Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.080739

ARTICLE

Semi-Automated Generation of Realistic Simulation Environments from Geospatial Data for Agricultural Robot Navigation

Sergio Sánchez de la Fuente*, Luis Prieto-López, Miguel Á González-Santamarta, Vicente Matellán-Olivera and Ángel Manuel Guerrero-Higueras

Grupo de Robótica, Universidad de León, León, Castilla y León, Spain

*Corresponding Author: Sergio Sánchez de la Fuente. Email: ssand@unileon.es

Received: 13 February 2026; Accepted: 23 April 2026; Published: 27 May 2026

ABSTRACT: The development and testing of autonomous agricultural robots requires realistic simulation environments that accurately represent field conditions and terrain features. Traditional manual scenario creation is time-consuming, expensive and limits the diversity of testing conditions. This paper presents an integrated two-stage system for semi-automated generation of realistic 3D simulation scenarios. The first stage transforms publicly available geospatial data into high-fidelity 3D terrain models, supporting 23 discrete levels of detail (LoD), from 0 to 22, and generating simulation-ready models compatible with the Gazebo robotics simulator. The second stage provides a web-based tool that enables users to populate generated terrains with crop elements, configuring crop distributions, row patterns, and field geometries through a map interface. Detailed performance evaluation across multiple LoD levels identifies the optimal balance between visual fidelity and computational efficiency, with levels 19–20 providing sufficient geometric detail for accurate sensor simulation while maintaining real-time performance on standard hardware. The complete system integrates with the Robot Operating System (ROS) 2 and Gazebo, significantly reducing scenario creation time by eliminating the need for manual modelling of terrain and agricultural elements. Validation experiments were conducted using a Summit-XL robotic platform equipped with an additional RealSense depth camera. The results demonstrate the system's capability for developing and testing autonomous navigation algorithms in the generated scenarios.

KEYWORDS: Robotic simulation; scenario generation; geospatial data; agricultural robotics; level of detail; autonomous navigation; terrain modelling

1 Introduction

Autonomous agricultural robots require testing in diverse field conditions before deployment [1–3]. Field trials face limitations including seasonal constraints, equipment costs, safety concerns, and the difficulty of testing failure scenarios. Simulation environments address these limitations by enabling controlled, repeatable testing without the constraints of physical experiments [4]. However, creating realistic simulation environments that accurately represent real-world terrain topology and geometric features remains a significant challenge.

Simulation scenario creation traditionally involves manual 3D modelling using software such as Blender or Maya. This approach requires multiple days of work per scenario and specialised people, making it impractical for generating diverse test environments needed for algorithm testing. This process includes terrain modelling, texture mapping, material property assignment, and coordinate system configuration.

Thus, each scenario may require dozens of hours of skilled labor, limiting the number of test environments that can be feasibly created. Procedural generation methods can automate scenario creation but typically produce generic environments that lack correspondence to real geographic locations. This limitation becomes significant when testing Global Positioning System (GPS)-based navigation systems or validating simulation results against field-collected data from specific sites.

On the other hand, geospatial data from satellite imagery and elevation measurements [5] offers an alternative approach for creating realistic simulation environments. This data, available worldwide from government and commercial sources through services such as Azure Maps Application Programming Interface (API)[1], provides detailed information about real terrain surfaces, field boundaries, and geographic features. The data is typically organised in hierarchical levels of detail (LoD), with LoD 0 representing global-scale coverage and higher levels providing increasing spatial resolution. Transforming this data into robotics simulation environments requires processing pipelines that convert raw formats like GL Transmission Format (glTF)[2] into simulation-compatible formats such as COLLAborative Design Activity (COLLADA) for platforms like Gazebo [6] and Robot Operating System (ROS) 2 [7]. The processing must maintain coordinate accuracy for GPS sensor simulation, handle large geometric datasets efficiently through tools like Blender for mesh optimisation, and generate terrain models at appropriate levels of geometric detail to balance visual fidelity with computational performance.

For agricultural robotics specifically [8,9], precision agriculture increasingly depends on accurate geospatial representations of real field conditions. While geospatial data provides realistic terrain models, the representation of vegetation from these sources lacks the detail necessary for testing perception systems and navigation algorithms. Satellite imagery and elevation data capture terrain topology but do not include individual crop models with the appropriate geometric complexity required for sensor simulation. Agricultural fields present irregular surfaces from cultivation practices and natural topography. Crops exhibit variability in height, orientation, and spacing. Vegetation creates visual patterns that affect perception systems when distinguishing between crop rows, weeds, and soil. This limitation becomes critical when developing algorithms for crop row detection, furrow following, or obstacle avoidance in agricultural fields [10,11]. Existing agricultural simulation systems either use simplified geometric representations or require manual placement of individual crop models, neither of which scales to large field scenarios with thousands of plants. Recent advances in agricultural digital twins [12,13] have further increased the demand for simulation environments that are not only visually realistic but also geographically exact, enabling algorithm validation that transfers reliably to specific real-world deployments.

This paper addresses these limitations by presenting an integrated semi-automated framework that combines GPS-accurate geospatial terrain reconstruction with parametric crop-field population into a single coherent pipeline compatible with ROS 2 and Gazebo. The three main system-integration contributions are a terrain processing pipeline that preserves GPS coordinate fidelity across 23 configurable LoD levels, with systematic performance characterisation to guide practical configuration choices, a web-based configuration tool that replaces manual 3D crop modelling with a brief structured parametric setup step, and end-to-end deployment validated on a real agricultural robotic platform, demonstrating practical applicability for autonomous navigation research. The novelty of the proposed system resides in its capability to fuse accurate topographic environments with crop-level geometric complexity. Unlike other frameworks that only model terrain without field details or focus solely on procedural agricultural fields over flat surfaces, this work bridges both domains into an integrated pipeline.

