INFORMATION AND CONTROLLING SYSTEM

This study focuses on the reconstruction of missing GPS trajectory data. The principal issue relates to restoring geospatial coordinates in the absence of large volumes of labeled data and under conditions where conventional spatial-temporal models demonstrate limited generalization capabilities.

This paper proposes a large language model-based approach to address the reconstruction task without requiring prior training on specialized datasets. To reduce dependence on domain-specific features, the focus was on optimizing data preprocessing and constructing effective prompts. Three coordinate representations have been explored: original degree-based values (using the VPAIR dataset), the Earth-Centered, Earth-Fixed (ECEF) system, and ECEF coordinates shifted relative to the starting point of the trajectory.

Experimental results show that using centered ECEF coordinates reduces the mean absolute error (MAE) by 51–59% for both latitude and longitude compared to other representations. Conversion to the ECEF system also demonstrates selective advantages in latitude reconstruction. To mitigate the instability of autoregressive prediction, a multi-iteration reconstruction strategy with result aggregation has been implemented. The open-source model LLaMA 3.2 achieved the highest accuracy (MAE: 36.57 for latitude and 52.14 for longitude), outperforming both other open models and the commercial GPT-4o.

The proposed approach can be considered a viable post-processing tool, particularly in missions involving unmanned aerial vehicles or other mobile platforms where part of the GPS data has been lost during acquisition

Keywords: large language models, missing values, neural networks, imputation, prompt tuning

UDC 004.8, 004.93, 004.932
DOI: 10.15587/1729-4061.2025.335592

RECONSTRUCTING MISSING GLOBAL POSITIONING DATA WITH ZERO-SHOT LARGE LANGUAGE MODELS

Roman Ilechko
Corresponding author
PhD student*
E-mail: roman.i.ilechko@lpnu.ua
Orest Borovyi
PhD student*
Kyrylo Yemets
PhD student
Department of Artificial Intelligence Systems**
Yurii Tsymbal
PhD, Associate Professor*
*Department of Automated Control Systems**
**Lviv Polytechnic National University
Bandery str., 12, Lviv, Ukraine, 79013

Received 29.04.2025
Received in revised form 20.06.2025
Accepted date 07.07.2025
Published date 29.08.2025

How to Cite: *Ilechko, R., Borovyi, O., Yemets, K., Tsymbal, Y. (2025). Reconstructing missing global positioning data with zero-shot large language models. Eastern-European Journal of Enterprise Technologies, 4 (9 (136)), 6–18. https://doi.org/10.15587/1729-4061.2025.335592*

1. Introduction

Autonomous mobile systems, in particular unmanned aerial vehicles (UAVs), rely heavily on Global Positioning System (GPS) data to provide accurate positioning, navigation, etc. GPS data provides three-dimensional geospatial coordinates (latitude, longitude, altitude) and time information, which is critical for the correct operation of autonomous control algorithms in real time. However, due to technical or environmental limitations, continuous access to GPS is not always possible. Studies [1, 2] show that situations of GPS signal loss or degradation are common in urban environments, dense forests, tunnels, and under the influence of radio frequency interference. Loss or degradation of the signal creates significant operational risks.

Localization errors. In environments without access to GPS, UAVs often rely on inertial measurement systems (IMUs) to estimate position using inertial navigation. However, IMUs are prone to accumulating errors (such as accelerometer drift and gyro noise), which leads to exponentially increasing positioning errors over time.

Collision risks. In the absence of accurate geolocation data, obstacle avoidance systems may incorrectly match sensor data (e.g., LiDAR, cameras) to real coordinates, which increases the likelihood of collisions.

Mission failure. Applications in agriculture, search and rescue, or cargo delivery require high positioning accuracy. GPS signal interruptions prevent correct tracking of waypoints, making such tasks unreliable.

Existing approaches to GPS data recovery are typically based on the use of machine learning models, such as recurrent neural networks or statistical methods [3–5], which require special training on large, highly specialized GPS data sets. However, such methods have a number of significant limitations: requirements for a high-quality training dataset, risks of overtraining, and difficulties in transferring models to new geographical or contextual conditions.

In this context, a new class of models, large language models, such as GPT [6, 7], Llama [8–10], Gemma [11], and others, open up new perspectives. These neural networks, trained on large-scale corpora of sequential text data, have an architecture that supports generalization, memorization of long-term dependences, and processing of structured inputs. The ability of LLMs to zero-shot generalization [12], that is, solving problems without prior training on a specific example, makes them an attractive tool for the task of recovering missing values in time series, in particular GPS sequences.

Although LLMs were not specifically trained on geospatial data, their ability to reproduce patterns and logical dependences between sequence elements suggests their effectiveness in trajectory reconstruction tasks. In study [13], the importance of pre-processing, the choice of data representation system, as well as the construction of an effective text prompt that ensures the interpretability and correctness of the answer, is particularly emphasized.

Therefore, studies on zero-shot reconstruction of GPS data using large language models are relevant, especially in

the context of data post-processing for autonomous systems. These results could be applied in real-world scenarios where it is necessary to restore lost coordinates, in particular for UAVs or other mobile platforms that face problems of GPS signal loss due to technical or environmental limitations.

2. Literature review and problem statement

In [14], a real-time fuzzy motion control is proposed that instantly smooths out external GPS signal perturbations, providing robust navigation in the presence of obstacles. In contrast, in [15], a visual-inertial scheme is described that combines camera and inertial sensor data for navigation in the absence of GPS communication. These approaches demonstrate effectiveness in real-world scenarios but rely on complex hardware setups and hybrid sensor systems, which reduces their versatility.

In [16], splines are considered as a means to account for smooth signal changes, but the authors emphasize that the choice of spline order and node size can lead to overfitting. In [17], the authors caution that its effectiveness depends on the correct specification of models for each variable, as well as on the size and distribution of missing data in the multidimensional space. In [18], kriging is reported as a geostatistical method that optimally takes into account spatial correlation. However, the authors note that the effectiveness of kriging is based on strict assumptions about the stationarity and isotropy of the process, as well as on the quality of the variogram estimate, which can be resource-intensive in the case of complex spatiotemporal irregularities. Paper [18] also indicates that kriging under conditions of sparse data or the presence of anomalies may give inaccurate forecasts without the use of variogram correction methods. Thus, the listed methods show high computational efficiency and the ability to restore missing values, provided that the signal is smooth and stationary. However, their linear assumptions often do not correspond to the complex nonlinear dynamics of modern GPS trajectories, worsening the accuracy during sharp changes in speed or direction of movement.

In [19], the KNN method is considered easy to implement, but its efficiency decreases in the case of uneven distribution of gaps, since it does not take into account the temporal or spatial structure of the data. In [20], RNN approaches demonstrate a better ability to account for temporal dependence in multidimensional series. However, the authors emphasize that such models are sensitive to sequence length and may experience difficulties with gradient decay or gradient explosion. The approach in [21] is proposed as a means of simultaneously processing spatiotemporal correlation but tuning the architecture of the method requires significant computational resources and careful validation on different data sets. In addition, the model can overtrain on small samples and perform poorly with atypical gap patterns. The efficiency of the above methods depends on the availability of large, labeled data sets, optimization of hyperparameters, and the ability of the model to generalize results on data with gap patterns different from the training ones.

Hybrid models that combine spatiotemporal kriging techniques with deep neural networks or graph neural networks [22, 23] demonstrate higher imputation accuracy by integrating local spatial information with global temporal trends. However, their effectiveness depends on the availability of auxiliary data and high computational resources for fine-tuning parameters.

Despite significant progress, key challenges remain in GPS imputation:

1) dependence on large datasets;

2) limitations in the domain adaptability of spatiotemporal models;

3) non-generalizability to atypical omission patterns.

A likely option to overcome these difficulties is to use large language models under a zero-shot mode, which do not require retraining and are able to understand the semantics of spatial-temporal sequences through text prompts. All this gives grounds to argue that the study of zero-shot imputation of GPS data by large language models is a reasonable and promising area for the further development of spatial-temporal data restoration technologies.

3. The aim and objectives of the study

The aim of our study is to devise an approach to recovering lost GPS data using large language models without additional training. This will make it possible to implement an effective GPS data post-processing module for imputation of missing fragments in traffic monitoring systems, increasing the accuracy and consistency of trajectories without the need for additional training of profile models.

To achieve this aim, the following objectives were accomplished:

– to choose the optimal format for representing GPS coordinates;

– to determine the optimal version of the instruction;

– to perform a comparative assessment of the quality of recovery for different language models.

