NAVISP-EL1-087 contract no. 4000145024/24/NL/AK/kg

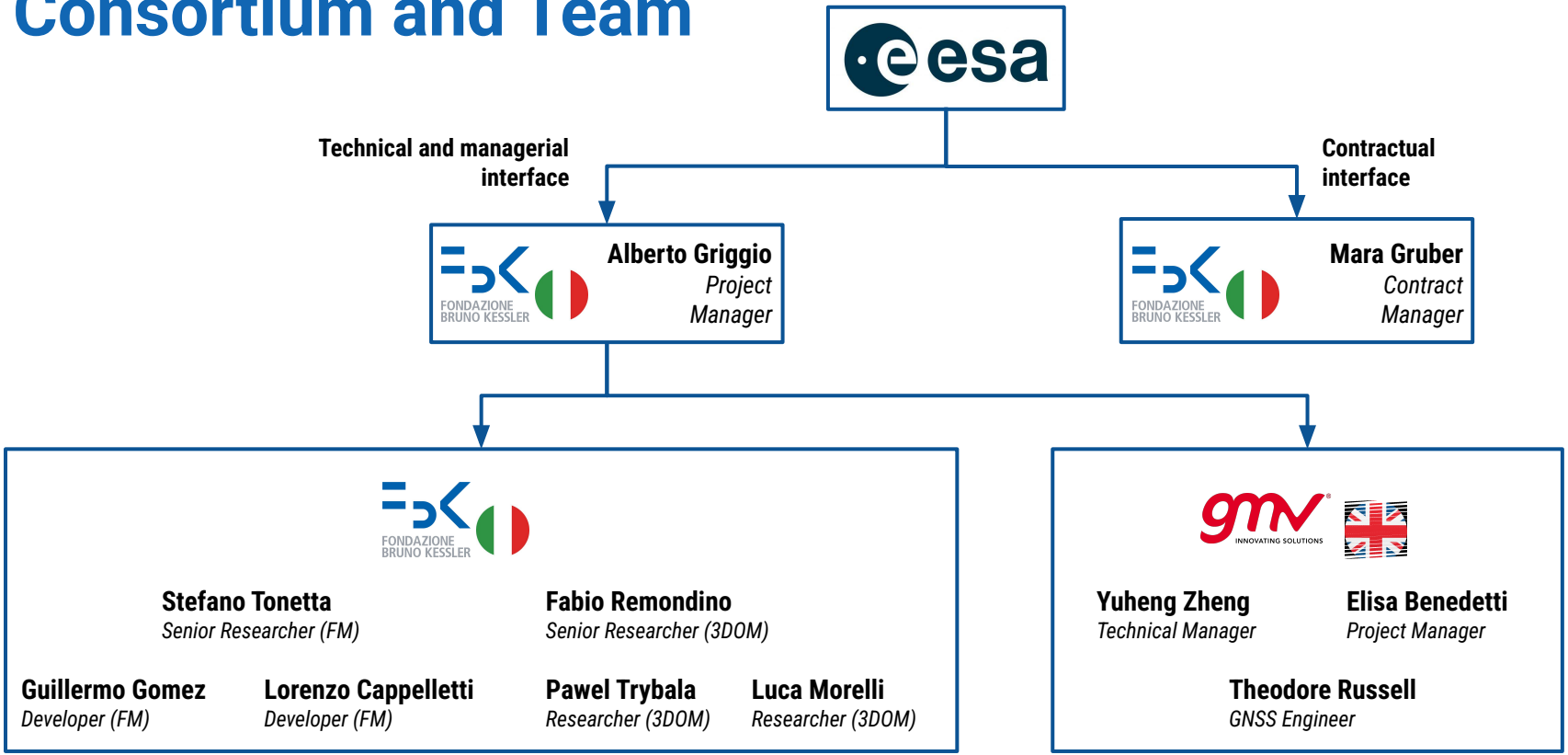
FONDAZIONE BRUNO KESSLER
FM FORMAL METHODS FOR SYSTEM & SOFTWARE DESIGN
3DOM 3D OPTICAL METROLOGY
gmv INNOVATING SOLUTIONS

Project Overview

"to develop a comprehensive, versatile and efficient toolkit tailored for the **verification of AI-based Positioning, Navigation, and Timing (PNT) systems**, capable to address a wide spectrum of AI-based PNT applications".

Focus	Problem	Approach	Expected Outcome
Verification and Validation (V&V) of advanced PNT solutions integrating AI/ML components	AI/ML very effective/ promising, but poses several new challenges for V&V	V&V methodology at multiple levels: • *unit* testing of ML algorithms • safety cage architecture for the PNT *component* • *system-level* V&V via simulation-based testing	• New AI-based PNT solution • V&V methodology at multiple levels • Demonstrator on relevant use case

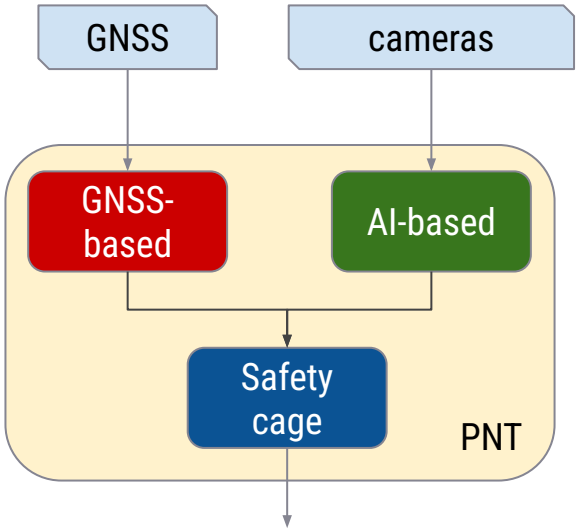
Consortium and Team



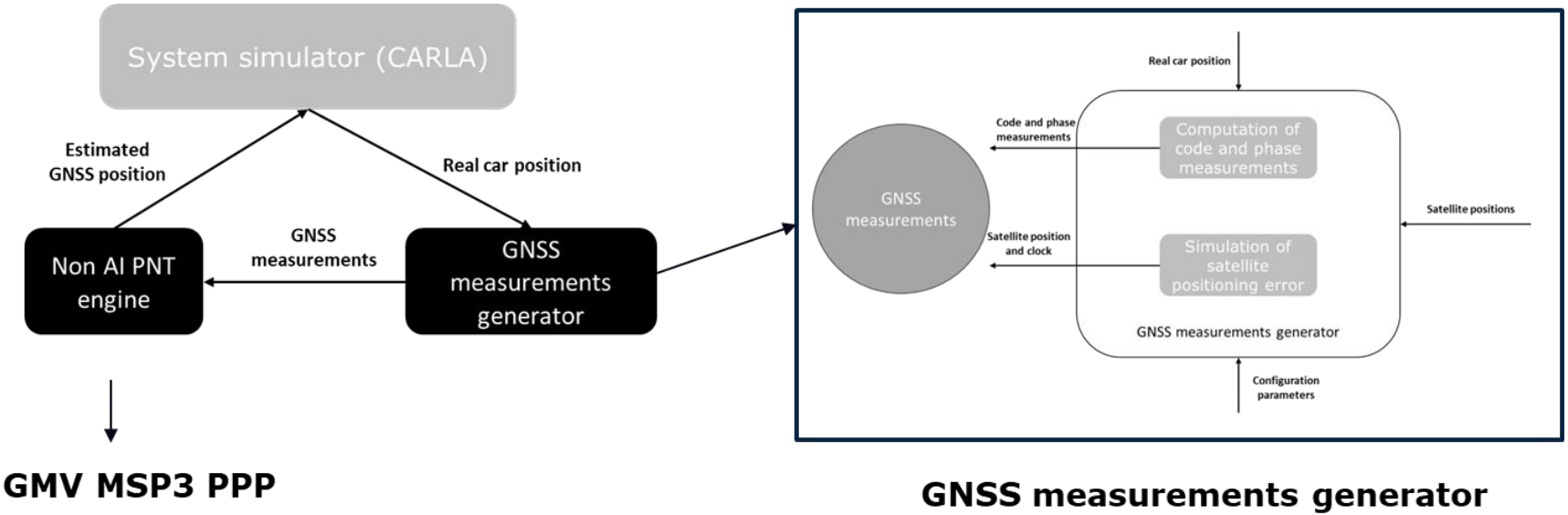
Verifiable AI algorithms for PNT

AI techniques to **complement rather than replace** traditional PNT algorithms to improve performance in non-nominal/degraded conditions.

