

FASCINATE Final Presentation

NAVISP-EL1-088

PROOF-OF-CONCEPT OF ADVANCED
NAVIGATION ALGORITHMS BASED ON
FACTOR GRAPH OPTIMIZATION

05/10/2026

©GMV 2026

The copyright in this document is vested in GMV.

This document may only be reproduced in whole or in part, or stored in a retrieval system, or transmitted in any form, or by any means electronic, mechanical, photocopying or otherwise, either with the prior permission of GMV or in accordance with the terms of
ESA Contract No. 4000146356/24/NL/MP

© GMV 2026



Agenda

1. Motivation and Objectives
2. FGO-Nav Architecture
3. FGO-Nav Software Development
4. Experimental Validation Campaign
5. Performance Assessment
6. Conclusions and Next Steps

Motivation and Objectives

FASCINATE Project

Overview

Motivation

- Growing demand for high-accuracy and robust navigation capabilities across multiple applications, particularly in safety-critical domains (i.e. automotive)
- Positioning and navigation challenges in urban environments
- Limitation of conventional WLS and EKF-based approaches under harsh GNSS conditions
- Growing adoption of multi-sensor navigation solutions to improve PNT resilience, robustness and availability

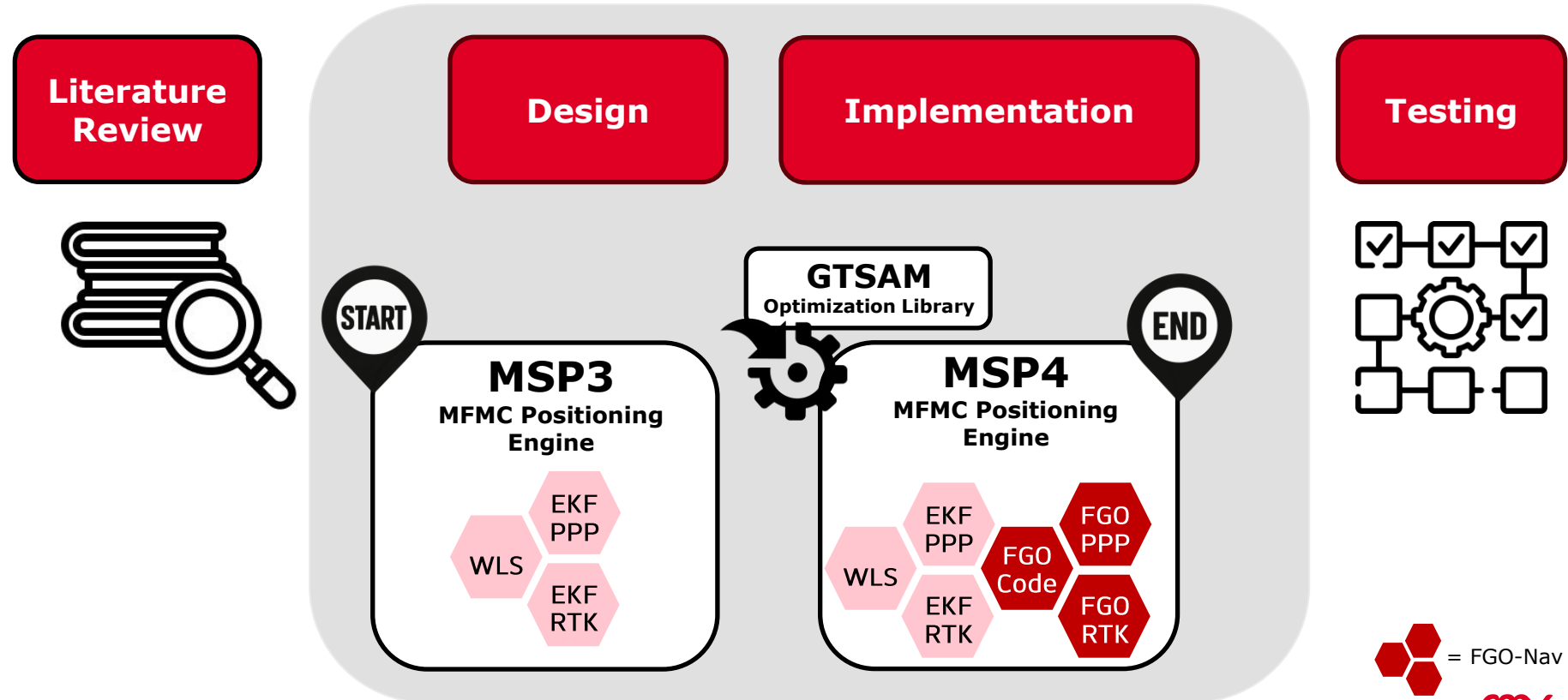
FASCINATE Contribution

- Exploitation of Factor Graph Optimisation (FGO) in GNSS navigation
- Creation of a unified framework supporting a consistent comparison between FGO and conventional WLS/EKF approaches
- Application of FGO to both code-based and carrier-phase-based positioning methodologies, including PPP and RTK
- Validation of the benefits and limitations of FGO across diverse environments and extended real-world GNSS+IMU+odometer data collection campaign (~25h)



FASCINATE Phases and Key Developments

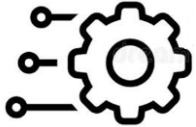
GMV Software Tools Evolution



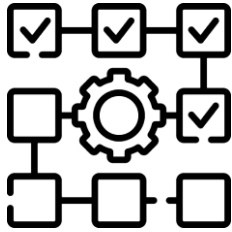
FASCINATE Phases and Key Developments

Testbed Integration and Data Collection

**Hardware
Integration**



**Hardware
Testing**



**Demonstration
Campaign**

**25h Data
Collection**

GNSS, IMU, Odometer,
in real-word scenario
(four main UK cities)



**Performance
Assessment**

25h PVT Solutions

Exploiting and comparing WLS
and FGO-Code, EKF-RTK and
FGO RTK, EKF-PPP and FGO-PPP



FGO-Nav Architecture

FGO Architecture

Design

FGO Core

- Fixed-window graph, (YAML) configurable as incremental updates or batch fixed-lag smoothing

Code & PPP Factors

- Pseudorange + carrier phase (+ optional Doppler)

RTK Factors

- Double-difference pseudorange + double-difference carrier-phase block factors

Dynamics Layer

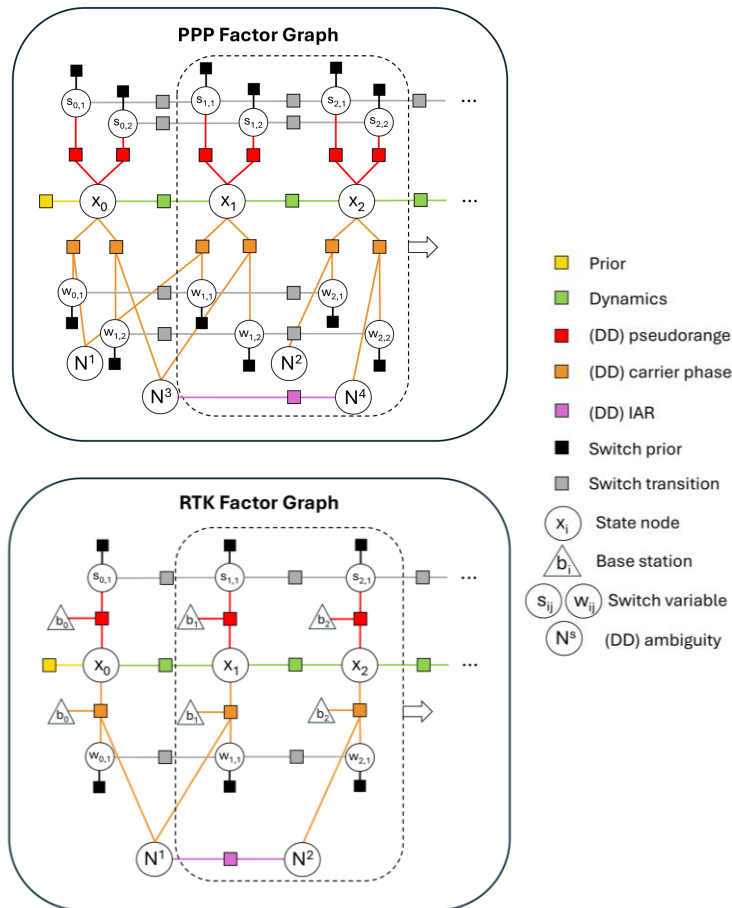
- IMU/odometer-based dynamics constraints

Miscellaneous Factors

- Process-noise links, clock priors/jump handling, tropo and iono priors, ambiguity continuity/reset logic

Robust Modes

- None, M-estimator, or switch-variable factors



FGO Architecture

Design

Robust Estimation

- Add configurable robust factor modeling including M-estimator kernels and switchable constraints

Code State Vector

- We estimate receiver position, receiver velocity, clock bias, atmospheric states, and SCs

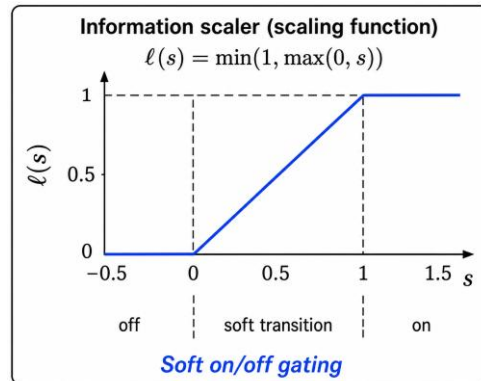
PPP State Vector (UD, uncombined PPP)

- We estimate receiver position, receiver velocity, clock bias/drift, atmospheric states, ambiguities, and SCs

RTK State Vector

- We estimate receiver position, receiver velocity, DD carrier-phase ambiguities, and SCs

Switchable Constraints:



Code: $\mathbf{x}_k^{\text{code}} = \left[\mathbf{p}_{i,k}^\top, \mathbf{v}_{i,k}^\top, \mathbf{c}_k^\top, T_{i,k}, \{I_{i,j,k}\}_{j \in \mathcal{S}_k}, \{s_{i,j,k}\}_{j \in \mathcal{S}_k} \right]^\top$

PPP: $\mathbf{x}_k^{\text{PPP}} = \left[\mathbf{p}_{i,k}^\top, \mathbf{v}_{i,k}^\top, \mathbf{c}_k^\top, \mathbf{d}_k^\top, T_{i,k}, \{I_{i,j,k}\}_{j \in \mathcal{S}_k}, \{N_{i,j,k}^q\}_{(j,q) \in \mathcal{O}_k^\phi}, \{s_{i,j,k}\}_{j \in \mathcal{S}_k} \right]^\top$

RTK: $\mathbf{x}_k^{\text{RTK}} = \left[\mathbf{p}_{i,k}^\top, \mathbf{v}_{i,k}^\top, \{N_{i,b,j,k}^{\Delta \nabla, q}\}_{(j,q) \in \mathcal{O}_k^{\Delta \nabla \phi}}, \{s_{i,b,j,k}\}_{j \in \mathcal{S}_k} \right]^\top$

FGO Architecture

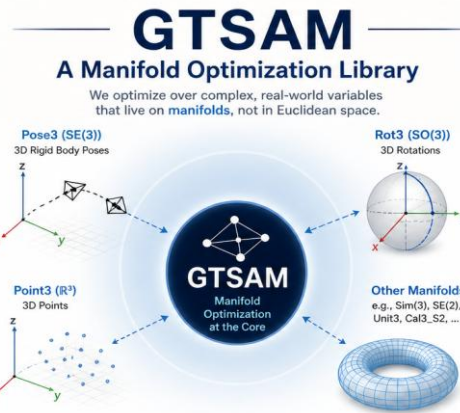
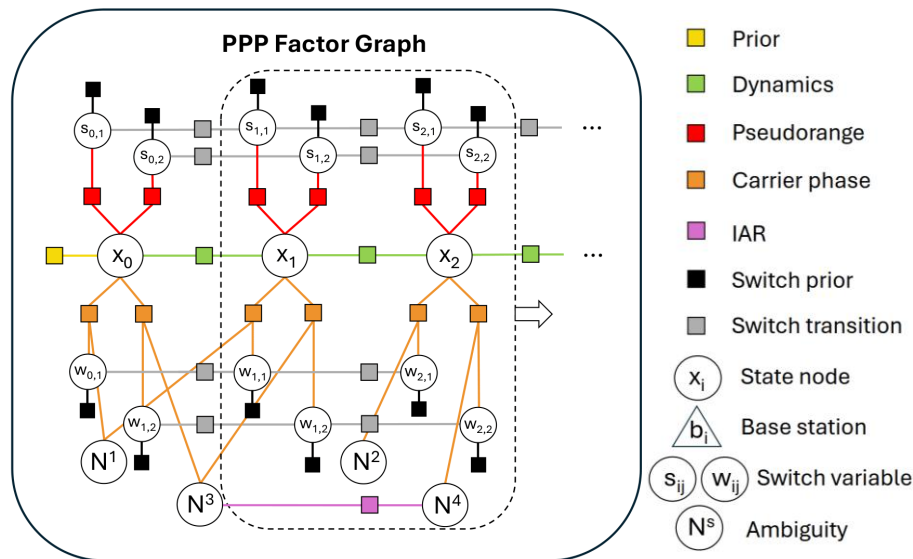
Design