[1] https://azure.microsoft.com/en-us/products/azure-maps
[2] https://www.khronos.org/gltf/

The proposed two-stage workflow begins with semi-automated terrain generation from geospatial sources, followed by interactive agricultural population through map-based controls. Although the initial data extraction and all terrain transformation steps execute without manual intervention, the agricultural population stage provides a guided parametric interface through which users define planting polygons and configure crop layout parameters. Once specified, the scenario is compiled and deployed fully automatically. This design deliberately replaces hours of manual 3D modelling with a brief, structured configuration step. This design enables users to configure crop distributions with variable row spacing, plant density, field geometries, and growth stages, while preserving the GPS-accurate coordinates established during terrain generation. The resulting complete scenarios combine realistic topography from satellite data with user-defined crop layouts, providing comprehensive testing environments for mobile robot navigation algorithms in agricultural settings.

The remainder of this paper is organised as follows. Section 2 reviews related work in scenario generation approaches, geospatial data utilisation, and agricultural robotics simulation. Section 3 describes the system architecture, detailing the terrain generation pipeline, agricultural scenario tool, and simulation deployment framework. Section 4 presents the experimental validation methodology, including hardware and software configuration, test scenarios, evaluation metrics, and experimental procedures. Section 5 presents experimental results, including level of detail analysis, visual quality comparison, resource consumption measurements for both terrain generation and agricultural scenarios, and validation of crop distribution and height interpolation capabilities. Section 6 interprets the results, analyzing performance trade-offs and providing optimal configuration recommendations. Finally, Section 7 summarises the key contributions and discusses future research directions.

2 Related Work

In recent years, robotic simulation technologies have become a significant research topic in both academic and industrial fields, particularly for agricultural automation. Realistic simulation environments, such as those provided by Gazebo [6], CARLA [14], and AirSim [15], are essential for developing and validating autonomous vehicles capable of operating safely in complex outdoor environments. These systems offer significant advantages in terms of cost reduction, accelerated development cycles, and comprehensive testing capabilities before field deployment. The literature emphasizes that high-fidelity simulation can significantly improve algorithm robustness by enabling extensive testing across diverse environmental conditions that would be impractical or impossible to replicate in physical trials. Therefore, many research institutions and technology companies are increasing their investments in simulation-based development methodologies, including digital twin concepts for agricultural systems [12,13,16,17], and conducting multidisciplinary studies in geospatial data integration, procedural generation, and agricultural scenario modelling.

To achieve realistic simulation environments, researchers integrate data from multiple sources, including satellite imagery, elevation models, vegetation databases, and cadastral maps. These data are processed through specialised pipelines involving format conversion, mesh optimisation, texture mapping, and coordinate transformation to generate simulation-ready 3D environments [4]. This process requires the integration of many technological components, such as geospatial data processing, procedural modelling, level-of-detail management, and physics-based rendering.

2.1 Terrain Generation and 3D Reconstruction Techniques

When reviewing studies based on geospatial data processing for robotic simulation, applications involving terrain reconstruction from elevation models, satellite imagery integration, and automated mesh generation stand out. Abbyasov et al. [18] proposed an automatic tool for Gazebo world construction that converts grayscale heightmap images into 3D COLLADA models. The LIRS-RSEGen system processes 2D elevation data through a multi-stage pipeline, generating textured terrain meshes directly compatible with physics simulators. Validation in outdoor environments demonstrated successful integration with ROS-based robotic platforms, though limitations in texture quality and geometric detail were noted for highly irregular terrain features.

Scholz [19] developed a proof-of-concept implementation for procedural terrain generation using model synthesis techniques. The approach employs example-based terrain generation to create large-scale landscapes from small input patches, incorporating Physically Based Rendering (PBR) for realistic material appearance. While demonstrating computational efficiency, the method struggles with preserving specific geographic features and maintaining coordinate accuracy required for GPS-based navigation testing. The study acknowledged that purely procedural approaches introduce a reality gap when simulating actual field locations, as stochastic generation cannot guarantee reproduction of real terrain characteristics.

These terrain reconstruction techniques provide the foundation for agricultural simulation. However, for agricultural applications specifically, terrain generation must account for field-scale topographic variations that affect robot traversability and sensor performance. Recent work on outdoor environment simulation has emphasised the importance of high-resolution elevation data [5] and accurate material properties for the realistic simulation of vision sensors, Light Detection and Ranging (LiDAR), and radar systems operating in natural settings. The integration of geospatial data with agricultural-specific elements thus requires specialised approaches that combine accurate terrain representation with crop-level modelling capabilities.

2.2 Procedural Generation and Domain Randomisation

In the study of automated scenario generation through procedural methodologies and deep learning–based approaches, tasks including environment synthesis, texture generation, and scene composition constitute the most prominent application domains. Tobin et al. [20] introduced domain randomisation as a technique for training deep neural networks that can transfer from simulation to real-world applications. By randomising visual appearance, lighting conditions, and object textures during the training process of perception models, the method enables trained neural networks to treat the real world as just another variation of the simulated environment. Experiments on object detection and robotic grasping demonstrated successful sim-to-real transfer, with models trained exclusively in randomised simulation achieving comparable performance to those trained on real data. Complementary approaches have extended sim-to-real transfer through dynamics randomisation for robotic locomotion [21] and learned visual representations for autonomous aerial navigation [22]. In the agricultural domain specifically, recent work has demonstrated that high-quality synthetic datasets generated in simulation can effectively train deep perception models for row-based crop navigation, significantly reducing the dependency on costly real-world data collection campaigns [23].