4. The study materials and methods

4. 1. The object and hypothesis of the study

The object of our study is the restoration of a sequence of GPS coordinates from a real open dataset VPAIR, which contains vehicle trajectories. The principal hypothesis of the study assumes that the use of large language models with an optimal data format, well-formulated instructions, and a retry mechanism allows for the correct reconstruction of missed fragments of GPS sequences without additional training.

The following assumptions were adopted:

– the influence of atmospheric or satellite failures on the initial GPS data is not taken into account;

– the selected LLM model is able to interpret digital sequences in context, regardless of geographical reference;

– no additional training or tuning of the LLM is performed.

4. 2. Large Language Models

Current large language models have revolutionized natural language processing through the use of a revolutionary architecture known as a transformer. The transformative potential of a transformer lies in its innovative structure, which replaces sequential processing with a mechanism capable of simultaneously covering all parts of the input sequence. This is implemented using the attention mechanism, a key innovation that allows models to dynamically evaluate the relevance of different tokens, capture complex long-range relationships, and build rich contextual representations. It is this mechanism that underlies the effectiveness of trans-

former-based models, making them indispensable for a wide range of applications.

The attention mechanism, proposed in [24] and subsequently improved in [25], has become the foundation of modern natural language processing. It addresses a critical limitation of previous models, such as recurrent neural networks (RNNs), by allowing dynamic focus on individual parts of the input sequence. Unlike RNNs, which process data sequentially and often struggle with long-range relationships, the attention mechanism makes it possible to directly capture relationships between distant elements regardless of their positional distance. The mechanism computes a context-sensitive representation of the input sequence, assigning importance weights to its elements. Given a query vector Q, which represents the current focus of the model, and a set of key-value pairs (K, V), the mechanism computes a compatibility coefficient between the query and each key. These coefficients are normalized using a softmax function, as shown in equation (1), to obtain weights that determine how much each value contributes to the final output

$$A(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V, \quad (1)$$

where d_K is the dimensionally of the key vector. The scaling factor is used to prevent excessive growth of scalar products, which can lead to instability of gradients during training. To improve the model's ability to capture various types of dependences in the data, researchers proposed a multi-head attention mechanism. Instead of computing a single set of attention weights, this mechanism uses multiple attention "heads", each of which has its own training linear transformations for the query, key, and value vectors. The outputs of all heads are concatenated and projected back into the original feature space, which allows the model to focus on different aspects of the input data simultaneously. Building on these improvements, the authors of [25] proposed a transformer architecture that has become a real paradigm shift in sequence modeling. Unlike conventional models based on recurrent or convolutional layers, transformers are fully based on the attention mechanism, which allows them to process entire sequences in parallel. This parallel processing significantly increases computational efficiency, making transformers particularly suitable for training on large amounts of data using modern hardware. The transformer architecture consists of two main components: an encoder and a decoder. The encoder processes the input sequence and generates a set of contextualized representations, while the decoder generates the output sequence, guided by both the encoder outputs and its own prior predictions. Each encoder and decoder layer contains two key substructures: a multi-head attention mechanism and a position-wise feed-forward network. Multi-head attention allows the model to capture complex dependences in the input data, while the fully connected network applies a nonlinear transformation to each position separately, further refining the representation.

Large language models are a revolutionary advancement in natural language processing, fundamentally changing the capabilities and applications of artificial intelligence. LLMs are built on a transformer architecture that uses deep neural networks with billions or trillions of parameters. This architectural innovation allows the model to dynamically assess the importance of different tokens in a sequence, capturing complex patterns and long-range dependences in text data. The effectiveness of LLMs is based on a pre-training paradigm, where models are trained on huge and diverse corpora spanning hundreds of terabytes of text. This allows LLMs to develop complex representations of language structure, semantics, and even implicit knowledge about the world. By training on such large data sets, these models gain the ability to generalize to a wide range of linguistic tasks, from text generation and summarization to question answering and machine translation. Pre-training is usually accompanied by additional tuning or prompt-based adaptation, where the model is tuned to specific tasks, allowing for further improvements in its performance and versatility. One of the most significant achievements of LLMs is their ability to perform few-shot or zero-shot training, where the model can generalize to new tasks with minimal or no specific training data for those tasks. This ability is largely due to the scale of the models and the amount of pre-training that gives them a form of meta-learning. For example, models like GPT-3 [6] have demonstrated an impressive ability to generate coherent and contextually relevant text, even for tasks for which they were not explicitly trained.

4. 2. 1. Llama

Llama has been selected as the base model among open language models due to its technical characteristics and architectural innovations. The Meta AI family of large language models developed by Meta is a significant contribution to the field of natural language processing. Since its inception, Llama has gone through several stages of evolution, each of which has brought improvements in architecture, training methods, and performance.

Llama v1 [8] laid the foundation for the Llama family by introducing a robust transformer-based architecture. The model had a standard transformer-decoder with multi-head self-attention and pre-normalization, which allowed it to effectively capture contextual relationships between words. Llama v1 was trained on a diverse corpus of text data, including books, web pages, and academic papers, using a holistic language model. This approach allowed the model to predict the next token in a sequence, thereby learning the statistical structure of language. The efficient use of computational resources made the model a major advance in the field of large language models.

Building on Llama v1, Llama v2 [9] introduced several key improvements that improved both efficiency and performance. Architecturally, Llama v2 incorporated the Grouped Query Attention (GQA) technique, which reduced the computational complexity and memory requirements of the self-attention mechanism. This innovation allowed the model to work more efficiently with longer sequences and larger data sets. The training dataset for Llama v2 was expanded and carefully selected, including synthetic data and instructional tuning datasets to improve performance for specific tasks. Additionally, Llama v2 introduced instructional tuning and human feedback-based reinforcement learning (RLHF), which allowed the model to more accurately respond to human preferences and increased its utility in real-world applications. Llama v2 also expanded the context window, allowing the model to process and generate longer sequences of text. This improvement was particularly useful for tasks that required consistency across large texts, such as document summarization or long-form text generation.

Llama v3 [10] is the latest advancement in the Llama series, focusing on scalability, efficiency, and multimodal

capabilities. Although the architectural changes between Llama v2 and Llama v3 are relatively minor, the latest version introduces a larger context window and improved rotational embedding (RoPE) parameters, which allow for better handling of long sequences. The biggest improvements in Llama v3 relate to the training data and learning methods. The model was pretrained on a dataset exceeding 15 trillion tokens, seven times larger than Llama v2, and includes four times more code. In addition, over 5% of the training data consists of high-quality content in non-English languages, covering over 30 languages, significantly enhancing its multilingual capabilities. Meta has implemented advanced data filters for Llama v3, including heuristic filters, NSFW filters, semantic deduplication, and text classifiers trained with Llama 2 to ensure data quality. Llama v3 also benefits from improved post-training tuning techniques that reduce false positives, improve consistency with user intent, and diversify model responses. One notable feature of Llama v3 is the consistent logarithmic performance improvement, even after training on significantly larger data sets than expected. This trend challenges conventional scaling laws, such as optimal computational scaling using the Chinchilla model, as the model continues to improve beyond a typical performance plateau. To ensure reliable and unbiased evaluation, Meta has introduced a new human scorer set consisting of 1,800 cues across 12 critical use cases, including advice-giving, brainstorming, and creative writing. The training methodology includes improvements in parametric efficiency techniques, such as optimized attention mechanisms, which reduce computational costs while maintaining performance. The data selection process for training prioritizes high-quality multilingual content that provides strong cross-linguistic capabilities. This careful data selection, combined with improved pre-training objectives, allows Llama v3 to achieve performance comparable to much larger models while maintaining a more efficient number of parameters.

4. 2. 2. GPT-4o

The selection of GPT-4 [7] as the reference model for closed models reflects its position at the forefront of language modeling capabilities. The model demonstrates exceptional performance across a wide range of natural language processing tasks, achieving best-in-class results without task-specific architectural changes or refinements to logical reasoning. The GPT-4 architecture builds on previous versions, with sophisticated improvements to the attention and parameter scaling mechanisms. The model exhibits impressive stability in multitasking scenarios, demonstrating robust generalization capabilities across a variety of application domains. Its token processing system allows it to handle complex inputs with increased accuracy, supporting complex language understanding tasks with high reliability. The model's documented performance on standard benchmarks demonstrates significant gains in natural language understanding and generation. These metrics include performance on tasks requiring deep contextual understanding, abstract reasoning, and complex instruction execution. The stability of these results across different levels of task complexity and application domains establishes GPT-4 as a representative benchmark for modern closed language models.