Marginalization

- Follow marginalization-based carrier phase ambiguity propagation method from Wang et al. (2023) and Xin et al. (2023)
 - Four operations for the carrier phase ambiguities: adding, merging, separating, and sliding

Backend Implementation

- Use iSAM2 backend (Kaess et al., 2012) in GTSAM
 - converts the complex factor graph into a specialized data structure called a Bayes Tree



Adapted from
<https://borglab.github.io/gtsam/>

FGO Architecture

Warmup And Settings

Warmup at Startup

- Seeds iSAM2 (including ambiguity-key history and marginalization schedule) from a consistent solution before incremental updates

Empirical Settings

- We summarize a priori covariance and process noise covariance settings of different state components
- Ranges are practical tuning intervals used during validation

Parameter	Initial Noise σ	Process Noise σ	Initial Guess
Receiver position (Code, PPP, RTK)	10–100 m	1–40 m/ $\sqrt{s}$	SPP
Receiver clock bias (Code, PPP)	5–15 m	80–120 m/ $\sqrt{s}$	SPP
Receiver clock drift (PPP)	0.5–1.5 m/s	0.0001–0.0005 m/s/ $\sqrt{s}$	SPP
Receiver velocity (Code, PPP, RTK)	5–15 m/s	0.05–0.2 m/s/ $\sqrt{s}$	Doppler/DR
Tropospheric delay (Code, PPP)	0.1–0.5 m	10^{-5} – 10^{-7} m/ $\sqrt{s}$	Saastamoinen
Ionospheric delay (Code, PPP)	0.1–5 m	0.0005–0.0015 m/ $\sqrt{s}$	Broadcast/IONEX
Phase ambiguity (PPP, RTK)	50–150 cycles	0.5–50 cycles/ $\sqrt{s}$	LC of raw measurements
Switch variable (Code, PPP, RTK)	0.1–0.3	0.05–0.1	1.0

FGO Architecture

Motion Model And Optimization

Motion Model

- Use planar kinematic model propagating 10 Hz horizontal speed (from odometer) and yaw measurements (from IMU)

$$\mathbf{r}_{k-1,k}^{\text{dyn}} = \begin{bmatrix} p_{E,k} - p_{E,k-1} - \bar{V}_{E,k-1,k} \Delta t \\ p_{N,k} - p_{N,k-1} - \bar{V}_{N,k-1,k} \Delta t \\ p_{U,k} - p_{U,k-1} \end{bmatrix}$$

Optimization Modes

- Each approach balances accuracy, computational cost, memory usage, and correlation retention differently

Optimization Mode	Accuracy	Computational Cost / Latency	Memory Usage	Temporal Correlation	Typical Use Case
Batch Smoothing	★★★★★ (High)	High	High (unbounded)	Exact	Offline high-accuracy post-processing
Incremental	★★★★☆	Low-Medium	High (unbounded)	Exact in principle	Full-history real-time navigation for robots / UAVs / vehicles
Batch Fixed-Window	★★★★☆	Medium-High	Medium	Exact within window	Embedded GNSS/INS with moderate update rates
Incremental Fixed-Window	★★★☆☆	Medium	Low-Medium	Exact within window in principle	Embedded GNSS/INS with high update rates

FGO-Nav Software Development

FGO-Nav Software Development

Code Architecture

MSP4

- Top-level orchestrator (engine dispatch, config load/check)

FGO

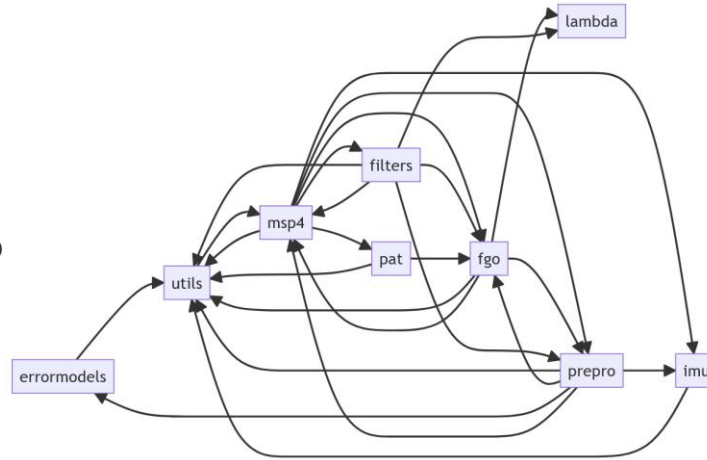
- Factor-graph core (factor construction, sliding-window management)

Filters

- EKF filter backends and adaptive measurement-noise scaling

Preprocessing

- Quality checks, cycle slips, rough positioning, DD



IMU

- IMU/odometer integration for DR velocity propagation

LAMBDA

- Integer ambiguity-resolution algorithms

PAT

- Post-analysis and diagnostics

Utilities

- RINEX, atmosphere, ephemeris I/O etc.

Error Models

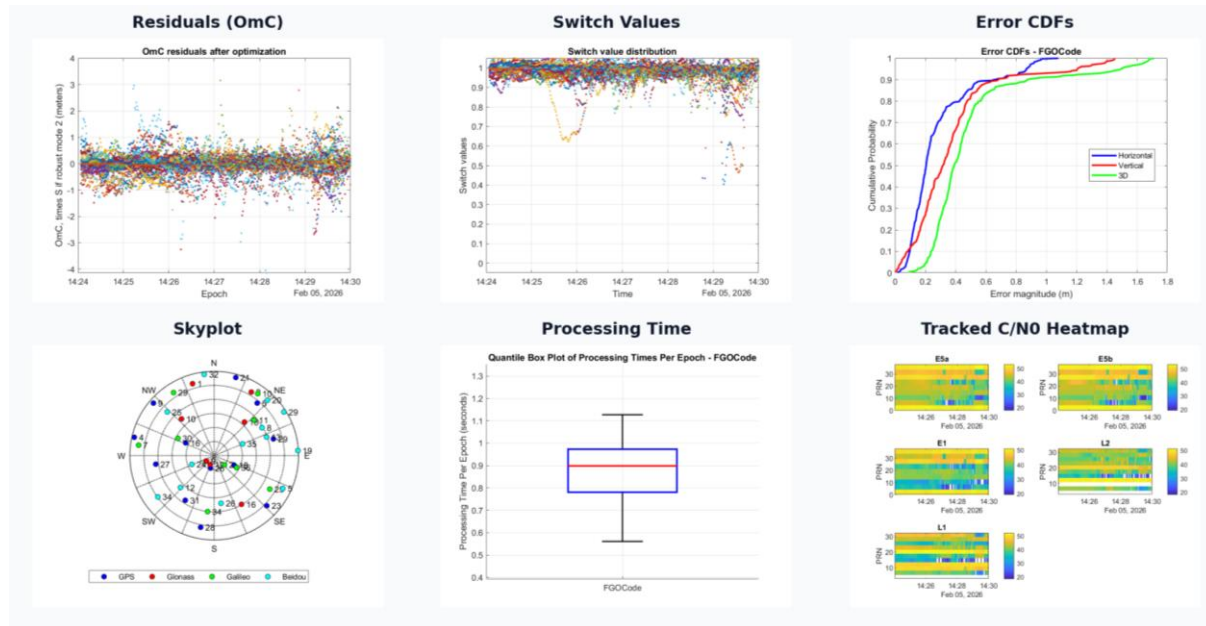
- GNSS correction models

FGO-Nav Software Development

Software Output

MSP4 Outputs Include:

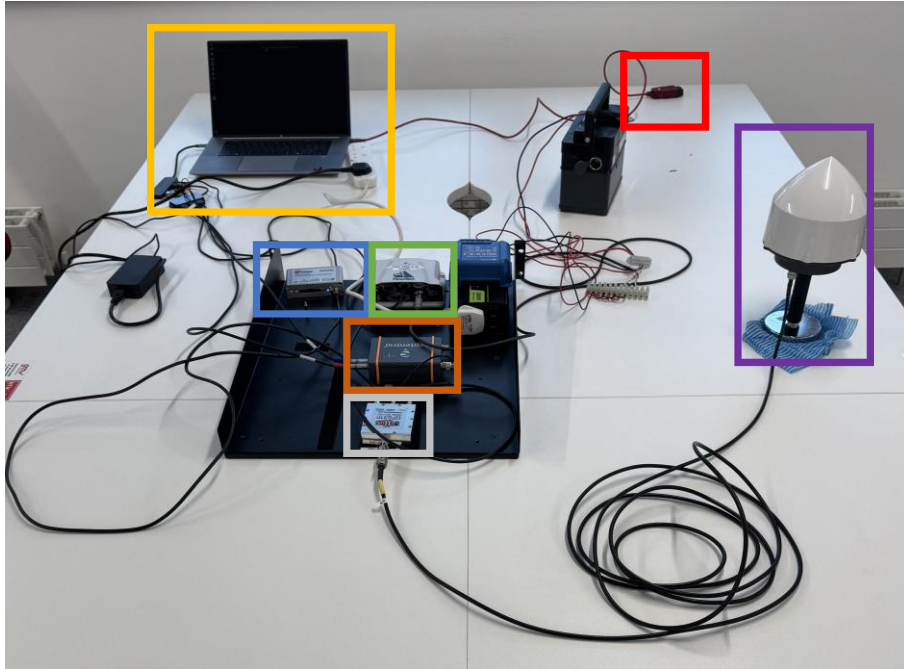
- Position and velocity statistics
- Residual diagnostics
- Trajectory diagnostics, including KML export
- Satellite/observation availability plots
- Signal-quality heatmaps
- FGO state diagnostics, including switch values
- Processing time diagnostics



Experimental Validation Campaign

Data Collection Campaign

Testbed Components



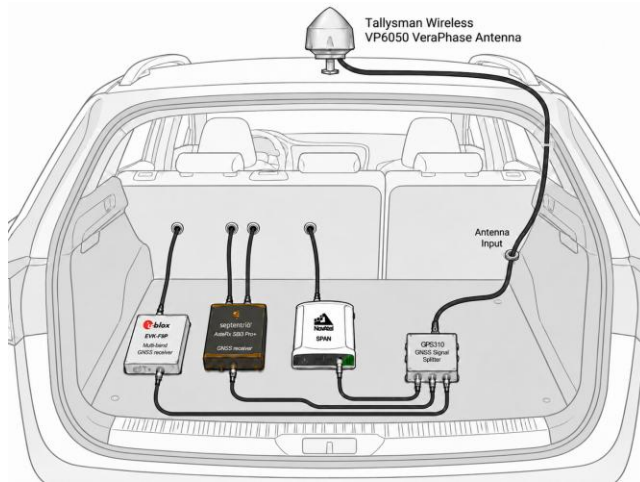
Placement	Element	Colour
Vehicle Roof	GNSS Antenna: VP6050 Antenna	Purple
Vehicle Boot	Ground Truth receiver: Novatel PwrPak7D-E2	Light Green
	High-end receiver (+IMU): AsteRx SBI3 Pro+	Orange
	Low-cost receiver: u-blox EVK-F9P	Blue
	Splitter: GPS310	Light Grey
Rear Passenger Seats	Processing Unit: HP ZBook Studio	Yellow
Front Passenger Seat	Odometer: OBDLink SX USB	Red