- **Building blocks:**
  - Classical GNSS-based PNT engine
  - Image-based PNT (through COLMAP-SLAM) using AI to support image matching in challenging environments
  - Safety-cage architecture for anomaly detection via formal methods and run-time monitoring



Classical engine for PNT from data

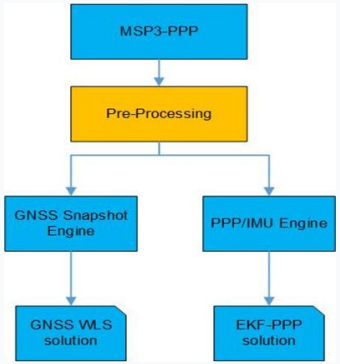


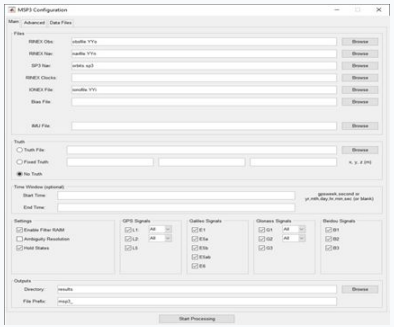
- Classical GNSS-based PNT engine is composed of the GMV GNSS measurements generator and the non-AI PNT engine

GMV tools

MSP3 PPP

GMV NSL Precise Point Positioning (PPP) engine

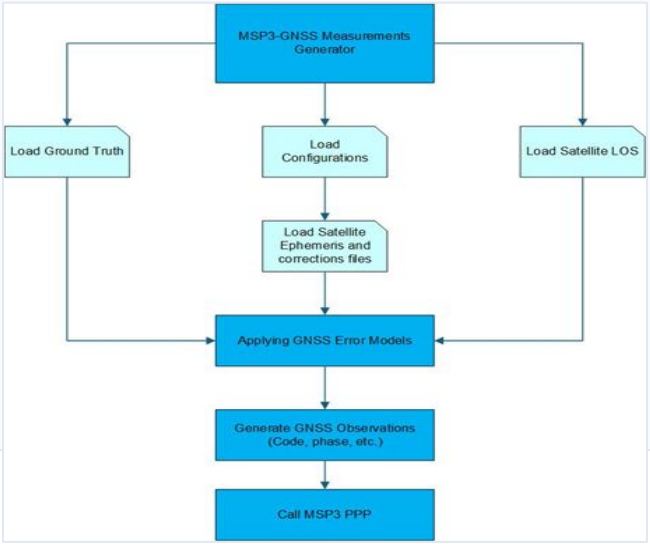




Compatible with multiple constellations (GPS, Galileo, GLONASS, Beidou) on multiple frequencies.

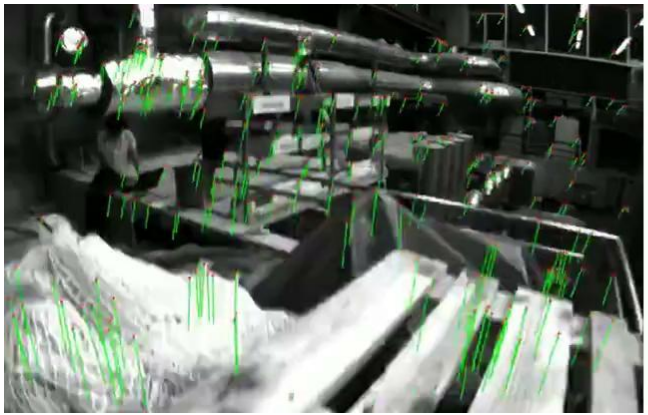
MSP3 GNSS Measurements Generator

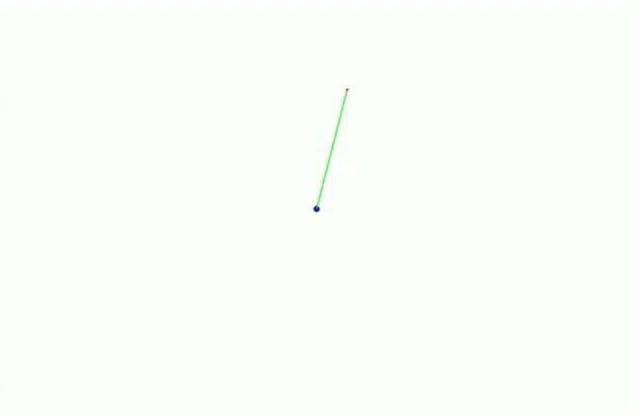
Generate GNSS observations for MSP3 PPP



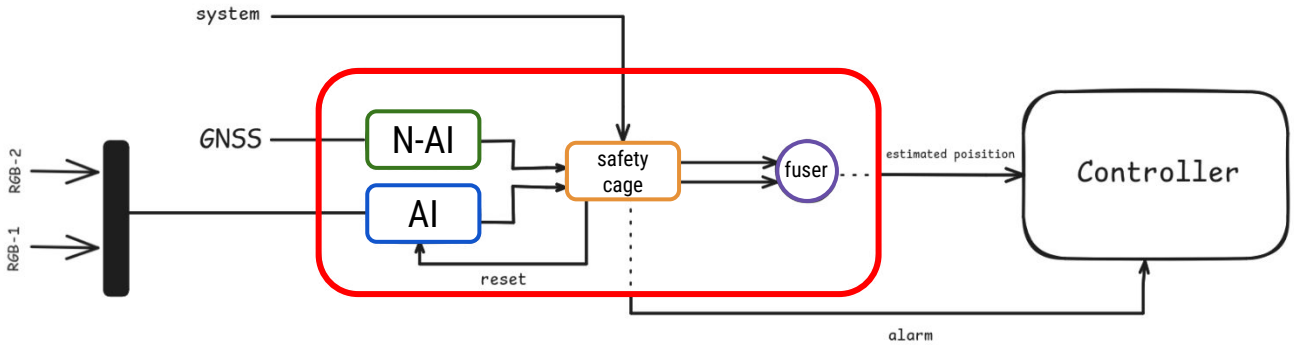
Deep learning aided image-based PNT engine

- Image-based (mono or multi-view) approach (videos from CARLA simulator)
- **COLMAP-SLAM** framework integrated with deep learning features to find image correspondences in complex scenarios (lack of texture, low illumination, reflective surfaces, etc.)
- Camera position / trajectory recovered through least squares bundle adjustment