Advanced procedural generation techniques such as rule-based city generation [24], vegetation placement algorithms [25,26], and terrain synthesis [27] have been widely used to create diverse testing environments. However, these approaches face significant limitations when applied to agricultural robotics. Agricultural fields exhibit structured yet irregular patterns, crop rows with variable spacing and gaps, and terrain features resulting from cultivation practices that are difficult to capture through purely stochastic

generation. The deterministic nature of many agricultural layouts conflicts with randomisation-based approaches, requiring hybrid methods that combine procedural flexibility with geographic grounding.

2.3 Geospatial Data Integration and Map-Based Simulation

Several research efforts have explored integrating real-world geographic data into simulation environments to improve location-specific realism. OpenStreetMap (OSM)[3] has been utilised for robot navigation research, providing vector representations of roads, buildings, and land use. However, studies examining OSM accuracy have reported mean positional errors ranging from 5 to 15 m when compared to authoritative cadastral datasets, with significantly larger deviations in rural and agricultural areas where map coverage is sparse and update frequency is low [28]. These accuracy limitations constrain the utility of OSM for high-precision simulation requiring accurate GPS coordinate alignment.

Digital Elevation Models (DEMs) from sources such as the Shuttle Radar Topography Mission (SRTM) [5], the Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) and regional LiDAR surveys provide elevation data at various resolutions, typically ranging from 30 m to sub-meter precision. Processing DEM data into simulation-compatible formats requires coordinate system transformations [29], mesh generation from elevation grids, and texture mapping from aerial or satellite imagery. The Azure Maps API provides programmatic access to global satellite imagery and elevation data, enabling automated retrieval of terrain information for arbitrary geographic locations.

2.4 Agricultural Robotics Simulation Frameworks

Table 1 provides a comparative overview of the most relevant works in geospatially grounded agricultural robotics simulation, whose individual contributions and limitations are discussed below.

Simulation in agricultural robotics introduces domain-specific challenges, including crop-row detection, furrow following [30], variable terrain traversability, and perception in cluttered vegetation. The AgROS framework [31] enables geospatially accurate agricultural simulation by allowing users to select real field boundaries from map interfaces and automatically import corresponding terrain into ROS-based environments [32]. The system integrates elevation data, satellite imagery, and field boundary vectors to create site-specific simulation scenarios. While representing a significant advance in agricultural simulation accessibility, AgROS focuses primarily on terrain generation and lacks integrated tools for populating fields with crop models at appropriate spatial scales and geometric detail. Recent work on deep-learning-based crop row detection has reinforced the importance of realistic simulated environments for training and evaluating infield navigation algorithms [33].

Table 1: Summary of the most relevant works in agricultural robotics simulation and geospatial scenario generation.

Reference	Approach	Target Domain	Key Method	Limitations
Tobin et al. [20]	Domain randomisation	Object detection, grasping	Texture/geometry randomisation	Purely stochastic, lacks geographic grounding
Scholz [19]	Procedural generation	Generic terrain	Example-based synthesis, PBR	Cannot reproduce specific geographic locations

(Continued)

[3] https://www.openstreetmap.org/

Table 1 (continued)

Reference	Approach	Target Domain	Key Method	Limitations
OSM-based systems [28]	Vector map data	Urban navigation	Road network extraction	5–15 m positional errors, poor rural coverage
Abbyasov et al. [18]	Image-based reconstruction	General outdoor robotics	Heightmap to COLLADA conversion	Limited texture quality, manual heightmap creation
Tsolakis et al. [31]	Geospatial integration	Agricultural fields	Map interface, terrain import	No crop modelling, limited to terrain only
Forest3D [34]	Automated pipeline	Forested environments	DEM to Gazebo, procedural crops	Optimised for forests, not structured crops
Virtual Maize Field [35]	Procedural generation	Crop fields	Parametric row generation	Flat terrain only, no real-world coordinates
Afzal et al. [36]	Probabilistic programming	General simulation	Scenic language, spatial constraints	Requires manual scenario specification

Forest3D [34] addresses vegetation modelling by providing an automated pipeline from DEM data to populated outdoor environments. The framework processes elevation data through format conversion, generates base terrain meshes, and procedurally places tree and crop models based on land cover classification. Integration with Gazebo enables physics-based simulation of robots navigating forested environments, though the system is optimised for natural forests rather than structured agricultural crops.

For crop-specific scenarios, the `virtual_maize_field` package [35] offers specialised tools for generating maize fields with configurable parameters including row spacing, plant density, growth stage, and weed distribution. The ROS-native implementation enables direct integration with robot navigation stacks and supports randomised field generation for algorithm training. However, terrain is represented as flat ground planes, limiting applicability to fields with topographic variation. Complementary frameworks such as GzScenic [36] employ probabilistic programming to define spatial relationships and constraints, enabling automatic generation of diverse scenes while maintaining physical plausibility. These tools demonstrate the value of domain-specific modelling but lack integration with real-world geospatial data sources.

Furthermore, the proliferation of digital twin paradigms in global agriculture has demonstrated the value of replicating precise farm environments to track, simulate, and predict agronomic processes [12,13,16,17]. However, developing these comprehensive Digital Twins traditionally depends on extensive manual 3D modelling. Recent advances have investigated Large Language Model (LLM)-driven mechanisms to automatically parse textual descriptions into coherent 3D agricultural layouts [37]. Similarly, exploring intricate physical plant-robot interactions within standard simulators has driven the creation of specialised physics plugins utilising Cosserat rod models to evaluate non-rigid deformations during harvesting [38]. Despite these powerful isolated innovations in procedural logic and physical manipulation,

constructing an automated continuous pipeline that bridges real-world geospatial topology with explicit agricultural element deployments remains an unresolved challenge.