Data Collection Campaign

Testbed Setup

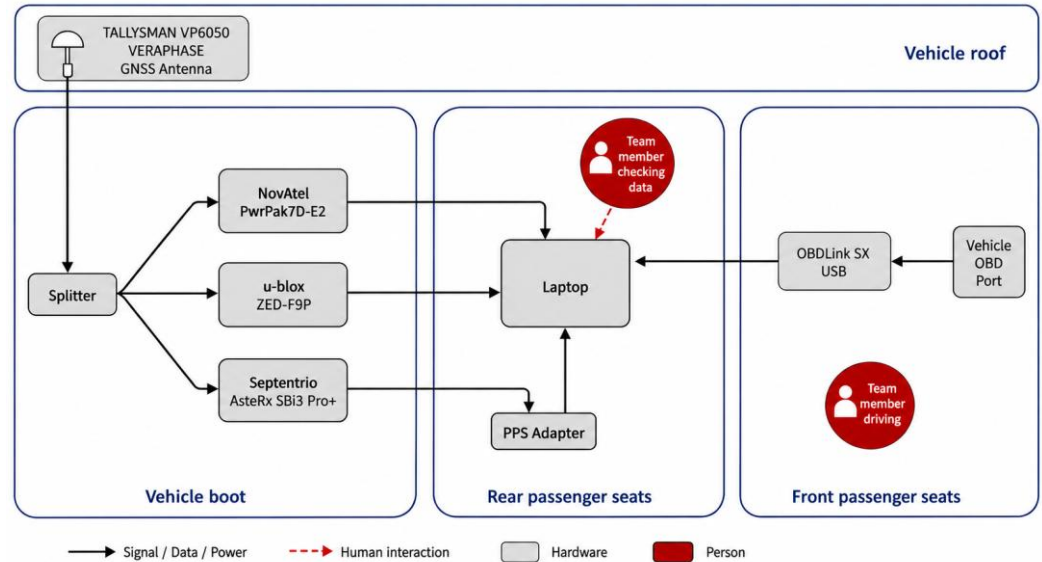
In-vehicle Hardware Installation:

- Vehicle: Vauxhall Mokka



Hardware testbed model:

- Highlights signal flows from sensors to the processing unit



Data Collection Campaign

Synchronization

Septentrio AsteRx SBi3 Pro+ Output Used

- **PPS signal**: precise one-pulse-per-second timing reference
- **NMEA message**: absolute GNSS time information used to identify each PPS epoch



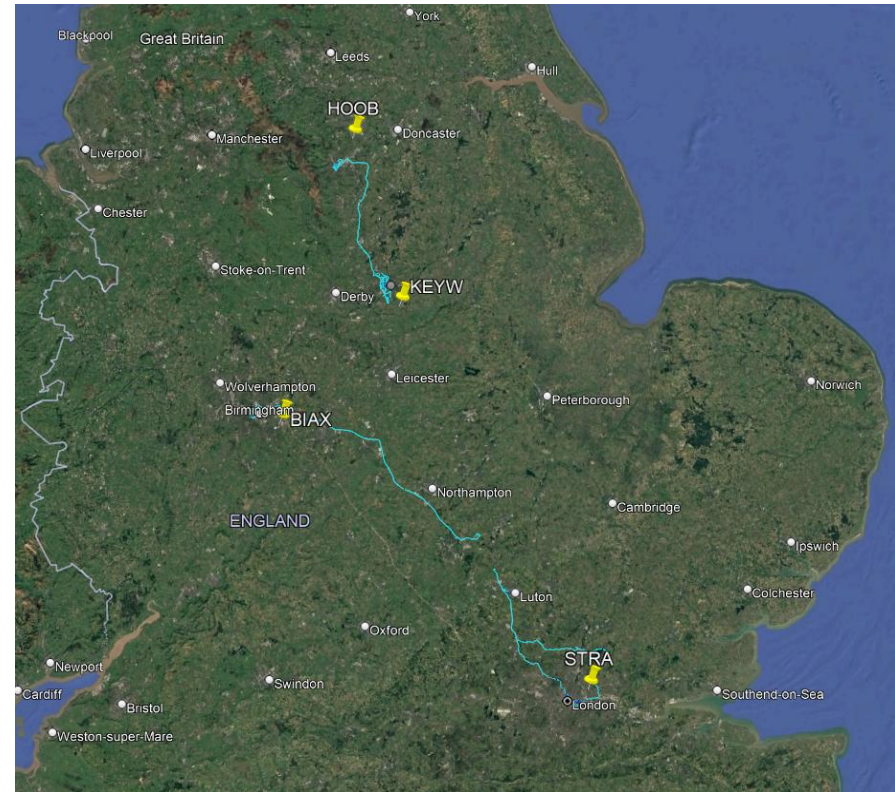
Synchronization Steps

- FT232H converts the PPS electrical signal into a USB-readable input → exposed as `/dev/pps*`
- gpsd provides the NMEA time source to **Chrony** software
- Chrony adjusts the laptop clock using the PPS signal and NMEA time reference
- Synchronization status verification:
 - *ppstest* to confirm PPS pulses
 - *chronyc sources* to verify active timing sources
 - *chronyc tracking* to check clock stability
- Note: The ms-level odometer timestamping error (compared to GPS time) is sufficient for coarse synchronization

Data Collection Campaign

Experimental Campaign

ID	Scope	Time [h]	Environment characteristics
LDN	Cranfield–London–Cranfield corridor	~7.5	Long urban campaign with extended central-London coverage; includes repeated Rotherhithe tunnel traversals (GNSS-denied).
BHM	Cranfield–Birmingham corridor	~4.5	Mixed open-sky motorway and central-Birmingham urban driving; includes localized degradation in city intervals, especially during low-dynamic periods.
NTH	Nottingham campaign	~6.1	Moderately challenging (sub -)urban environment with non-open-sky initialization segments, static phases, and changing visibility conditions.
SFL	Sheffield campaign	~2.5	Alternating urban and inter-urban segments, including dense Sheffield city-centre sections with tall-building masking and restricted sky visibility.
MID	Midlands transfer route (Sheffield–Nottingham)	~4.5	Long mixed-environment trajectory with degraded visibility at the beginning, followed by urban and inter-urban/highway segments.



Data Collection Campaign

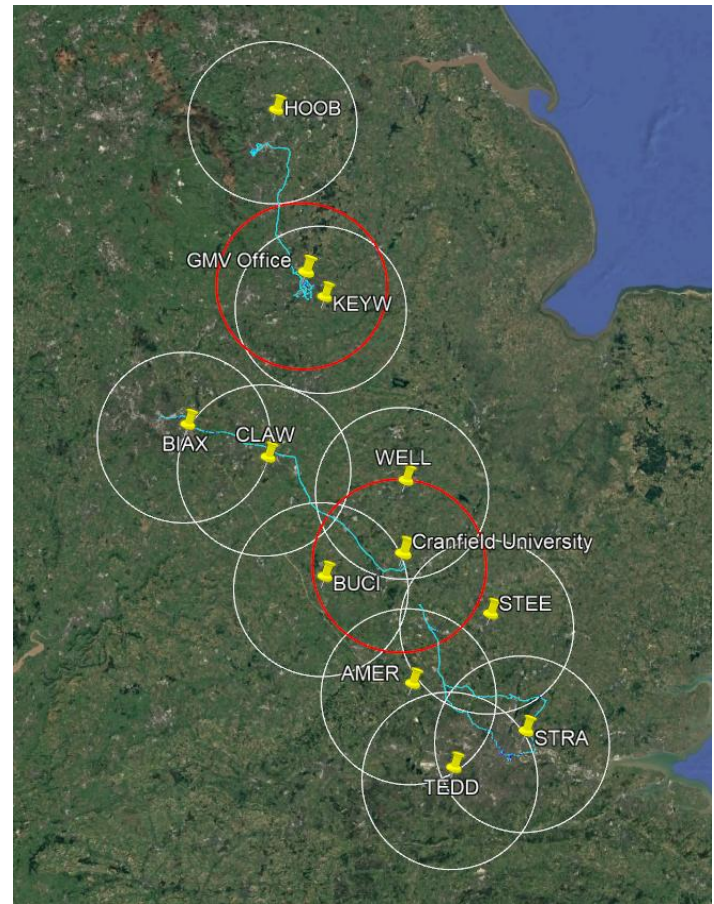
Datasets

Input Data for MSP4:

- Raw GNSS observations from Septentrio and u-blox receivers: GPS L1/L2 and Galileo E1/E5b
- GNSS RINEX navigation file
- PPP: Precise products for ionosphere, clocks, orbits, code and phase biases (from CODE)
- DR data (IMU + Odometer) at 10 Hz
- Ground-truth (from Inertial Explorer v8.6, using network-RTK corrections from OSNET, GMV, and Cranfield University CORS)
- RTK: GNSS CORS data of nearby OS station (1 Hz)

CORS Data Used for RTK MSP4 Processing

- STRA for London (February 5th)
- BIAx for Birmingham (February 6th)
- KEYW for Nottingham (February 7th and 8th)
- HOOB for Sheffield (February 10th and 11th)



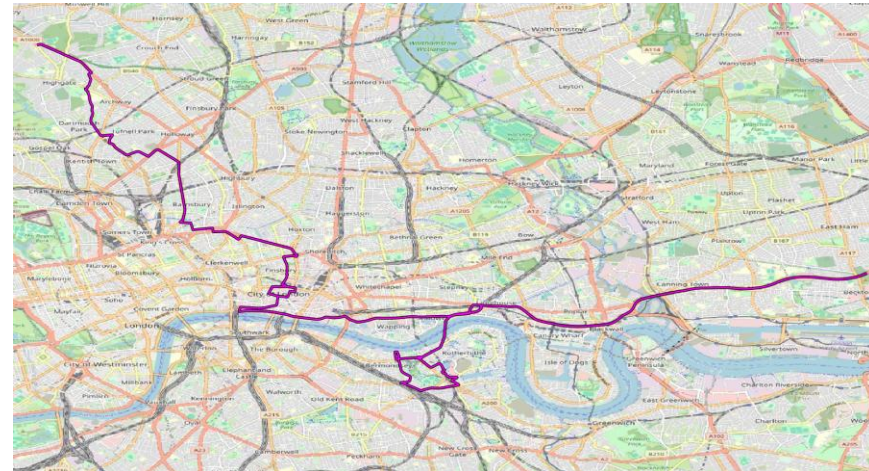
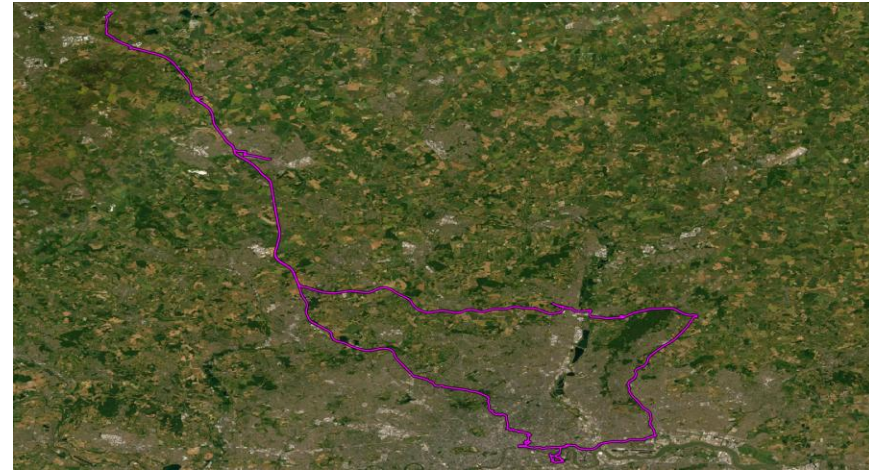
Test-campaign vehicle trajectories followed, along with OSNET, GMV, and Cranfield University reference stations used to obtain the ground truth. Each reference station is highlighted with a circle of 30 km.

