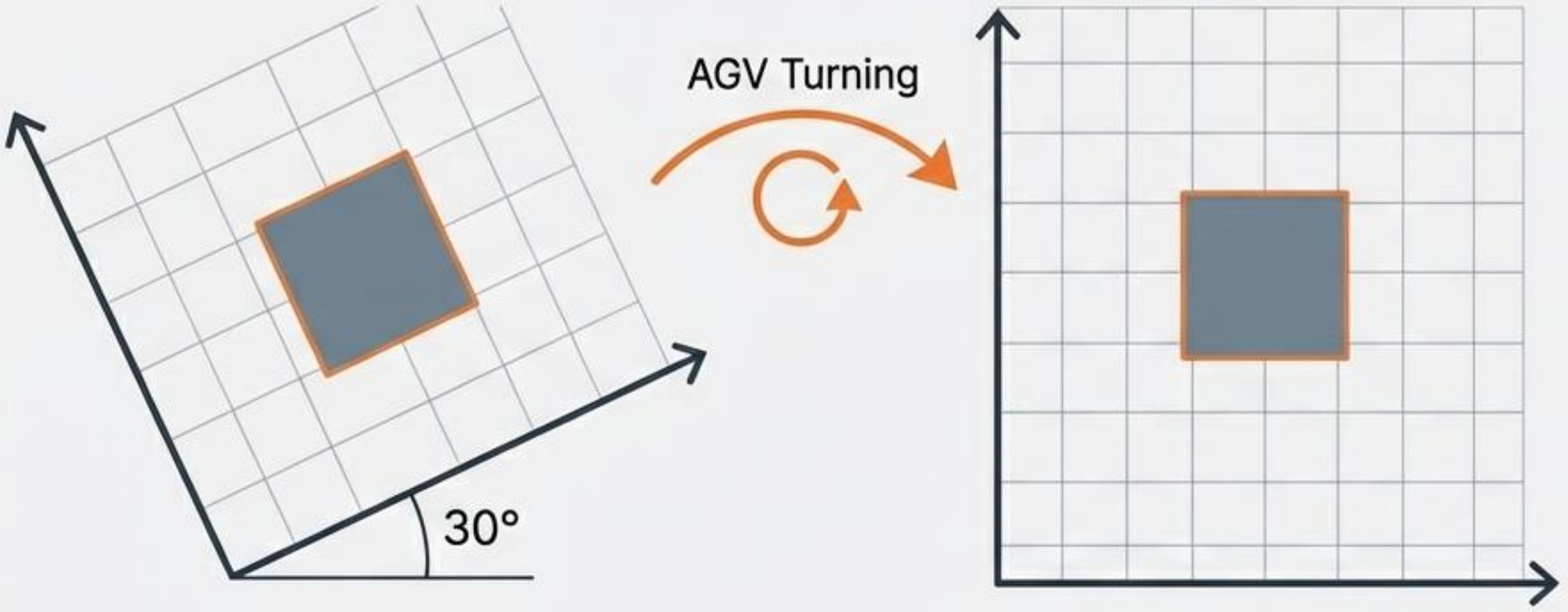
Output

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

Step 2: Ego-Motion Compensation

Solving the Rotation Problem.

Local AGV Frame (x_L, y_L) Global World Frame (X_G, Y_G)