4. 2. 3. Gemma

The open-source Gemma family of language models [11], developed by Google DeepMind, represents a significant breakthrough in building lightweight yet high-performance large-scale language models. Designed to strike a balance between efficiency and accuracy, the first version of Gemma introduced models with 2 billion and 7 billion parameters trained on data of up to 6 trillion tokens with a context length of 8,192 tokens. These models, available in basic and instruction-tuned versions, have been optimized for deployment on a wide range of hardware platforms, including consumer devices, without the need for significant amounts of quantization. Building on this version, Gemma 2 introduced several architectural improvements, as well as expanded model sizes: 2 billion, 9 billion, and 27 billion parameters. One of the most significant improvements in Gemma 2 is the integration of Grouped Query Attention (GQA) technology, which optimizes the self-attention mechanism, reducing memory and computational costs, while maintaining high quality of the output data. The model uses 16 shared key-value heads of size 128, as well as 32 query heads and output projections of the same size, which results in a more efficient attention mechanism. In addition, Gemma 2 includes an additional layer of root mean square normalization (RMSNorm), which improves the stability of training and overall reliability of the model. The increase in the number of parameters and the improved architecture of Gemma 2 have contributed to significant improvements in language modeling tasks, including question answering, summarization, and common sense reasoning. While the original Gemma models established a solid foundation for compact high-performance LLMs, the implementation of GQA and improved normalization methods in Gemma 2 has significantly improved the scalability and reasoning capabilities.

4. 3. Missing data recovery with Large Language Models

Missing value recovery was implemented using several large language models under a zero-shot mode, including both open-source and closed-source models (LLaMA 3.2, Gemma with 2B and 7B parameters, and GPT-4o). For comparison, a basic mean-filling recovery technique was also implemented. Input sequences were shaped using a fixed-length window of 100 data points, and a rolling window strategy of length 50 was used to generate overlapping segments. This approach provided sufficient contextual information for the models during prediction. An important component of the methodology was hint engineering. Different hint formulations were systematically evaluated to determine their impact on the ability of LLM to accurately predict missing GPS values. The performance was quantified using the following metrics: mean absolute error (MAE), mean square error (MSE), and root mean square error (RMSE). These metrics allowed for a comparison between the LLM result and the baseline method, which helped reveal the relative advantages and limitations of the proposed imputation technique. The imputation algorithm is designed so that the number of predicted values exactly matches the number of missing values in the data sequence. The algorithm starts with the initialization of the prompt. First, a predefined prompt template is called, which instructs the LLM to perform the imputation. After that, a segment of GPS data that contains both observed data points and missing values is selected and inserted into the prompt template. The next step is to count the number of missing values in the sequence, which determines the target number for the LLM imputation process.

Once the prompt is filled with the appropriate data sequence, it is fed to the LLM for prediction. The model is ex-

plicitly instructed to return exactly the number of values that are missing. After receiving the output, the algorithm checks whether the number of values returned matches the expected number. If the numbers match, the imputed values are directly inserted into the sequence. However, if a mismatch occurs, the algorithm enters the retry phase. The retry mechanism uses a secondary prompt that provides additional context, including the previously failed sequence as an example, and repeats the requirement for the accuracy of the number of missing data. This retry process is performed up to N times or until the LLM output exactly matches the required number. If the mismatch persists after N retries, a final adjustment is made. At this point, if the absolute difference between the expected and predicted numbers is within a predefined range M, the algorithm completes the imputation, taking the minimum of the two values. If the difference exceeds this range, the retry process is repeated for N more attempts.

The process can be described in pseudocode as follows:

```
Algorithm ImputeMissingValues(sequence, promptTemplate, retryPrompt, N, M)
    missingCount ← countMissing(sequence)
    prompt ← fillTemplate(promptTemplate, sequence)
    predicted ← LLM(prompt)
    predictedMissingCount ← countMissing(predicted)

    retries ← 0
    // Initial attempt and iterative retries
      while predictedMissingCount ≠ missingCount and retries < N do
        prompt ← fillTemplate(retryPrompt, sequence, predicted)
        predicted ← LLM(prompt)
        predictedMissingCount ← countMissing(predicted)
        retries ← retries + 1
    end while

    // Final check if count still does not match
    if predictedMissingCount ≠ missingCount then
      if |missingCount - predictedMissingCount| ≤ M then
        // Adjust prediction to use the minimum valid count
          predicted ← adjustPrediction(predicted, missingCount)
      else
          // Additional retries if the adjustment condition is not met
          for i from 1 to N do
            prompt ← fillTemplate(retryPrompt, sequence, predicted)
            predicted ← LLM(prompt)
            predictedMissingCount ← countMissing(predicted)
            if predictedMissingCount = missingCount then
              break
            end if
          end for
      end if
    end if

    return predicted
```

Analysis of the completion results revealed that discrepancies in sequence length can arise for two main reasons: model hallucinations or tokenizer-related problems. In the first case, which was observed mainly in models with less than 7 billion parameters, the phenomenon of model hallucinations led to the output of sequences that were significantly longer than expected. In the second case, the discrepancies arose due to the inherent differences between the number of tokens and the actual sequence length; a large language model could generate a different number of tokens than the expected sequence length, leading to minor variations in the final result. Choosing a minimum length between the expected and generated sequence turned out to be an effective compromise that increases the consistency of the recovery process and maintains high accuracy.

4. 4. Dataset description

Our study began by selecting the VPAIR dataset [26] as the primary source of GPS information. In order to comprehensively assess the impact of the data representation technique on the efficiency of imputation procedures, three dataset variants were generated:

1) original GPS coordinates [27];

2) GPS coordinates transformed into the Earth-Centered, Earth-Fixed (ECEF) coordinate system to ensure spatial consistency;

3) shifted coordinates in the ECEF system to standardize the range of input data.

To simulate real-world scenarios with incomplete data, gaps were systematically introduced into the generated datasets. Sequences with different levels of missing data and sequence lengths were generated, which allowed for a thorough assessment of the robustness of the models to different levels of data degradation.

This study uses the VPAIR dataset, an innovative dataset specifically designed to address Visual Place Recognition (VPR) and Visual Localization (VL) tasks in large-scale open environments. The VPAIR dataset fills a significant gap in existing studies that have mostly focused on ground-based or low-altitude aerial platforms. It contains images acquired from a light aircraft at an altitude of 300–400 meters over a 107 km route that passes through urban, agricultural, and forested landscapes (Fig. 1). The images are accompanied by high-quality reference renders, dense depth maps, and accurate six-degree-of-freedom (6-DoF) poses computed from GNSS/INS data. Fig. 2 shows the spatial coordinates versus time, for each variable separately.