Performance Assessment

London Performance Assessment

Detailed Description of Route

- About **7.5 hours** GNSS data (Septentrio and u-blox)
- Test Route: Cranfield-London-Cranfield
- Includes open sky, suburban, urban and a **1.5 km tunnel** (Rotherhithe)
- About 2.5 hours subset in deep urban London



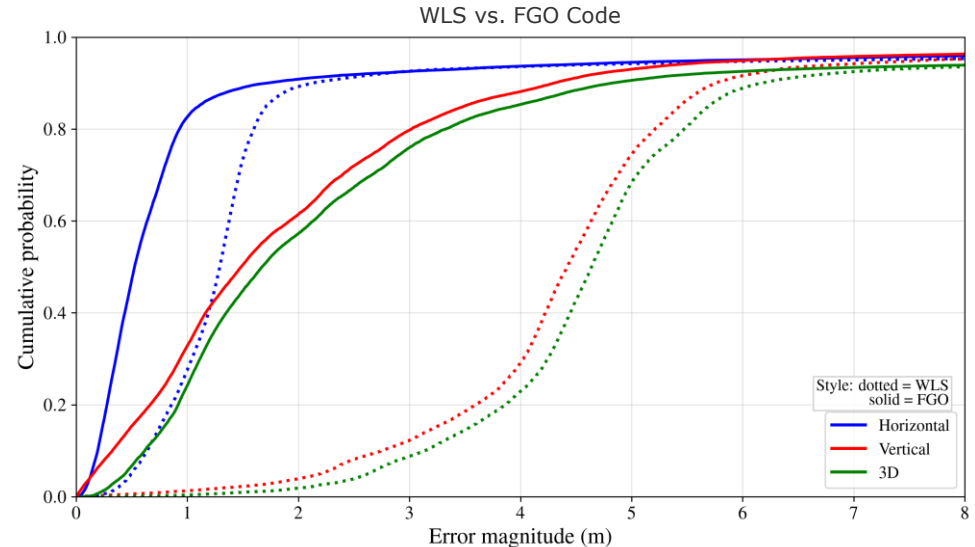
London Code Results

Septentrio

Septentrio	Horizontal Position Error (m)					3D Position Error (m)				
	50%	68%	95%	99%	RMS	50%	68%	95%	99%	RMS
WLS	1.29	1.44	6.97	32.54	7.03	4.64	4.99	12.17	46.23	16.81
FGO-Code	0.53	0.75	5.90	27.99	4.59	1.67	2.54	10.64	32.91	6.24

Discussion

- **RMS change:** 34.7% (H), 62.9% (3D)
- Consistent performance **gains** observed across **all metrics**
- **Lower percentile improvements** (50th–68th) likely driven by the explicit estimation of tropospheric and ionospheric states
- **Higher percentile improvements** (95th–99th) indicate that FGO effectively mitigates large positioning errors

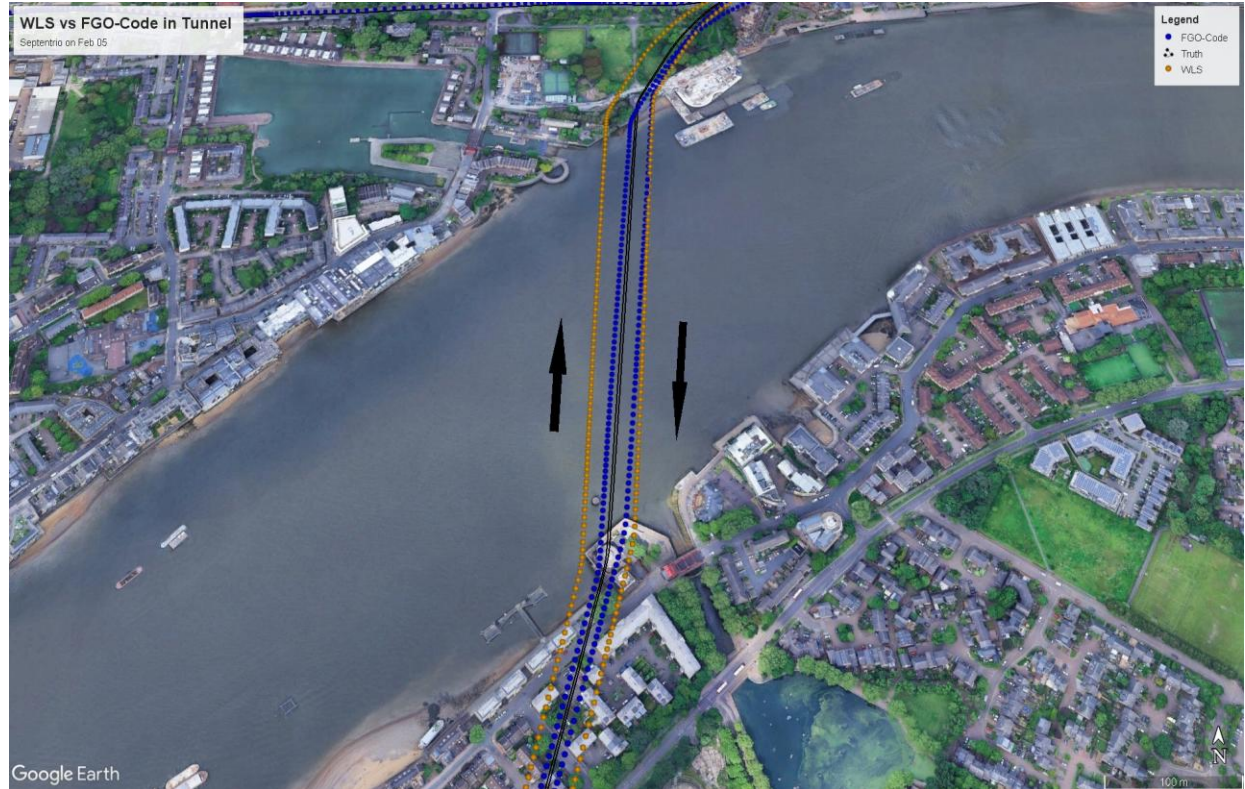


London Code Results, Tunnel Example

Septentrio

Observations

- In the two tunnel crossings (**1.5 km each**), GNSS signals are unavailable, hence requiring dead reckoning (DR)
- WLS exhibits **significant error growth** during both directions of travel due to DR drift
- FGO **mitigates error growth** through improved integration and smoothing, enhancing performance during GNSS outages



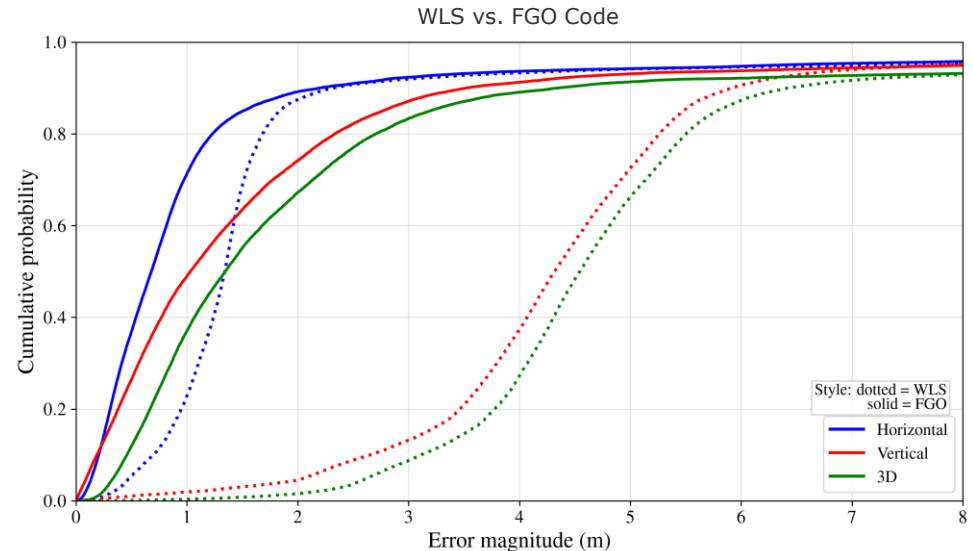
London Code Results

u-blox

u-blox	Horizontal Position Error (m)					3D Position Error (m)				
	50%	68%	95%	99%	RMS	50%	68%	95%	99%	RMS
WLS	1.34	1.49	7.38	43.63	11.80	4.55	5.06	13.89	64.69	30.70
FGO-Code	0.68	0.94	6.36	26.69	4.51	1.35	2.04	12.60	39.98	7.80

Discussion

- **RMS improvement:** 61.8% (H), 74.6% (3D)
- **Consistent improvements** observed across all percentiles and RMS
- **Noticeable reductions at the higher percentiles** (99th) confirm mitigation of large error excursions
- FGO delivers both improved accuracy and robustness, with **larger relative gains** than in the Septentrio case

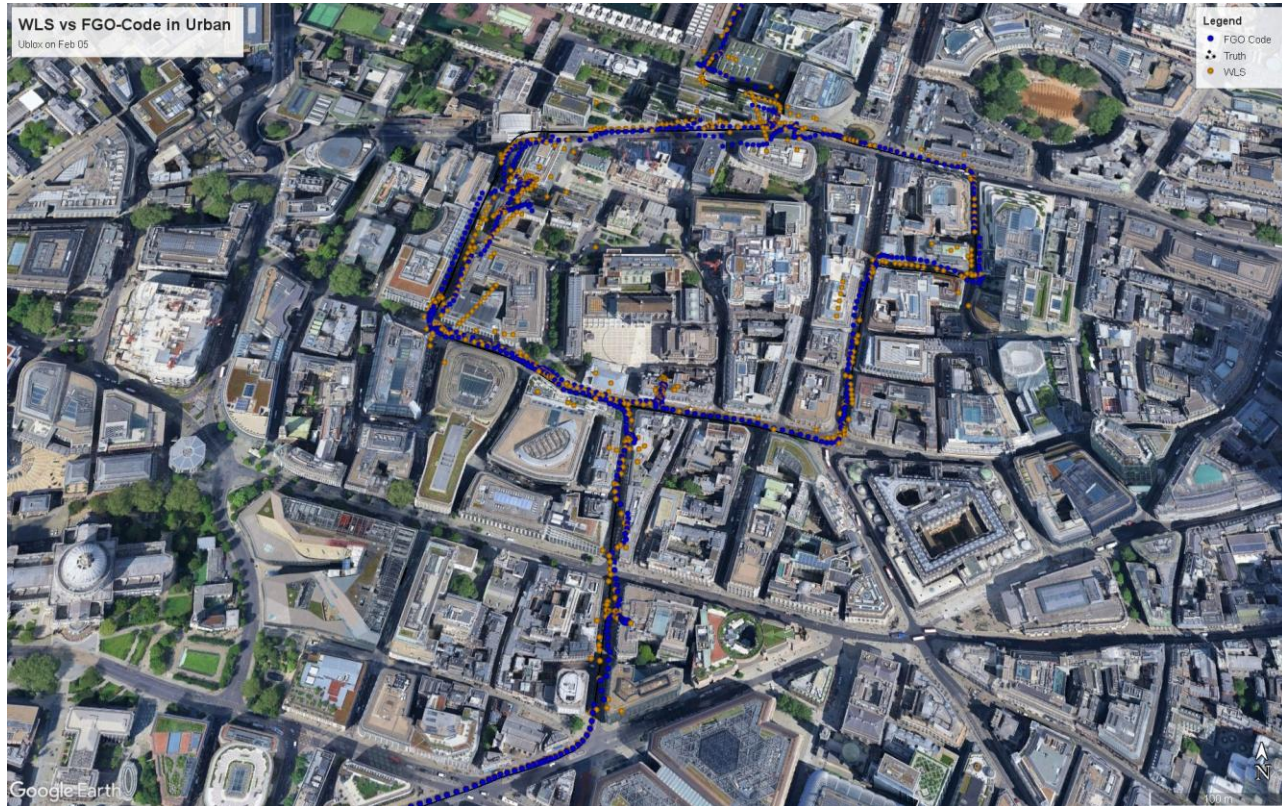


London Code Results, Urban Example

u-blox

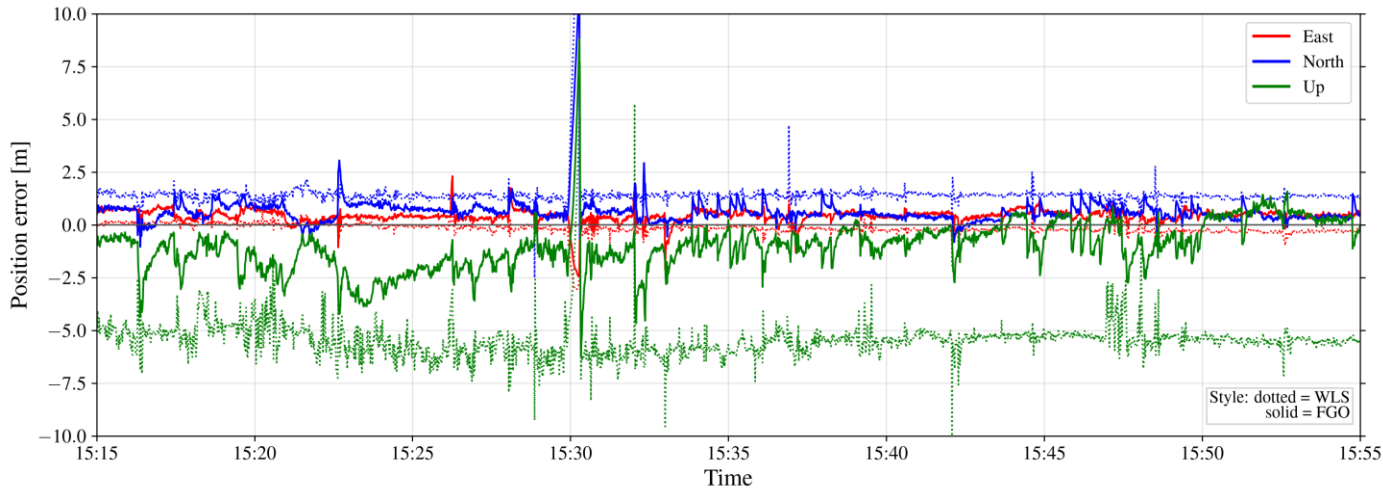
Observations

- FGO-Code delivers a **smoother and more accurate** trajectory in these challenging conditions
- The improvement is driven by **robust handling of degraded and inconsistent GNSS** measurements
- FGO substantially **enhances both accuracy and robustness** in dense urban environments