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

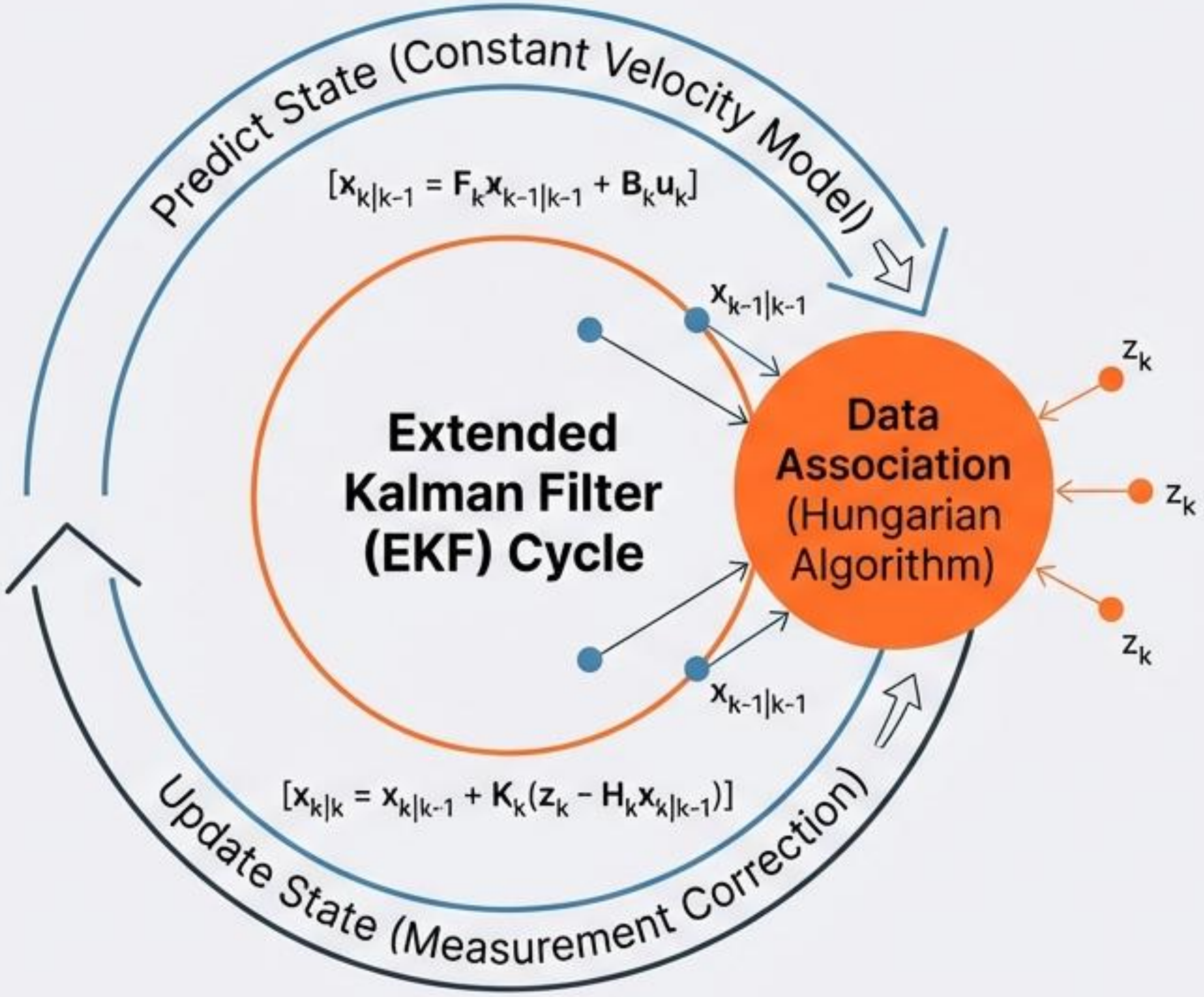
The Challenge: When the AGV turns, static objects appear to move relative to the sensor.

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.

Milad Jafari Barani - Milad.jafare@gmail.com

Step 3: Multi-Object Tracking & Estimation

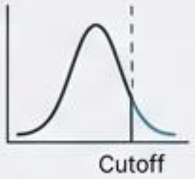


Algorithm: EKF utilizing a Constant Velocity motion model.

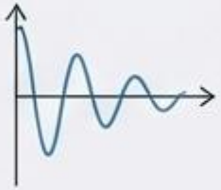
Association: Hungarian Algorithm matches new centroids to known tracks.

Stability Mechanisms:

- **Mahalanobis Distance Gating*: Statistical outlier rejection to prevent false tracks.