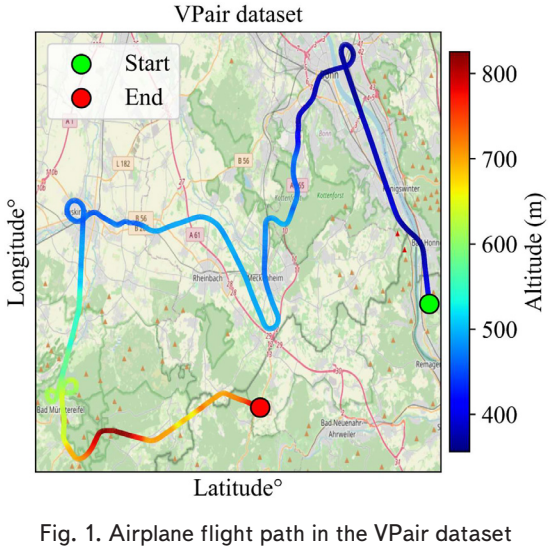

Fig. 1. Airplane flight path in the VPair dataset

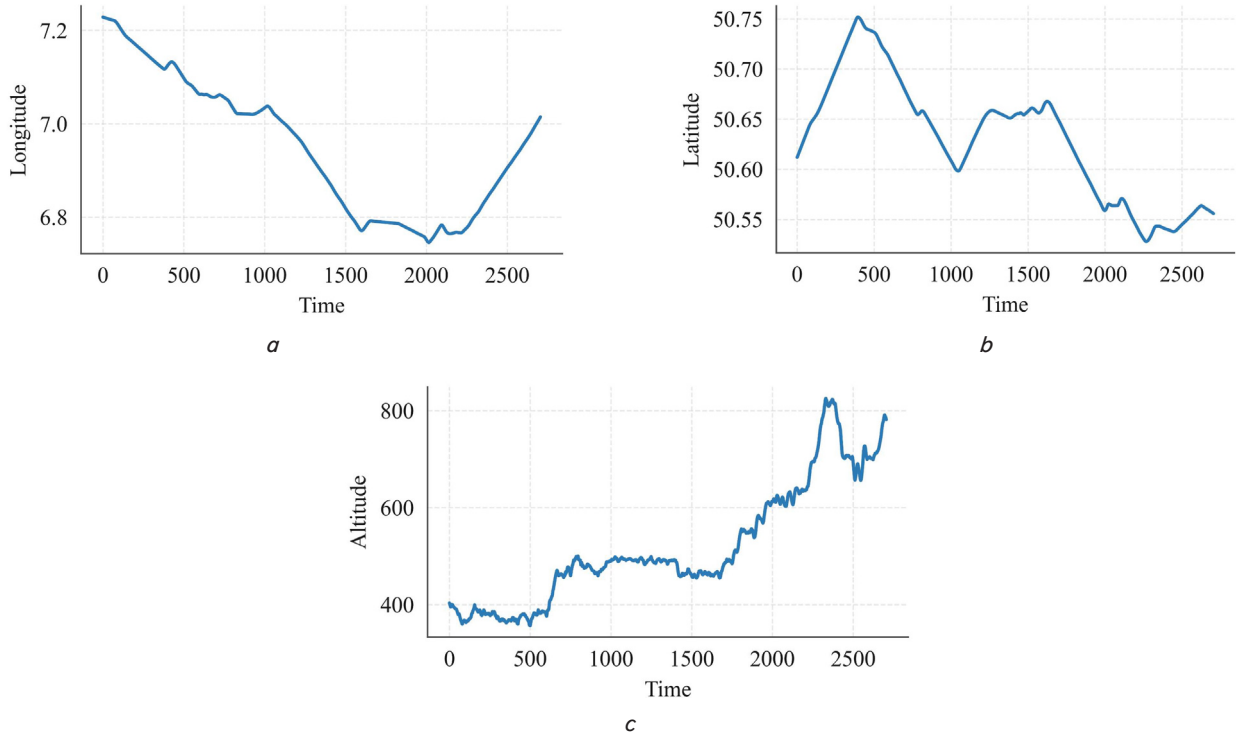

Fig. 2. Change in spatial coordinates over time: *a* — longitude; *b* — latitude; *c* — altitude

The authors of the dataset show that conventional VPR/VL methods, such as NetVLAD and D2-Net, have limited performance due to in-plane rotations. These rotations are typical of high-altitude images. In contrast, rotation-resistant descriptors provide higher recognition quality. The main advantages of VPAIR are its scale, environmental diversity, and integration of publicly available geospatial data to create reference renders, which contributes to the reproducibility of studies. High reliability of the data is additionally ensured by careful sensor calibration using Kalibr and hardware synchronization. Among the limitations, it should be noted that the data were collected during a single day, which makes it impossible to take into account seasonal and lighting changes, as well as possible discrepancies between the rendering of reference images (with model accuracy up to 0.5 m) and real-world conditions. Furthermore, the simplification of the reference poses to fixed orientations with the camera pointing vertically downwards may limit the application of the dataset to dynamic aerial platform tasks.

Although experiments have shown the computational inefficiency of local feature-based approaches, the practical deployment of such systems remains an open question. The VPAIR publication provides a strong benchmark for further development of air navigation technologies under conditions of limited GNSS availability, which is especially relevant in the context of emerging applications such as Urban Air Mobility.

Our study used GPS data to solve the problem of filling in missing values in spatial localization tasks. The GPS structure of the dataset provides absolute positional references and integrates with the North, East, Down (NED) convention, which allows for a full-fledged estimation of positions. These positional data are critical for the development and validation of methods for interpolating missing location information, especially given the significant coverage of trajectories in the dataset. The structured nature of GPS data creates an ideal basis for devising reliable methods for estimating missing values, providing the ability to evaluate the effectiveness of interpolation approaches at different spatial scales.

The integration of high-precision GPS data creates a reliable basis for the development of algorithms aimed at ensuring continuous spatial tracking, which is critical for navigation and mapping applications in conditions of positional continuity disruption. The scalability of the dataset and the applied technical synchronization mechanisms ensure its suitability for the development of reliable localization systems in aeronautical scenarios.

4. 5. Dataset preprocessing

For the missing value imputation task, a portion of the VPAIR dataset containing a sequence of GPS flight path coordinates was used, pre-processing it in several ways for the experiments. The original data were represented in a geodetic coordinate system in the LLA format (latitude, longitude, altitude), with reference to the WGS 84 (World Geodetic System 1984) ellipsoid, which is the global standard for GPS and GNSS applications. Latitude (ϕ) defines the angular position north or south of the Earth's equator in the range from −90° (South Pole) to +90° (North Pole). Longitude (λ) defines the angular position east or west of the prime meridian (Greenwich, United Kingdom) in the range from −180° to +180°. Height (h) represents the height above the WGS84 ellipsoid, a mathematically defined surface that approximates the shape of the Earth with a semi-major axis length of 6,378,137 meters and an oblateness factor of 1/298.257.

To ensure reliable imputation, three separate datasets were generated based on the original LLA coordinates. LLA dataset. The original geodetic coordinates were retained for basic analysis. ECEF dataset. The LLA coordinates were transformed into the Cartesian Earth-Centered, Earth-Fixed (ECEF) (X, Y, Z) system using the WGS84 ellipsoid parameters. The transformation is performed using formula (2), the radius of the principal vertical of curvature

$$N=\frac{a}{\sqrt{1-e^2\sin^2\phi}}, \tag{2}$$

where a = 6,378,137 (semi-major axis), $e^2=2f-f^2$ (eccentricity squared), and f = 1/298.257 (flattening).

Then the new ECEF coordinates are calculated from formulas (3) to (5):

$$X=\left(N+h\right)\cos\phi\cos\lambda, \tag{3}$$

$$Y=\left(N+H\right)\cos\phi\sin\lambda, \tag{4}$$

$$Z=\left[\left(1-e^2\right)N+h\right]\sin\phi. \tag{5}$$

The transformations (3) to (5) simplify spatial computations by representing positions in a three-dimensional global coordinate system with the X, Y, and Z axes centered at the Earth's core. An example of spatial coordinates with generated missing values is shown in Fig. 3.

Shifted ECEF dataset. Shifting the ECEF coordinates to a relative coordinate system by moving the origin to the spatial minimum of the trajectory. For each ECEF component X, Y, Z, the minimum value over the entire dataset is first computed:

$$X_{\min}=\min\left(\left\{X_i\right\}\right), \tag{6}$$

$$Y_{\min}=\min\left(\left\{Y_i\right\}\right), \tag{7}$$

$$Z_{\min}=\min\left(\left\{Z_i\right\}\right), \tag{8}$$

where X_i, Y_i, Z_i are the ECEF coordinates of the i-th data point. Using the result of calculations (6) to (8), the shifted coordinates are calculated from formulas (9) to (11):

$$X_i'=X_i-X_{\min}, \tag{9}$$

$$Y_i'=Y_i-Y_{\min}, \tag{10}$$

$$Z_i'=Z_i-Z_{\min}. \tag{11}$$

The transformation according to formulas (9) to (11) ensures that all coordinates are non-negative and relative to point 0, which effectively linearizes positional shifts and reduces numerical instability in algorithms sensitive to large absolute values.

Also, for each data set, aggregated sequences are created by combining N consecutive records and calculating their average. For a component of the coordinate x_i (for example, latitude, longitude, or altitude), the aggregated average for the kth group is determined using formula (12) as

$$\bar{x}=\frac{1}{N}\sum_{i=kN}^{(k+1)N-1}x^i, \tag{12}$$

where $k \in \{0, 1, ..., \lfloor M/N \rfloor - 1\}$ and M is the total number of data points. The computed aggregation using a sliding window smooths out high-frequency noise and reduces random GPS errors while preserving the trajectory structure.

The resulting datasets – LLA, ECEF, and shifted ECEF – provide complementary representations of the flight path, allowing for a systematic evaluation of imputation methods in different coordinate systems with different scales.