London Code Results, Motorway Example

u-blox

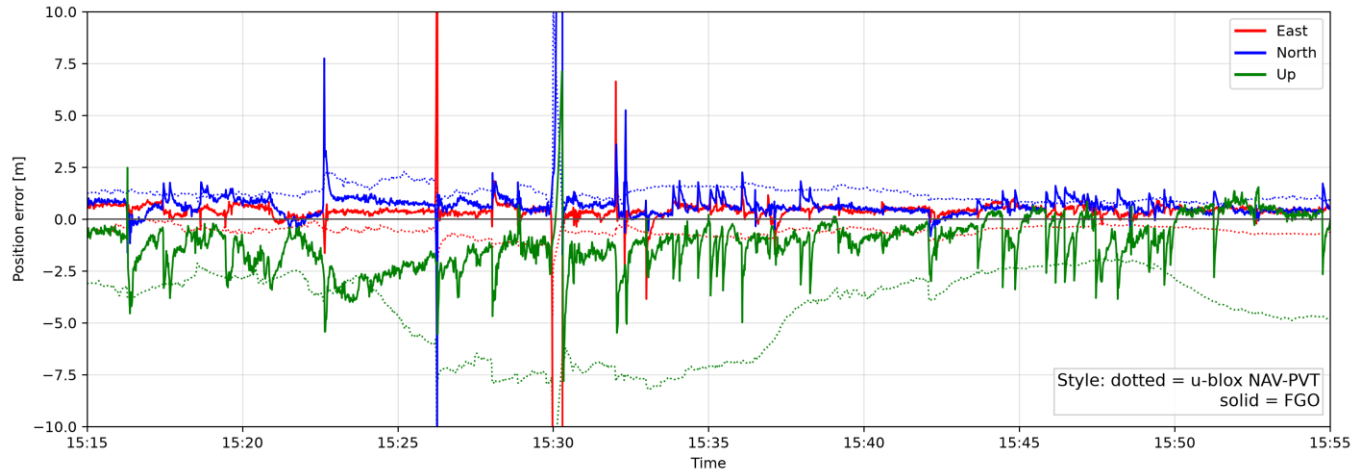


Observations

- We show a motorway section of the LDN scenario, using u-blox GNSS and DR data
- FGO shows a **lower** Up and North bias and **fewer** high-amplitude **peaks**
- Around 15:30, there is a short motorway tunnel (25 s), where both engines behave similarly

Comparing Against u-blox NAV-PVT

FGO GNSS-only Code vs u-blox NAV-PVT



Test Results

- While NAV-PVT results are smoother, FGO **biases are lower** in all ENU directions
- FGO results are **continuous** through the motorway tunnel where NAV-PVT shows a gap
- Caveat:** u-blox NAV-PVT was obtained using Portable/Default dynamics model and 1 Hz data

u-blox	Horizontal Position Error (m)					3D Position Error (m)				
	50%	68%	95%	99%	RMS	50%	68%	95%	99%	RMS
NAV-PVT	1.35	1.51	2.00	2.26	1.44	3.97	4.96	7.87	8.23	5.02
FGO-Code	0.77	0.93	1.45	5.30	20.92	1.37	1.82	3.59	6.52	20.98

Note: FGO continuity harms higher percentiles!

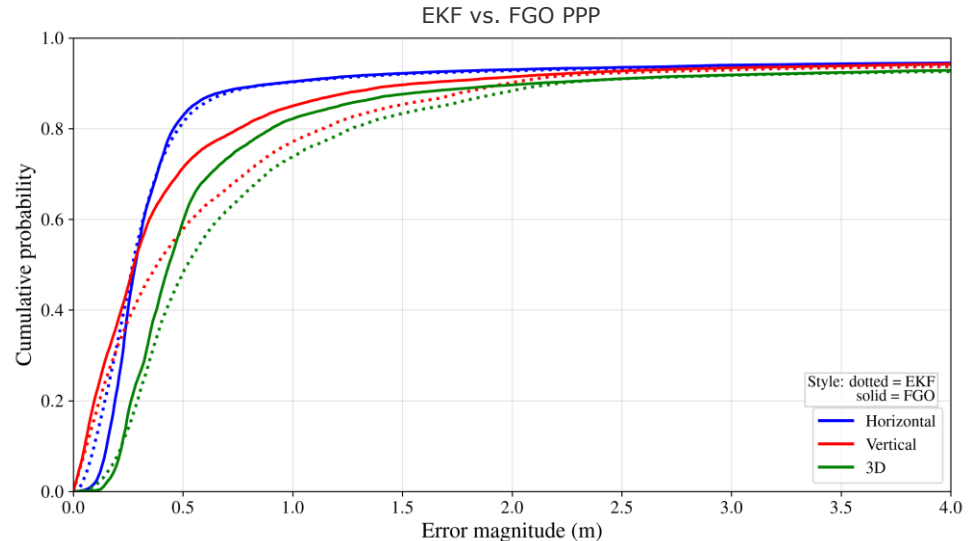
London PPP Results

Septentrio

Septentrio	Horizontal Position Error (m)					3D Position Error (m)				
	50%	68%	95%	99%	RMS	50%	68%	95%	99%	RMS
EKF-PPP	0.27	0.37	6.35	33.61	5.86	0.52	0.83	12.06	69.33	11.76
FGO-PPP	0.28	0.37	5.50	27.53	4.39	0.44	0.58	10.17	32.97	5.98

Discussion

- **RMS change:** 25.1% (H), 49.2% (3D)
- Limited differences are observed at the **lower percentiles**, indicating similar central accuracy
- Clear **improvements** at the **higher percentiles** demonstrate reduced large-error occurrences
- The benefit is **more pronounced in 3D**, reflecting improved vertical and overall solution stability

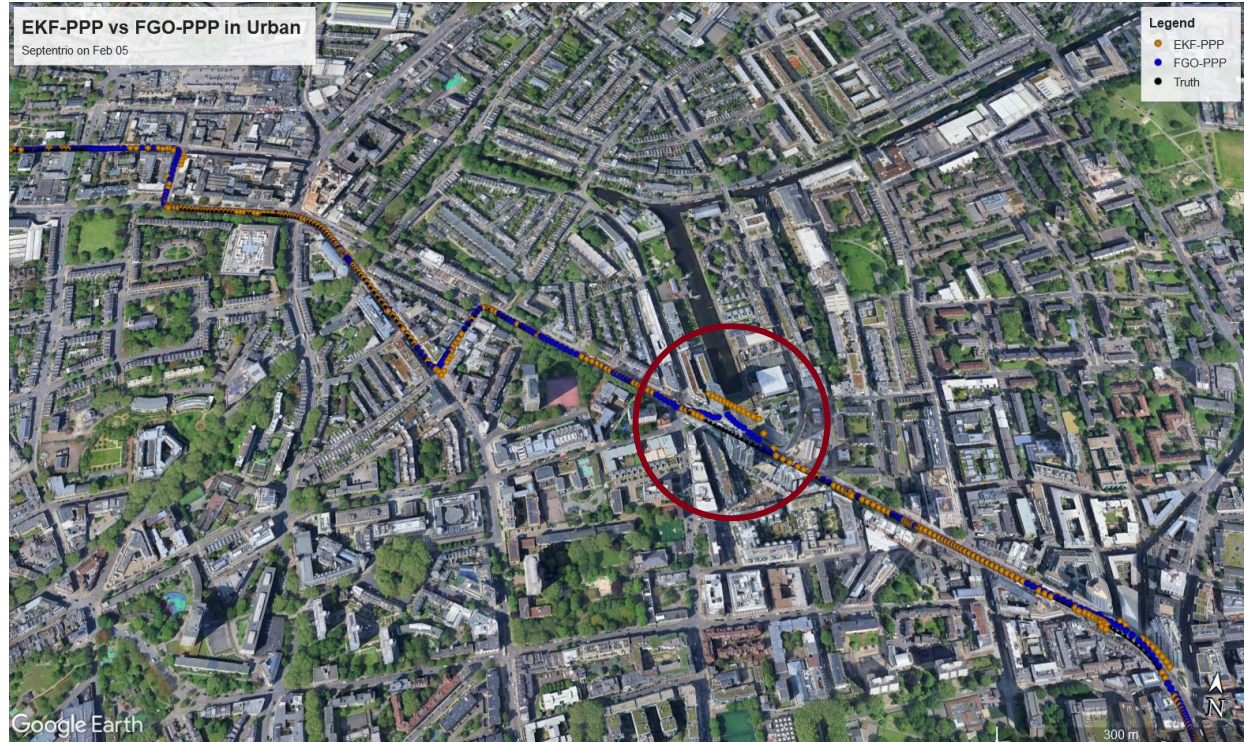


London PPP Results, Sub-Urban Example

Septentrio

Observations

- GNSS measurements are slightly **degraded** by partial signal blockage from surrounding vegetation and buildings
- EKF-PPP shows **noticeable trajectory drift** in the highlighted area
- FGO-PPP maintains a **more stable trajectory** with **reduced drift**, benefiting from sensor fusion and temporal correlations



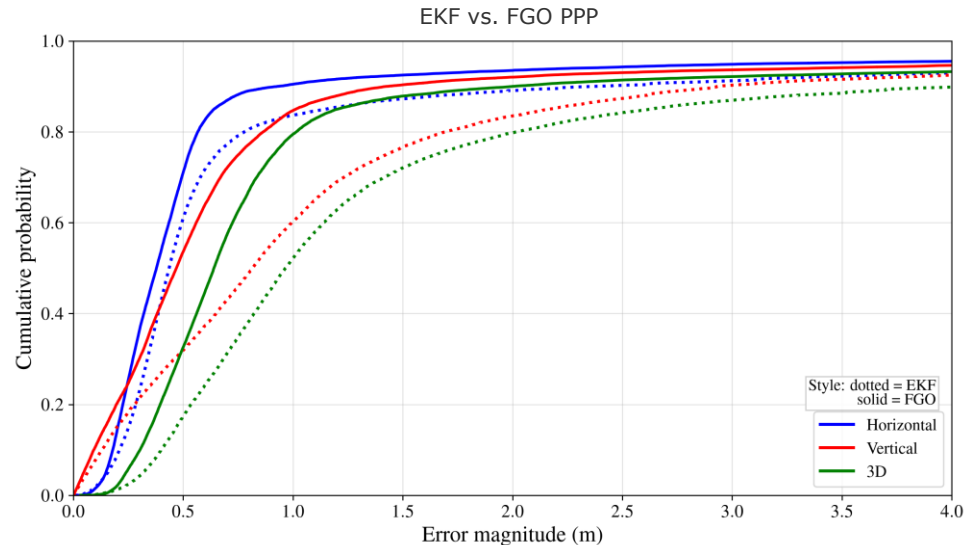
London PPP Results

u-blox

u-blox	Horizontal Position Error (m)					3D Position Error (m)				
	50%	68%	95%	99%	RMS	50%	68%	95%	99%	RMS
EKF-PPP	0.44	0.56	7.42	32.92	5.75	0.97	1.34	11.73	43.06	10.65
FGO-PPP	0.38	0.48	3.16	20.46	3.69	0.64	0.81	6.29	32.94	5.57

Discussion

- **RMS change:** 35.8% (H), 47.7% (3D)
- **Consistent improvements** are observed across both central and extreme percentiles
 - Enhanced central accuracy
 - Improved robustness against large errors
- The benefit is **more uniform across percentiles** compared to the Septentrio case