The comparative analysis in Table 1 highlights that existing approaches fall into three categories: systems like AgROS and Forest3D that provide geospatially accurate terrain generation without agricultural detail, tools such as `virtual_maize_field` that offer detailed crop modelling on simplified flat terrain, and generic procedural or randomisation-based methods (domain randomisation, GzScenic) lacking both geographic accuracy and agricultural specificity. No current system simultaneously provides geospatially accurate terrain reconstruction, flexible agricultural element modelling, and deployment-ready ROS 2/Gazebo integration. This gap constrains the development of robust agricultural robotics algorithms, as testing environments either lack realistic topography or fail to represent crop-level geometric complexity. To address this challenge, we propose an integrated semi-automated two-stage system that combines GPS-accurate geospatial terrain generation with interactive agricultural scenario configuration, eliminating the manual modelling bottleneck while maintaining the fidelity necessary for sensor-based navigation algorithm validation.

3 System Architecture

The proposed system implements a modular two-stage workflow consisting of three integrated components, a terrain generation pipeline that transforms geospatial data into simulation-ready 3D base environments, a web-based agricultural scenario tool for populating generated terrains with crop elements, and an automated simulation deployment framework. This architecture enables researchers to rapidly create complete agricultural testing scenarios without manual modelling expertise. The workflow begins with terrain generation from any worldwide location with available data, proceeds through interactive agricultural population via web interface, and concludes with automated deployment for robot navigation testing in ROS 2 and Gazebo environments. Fig. 1 provides a high-level overview of this workflow, while Fig. 2 details the individual processing stages within each phase.

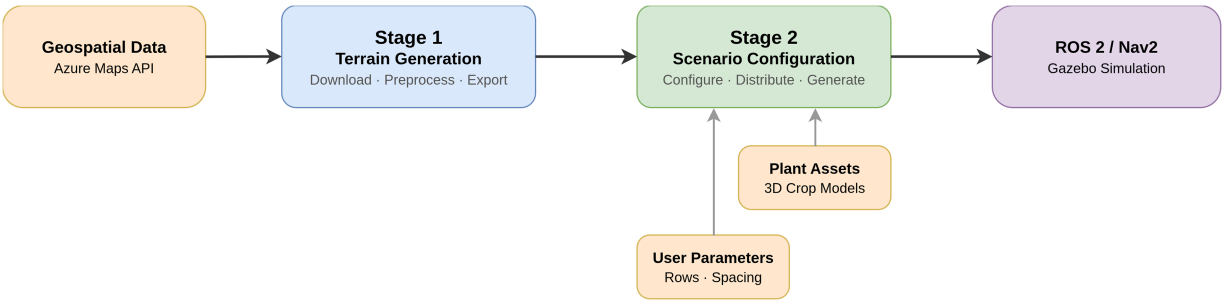


Figure 1: High-level overview of the two-stage system pipeline. Geospatial data from Azure Maps feeds into Stage 1, which downloads, preprocesses, and exports a COLLADA terrain model. Stage 2 combines this terrain with user-defined crop parameters and 3D plant assets to produce a deployable Gazebo world file for ROS 2/Nav2 simulation.

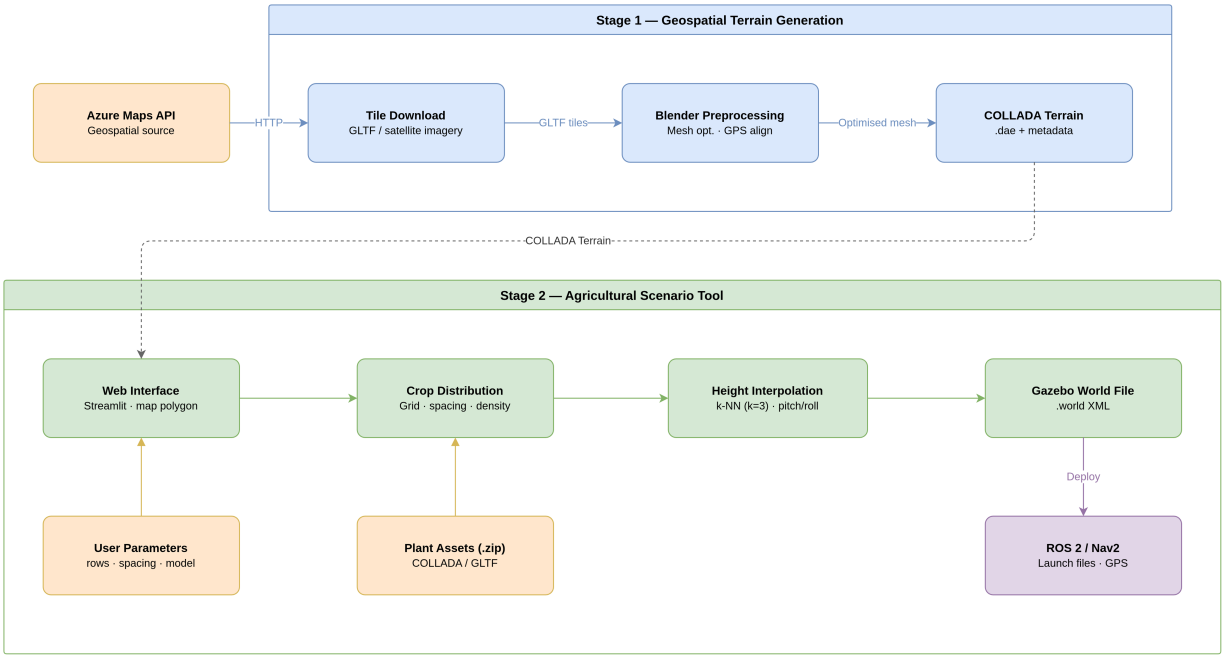


Figure 2: Detailed architecture of the pipeline introduced in Fig. 1. Node colours encode functional roles: blue for Stage 1 automatic processes (tile retrieval, Blender preprocessing, COLLADA export), green for Stage 2 processes (web-based configuration, crop distribution, k-NN height interpolation, world file generation), orange for external inputs (Azure Maps API, user parameters, plant assets), and purple for the final ROS 2/Nav2 simulation output.