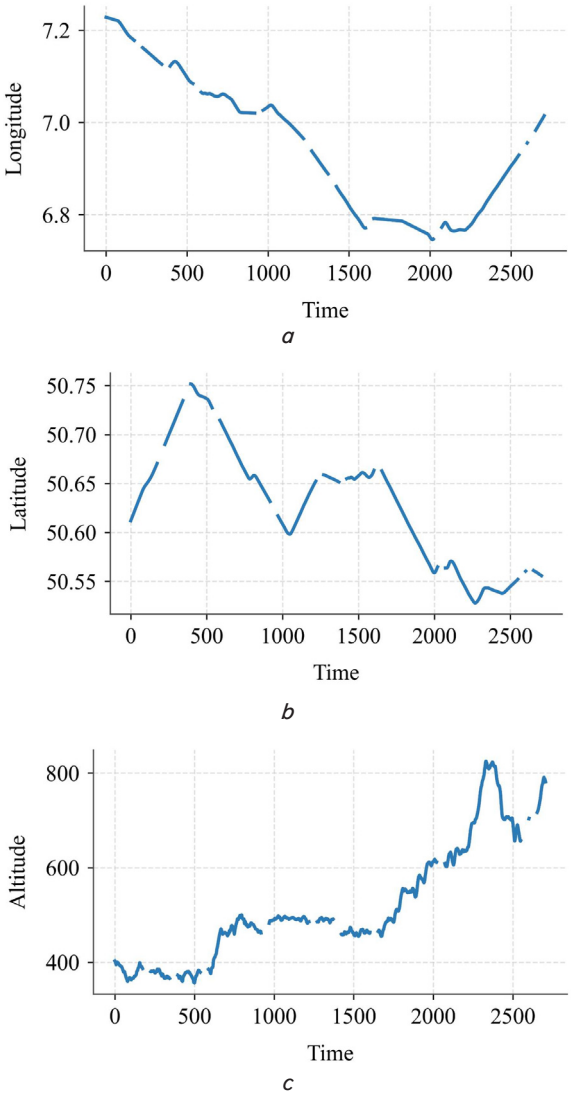

Fig. 3. Spatial coordinates after generating missing values: *a* – longitude; *b* – latitude; *c* – altitude

4. 6. Generation of missing values

Missing values are introduced into the GPS sequence by replacing adjacent data blocks with *NaN* values, which simulates situations such as sensor failures or signal loss. The mathematical model of this process is as follows.

Parameters: number of data points in the sequence – N, L – length of the lost data block. Number of lost data blocks – m, calculated from formula (13)

$$m=\frac{\text{total missing values}}{L}. \tag{13}$$

Constraints:

1. No intersection of blocks. For any two blocks starting with indices s_i and s_j, condition (14) must be satisfied so that

$$\left|s_i-s_j\right|\geq L, \forall i\neq j. \tag{14}$$

2. Generation limits. For each block, condition (15) must be satisfied, the block must fit within the sequence

$$s_k+L\leq N, \forall k\in\left\{1, ..., m\right\}. \tag{15}$$

Procedure:

1. Checking the feasibility of generation: If the required space for m blocks satisfies condition (16)

$$m \times L > N - L + 1, \quad (16)$$

the process is impossible because there is not enough space for non-overlapping blocks.

2. Determining the start indices of the sequences: for each block $k \in \{1, ..., m\}$. Randomly choose s_k uniformly from $\{0, 1, ..., N - L\}$, such that s_k is at least L units away from all previous start indices.

3. Generating missing values: for each block k, values from the $[s_k, s_{k+L}]$ interval are replaced by NaN, using formula (17)

$$x_{s_k:s_{k+L}} \leftarrow NaN. \quad (17)$$

Mathematical guarantees:

1. Total number of lost data: $m \times L$.

2. Non-intersection of blocks is ensured by interval constraints.

3. Determinism is achieved by pseudo-random filling for the reproduced sample.

Example.

For $N = 1000$, $L = 20$. Number of lost sequences: $1000/20 = 50$. Initial indices $s_1, s_2, ..., s_{50}$ are generated such that $|s_i - s_j| \geq 20 \; \forall \; i \neq j$.

Using the aforementioned approach to generating lost data and grouping (12), the generator settings were formed and given in Table 1.

Fig. 2 shows the change in coordinates over time, while Fig. 3 demonstrates a variant of the set with lost data, where $N = 500$, $L = 50$.

Table 1

Lost data generation settings values

Number of missing values N	Sequence length of missing values L	Number of elements in the group
120	10	20
250	20	10
500	25	0
500	30	0
500	50	0
700	10	0
700	30	0
700	70	0

4. 7. Metrics

For the comparative analysis of filling in missing GPS values, typical metrics for assessing the accuracy and reliability of forecasting in time series were used. The selected metrics (18) to (20) provide a comprehensive analysis of various aspects of forecasting performance

$$MSE = \sum_{i=1}^{n} \frac{(y_i - x_i)2}{n}. \quad (18)$$

The mean squared error (MSE) emphasizes larger deviations by squaring the errors before averaging, making it particularly sensitive to outliers and large discrepancies in forecasts

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(y_i - x_i)2}{n}}. \quad (19)$$

The root mean square error (RMSE) converts MSE to the original coordinate units, which facilitates intuitive interpretation while maintaining sensitivity to larger errors. RMSE provides a balanced estimate of the accuracy of the interpolation, taking into account both systematic and random components of the prediction error

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - x_i|. \quad (20)$$

The mean absolute error (MAE) is the average of the error magnitudes in the predictions, providing easy interpretability in the original coordinate units. MAE linearly scales all deviations, allowing for an assessment of the absolute accuracy of the interpolation.

5. Results of zero-shot GPS data reconstruction with large language models

5. 1. Choosing the optimal GPS coordinate representation format

To assess the impact of data representation on the accuracy of GPS data reconstruction, a comparative analysis was conducted using three variants of the VPAIR dataset: the original degree-based format (VPair), a version converted to Earth-centered coordinates (ECEF) (VPair ECEF), and a shifted variant (VPair Shifted). The process involved shifting coordinate values to zero to reduce the instability associated with accuracy, which is a critical adjustment given the sensitivity of degree-based GPS data to small decimal errors. For the baseline comparison, missing values were imputed using the simple mean method, which served as a benchmark for assessing the improvements achieved by the large language models. Experiments were conducted on three leading large language models – Llama 3.2, Gemma, and GPT-4o – to ensure model-independent validity, with the results aggregated and averaged across all models. Fig. 4 illustrates the percentage improvement in reconstruction accuracy for altitude, latitude, and longitude relative to the baseline model.

Tables 2–4 compare three LLMs in latitude, longitude, and altitude under three preprocessing regimes.

Table 2

Average performance of tested models on the VPair set

Model	Latitude			Longitude			Altitude		
	mae	mse	rmse	mae	mse	rmse	mae (10^3)	Mse (10^5)	Rmse (10^4)
gemma:7b	**7.55**	0.111	**30.6**	**11.5**	**0.274**	**52.2**	**7.1**	**2.16**	**4.06**
gpt-4o	17.8	0.246	49.5	35.3	0.786	88.4	45.4	18.9	13.6
llama3.2	16.2	**0.102**	220	12.3	2.96	99.6	34.5	109	23.9

Table 3

Average performance of tested models: on the Vpair ECEF set

Model	Latitude			Longitude			Altitude		
	mae	Mse (10^5)	rmse	mae	Mse (10^5)	rmse	mae	Mse (10^5)	rmse
gemma:7b	**58.8**	**7.94**	**282**	153	26.6	516	**79.8**	**6.20**	**249**
gpt-4o	147.0	17.2	415	192	29.7	545	11.1	12.2	350
llama3.2	61.5	34.9	425	**60.7**	**16.0**	**368**	100.0	77.4	717

The shifted dataset (VPair Shifted) demonstrated consistent superiority in two out of three geospatial dimensions. Giv-

en these results, further research is focused on this variation of the dataset. This solution fits the broader goals of untrained LLM applications, where minimizing preprocessing complexity while maximizing robustness to errors is a top priority.