London PPP Results, Urban Example

u-blox

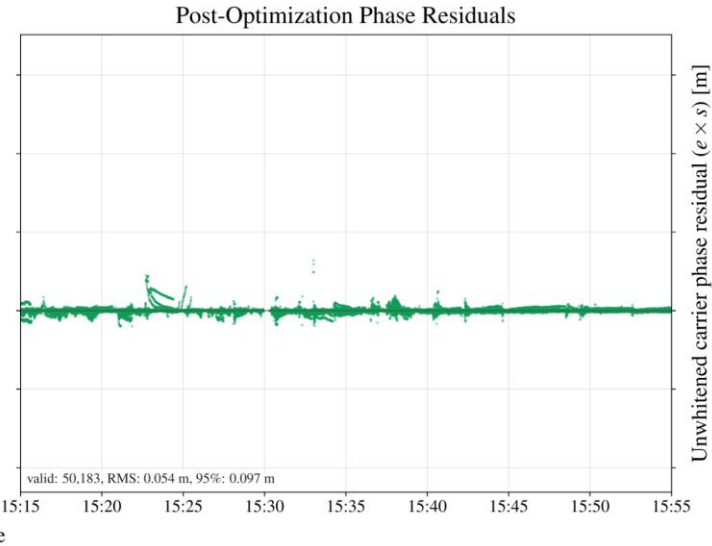
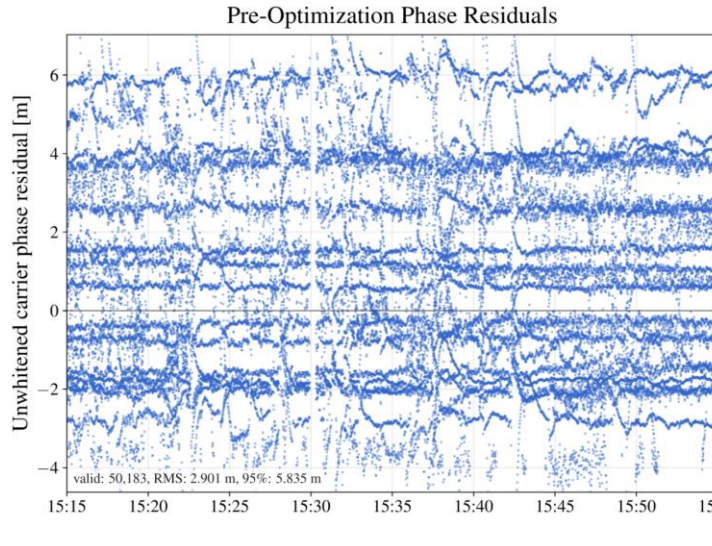
Observations

- **GNSS** measurements are significantly **degraded** across the whole trajectory shown in the City of London
- **FGO follows** the ground truth better than EKF, including street corners



London Results: Phase Residuals

u-blox



Before Optimization (After Preprocessing)

- Residual RMS: 2.901 m
- 95th percentile: 5.835 m

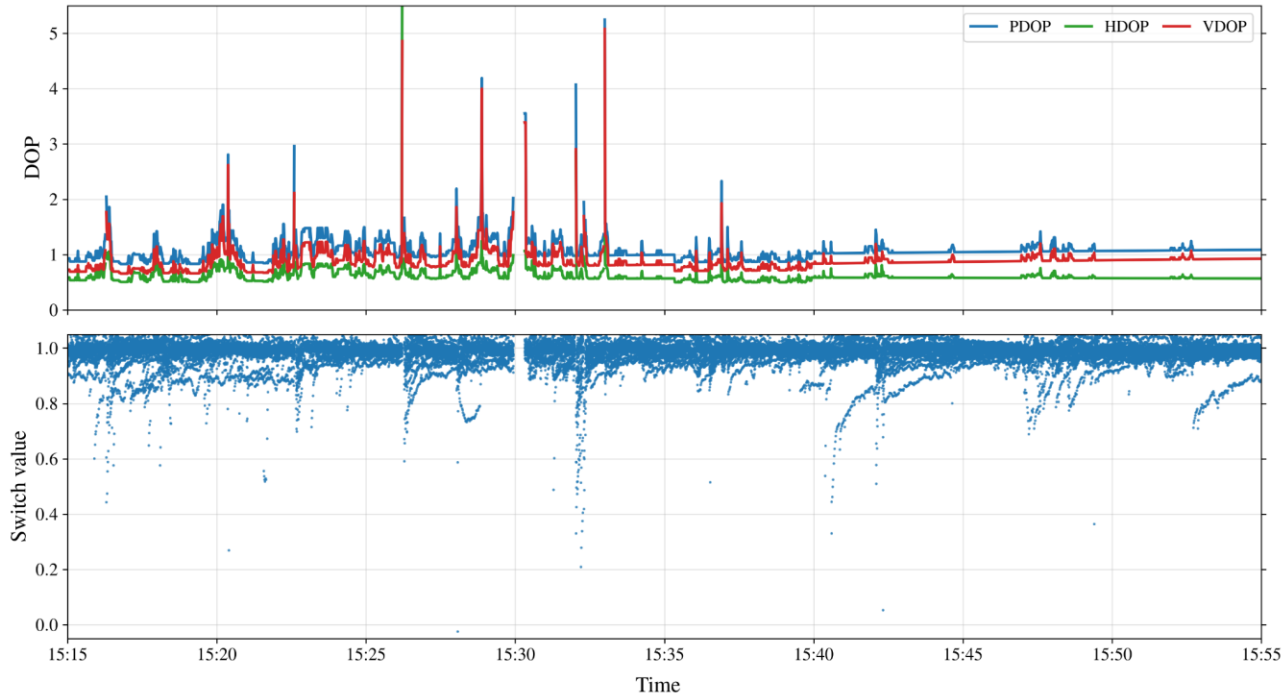
After Graph Optimization

- Residual RMS: 0.054 m (97.4% reduction)
- 95th percentile: 0.097 m (97.7% reduction)

Contraction reflects both improved state consistency from optimization and deliberate **suppression of outliers** via switches.

London Results: Robustness Mechanism

u-blox



Switch Value Mechanism

- Increased DOP suggests degraded GNSS measurement quality (multipath and NLOS)
- Lower switch value indicates less reliable GNSS observation
- Observe: Increased DOP → reduced switch values
- Hence: graph can **adaptively down-weight** unreliable GNSS observations

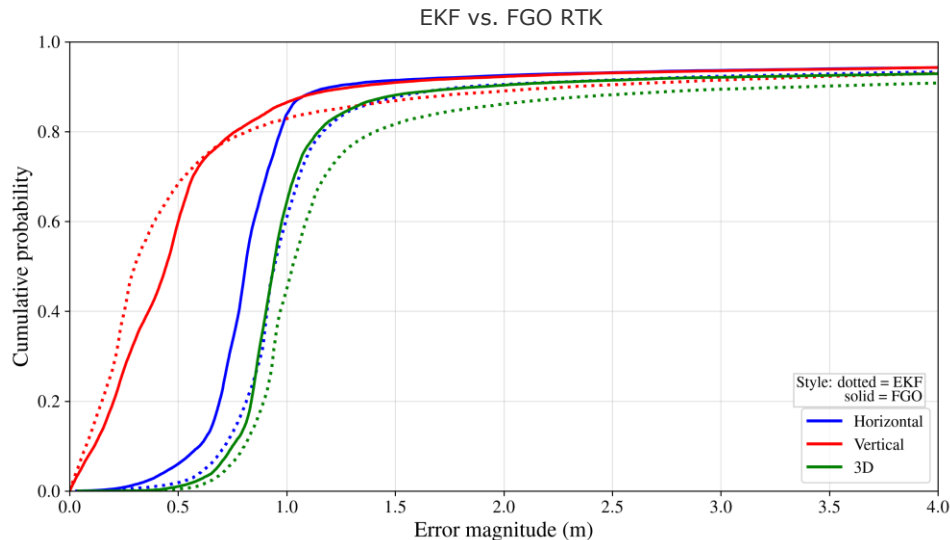
London RTK Results

Septentrio

Septentrio	Horizontal Position Error (m)					3D Position Error (m)				
	50%	68%	95%	99%	RMS	50%	68%	95%	99%	RMS
EKF-RTK	0.94	1.05	6.38	34.76	6.19	1.03	1.15	11.71	54.00	11.27
FGO-RTK	0.81	0.89	5.94	27.99	4.48	0.94	1.02	11.13	34.05	7.05

Discussion

- **RMS change:** 27.6% (H), 37.4% (3D)
- **Moderate improvements** are observed across 50th to 95th percentiles
- More **noticeable reductions** at the **99th** percentile highlight improved robustness against large errors
- Primary benefit of the FGO-RTK solution lies in enhancing **robustness and stability**

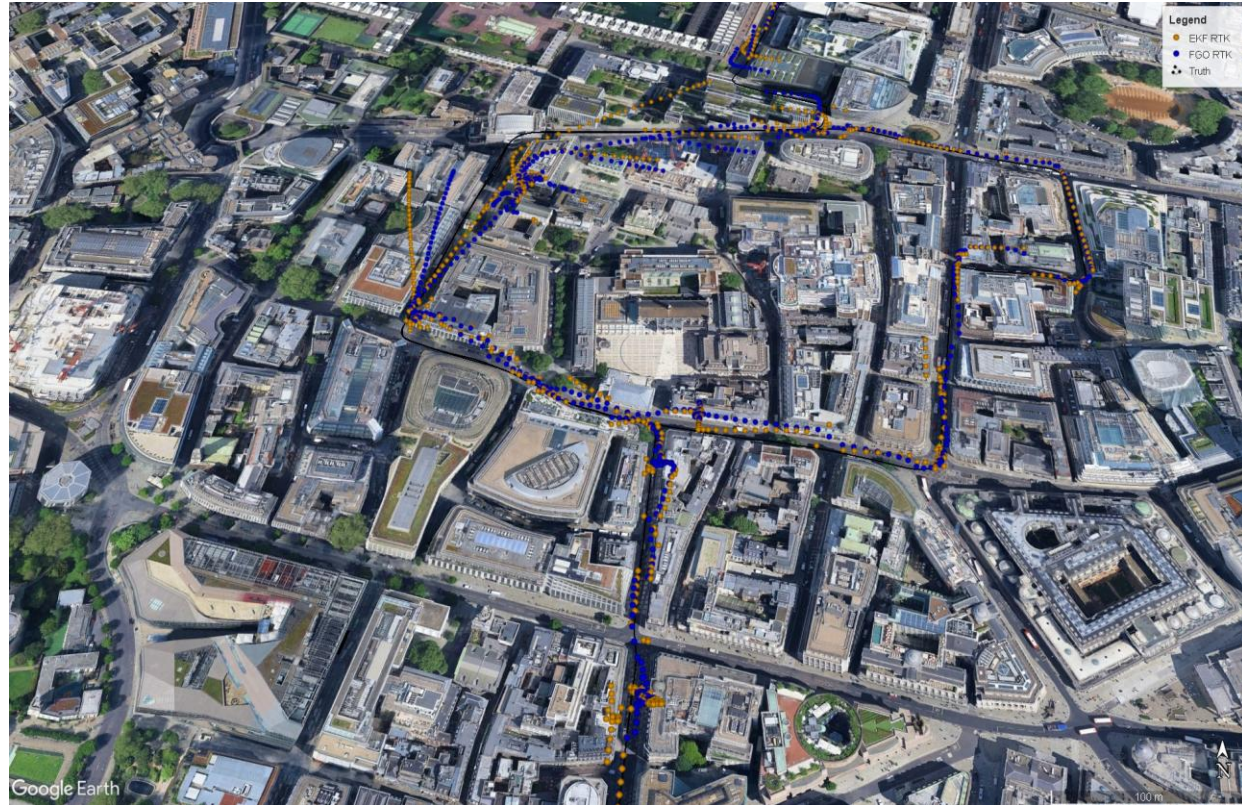


London RTK Results, Urban Example

Septentrio

Observations

- RTK performance is significantly challenged by **poor satellite visibility** and **outliers**
- EKF-RTK shows **large** positioning **errors** and frequent **discontinuities**
- FGO-RTK significantly improves trajectory **smoothness** and positioning **consistency**, even under harsh conditions