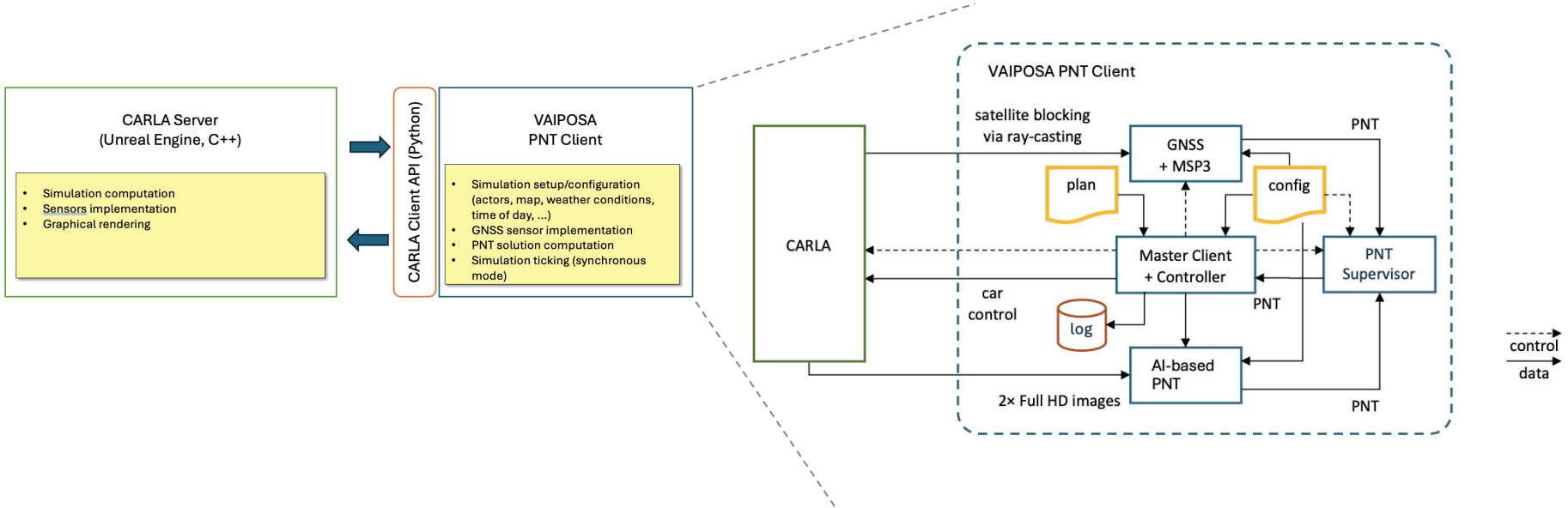
Logical architecture overview



PNT engine components:

- Classical GNSS-based PNT
- AI-based PNT using Visual SLAM techniques
- Safety Cage for fault detection and recovery
- Fuser for combining the outputs

Software architecture overview



GNSS measurements generator & non-AI PNT Engine

Objective

Generate and use synthetic GNSS signal observations to calculate the non-AI PNT solution of the Ego's position using **GMV's COTS** GNSS Measurement Generator and PPP Engine.

Interfaces and Data Exchanged

Satellite LOS

Precise satellite positions computed each epoch and passed to the CARLA python layer.

Casts rays from Ego toward satellites to determine **'Line-Of-Sight'** (environmental blocking).

Environment Data

Data from CARLA used to augment GMV's GNSS Measurement Generator.

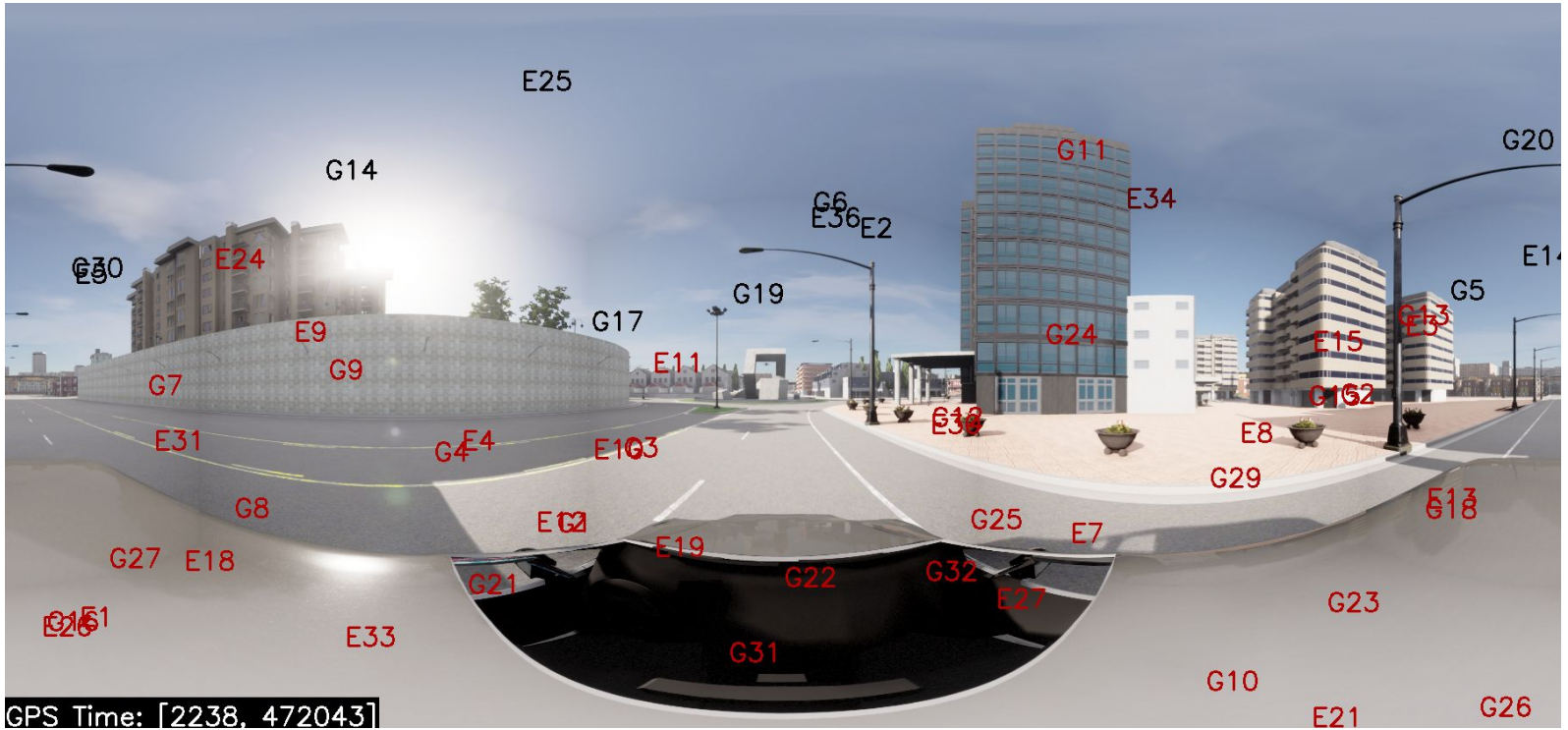
Improves realism in **non-open-sky** scenarios by integrating urban geometry.

Data Sync

Ground-truth and PNT result exchanged via **MAT-file interface**.