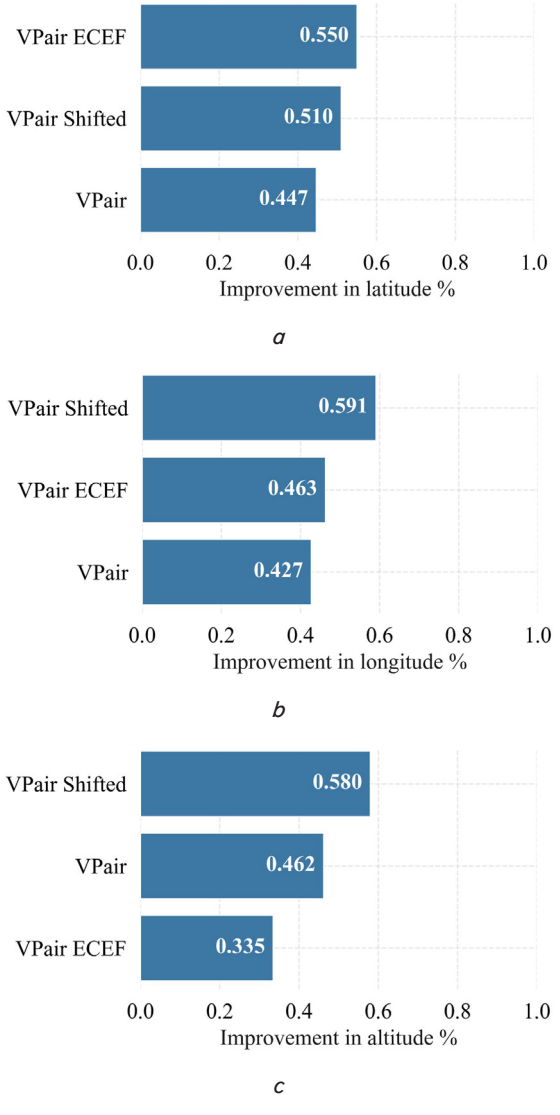

Fig. 4. Spatial coordinates after generating missing values: *a* — latitude; *b* — longitude; *c* — altitude

Table 4

Average performance of those tested on the VPair Shifted set

Model	Latitude			Longitude			Altitude		
	mae	Mse (10^4)	rmse	mae	Mse (10^4)	rmse	mae	Mse (10^4)	rmse
gemma:7b	96.0	13.1	358	114.0	20.2	447	79.2	**8.68**	**292**
gpt-4o	124.0	15.2	389	183.0	27.8	525	90.3	8.31	280
llama3.2	**41.0**	**12.1**	**264**	**57**	**11.5**	**320**	**54.3**	3.18	312

5. 2. Determining the optimal instruction variant

Effective prompt design is critical for LLMs that will be used without training, especially in accuracy-sensitive tasks such as geospatial data reconstruction, where ambiguous instructions can exacerbate error propagation. The investigated prompts were clearly structured around the task of reconstructing sequences containing *NaN* values, with the requirement: replace only a specified number of missing values (nan_amount).

As shown in Table 5, each prompt candidate included these constraints but differed in the details. For example, Prompt 1 – generated via ChatGPT – included an explicit I/O example ("[4915856.190, nan, 4915908.055]" → "[4915856.190, 4915882.090, 4915908.055]") to enforce both nan substitution and sequence length.

Table 5

Examples of prompt candidates

Prompt	Context
Prompt 1	You are a system designed to accurately predict and replace missing values ('nan') in GPS sequences in the Earth-centered, Earth-fixed coordinate system. Replace 'nan' values only with accurate predictions based on patterns in the provided sequence. The output must contain exactly {nan_amount} values. Any deviation from this length is unacceptable. Respond only with the corrected sequence in square brackets, with values separated by commas. For example, if the input is [4915856.190, nan, 4915908.055], the output should be [4915856.190, 4915882.090, 4915908.055]. Do not include explanations, comments, or any extra text only the corrected sequence in the specified format. Here is the input data: {content}. The output must contain exactly {nan_amount} values. Recalculate and regenerate predictions until this requirement is met
Prompt 2	You are a system designed to predict and replace missing values ('nan') in GPS sequences in Earth-centered, Earth-fixed coordinate system. Follow these strict rules:1. Replace 'nan' values ONLY with accurate predictions based on the given sequence. 2. The output MUST have exactly N values, where N is specified. Any deviation from this length is incorrect. 3. Respond ONLY with the corrected values in square brackets, separated by commas. 4. DO NOT include explanations, extra text, or code in your response only the corrected values. Here is the input data:\n{content}The output MUST contain exactly {nan_amount} corrected values. Re-calculate and regenerate until the response has {nan_amount} values
Prompt 3	You are a system designed to predict and replace missing values ('nan') in GPS sequences in Earth-centered, Earth-fixed coordinate system. Follow these strict rules: Replace 'nan' values ONLY with accurate predictions based on the given sequence. Respond ONLY with the corrected values in square brackets, separated by commas. Example of input: [4915856.190, nan, 4915908.055] Example of prediction: [4915856.190, 4915882.09, 4915908.055]. DO NOT include explanations, extra text, or code in your response only the corrected values. Here is the input data:{content} The output MUST contain exactly {nan_amount} corrected values. Re-calculate and regenerate until the response has {nan_amount} values
Prompt 4	1. You are a system designed to predict and replace missing values ('nan') in GPS sequences in Earth-centered, Earth-fixed coordinate system. Follow these strict rules:. Replace 'nan' values ONLY with accurate predictions based on the given sequence. The output MUST have exactly N values, where N is specified. Any deviation from this length is incorrect. 3. Respond ONLY with the corrected values in square brackets, separated by commas. Example of input: [4915856.190, nan, 4915908.055] Example of prediction: [4915856.190, 4915882.09, 4915908.055]. DO NOT include explanations, extra text, or code in your response only the corrected values. Here is the input data:\n{content}. The output MUST contain exactly {nan_amount} corrected values. Re-calculate and regenerate until the response has {nan_amount} values

Prompt 4, which was hand-crafted, was reduced to concise rules, retaining the example, while Prompts 2–3 omitted or simplified the instructions. Importantly, all prompts emphasized the need to restore predictions to the nan_amount condition, which provided strict control over

preserving the integrity of the dataset. In case of a mismatch between predicted and missing values, re-prompts were used.

Table 6 collates the performance by latitude, longitude, and altitude using various performance metrics.

Based on the results of this set of experiments, Prompt 4 was selected for further development of an approach to recovering lost GPS data.

Table 6

Prompt candidate productivity

Prompt	Latitude			Longitude			Altitude		
	mae	mse (10^4)	rmse	mae	mse (10^4)	rmse	mae	mse (10^4)	rmse
Prompt 1	33.9	4.9	**203**	87.0	18.2	422	57.6	7.15	264
Prompt 2	39.2	51.2	401	72.2	15.3	386	50.8	6.44	253
Prompt 3	50.1	5.48	228	52.1	9.95	304	**22.9**	**2.10**	**134**
Prompt 4	**27.9**	**4.29**	205	**33.8**	**5.63**	**237**	43.2	3.29	180

5. 3. Comparative assessment of the quality of recovery for different language models

The evaluation system included strict measures to account for two main sources of stochasticity: variability in the generation of sequences of missing values and the inherent diversity of predictions in autoregressive models. To reduce the influence of the first factor – random patterns of missing values that affect the complexity of the task – the results were aggregated across several independent runs for each model, with the average values of performance metrics (e.g., MAE) to ensure reliability. The boxplots (Fig. 5) visualize the distribution of MAE values for latitude, longitude, and altitude, highlighting central tendencies (mean, median) and ranges of variability.

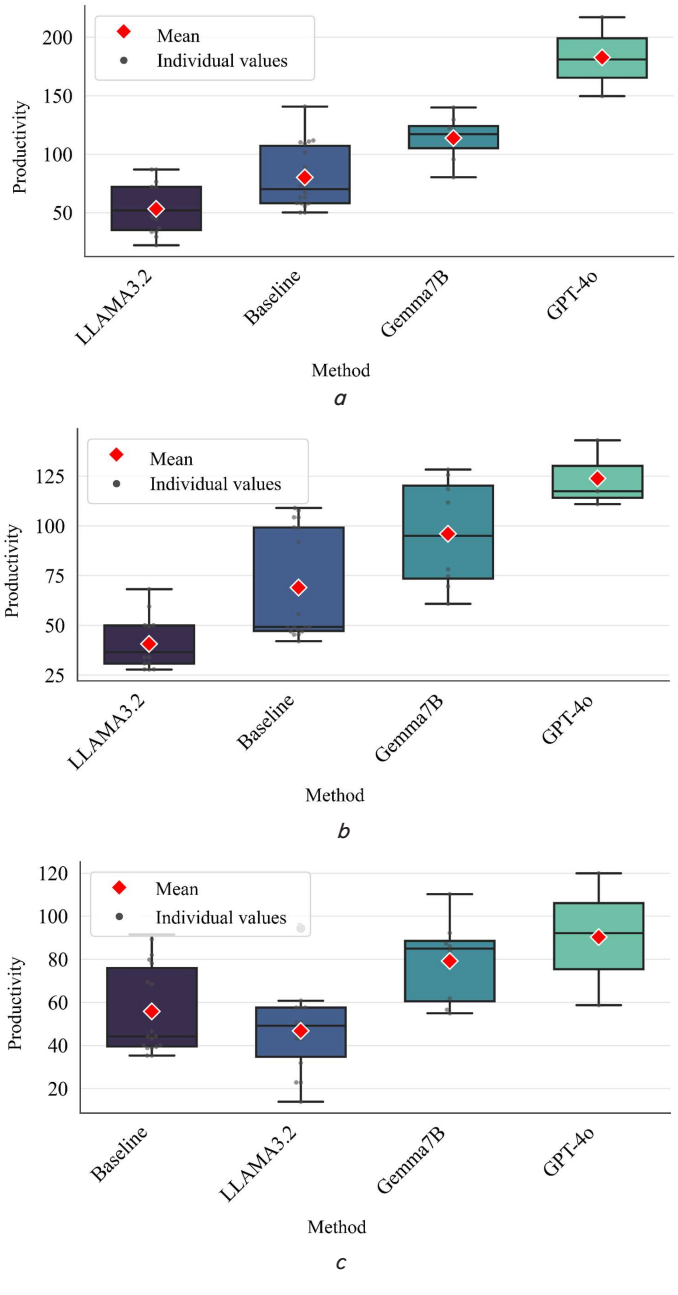

Fig. 5. Model results in terms of geodetic variables: *a* – latitude; *b* – longitude; *c* – height

A key factor in the application of autoregressive LLMs is the inherent variability of the predictions due to stochastic generation. For quantitative measurement, the stability of the models was assessed by performing 5–7 parallel inference iterations for each sample and measuring the variance of the output across latitude, longitude, and altitude. Fig. 6 demonstrates the trade-off between mean efficiency (MAE) and variance.