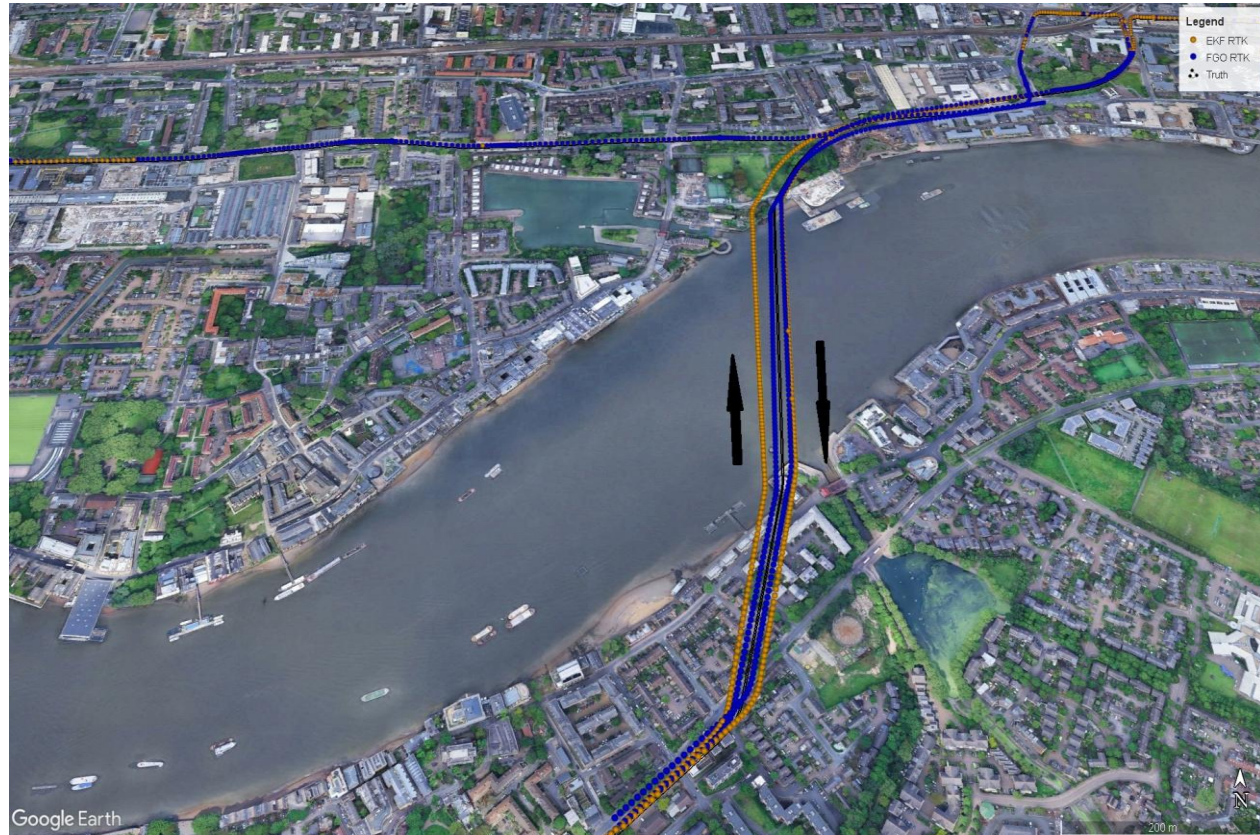


London RTK Results, Tunnel Example

Septentrio

Observations

- GNSS signals are unavailable, and both solutions rely on **dead reckoning (DR) propagation**
- EKF-RTK exhibits **faster error accumulation** due to drift during GNSS outages
- FGO-RTK **mitigates drift** through multi-epoch optimisation and better motion constraint handling, resulting in **improved trajectory continuity**



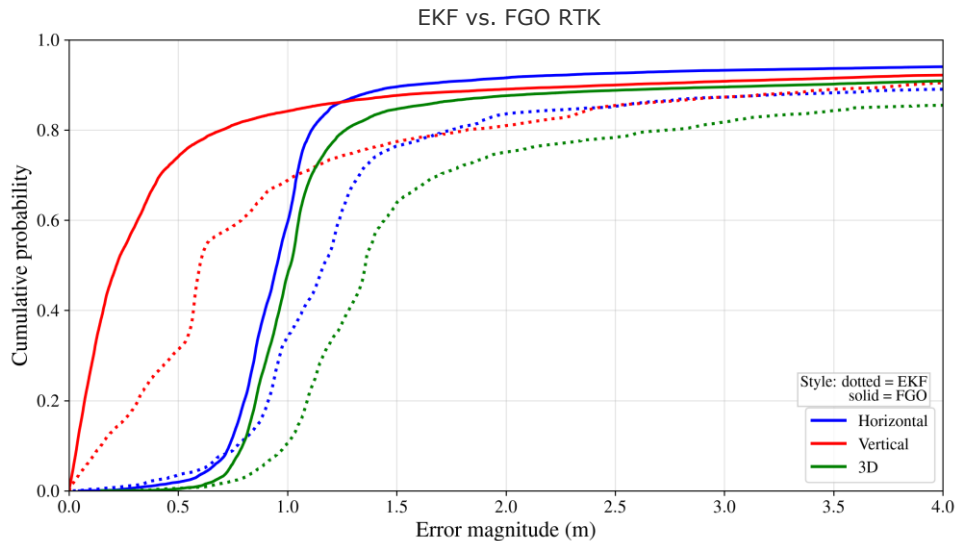
London RTK Results

u-blox

u-blox	Horizontal Position Error (m)					3D Position Error (m)				
	50%	68%	95%	99%	RMS	50%	68%	95%	99%	RMS
EKF-RTK	1.16	1.30	20.90	59.89	11.64	1.36	1.62	27.90	79.70	15.21
FGO-RTK	0.95	1.04	6.08	26.20	4.50	1.01	1.09	13.93	39.56	8.04

Discussion

- **RMS improvement:** 61.3% (H), 47.1% (3D)
- **Consistent improvements** are observed across all percentiles and RMS
 - Moderate gains in central accuracy
 - Effective mitigation of degraded GNSS
- FGO-RTK **improves robustness**, with **additional gains** in central accuracy compared to the Septentrio case

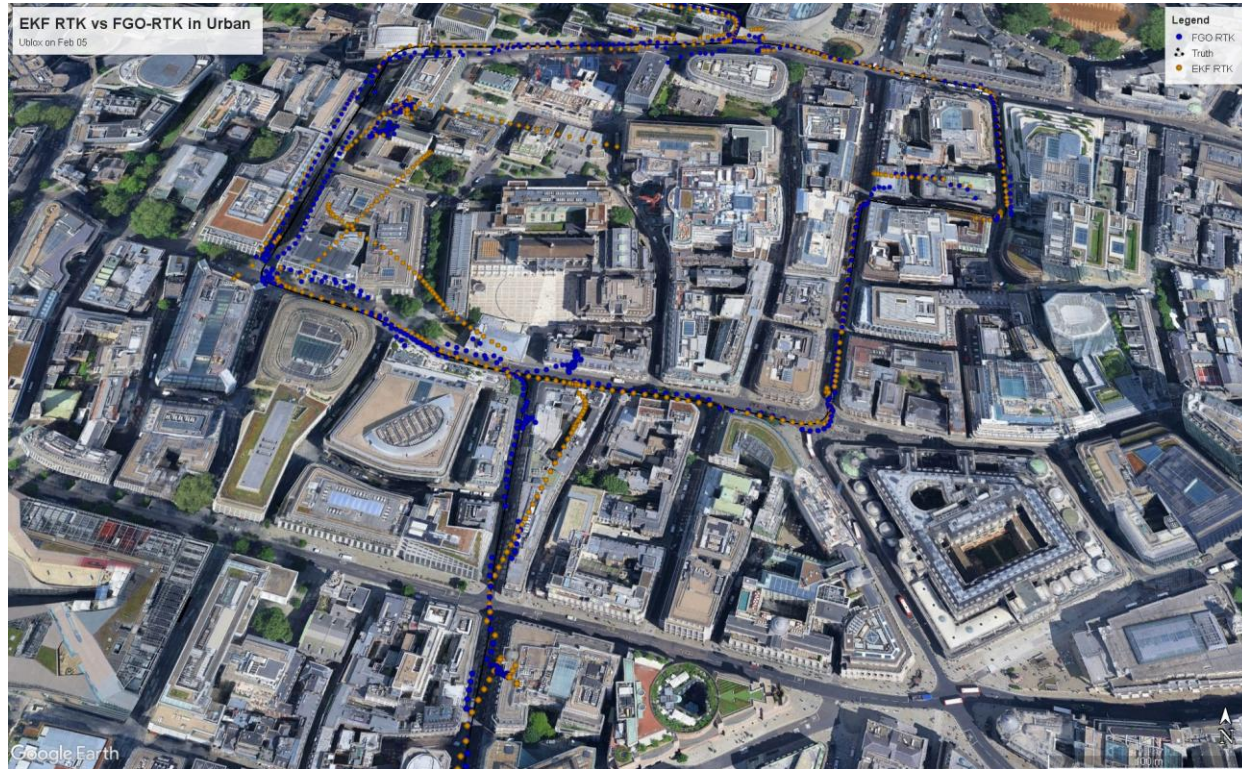


London RTK Results, Urban Example

u-blox

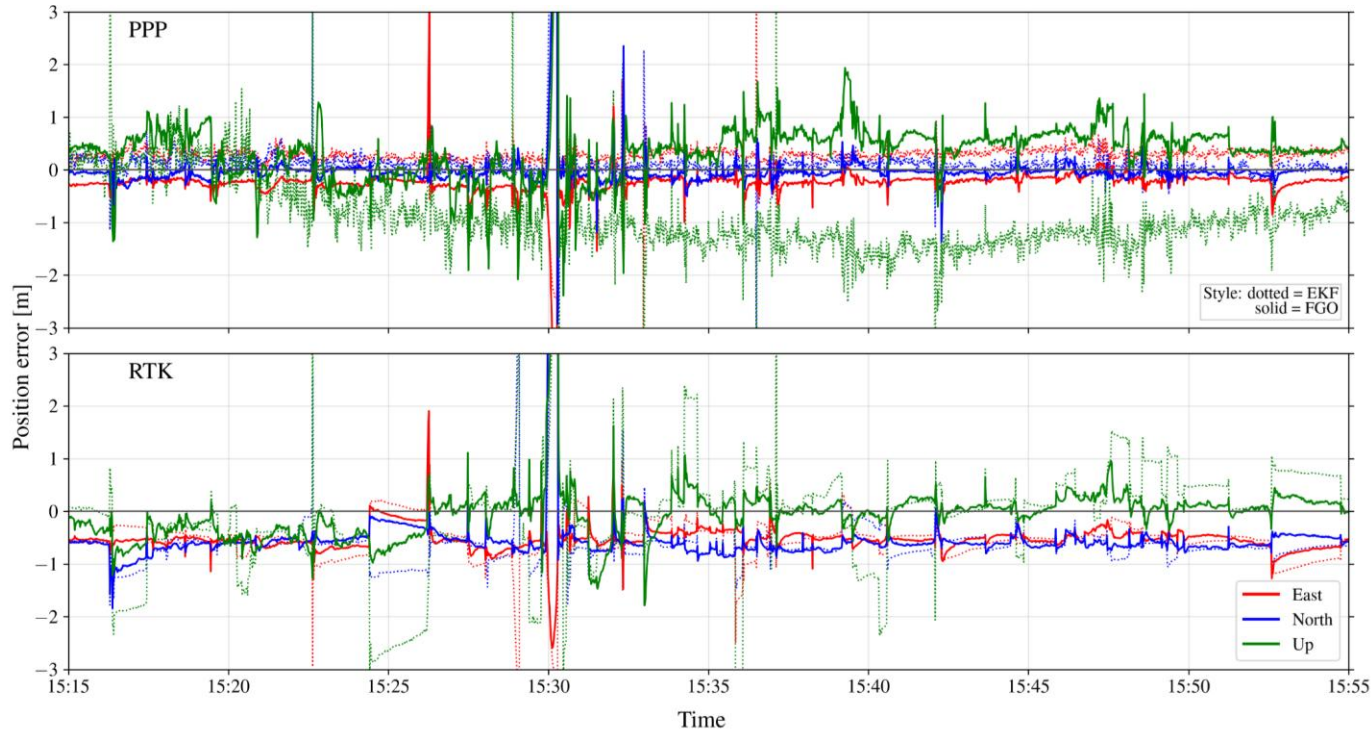
Observations

- We show the same aerial view as in the Septentrio case
- EKF-RTK again shows **large** positioning **errors** and frequent **discontinuities**
- FGO-RTK substantially improves **robustness** and maintains better **solution continuity**
- FGO error reductions are **significant** across various parts of this urban section (see Figure)