3.1 Geospatial Terrain Generation

The terrain generation subsystem converts publicly available geospatial data into Gazebo-compatible 3D models through a multi-stage processing pipeline. The system architecture comprises four primary components: data acquisition, preprocessing, format conversion, and optimisation. This modular design enables flexible scenario generation while maintaining computational efficiency and visual fidelity across diverse geographic contexts.

3.1.1 Data Acquisition and Interface

Data acquisition is handled through an application interface utilising Azure Maps API for geographic visualisation and area selection. Users can navigate to any location globally, with zoom capabilities enabling precise definition of scenario boundaries at resolutions down to 25 m. The interface provides satellite imagery overlays that facilitate accurate alignment between visual features and coordinate selections, ensuring that generated terrains correspond precisely to real-world locations. This capability is particularly important for applications requiring GPS-based navigation validation, where coordinate accuracy directly impacts algorithm performance. Fig. 3 illustrates the system interface, showing both the global navigation view centred on Spain and a detailed area selection example.

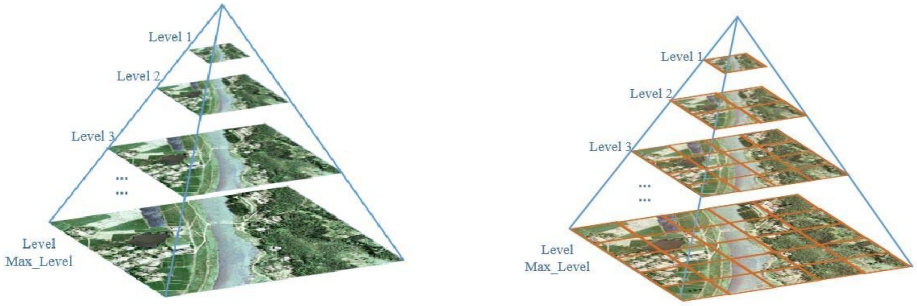
3.1.2 Level of Detail Configuration

The system supports 23 discrete LoD levels (0–22) following hierarchical tile pyramid structures common to major mapping services [39]. Each increment approximately doubles geometric resolution,

ranging from 200 m/element (LoD 0) to 1 cm/element (LoD 22). Fig. 4 illustrates the hierarchical tile pyramid structure.

(**A**) (**B**)

Figure 3: Geospatial data acquisition interface built on the Azure Maps API. (**A**) Global navigation view providing large-scale geographic context. (**B**) Detailed scenario selection view with interactive bounding box, allowing users to precisely determine the geographical coordinates and extent of the target area for 3D generation.



(**A**) (**B**)

Figure 4: Conceptual diagrams illustrating generic multi-resolution structures. (**A**) Standard single-image pyramid structure with decreasing resolution layers. (**B**) Hierarchical tile-pyramid structure used for geospatial data streaming, where each ascending Level of Detail (LoD) subdivides the parent tile into four higher-resolution child tiles [39].

3.1.3 Data Format and Geometric Representation

The system generates 3D surface-based terrain (not planar heightmaps) that captures topography and vertical structures. Source data (glTF with PNG textures) consists of triangulated meshes for terrain, buildings, and vegetation. Each LoD level retrieves data as tile collections: for example, a 0.5 km^2 campus spans 10 tiles at LoD 17, growing exponentially to 1800+ tiles at LoD 22.

3.1.4 Preprocessing and Format Conversion

glTF files are imported into Blender for automated preprocessing via Python scripts. Critical operations include mesh smoothing with adaptive Gaussian kernels to blend tile boundaries, geometry optimisation via

edge collapse [40] (40%–60% polygon reduction while preserving features), and texture enhancement for lighting consistency. The pipeline exports optimised geometry as unified COLLADA (.dae) files, preserving mesh geometry, materials, textures, and GPS coordinate metadata. Fig. 5 shows generated terrain examples.

Figure 5: Example output of the geospatial terrain generation pipeline at LoD 20, rendered in the Gazebo simulator. The model preserves the original satellite imagery as a draped texture over the reconstructed 3D mesh geometry.

3.2 Agricultural Scenario Tool

While the geospatial terrain generation pipeline produces realistic 3D environments with accurate topography and GPS-aligned coordinates, these base terrains lack the agricultural-specific elements necessary for testing crop-related navigation algorithms. The agricultural scenario tool addresses this limitation through an interactive web-based interface that enables users to populate generated terrains with crop models. Working directly with COLLADA terrain models and Gazebo world files from the first stage, the tool maintains coordinate system compatibility and requires no manual 3D modelling or format conversion. Users simply specify directories containing generated scenarios, and the system automatically identifies available terrains, extracts embedded metadata, and enables immediate crop placement.

3.2.1 Agricultural Scenario Configuration Interface

The web-based interface enables interactive crop placement on generated terrains through map visualisation. The system automatically extracts terrain boundaries from COLLADA mesh files and overlays them on satellite imagery, allowing users to define planting areas via polygon selection or standardised rectangular regions. Users configure four primary crop distribution parameters: *row spacing*, which sets the perpendicular distance between consecutive furrows, *plant spacing*, which controls the distance between successive plants along the same furrow, a *randomness factor*, which introduces a small lateral and longitudinal displacement to each plant so that individuals do not appear perfectly aligned, producing the natural irregularity observed in real cultivated fields, and *model selection*, which determines the 3D crop asset to instantiate (e.g., a specific corn growth stage or any user-supplied Gazebo-compliant model). The system ships with several built-in species (corn at five growth stages, tomatoes, cabbages, and sunflowers) and additionally supports a `.zip` asset ingestion workflow through which users can import any Gazebo-compliant 3D model

(COLLADA), for instance those sourced from online repositories such as Sketchfab[4], without architectural limitations. The interface provides real-time pattern preview before final scenario generation.