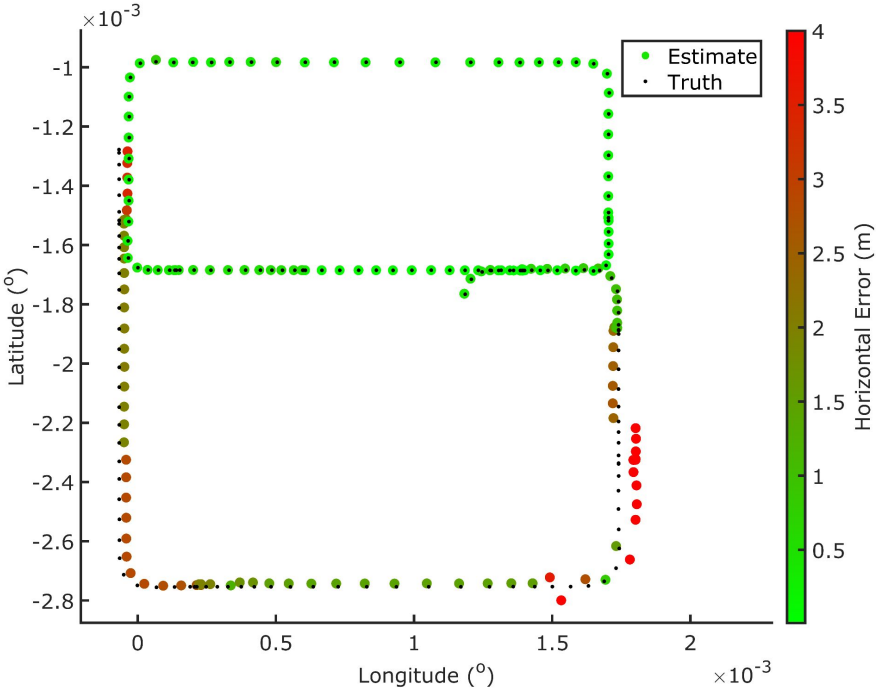
Reliable communication ensured using a robust **file-lock system**.

GNSS measurements generator & non-AI PNT Engine



Sub-urban scenario

GNSS measurements generator & non-AI PNT Engine



AI-based PNT engine

COLMAP-SLAM

Based on the **COLMAP API**, a premier open-source SfM framework for photogrammetry and computer vision.

AI-Driven Visual Feature Extraction & Matching

Local Feature Extraction

SuperPoint or ALIKED

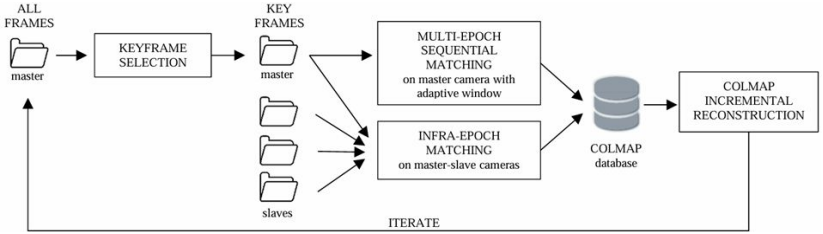
- Optimal balance of accuracy and efficiency
- Resilient to **illumination** and **viewing angle** changes

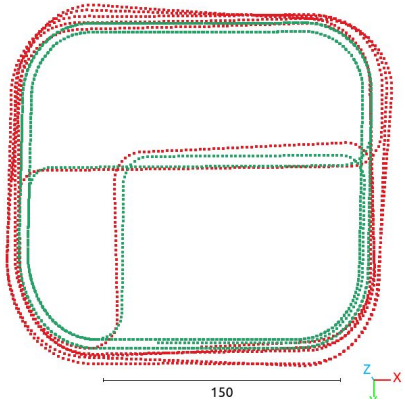
Feature Matcher

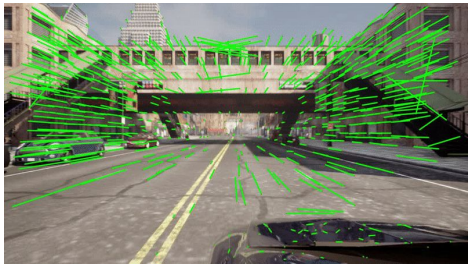
LightGlue

- Graph-based neural network (SuperGlue inspired)
- Superior reliability over classical RANSAC on complex data

AI-based PNT engine







- VO-only trajectory
- Ground truth

Engine I/O

Input Data

- Synchronized stereo image pairs

Output

- Relative pose change of the vehicle
- Expected frequency ~5 Hz

Local Features & NN Matchers

Feature Extraction: SuperPoint / ALIKED

Feature Matching: LightGlue

Safety cage

Situation Detection

Detect **critical situations** by checking failure properties and calculating weighted confidence scores.

Run-time Monitoring

- Check extended temporal conditions using monitoring techniques.
- Raise **alarms** in case of critical situations (e.g., faults).

Weight Computation for PNT Engines

Based on feature evaluation and fault conditions to encode **confidence on the solution quality**

GNSS-based engine:
Horizontal Protection Level

AI-based engine:
Covariance matrix, luminosity, feature counts, and extraction time

Safety cage flow

01. Read Inputs

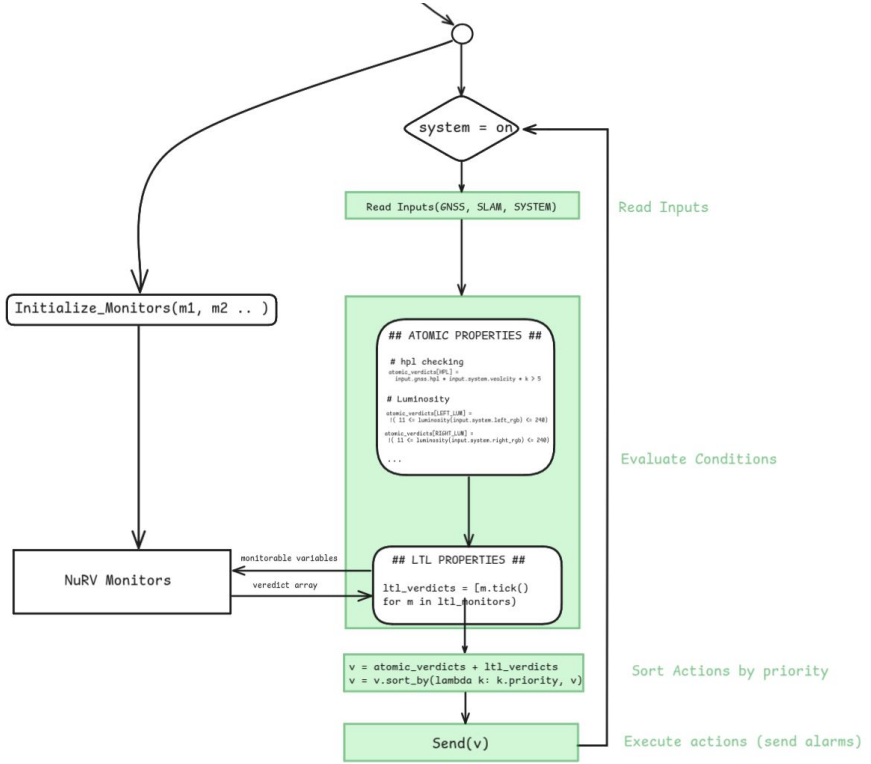
GNSS-based PNT, AI-based PNT, System

02. Conditions Evaluation

Evaluate Atomic & Temporal Failure Conditions

03. Perform Actions

- Configure fuser inputs
- Send alarms to controller
- Reinitialize AI component



Safety cage weight computation

Objective: Calculate weights for the solutions produced by individual PNT engines

- Encode the confidence on the solution quality

GNSS-based Engine

Evaluation features and conditions:

- Number of satellites in view of the receiver
- Horizontal Protection Level (HPL)

AI-based Engine

Evaluation features and conditions:

- Number of features extracted per image
- Feature extraction time
- Covariance matrix of the relative pose change

Safety cage weight computation

Dynamic weight $w \in [0, 1]$ for each engine, computed as follows:

1. Qualitative Cutoffs (Hard-Gating)

Check fault conditions affecting engines are detected.