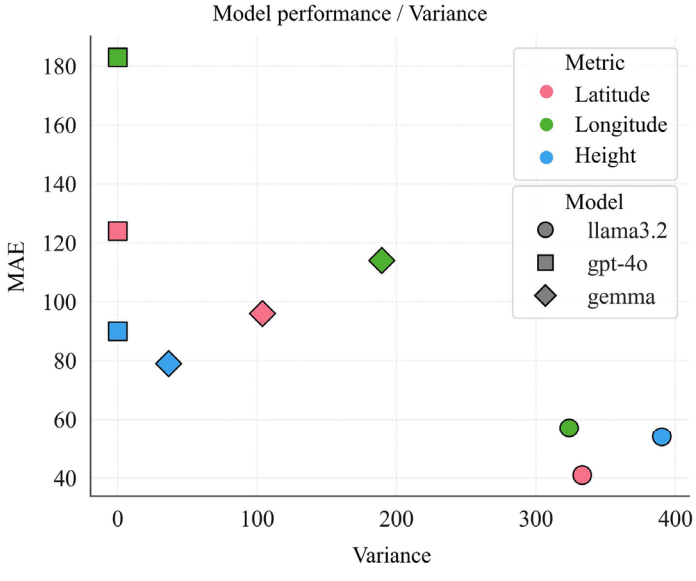

Fig. 6. Comparing model performance with its stability

Although variability is a potential problem, it was reduced by averaging the results. Aggregation across parallel runs helps detect hallucinations and reduce the impact of outliers. Combining with intact context further increases the robustness of the results.

6. Discussion of results based on zero-shot imputation of GPS data using LLM

Our experimental results (Fig. 4) clearly show the advantage of moving away from the use of a degree representation of the data. Thus, for the VPair Shifted set, an improvement of 58.0% was achieved for the reconstruction of the height, which significantly exceeded the indicators of the original VPair (46.2%) and VPair ECEF (33.5%). Similarly, for the longitude, VPair Shifted achieved an improvement of 59.1%, which exceeded the indicators of VPair (42.7%) and VPair ECEF (46.3%). The reconstruction of the latitude showed a different pattern: VPair ECEF achieved the highest improvement (55.0%), slightly exceeding VPair Shifted (51.0%), while the original VPair demonstrated 44.7%. These results highlight the relationship between the preprocessing of the data set and the geometric properties of the coordinates.

Analysis of our experimental results (Table 4) reveals that the optimal formulation of the instruction significantly affects the accuracy of the reconstruction. Prompt 4 achieved better accuracy in recovering latitude (MAE: 27.9, MSE: 4.29E+04), which is explained by the balance of clear rules and illustrative format. In contrast, the rigid structure of Prompt 1 led to an increased MSE for longitude (18.2E+04), indicating that excessive constraints inhibit adaptability. It is noted that Prompt 3 achieved good results for altitude (MAE: 22.9), probably due to its conciseness and inclusion of a guided example but showed a deterioration in the results for latitude (MAE: 50.1), indicating sensitivity to specific coordinates. The striking differences in performance demonstrate that precise engineering of prompts – in particular, explicit formulation of *NaN* handling and precise value counter – directly affects the reliability of reconstruction. These results indicate the advantage of using prompts that combine task specificity (e.g., nan replacement rules) with flexibility, allowing LLM to combine accuracy with contextual recovery of missing data. The results also confirm that explicit specification of the number of missing values and formatting in prompts is not only useful but also necessary for reliable reconstruction without prior training in GPS applications.

Unlike [20], in which recurrent neural networks are trained on labeled time series with gaps, the zero-shot LLM approach does not require any training. The proposed method uses only the prompt formulation to directly generate missing values in GPS trajectories. This is made possible by the ability of LLM to interpret a digital sequence in which the context of neighboring coordinates is encoded by tokens, and to model nonlinear spatiotemporal dependences without a specific architecture.

Compared to [21], convolutional-recurrent autoencoders spend significant resources on hyperparameter tuning. In addition, they require large, labeled sets for simultaneous extraction of spatial and temporal features. The proposed approach performs zero-shot imputation. In this case, prompt serves as an "instruction" for restoring coordinates. This is possible because LLMs trained on huge corpora already have internal mechanisms for predicting numeric tokens taking into account the context.

In contrast to [12], in which LLMs were used only for zero-shot prediction of time series, our approach specifically optimizes prompt for GPS data imputation. This approach makes it possible to fill spatiotemporal gaps in trajectories, rather than just predicting the next values. This is possible because we form prompt as a description of the spatiotemporal context, which allows LLMs to understand neighboring points and generate the most likely value of the gap.

While [27] uses the simple mean method as a basic strategy and obtains averaged "plateaus" over long gaps, the proposed method reconstructs the trajectory using the context of neighboring GPS coordinates. Thus, the Llama 3.2 model for longitude (Fig. 5) demonstrated the best performance, achieving the lowest median MAE value (median 52.14 units; IQR: 35.29–72.17), outperforming the mean method (median: 70.18 units; IQR: 58.2–107.24) and GPT-4o (median: 182.65 units; IQR: 165.42–199.09). Gemma performed slightly better (median: 117.17 units) and had a narrower range (IQR: 105.18–124.12). The mean method was most sensitive to sequence complexity. In terms of width, Llama 3.2 performed best with the lowest median MAE (36.57 units; IQR: 30.82–49.95), followed by the baseline method (median: 49.3 units; IQR: 47.21–99.17), and Gemma lagged significantly behind (median: 95 units; IQR: 73.52–120.31). The GPT-4o model performed worst (median: 117.41 units; IQR: 114.21–130.21). A different trend was found for altitude: The baseline method showed (median: 44.2 units; IQR: 39.57–75.96), Llama 3.2 – (median: 49.17 units; IQR: 34.76–57.61), Gemma – (median: 85.02 units; IQR: 60.55–88.6) and GPT-4o – (median: 90.2 units; IQR: 75.48–106.08). Analysis highlights the different performance profiles between the models, due to their robustness to

stochastic variables. Llama 3.2 was the most robust overall, achieving high results for longitude and latitude, although it was inferior to the baseline method for altitude. The baseline method, while competitive for altitude, showed increased sensitivity to the complexity of missing values for longitude and latitude, suggesting limited adaptability. Gemma demonstrated moderate stability for longitude but had significant difficulty for latitude and altitude. GPT-4o showed universally poor results, suggesting problems with generalizing geospatial patterns.

Based on our analysis of the variance of predictions (Fig. 6), the GPT-4o model showed the most stable reconstructions. This is explained by its deterministic caching of results, which eliminates stochastic randomness. However, this was accompanied by increased MAE values, suggesting that strict adherence to cached responses limits adaptive predictions. In contrast, the Gemma and Llama 3.2 models showed a gradual increase in variability. Gemma showed moderate variance but inconsistent results, reflecting the sensitivity of its stochastic intermediate layers to changes in input data. Llama 3.2 demonstrated the highest variance, especially in longitude, probably due to its flexible token sampling strategy. It was noted that this variability correlated with its high median MAE in latitude, indicating a trade-off: exploring a wider solution space increases both the potential for prediction and the instability.

The limitation of the proposed approach is the need to call a large language model for each gap or window of several points. Although LLM inference is faster than full training, processing large GPS datasets in real time requires significant computational resources. Therefore, such an approach is advisable to consider mainly as a post-processing stage, and not as a direct component of the real-time pipeline in constrained environments.