3.2.2 Coordinate Transformation and Crop Distribution

The system employs standard equirectangular projection for GPS-to-Cartesian coordinate transformation, maintaining sub-meter accuracy for agricultural-scale scenarios. The transformation accounts for latitude-dependent longitude scaling and optional heading angle rotations to align with terrain orientation.

Crop positioning employs a rotated grid algorithm with configurable row spacing (s_{row}), plant spacing (s_{plant}), and row angle (α). The algorithm generates candidate positions along a rotated orthogonal grid, applies optional randomness for realistic irregularity (typically r = 0.01–0.05 m), and filters positions using polygon containment tests to retain only plants within user-defined boundaries.

3.2.3 Terrain Height Interpolation

Accurate vertical placement of crop models on irregular geospatial terrain requires two complementary computations: estimating the elevation z at each target plant position, and computing the local surface orientation so that each model is tilted to match the slope rather than inserted vertically.

Elevation estimation. The system employs k-nearest neighbour (k = 3) weighted averaging with inverse-squared distance weighting [41,42]. For each candidate plant position (x, y), the k closest mesh vertices are retrieved via KD-tree lookup and their elevations are combined as a weighted average, giving greater influence to nearer vertices:

$$z(x,y) = \frac{\sum_{i=1}^{k} w_i \cdot z_i}{\sum_{i=1}^{k} w_i}, \quad w_i = \frac{1}{d_i^2 + \epsilon} \tag{1}$$

where d_i is the horizontal distance from the query point to the i-th neighbour and ϵ is a small regularisation constant to avoid division by zero.

Surface orientation (pitch and roll). Once the elevation is determined, the local terrain gradient must be estimated to align the plant mesh with the ground plane. Partial derivatives of the elevation field are approximated numerically using the central difference scheme, sampling the terrain height at two offset points along each axis with a fixed differential step Δx, Δy:

$$\frac{\partial z}{\partial x} \approx \frac{z(x+\Delta x, y) - z(x-\Delta x, y)}{2\Delta x}, \quad \frac{\partial z}{\partial y} \approx \frac{z(x, y+\Delta y) - z(x, y-\Delta y)}{2\Delta y} \tag{2}$$

These spatial derivatives are then converted into the two rotation angles that describe the tilt of the terrain surface. The *pitch* angle θ captures the forward/backward slope (rotation about the y-axis) and the *roll* angle ϕ captures the lateral slope (rotation about the x-axis):

$$\theta = \arctan\left(-\frac{\partial z}{\partial x}\right), \quad \phi = \arctan\left(-\frac{\partial z}{\partial y}\right) \tag{3}$$

Both angles are applied to each plant's Simulation Description Format (SDF) pose during world file generation, preventing the geometric "floating" artefacts that occur when models are placed with a purely

[4] https://sketchfab.com

vertical insertion on rugged topography. This ensures realistic ground contact and improves mechanical plausibility during physics simulation.

KD-tree spatial indexing [43] enables efficient queries (<0.2 ms per plant) on meshes containing 50,000–500,000 vertices, ensuring accurate vertical placement across irregular terrain.

3.2.4 Output Generation and Deployment

The system generates ROS 2-compatible outputs [44] including modified Gazebo world files or dynamic spawner nodes for runtime crop instantiation, plus launch files with GPS localisation configuration. JSON metadata accompanies each scenario, recording generation parameters and enabling reproducibility for batch scenario creation.

4 Experimental Validation

4.1 Experimental Setup

Experiments used a workstation (AMD Ryzen 9 7900X, 32 GB RAM, NVIDIA RTX 4070 Ti SUPER) running Ubuntu 22.04 LTS with ROS 2 Humble and Gazebo Classic and Ignition. Terrain generation utilised Blender 3.6 LTS, and agricultural scenarios used Python 3.10 with NumPy, SciPy, and Streamlit. Autonomous navigation was implemented using the Nav2 stack [44], configured with a behaviour tree-based planner and the Dynamic Window approach-Based (DWB) local controller for waypoint-following along crop rows. Validation experiments were performed using a Summit-XL mobile robot [45,46] equipped with an Intel RealSense D455 depth camera [47], a u-blox ZED-F9P Real-Time Kinematic (RTK) GPS module [48], and an Xsens MTi Inertial Measurement Unit (IMU) [49].

Three geographic locations were evaluated: the León university campus (0.52 km^2), a rural agricultural field (0.31 km^2), and a hillside vineyard (0.18 km^2, 45 m elevation variation). Three scenario scales from the León campus were tested: large (126,739 m^2), medium (41,610 m^2), and small (8578 m^2) at LoD 17, 19, and 20. Four plant densities: P1–P4 (275–2200 plants).

4.2 Evaluation Procedures

The evaluation is structured around two complementary assessment strategies. Quantitative metrics are applied to terrain generation efficiency (processing time and file count), runtime resource consumption (CPU, RAM, GPU usage during active navigation), and autonomous navigation accuracy (cross-track RMSE, MAE, and maximum deviation against the reference path). Qualitative assessment is reserved for aspects where numerical measurement is not directly applicable, such as visual fidelity across LoD levels, crop model ground contact on sloped terrain, and the diversity of agricultural configurations achievable through parameter variation. The intended evaluation scope is system-level integration for autonomous navigation, while perception algorithm benchmarking and physical robot–crop interaction lie outside the scope of this work.

Each scenario-LoD combination was generated three times. Resource consumption was measured during 10-min simulation sessions using system monitoring tools (htop, nvidia-smi), averaging over the duration excluding loading transients. Navigation performance was quantitatively evaluated using cross-track error, defined as the perpendicular distance between each recorded robot pose and the nearest segment of the reference path. The reference path (digital twin trajectory) was defined as the ideal continuous waypoint sequence generated by the parametric scenario tool. Robot poses were recorded via the ROS 2 odometry topic at the Nav2 controller frequency. Metrics reported include the global Root Mean Square

Error (RMSE), the Mean Absolute Error (MAE), segment-specific RMSE for in-furrow and headland phases separately, and the maximum deviation error.