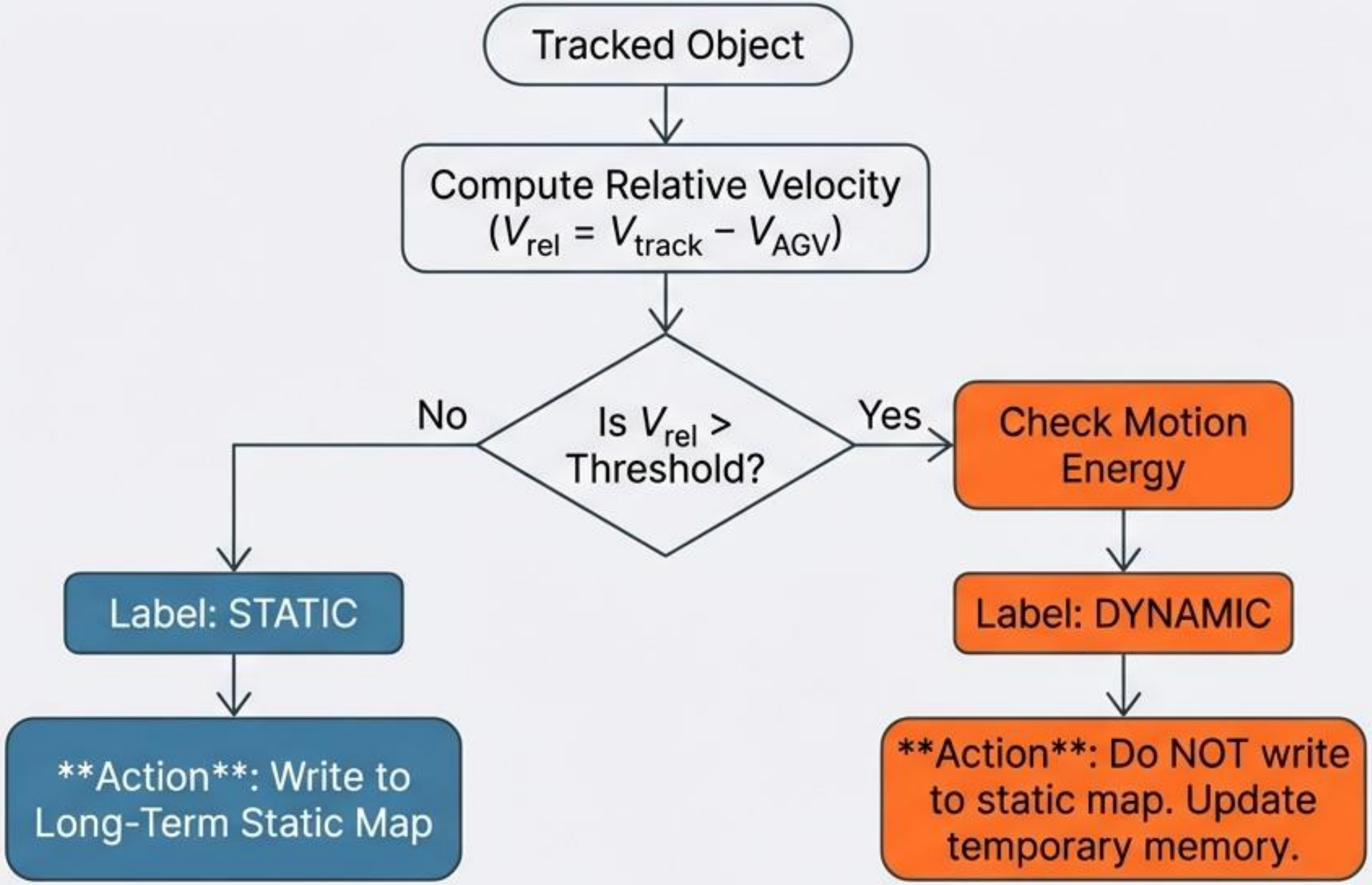


- **Adaptive Velocity Damping*: Smooths out jittery measurements for stable estimation.