- If "cutoff" properties are violated, engine is untrustworthy

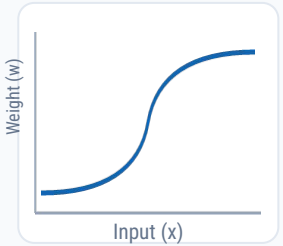
Raise alarm

Block engine output (set w=0)

2. Quantitative Scoring (Soft-Weighting)

Fine-grained weight calculation.

- Calculate weight using **sigmoid functions**
- Weight reflects current **"health"** of the engine



- Based on **rolling window statistics**

Qualitative cut-off properties

GNSS-based PNT

$G[0,3]\ hpl < th$ (where *th* is threshold based on road width)

Action: Raise high_hpl alarm if violated

AI-based PNT

Luminosity

$G[0,3]\neg low\_luminosity$

If violated, re-initialize PNT by swapping **SuperPoint** for **ALIKED**.

Frozen Image

$G[0,2]\neg frozen\_image\{l,r\}$

Raise alarm

Camera Liveness

$\neg cam\_\{l,r\}\_alive\ U[0,10]\ cam\_\{l,r\}\_alive$

Raise camera_dead alarm

Quantitative weighting properties

Compute the weight *w* using a **parametric sigmoid function**

$$f(x; c, \alpha) = \frac{1}{1 + e^{-k(x-c)}}$$

GNSS-based PNT

$x = -hpl(t)/W_{hpl}$, where W_{hpl} is the mean HPL of the last 20 steps

- **Rationale:** Detect "spikes" in the HPL
- Exploits MSP3 inner workings to stabilize HPL under nominal conditions

AI-based PNT

Take the **minimum value** among:

- Feature drop ratio
- Extraction time ratio
- Covariance spike

Rolling window of 10 steps for averages

Pose Information Fuser

Method: Extended Kalman Filter (EKF)

- Widely used and well-tested off-the-shelf component
- Supports temporary outage of selected sensors

15-Dimensional Vehicle State

Pos: XYZ, Ori: $\phi\theta\psi$ | Vel: XYZ, Ang Rates: $\phi\theta\psi$ | Acc: XYZ

Input Data

- Absolute positions (non-AI)
- Relative poses (AI)
- Safety Cage weights

Output

- Absolute vehicle pose
- Frequency: ~5 Hz (highest of input)

Implementation

CARLA Simulation Platform (http://www.carla.org)

- State-of-the-art engine with extensive documentation, APIs, and community support
- Autonomous driving use case — **Ideal for PNT**

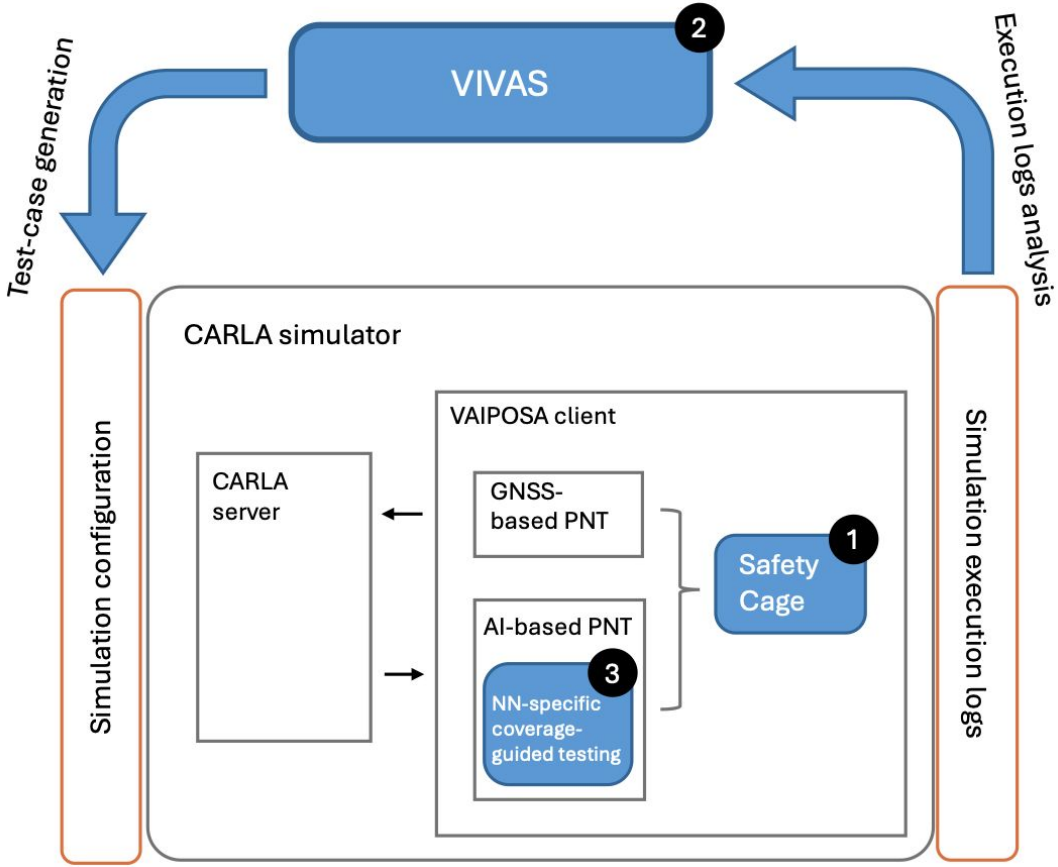
Docker Deployment

Custom image built with all dependencies necessary for the **VAIPOSA demonstrator** execution environment.

Verification engine

3 levels of V&V

1. **safety cage** component inside the VAIPOSA PNT
2. **system-level** simulation-based testing with VIVAS
3. NN-specific coverage-guided **component-based** testing



Component-level testing of NNs

The base AI-based PNT engine of VAIPOSA uses two different Neural Networks:

1. SuperPoint NN

Input: a single image

Output: set of keypoints (2D points stable under perspective/radiometric distortions) and descriptors.

2. LightGlue NN

Input: two sets of keypoints and descriptors from SuperPoint.

Output: corresponding points on the two images.

Combined (SuperPoint + LightGlue) network:
Input: a pair of images; Output: corresponding points on the two images

Tests performed either on SuperPoint or on the Combined Network

- Tests on LightGlue in isolation are not meaningful in our context

Data sets

Synthetic Data Generation with CARLA

The dataset consists of images from a car driving on a circuit, specifically designed to isolate factors affecting coverage scores.

Dataset Composition

- Divided into **5 subsets**
- **865 images** per subset
- Varying weather & traffic conditions

Primary Goal

Better isolate specific environmental factors that impact the overall coverage score obtained by the neural networks.