5 Results

This section presents the experimental results obtained from the validation procedures described in Section 4. Following the evaluation strategy outlined therein, quantitative measurements (generation times, resource consumption, and navigation cross-track error) are reported with numerical precision, while visual fidelity and crop placement realism are demonstrated through comparative figures. All terrain generation evaluations presented in this section utilise the university campus scenario depicted in Fig. 3B as the reference test environment, ensuring consistency across comparative analyses.

5.1 Terrain Generation Performance

It should be noted that terrain generation is an offline, one-time preprocessing step, so the times reported below do not affect the real-time performance of the simulation once the scenario has been built. They are included as a reference to guide users in selecting appropriate LoD levels for their hardware and time constraints.

5.1.1 Level of Detail Analysis

Table 2 presents generation time and file count measurements across LoD levels 20–22. Generation time ranged from 10.2 s (small scenario, LoD 20) to 3138.9 s (large scenario, LoD 22), exhibiting exponential growth with LoD level. File count increased from 18 (small, LoD 20) to 4120 (large, LoD 22).

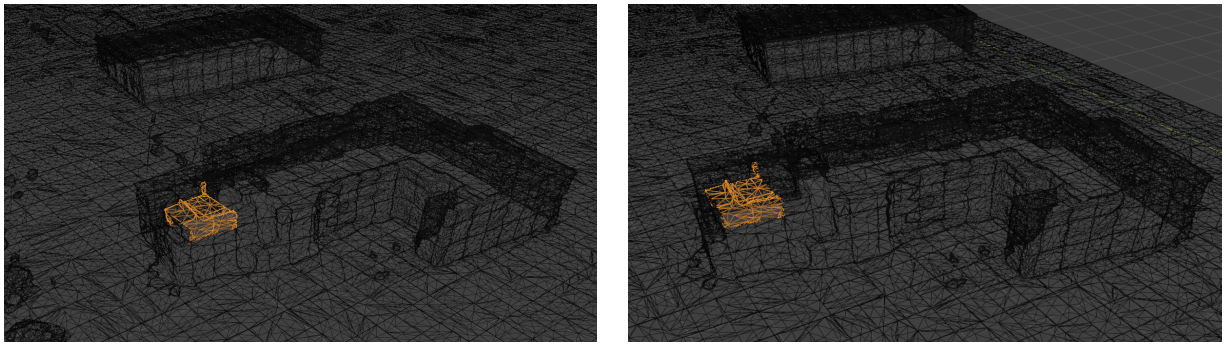
Table 2: Metrics across LoD levels for three university campus scenarios.

Scenario/Metric	LoD 20	LoD 21	LoD 22
Large scenario (126,739 m^2)			
File count (glTF + BIN + XML)	256	775	4120
Texture count (PNG)	281	860	4038
Generation time (s)	22.5	117.8	3138.9
Medium scenario (41,610 m^2)			
File count (glTF + BIN + XML)	67	221	1171
Texture count (PNG)	71	241	1137
Generation time (s)	13.4	28.3	567.3
Small scenario (8578 m^2)			
File count (glTF + BIN + XML)	18	45	216
Texture count (PNG)	18	45	207
Generation time (s)	10.2	12.4	27.4

5.1.2 Visual Quality and Resource Consumption

Visual quality assessment of the campus scenario showed model consistency at given LoD levels regardless of scenario size (Fig. 6). Both large and small scenarios generated at the same LoD exhibit equivalent geometric detail and texture quality when examining the same building structure visible in the reference area, confirming that LoD level, rather than scenario extent, determines visual quality. Fig. 7 presents detailed

views demonstrating that LoD 20 provides sufficient geometric detail for depth camera simulation while LoD 22 provides maximum fidelity at impractical computational cost.



(**A**) (**B**)

Figure 6: Visual quality consistency across different geographic extents at the same LoD level. (**A**) Large-area scenario. (**B**) Small-area scenario. Both maintain comparable texture resolution and mesh detail, demonstrating that the tile-based pipeline scales without quality degradation.

(**A**) (**B**)

Figure 7: Geometric detail comparison between LoD levels on the same terrain region. (**A**) LoD 20, providing sufficient detail for navigation sensor simulation. (**B**) LoD 22, showing finer surface features at the cost of significantly higher polygon count and memory consumption.

5.1.3 Resource Consumption Measurements

Resource consumption was measured during 10-min active simulation sessions at LoD levels 17, 19, and 20. LoD levels 21 and 22 were excluded due to frame rates below 10 frames per second during preliminary testing. Table 3 presents CPU usage, RAM consumption, GPU Video RAM (VRAM), and GPU utilisation measurements.

CPU usage increased from 98.2% (LoD 17) to 245.7% (LoD 20), indicating multi-core utilisation. RAM consumption ranged from 3.8 to 4.5 GB. GPU utilisation remained moderate (12%–38%), suggesting rendering is not a primary bottleneck.

Table 3: Resource consumption during simulation across LoD levels.