London PPP & RTK Results

Summary



Observations

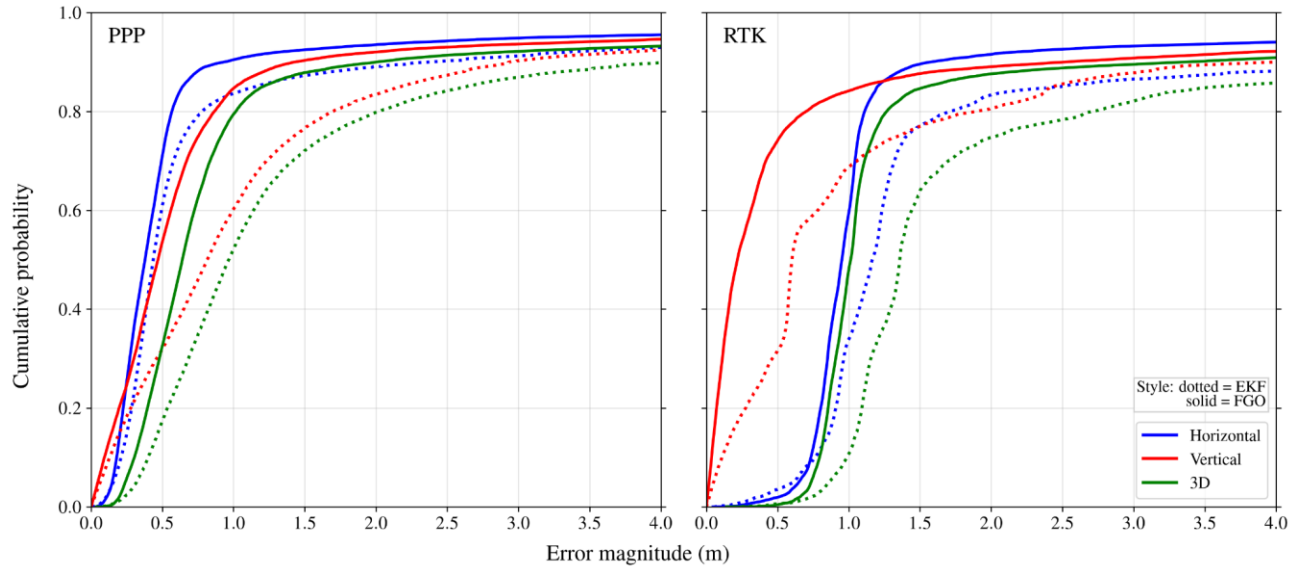
- We show an **ENU time series** plot for a subinterval of the LDN scenario, using **u-blox** GNSS and DR data
- PPP: FGO shows a lower UP bias and **fewer** high-amplitude **peaks**
- RTK: FGO maintains lower component-wise bias and **better boundedness** during overpass segment
- Around 15:30: short motorway tunnel
 - FGO **reconverges** more smoothly

London PPP & RTK Results

Summary

Observations

- We show **CDFs** for the full LDN route using **u-blox** GNSS and DR data
- This confirms **lower error magnitudes** across most of the distribution
- RTK shows very strong tail reduction
- **Max RTK baseline length:**
~70 km
 - Large position bias (see Figure)
- Min RTK baseline length:
~3 km



Mode	Solver	EN 50%	EN 95%	3D 50%	3D 95%	[in meters]
PPP	EKF	0.44	7.42	0.97	11.73	
PPP	FGO	0.38	3.16	0.64	6.29	
RTK	EKF	1.16	20.90	1.36	27.90	
RTK	FGO	0.95	6.08	1.01	13.93	

Campaign-Level Results

Across the 5 Routes

Relative Improvement (x100)	LDN	BHM	NTH	SFL	MID	LDN	BHM	NTH	SFL	MID
	Septentrio 2D RMS					u-blox 2D RMS				
Code	34.7	31.1	39.7	26.4	92.4	61.8	97.3	32.9	28.4	64.7
PPP	25.1	27.8	28.5	23.1	37.3	35.8	46.3	44.6	10.7	26.7
RTK	27.6	13.1	39.6	20.6	13.7	61.3	25.0	42.4	20.3	29.2
	Septentrio 3D RMS					u-blox 3D RMS				
Code	62.9	18.1	61.1	31.7	87.4	74.6	97.0	48.8	27.7	53.6
PPP	49.2	19.9	30.7	15.0	29.1	47.7	26.7	35.7	13.2	26.6
RTK	34.7	4.8	34.2	22.7	14.1	47.1	16.2	38.4	19.8	22.8

Discussion

- Gains **persist when aggregated** over total route duration
- Observed benefits are systematic at campaign scale
- RTK improvements are a bit more route-dependent
- Improvements are strongest where error conditions vary rapidly
 - Urban transitions, blockage/recovery phases, overpasses

Duration-Weighted RMS Average Improvements

x100	Septentrio		u-blox	
	2D	3D	2D	3D
Code	44.5	55.1	58.5	64.0
PPP	28.4	32.1	35.9	31.8
RTK	24.6	24.9	40.0	32.1

Processing Time and Window Size Analysis

Septentrio

Test	Input Data	PVT Engines	Environment	Route Selection
Processing Time Analysis CODE	Septentrio GNSS & IMU, Odometer, RINEX Navigation file	FGO-Code Vs WLS	Suburban & motorway	LDN Route 18:00:00 – 19:00:00
Processing Time Analysis PPP	Septentrio GNSS & IMU, Odometer, CODE Precise Products	FGO-PPP Vs EKF-PPP	Suburban & motorway	LDN Route 18:00:00 – 19:00:00
Processing Time Analysis RTK	Septentrio GNSS & IMU, Odometer, RINEX Navigation file, STRA Reference Station	FGO-RTK Vs EKF-RTK	Suburban & motorway	LDN Route 18:00:00 – 19:00:00
Window Size Test	Septentrio GNSS & IMU, Odometer, RINEX Navigation file	FGO-Code Vs WLS	Motorway	BHM Route 10:28:00 – 11:28:00

Computational Environment: Hardware & Software

Architecture	OS	CPU	Cores / threads	Memory	Container
x86-64	Ubuntu 24.04.4 LTS Kernel 6.8.0-100-generic	AMD EPYC 7443P Up to 4.036 GHz	24 physical / 48 hardware	62 GiB RAM	Docker. No explicit CPU or memory limits
GTSAM	GTSAM Compiler	MATLAB	MEX	Build Configuration	Flags
C++17, GNU++17	GCC/G++ 13.3.0	R2024b Update 9	g++ 13	Release	-O3 -DNDEBUG

Processing Time Analysis

Code, PPP, and RTK

Goal

- Compare FGO runtime against the corresponding benchmark engines

LDN	Processing Time Quantiles [ms]				
	5%	25%	50%	75%	95%
WLS	70	80	90	95	115
FGO-Code	130	160	205	260	325
EKF-PPP	260	280	310	340	440
FGO-PPP	160	210	280	360	450
EKF-RTK	175	190	200	210	240
FGO-RTK	210	250	300	340	400

Observations

- FGO-Code: **70-120% slower** than WLS, partly due to tropo/iono estimation overhead
- EKF-PPP: Runtime is currently limited by inefficient dynamic-state management
- FGO-RTK: Similar runtime to FGO-PPP and **30-70% slower** than EKF-RTK

Window Size Analysis

FGO-Code, Septentrio

Goal

- Evaluate how increasing the FGO sliding window size affects processing time and accuracy

BHM	Processing Time					
	WLS	WS2	WS4	WS10	WS18	WS30
Seconds	369	673	790	2405	10200	47365

What is the ideal window size?

- Exploratory runtime tests showed potential performance gains (though limited to 10% RMS improvements) with **larger windows**, but at a substantially **higher computational cost** (see table above)
- A **4-epoch** sliding window was therefore retained as a practical baseline configuration in FASCINATE. This value provides a good compromise between trajectory consistency, numerical stability and computational cost, which is in agreement with existing FGO literature

Environmental Dependence

Open Area, Suburban and Urban

Choice of Routes for Analysis

- Open Area
 - 1 ½ hour subset of LDN scenario
 - Mostly motorway driving, includes overpasses
- Sub Urban
 - Take NTH scenario, since Nottingham is characterized by suburban conditions
 - 6-hour total, diverse environment with tree canopies and residential areas
- Urban
 - 2 ½ hour subset of LDN scenario
 - Central London & City, includes canyons and tunnels

Results

- We find only a **minor** environmental dependence
- Note, in the limit of static, open-sky processing:
 - FGO and EKF results are very similar, as expected!

3D RMS Position Accuracy Improvement

% Change	PPP %	RTK %	Code %
Open Area	25-30	15-20	50-70
Sub Urban	25-35	15-35	25-55
Urban	20-30	20-30	30-70

We find **NO** significant difference in performance across environments!

Conclusions and Next Steps

Conclusions

Consistent FGO Gains

- FGO improved performance across the evaluated scenarios, receivers and processing modes, confirming the benefit of temporal graph optimization
- The largest gains were observed in RMS and **high-percentile** errors, showing improved robustness against multipath, NLOS reception and degraded satellite geometry
- Window size of 4 epochs provides a good compromise between accuracy, robustness and cost
- Gains hold at aggregated level
 - Relative improvements for code-based processing are **30-60%** in 2D and 3D RMS
 - Relative improvements for PPP and RTK are **25-40%** in 2D and 3D
 - **u-blox** generally **benefits more** from FGO than Septentrio in both 2D and 3D

FGO offers *significant* position accuracy improvements for code, PPP and RTK across motorway, sub-urban and urban driving scenarios.

Next Steps

Algorithm Consolidation and Target Optimization

Evolution to Full Tightly Coupled (TC) GNSS/INS Integration

- Extend current architecture to a full 3D tightly coupled GNSS/INS factor graph
- Integration of full 6-DoF IMU preintegration
- Augmented state vector: vehicle attitude, IMU biases (gyro and accelerometer), and odometer scale factors

Partial Integer Ambiguity Resolution (PIAR)

- Implement partial fixing strategy with backfeeding of integer constraints into the factor graph
- Fix only the subset of highly reliable carrier-phase ambiguities

FGO-Based Integrity Monitoring

- Develop dedicated integrity monitoring frameworks
- Develop a reliable error bound algorithm for the FGO-based multi-sensor solutions

Next Steps

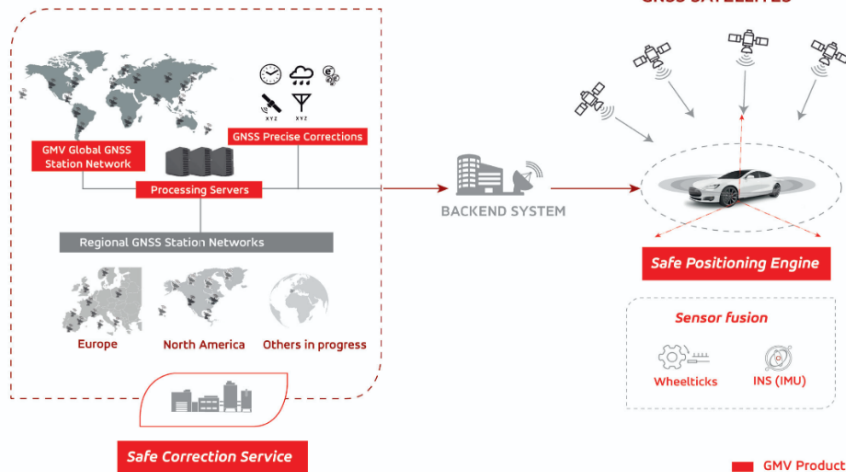
Industrialisation and Portfolio Integration

Potential FGO Approach Integration into GMV products

- Adapt the software considering real-time PVT requirements and customer-specific targets
- Explore integration of the FGO software into existing GMV positioning solutions, such as GSharp, as a complementary estimation mode, leveraging its modular and multi-sensor architecture to enhance robustness in challenging GNSS environments

GMV GSharp®

SAFE HIGH-ACCURACY RELIABLE POSITIONING
FOR AUTOMOTIVE



Thank you

FASCINATE Team



088 QUATER - PROOF-OF-CONCEPT OF ADVANCED NAVIGATION ALGORITHMS BASED ON FACTOR GRAPH OPTIMIZATION

Status: On Going

Activity Code: NAVISP-EL1-o88 quater

Start date: 07/11/2025

Duration: 18 Months

Factor graphs are widely used in robotics, especially for simultaneous localization and mapping, but also in filtering and smoothing, model-based signal processing, artificial intelligence, coding and decoding. They serve as probabilistic models where variable nodes are associated with system states, and factors are associated with measurements.

The factor graphs techniques have been applied to the positioning algorithms in the recent years, with very positive results.

The objective of FASCINATE is to investigate how to exploit Factor Graph Optimization (FGO) for PNT positioning, and to benchmark its performances with respect to the widely used high-accuracy positioning approaches such as Extended Kalman Filter.

To accomplish this, a FGO navigation engine will be developed and integrated into a comprehensive testbed that includes multiple Global Navigation Satellite System (GNSS) receivers, antennas, and an Inertial Measurement Unit (IMU). This testbed, combined with previously recorded data, will be utilized to assess the performance of the FGO through an experimental campaign.

PRIME CONTRACTOR



Name: GMV NSL Ltd

Country: United Kingdom

Website: <https://www.gmv.com/en-es>

SUBCONTRACTORS



Name: Cranfield University

Country: United Kingdom

Website: <https://www.cranfield.ac.uk/>