Tested properties

SuperPoint Network Property

Given a pair of images in different context conditions (e.g., day/night), captured at the same time and pose:

Given keypoint PD on image D, and point PN = (xD, yD) on image N, then ∃ QN s.t. || PN - QN || < ε

Combined Network Properties

1. Left and right camera pictures, same timestamp:

Given keypoint P on (L) and corresponding Q on (R), then: || yL - yR || < ε

2. Different context conditions, same timestamp/pose:

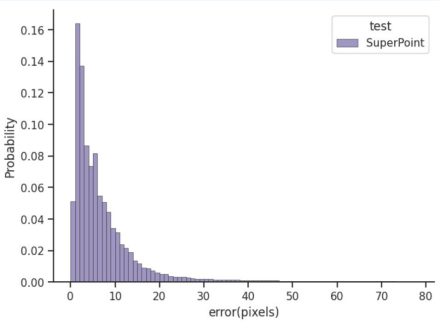
Given keypoint PD and corresponding QN extracted from N: || PD - QN || < ε

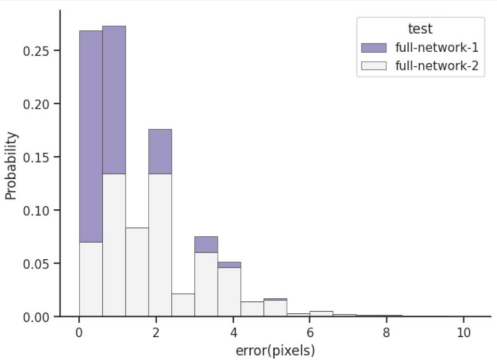
NN component level testing results

Coverage Results

- **Consistent results** across all metrics considered
- **No significant differences** between traffic and no traffic datasets
- **Significant drop** under low visibility (dark, rainy / foggy) - images contain less extractable information

Robustness Checks Results





Visual confirmation of NN performance stability

SuperPoint-1: >60% of errors are < 5px

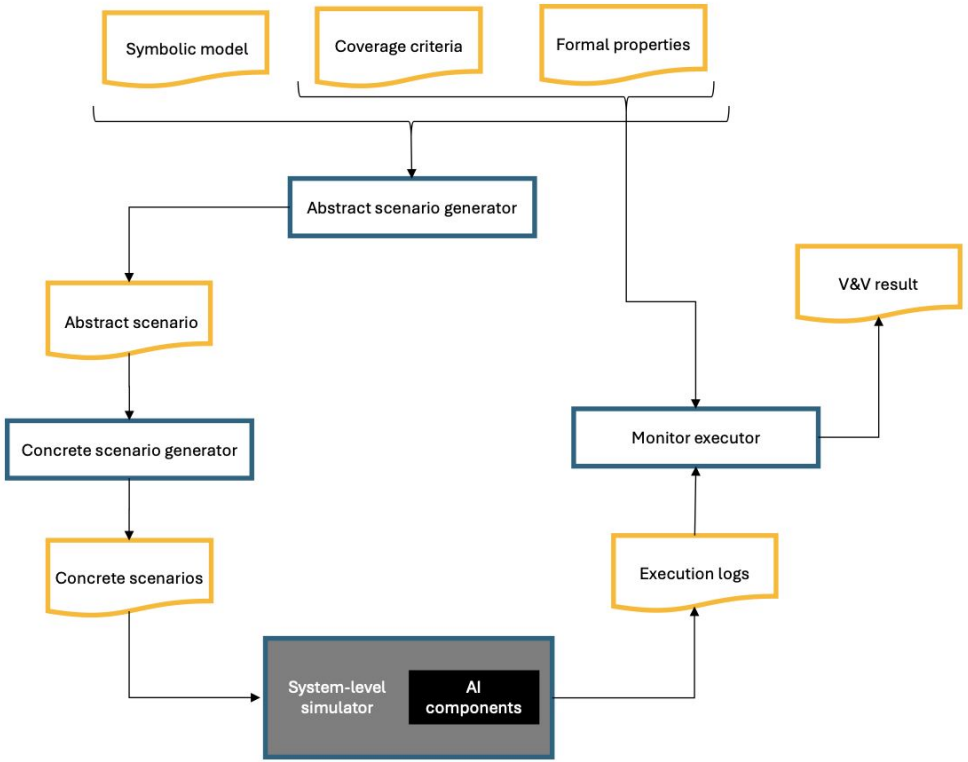
Combined network:

1. 85% of errors are < 1px
2. 60% of errors are < 3px

stability under radiometric changes induced by different light and weather conditions.

Simulation-based system-level V&V with VIVAS

- Enumeration of abstract scenarios with *symbolic techniques*
  - **Covering all situations of interest**
- Translation into concrete simulator configurations
- System-level simulator (**CARLA**)
  - Integration of the new PNT engine and high-fidelity GNSS sensor
- Execution Monitor
  - Evaluation of properties and coverage
- ***Reuse of ESA-funded VIVAS project***



Coverage criterion definition

Use the idea of **situation coverage**: focus on interesting situations for PNT navigation

Expressed as a combination of predicates over environment dimensions

Map Coordinates

Choose among a fixed set of locations of relevant cities

Weather & Time

Single predicate ranging over conditions

- `ClearNoon`
- `CloudySunset`
- `HardRainNight`

Route & Obstacles

- Abstract POIs with semantic tags
- Independent from map content
- Manual user annotation

Navigation task: Sequence of abstract POIs

Coverage criterion definition

Waypoints to follow and obstacles on the route (cont'd)

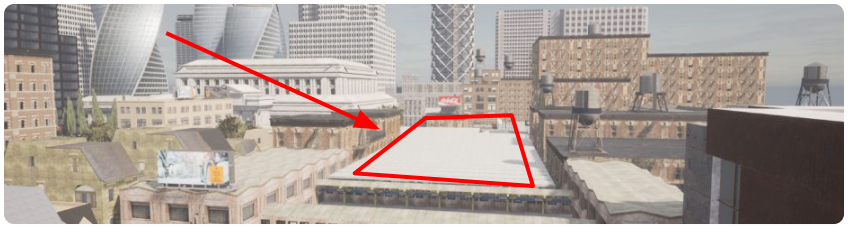
Map Annotation

Developed a **map editor** to annotate CARLA maps with semantic tags.

Tunnel Emulation

Emulate tunnels by placing concrete blocks as tiles.

Workaround for CARLA map editing limitations.