Milad Jafari Barani - Milad.jafare@gmail.com

Step 4: Classification Logic & Decision Fusion



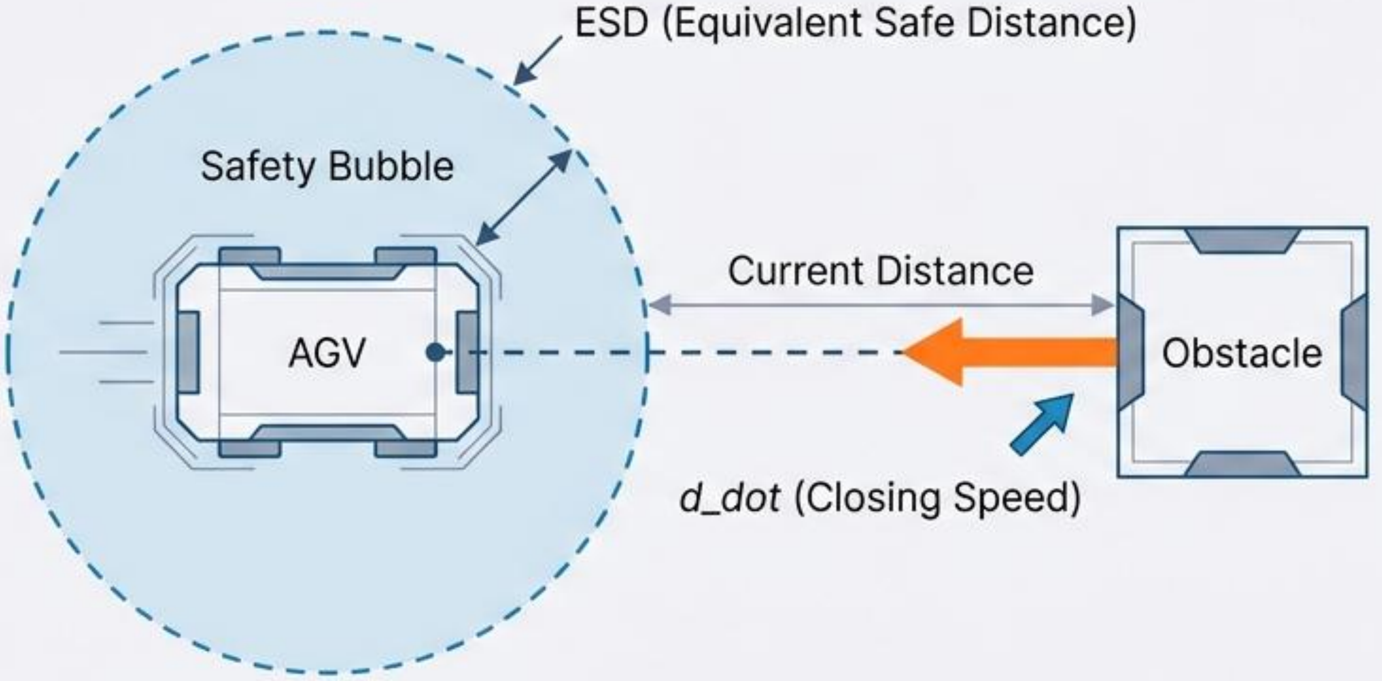
Sidebar Text

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

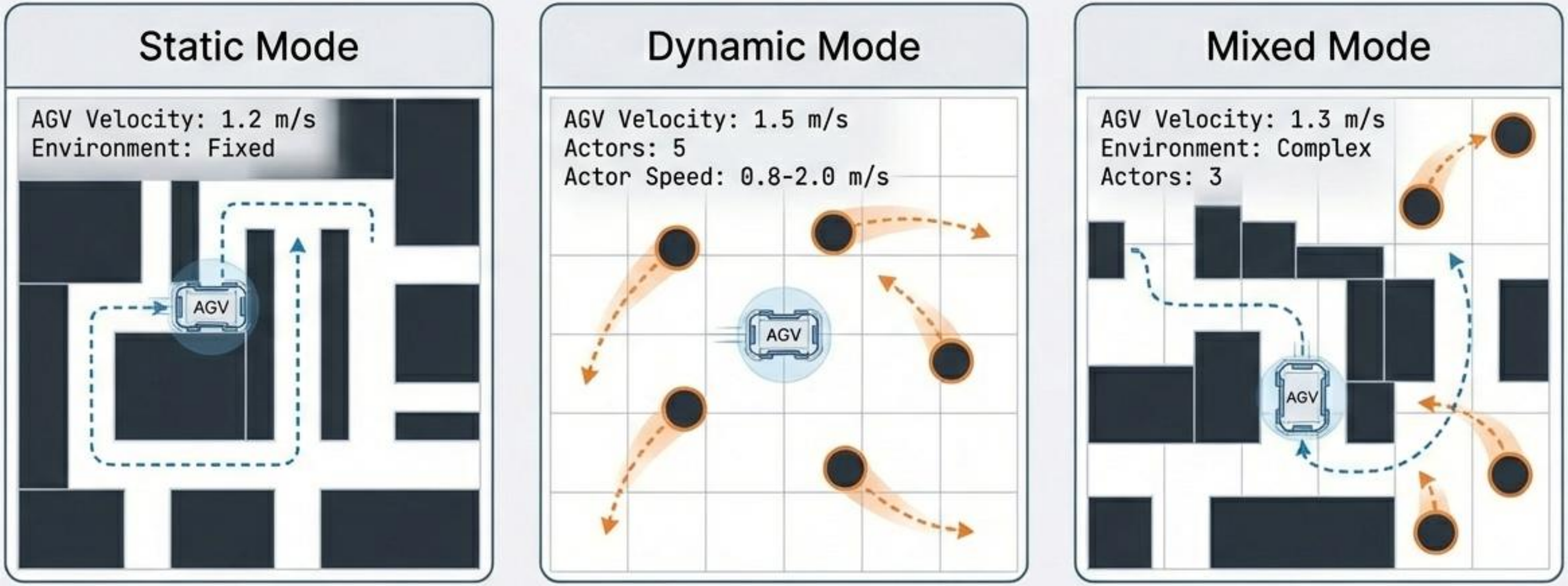


ESD (Equivalent Safe Distance): A dynamic safety buffer computed based on current speed and proximity.

***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



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

Thank You!

Questions & Discussion

Team #2 — LM-RtA Workshop

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

Lightweight Models for Real-Time Applications • January 2026