The disadvantage of the proposed approach is the sensitivity of LLM to the length of the input sequence. In the case of long gaps (>30 points), LLM sometimes focuses on the general trend of the local context, ignoring small variations, which leads to averaged trajectories. This reduces the accuracy of the recovery and demonstrates the need for a differentiated approach to processing different gap lengths.

Further research may aim at devising an adaptive approach to prompt window generation. Automatic determination of the number of contextual GPS points depending on the sequence length. In addition, it is advisable to experiment with prompt engineering for short and long gaps separately to maximize the reconstruction accuracy in each case. Expanding the application domain of zero-shot LLM to other types of spatiotemporal data. For example, vehicle trajectories or meteorological phenomena dynamics. In combination with the integration of multimodal inputs, it could contribute to increasing the robustness of imputation under complex conditions.

7. Conclusion

1. Three strategies for representing coordinates have been implemented. Further analysis revealed that the most effective strategy is the ECEF Shifted strategy, where the coordinates are shifted using a shift to the starting point. This approach reduces the error by 51–59% for latitude and longitude, which demonstrates its effectiveness in restoring GPS data.

2. The optimal option for the initial instruction (prompt) was selected for the zero-shot method of reconstructing missing GPS data using large language models. A re-query mechanism was also implemented, which makes it possible to reduce the probability of inaccurate predictions in the event that the number of restored coordinates does not meet expectations. The re-query mechanism provides reliability in data restoration, which is important for complex scenarios where the accuracy of coordinates is critical.

3. A comparative assessment of the accuracy of different language models has shown that Llama 3.2 provides the best results for GPS data reconstruction, outperforming GPT-4o and Gemma. For latitude, Llama 3.2 achieved MAE = 36.57, and for longitude, MAE = 52.14, which are 2.3 times better results compared to GPT-4o. This can be explained by the better spatial thinking of the model, which is provided by specialized query tuning. Such results allow us to conclude that open models are superior in geospatial analytics tasks, where flexibility and adaptability are critical.

Conflicts of interest

The authors declare that they have no conflicts of interest in relation to the current study, including financial, personal, authorship, or any other, that could affect the study, as well as the results reported in this paper.

Funding

The study was conducted without financial support.

Data availability

The manuscript has associated data in the data warehouse.

Use of artificial intelligence

The authors used artificial intelligence technologies within acceptable limits to provide their own verified data, which is described in the research methodology section.

References

1. Chang, Y., Cheng, Y., Manzoor, U., Murray, J. (2023). A review of UAV autonomous navigation in GPS-denied environments. Robotics and Autonomous Systems, 170, 104533. https://doi.org/10.1016/j.robot.2023.104533
2. Khan, S. Z., Mohsin, M., Iqbal, W. (2021). On GPS spoofing of aerial platforms: a review of threats, challenges, methodologies, and future research directions. PeerJ Computer Science, 7, e507. https://doi.org/10.7717/peerj-cs.507
3. Gao, Q., Molloy, J., Axhausen, K. W. (2021). Trip Purpose Imputation Using GPS Trajectories with Machine Learning. ISPRS International Journal of Geo-Information, 10 (11), 775. https://doi.org/10.3390/ijgi10110775

4. Zhang, S., Gong, L., Zeng, Q., Li, W., Xiao, F., Lei, J. (2021). Imputation of GPS Coordinate Time Series Using missForest. Remote Sensing, 13 (12), 2312. https://doi.org/10.3390/rs13122312
5. Liu, H., Li, L. (2022). Missing Data Imputation in GNSS Monitoring Time Series Using Temporal and Spatial Hankel Matrix Factorization. Remote Sensing, 14 (6), 1500. https://doi.org/10.3390/rs14061500
6. Floridi, L., Chiriatti, M. (2020). GPT-3: Its Nature, Scope, Limits, and Consequences. Minds and Machines, 30 (4), 681–694. https://doi.org/10.1007/s11023-020-09548-1
7. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L. et al. (2023). GPT 4 Technical Report. arXiv. https://doi.org/10.48550/arXiv.2303.08774
8. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T. et al. (2023). LLaMA: Open and Efficient Foundation Language Models. arXiv. https://doi.org/10.48550/arXiv.2302.13971
9. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y. et al. (2023). LlaMA 2: Open foundation and finetuned chat models. arXiv. https://doi.org/10.48550/arXiv.2307.09288
10. Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., AlDahle, A. et al. (2024). The Llama 3 herd of models. arXiv. https://doi.org/10.48550/arXiv.2407.21783
11. Riviere, M., Pathak, S., Sessa, P. G., Hardin, C., Bhupatiraju, S., Hussenot, L. et al. (2024). Gemma 2: Improving open language models at a practical size. arXiv. https://doi.org/10.48550/arXiv.2408.00118
12. Gruver, N., Finzi, M., Qiu, S., Wilson, A. G. (2023). Large language models are zeroshot time series forecasters. NeurIPS 36, 19622–19635. https://doi.org/10.48550/arXiv.2310.07820
13. Gu, Y., Han, X., Liu, Z., Huang, M. (2022). PPT: Pre-trained Prompt Tuning for Few-shot Learning. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin: Association for Computational Linguistics, 8410–8423. https://doi.org/10.18653/v1/2022.acl-long.576
14. Tsmots, I., Teslyuk, V., Opotyak, Y., Rabyk, V. (2023). Intelligent motion control system for the mobile robotic platform. 7th International Conference on Computational Linguistics and Intelligent Systems. Kharkiv: CEUR Workshop Proceedings. Available at: https://ceur-ws.org/Vol-3403/paper42.pdf
15. Wang, Y., Feng, Z., Zhang, H., Gao, Y., Lei, J., Sun, L. et al. (2024). Angle Robustness Unmanned Aerial Vehicle Navigation in GNSS-Denied Scenarios. Proceedings of the AAAI Conference on Artificial Intelligence, 38 (9), 10386–10394. https://doi.org/10.1609/aaai.v38i9.28906
16. Junninen, H., Niska, H., Tuppurainen, K., Ruuskanen, J., Kolehmainen, M. (2004). Methods for imputation of missing values in air quality data sets. Atmospheric Environment, 38 (18), 2895–2907. https://doi.org/10.1016/j.atmosenv.2004.02.026
17. van Buuren, S., Groothuis-Oudshoorn, K. (2011). mice: Multivariate Imputation by Chained Equations in R. Journal of Statistical Software, 45 (3), 1–67. https://doi.org/10.18637/jss.v045.i03
18. Oliver, M. A., Webster, R. (2014). A tutorial guide to geostatistics: Computing and modelling variograms and kriging. CATENA, 113, 56–69. https://doi.org/10.1016/j.catena.2013.09.006
19. Murti, D. M. P., Pujianto, U., Wibawa, A. P., Akbar, M. I. (2019). K-Nearest Neighbor (K-NN) based Missing Data Imputation. 2019 5th International Conference on Science in Information Technology. Yogyakarta: IEEE, 83–88. https://doi.org/10.1109/icsitech46713.2019.8987530
20. Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y. (2018). Recurrent Neural Networks for Multivariate Time Series with Missing Values. Scientific Reports, 8 (1). https://doi.org/10.1038/s41598-018-24271-9
21. Asadi, R., Regan, A. C. (2019). A convolution recurrent autoencoder for spatiotemporal missing data imputation. arXiv. https://doi.org/10.48550/arXiv.1904.12413
22. Nag, P., Sun, Y., Reich, B. J. (2023). Spatio-temporal DeepKriging for interpolation and probabilistic forecasting. Spatial Statistics, 57, 100773. https://doi.org/10.1016/j.spasta.2023.100773
23. You, J., Ma, X., Ding, D. Y., Kochenderfer, M., Leskovec, J. (2020). Handling missing data with graph representation learning. NIPS'20, 19075–19087. https://doi.org/10.48550/arXiv.2010.16418
24. Chorowski, J., Bahdanau, D., Serdyuk, D., Cho, K., Bengio, Y. (2015). Attentionbased models for speech recognition. NIPS'15, 577–585. https://doi.org/10.48550/arXiv.1506.07503
25. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gómez, A. N. et al. (2017). Attention is all you need. NIPS'17, 6000–6010. https://doi.org/10.48550/arXiv.1706.03762
26. Schleiss, M., Rouatbi, F., Cremers, D. (2022). VPAIR – Aerial visual place recognition and localization in large-scale outdoor environments. arXiv. https://doi.org/10.48550/arXiv.2205.11567
27. Enge, P. K. (1994). The Global Positioning System: Signals, measurements, and performance. International Journal of Wireless Information Networks, 1 (2), 83–105. https://doi.org/10.1007/bf02106512