Coverage criterion definition

Fault predicates for GNSS sensors and onboard cameras

GNSS Fault

- Upper bound N on available satellites and constellation
- Randomly pick N satellites at each tick during fault
- *Significant impact on MSP3 solution quality*

SLAM Fault

- Camera sensors (L, R, or both) are broken - black images
- Camera sensors are stuck - no image refresh

Evaluation - system-level experiments

Evaluate quality of the **PNT solution**, regardless of the capability/adequacy of the autonomous driving controller

Trajectory Error Metrics

ATE (Absolute Trajectory Error)
Global consistency vs. Ground Truth

RPE (Relative Pose Error)
Local accuracy (drift) between frames

Errors computed wrt. ground truth

Simulation Statistics

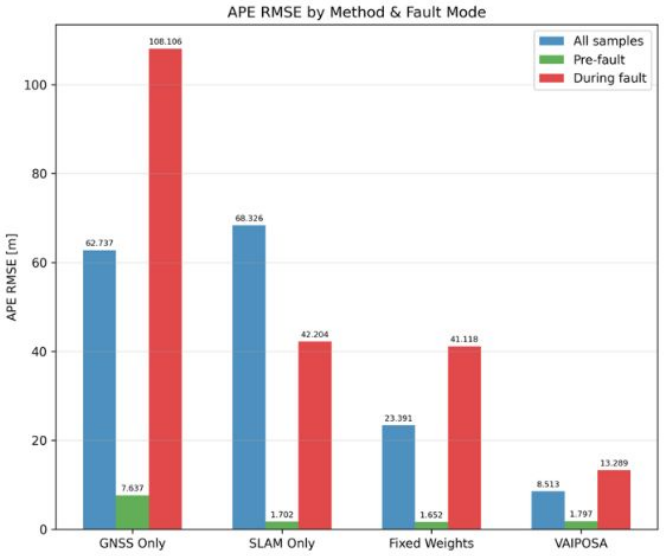
Percentiles, mean, median, **RMSE**

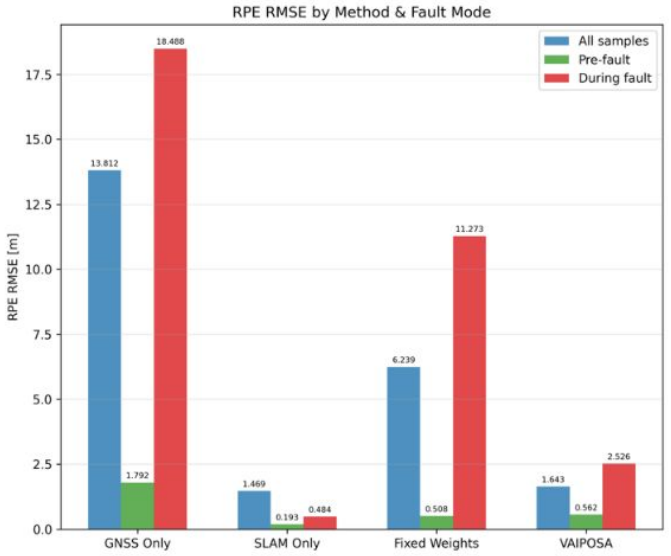
- **Expected outcome:** Lower error vs. GNSS-only solution.
- **Safety cage impact:** Default VAIPOSA (*dynamic weights*) vs. Fixed Weights (*disabled cage*).

System-level experiments - results summary

Default VAIPOSA obtains **very good performance** across different metrics

- *Improved ATE over baselines* (both median and RMSE)
- Improved MAD over all metrics
- RPE comparable to SLAM-only
- Safety cage essential in faulty situations (as expected)





Demo

https://youtu.be/iLMbVhxJ7XI

System-level experiments - safety cage impact

Analysis of Correlations: Trajectory Errors vs. Safety Cage Alarms

KPI	Min	Max	Median	Mean	MAD	10th percentile	90th percentile
Q (recall)	0.42	0.99	0.91	0.89	0.04	0.79	0.95
P (precision)	0.03	0.99	0.50	0.49	0.21	0.19	0.86

Recall Analysis

Recall is very good

Safety cage effectively detects trajectory errors across most test scenarios.

Precision Analysis

Precision could be improved:

- Sometimes too eager in raising alarms
- No distinction between warning and severe alarms

Conclusion: Results show no negative impact on quality in nominal situations.

Impact of Safety Cage

Appropriateness of the safety cage alarms

- analyze possible correlations between alarms and trajectory errors

Recall

"whenever there is a fault, an alarm will be triggered within a time limit"

Quantitatively:

$$Q = \inf_t \max(m_{err}(t), m_{alm}(t))$$

where:

$$m_{err}(t) = 1 - \frac{error(t)}{max\_error} \qquad m_{alm}(t) = 1 - \frac{time\_distance\_to\_alarm(t)}{max\_time}$$

Precision

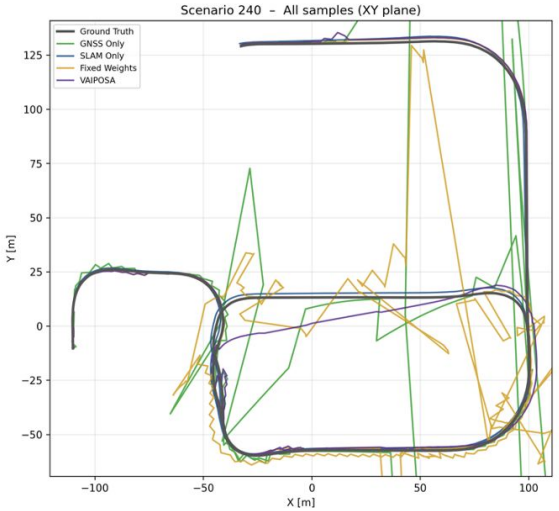
"whenever there is an alarm, a fault occurred within a time limit"

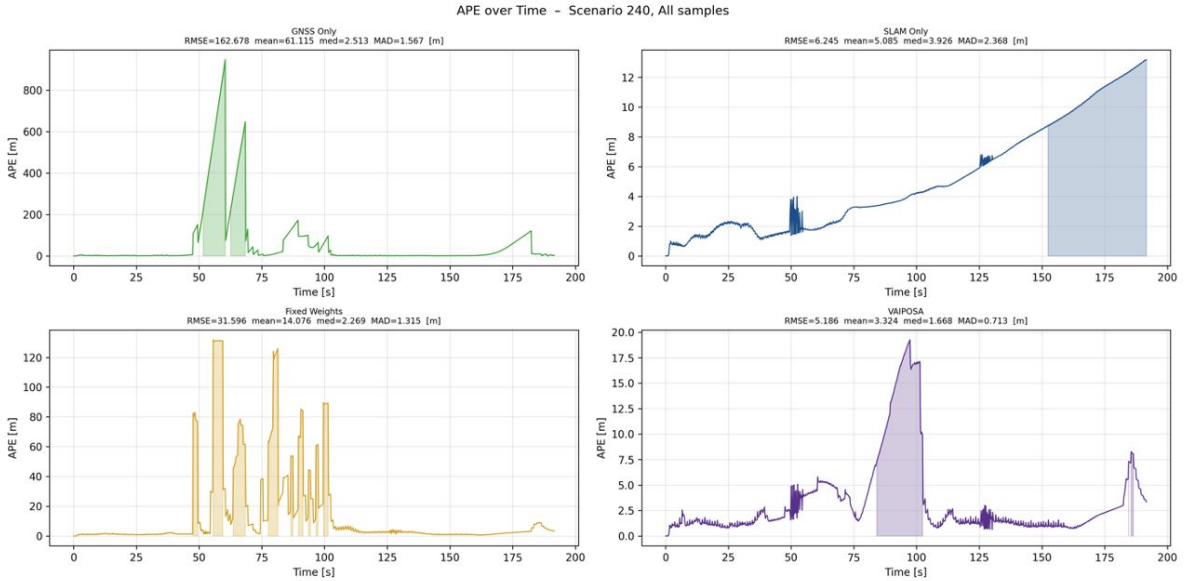
Quantitatively:

$$P = \inf_t(\max_s(error(s) * \exp(-(t - s)/max\_time)^2)$$

Safety cage impact - scenario with a GNSS fault

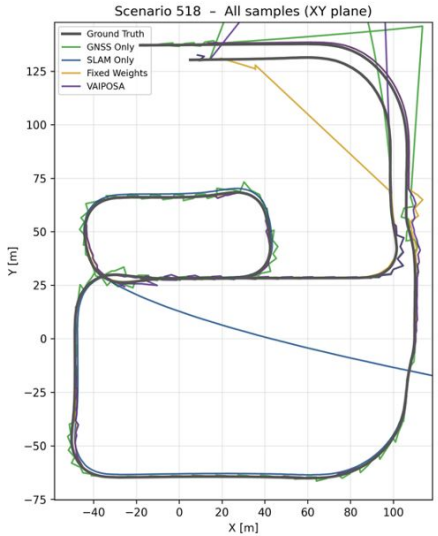
Safety cage detects the fault, effectively reduces the GNSS-based PNT weight, and significantly improve the ATE

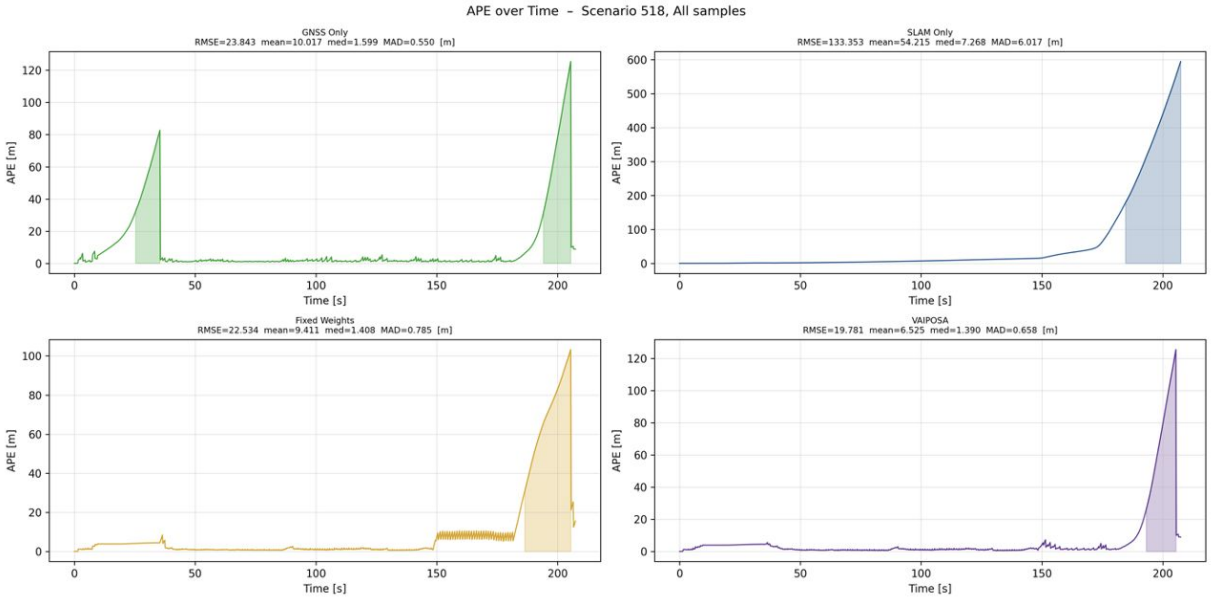




Safety cage impact - scenario with multiple faults

Navigation inside a tunnel, and camera fault: error increases when both engines fail, but safety cage still mitigates it





Lessons learned and insights

Monitoring & Methodology Monitoring property definition lacks established standards; parameter tuning relies on intensive 3D simulation experimentation.

- *Stateful design: Monitors must track temporal trends (e.g., HPL stabilization) to ensure robustness against transient errors.*

Performance Fusing global and local PNT solutions key to enable robust performance, even with a simple filtering algorithm. A fundamental trade-off exists between absolute global accuracy (filter-based) and trajectory smoothness (AI-based solutions).

- *Potential improvement: bypass the internal filtering done by the GNSS-based engine to access the raw measurements for better control by the safety cage*

Lessons learned and insights

Simulation-based V&V Using a simulator with 3D environment models, jointly with the automatic testing procedures of VIVAS, was essential for debugging and created a realistic and high-performance benchmarking environment

Further research and enhancements

- The safety cage is highly conservative (high recall), resulting in frequent false positives and moderate precision.
- **Quantifying final safety guarantees, including Time-to-Alarm and HPL for fused outputs, remains a critical challenge.**

Exploitation of innovation

Product Development Plan

Current TRL is 4

A plan was developed to bring it to level 6-7, with steps including:

- **Integrate sensors:** Add IMU and odometry to minimize SLAM drift.
- **Refine safety cage:** Improve alarm precision to reduce false positives.
- **Unify integrity concept:** Develop a unified multi-sensor uncertainty and integrity framework.
- **Test HIL and real world data:** Deploy hardware-in-the-loop for testing (i.e. failure scenarios), test with real world data.

Market Analysis

PRIMARY MARKET

SAE Level 3/4 autonomous vehicles requiring high-integrity localization.

COMPETITIVE EDGE

Unique "Safety Cage" providing transparency/diagnostics for AI certification.

BARRIERS

Data validation, AI safety standards, and hardware resource limits.

Conclusions

The VAIPOSA project successfully demonstrated that AI can be part of a safety-critical navigation system when governed by a verifiable supervisory architecture.

Architecture

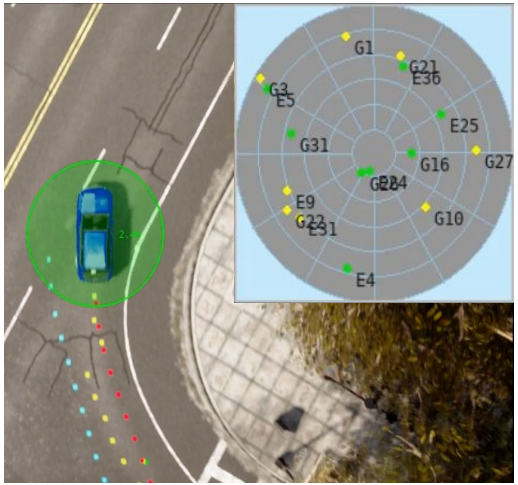
Defined a functional, integrated hybrid PNT engine using COTS and AI components.

Safety Cage

Proved runtime monitoring of AI mitigates "black box" risks effectively.

V&V Pipeline

Created a robust, automated testing environment using CARLA and VIVAS.



Future Extensions & Path Forward

- **Addressing False Alarms:** Refining "stateful" monitors to reduce conservative false positives.
- **Sim-to-Real Gap:** Transitioning from CARLA simulation to real-world vehicle testing for sensor noise.
- **IMU Integration:** Adding Inertial Measurement Units for stability during rapid maneuvers or outages.
- **Safety guarantees:** develop a unified uncertainty and integrity framework for the fused output.

Dissemination

Publications in relevant international conferences:

Deep Learning in Visual Odometry for Autonomous Driving

Morelli, L., Trybała, P., et al. | Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., 2025

https://doi.org/10.5194/isprs-archives-XLVIII-1-W5-2025-93-2025

A Safety Cage for Reliable GNSS and AI-based PNT: an Experience Report

Gomez, G., Griggio, A., et al. | Accepted at 45th International Conference on Computer Safety, Reliability and Security (SAFECOMP), 2026

Thank you

https://fm.fbk.eu https://3dom.fbk.eu https://www.gmv.com/en

FONDAZIONE BRUNO KESSLER

FM FORMAL METHODS FOR SYSTEM & SOFTWARE DESIGN

3DOM 3D OPTICAL METROLOGY

gmv INNOVATING SOLUTIONS