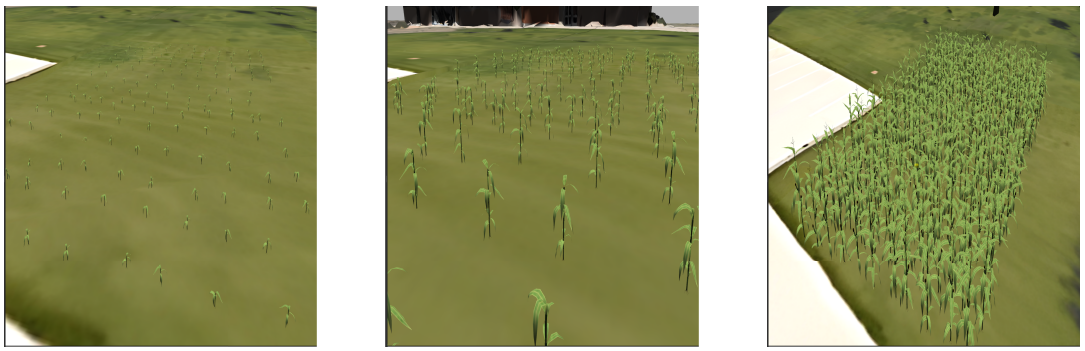
LoD Level	CPU Usage	RAM Usage	GPU VRAM	GPU Utilisation
17	98.2%	3812 MB	652 MB	12%
19	133.3%	4046 MB	905 MB	24%
20	245.7%	4523 MB	1247 MB	38%

5.2 Agricultural Scenario Results

5.2.1 Crop Model Placement and Configuration Flexibility

The system does not dynamically construct procedural meshes at runtime. The 3D plants are instantiated from static, pre-modelled Gazebo assets (COLLADA/.dae geometries accompanied by their structural SDF definitions). Our custom parametric generator computes the required positioning matrix and explicitly compiles a monolithic XML environment file that spawns individual instances of these meshes at the designated Cartesian coordinates, applying corresponding scale and collision parameters. The pipeline is fully crop-agnostic. Users can supply any Gazebo-compliant 3D model (COLLADA) via a `.zip` ingestion workflow in the web interface, and the distribution and height-interpolation logic applies unchanged to any crop geometry.

Five corn models (0.3–2.5 m height) representing different growth stages were validated (Fig. 8). Visual inspection confirmed proper ground contact across sloped terrain. Parameter variations (spacing, randomness, burial depth) successfully generated diverse scenarios ranging from dense mature fields ($\approx$1 plant/m^2) to sparse early-season plantings, with doubling spacing parameters reducing plant count by 50%.

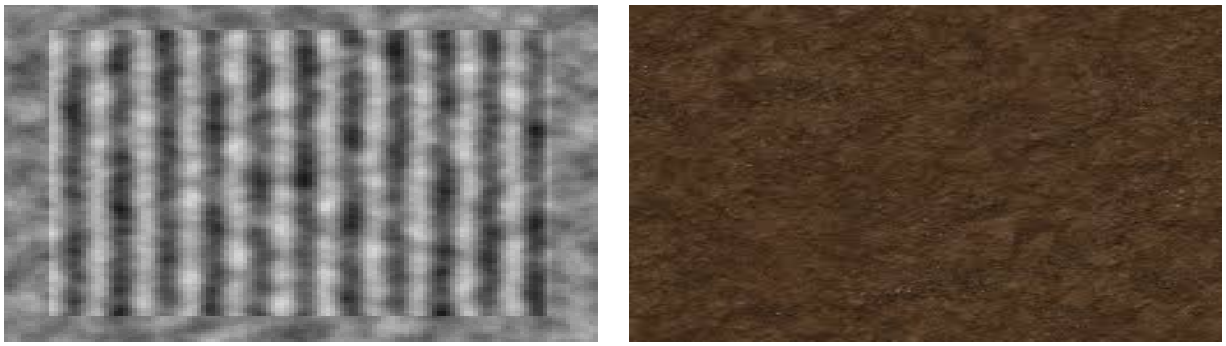


(**A**) (**B**) (**C**)

Figure 8: Visualisation of the 3D corn models used in the scenario generation, illustrating structural diversity across different growth stages. (**A**) Early growth stage model (approximately 0.3 m height) suitable for initial furrow navigation. (**B**) Intermediate stage model (1.2 m height). (**C**) Mature stage model (2.5 m height), which presents significant visual occlusion challenges for perception sensors.

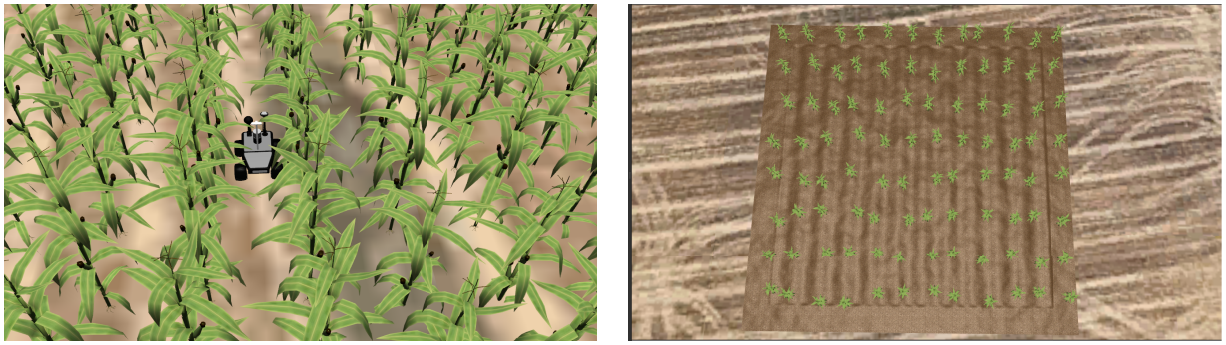
5.2.2 Heightmap-Based Furrow Integration

Heightmap-based terrain integration provides standardised furrow geometry for controlled experiments. The approach addresses geospatially-derived terrain limitations by adding centimeter-scale surface variations (furrows) characteristic of cultivated fields. Fig. 9 shows the heightmap encoding components, while Fig. 10 demonstrates dense and sparse crop configurations aligned with furrow ridges.



(**A**) (**B**)

Figure 9: Heightmap-based furrow generation. (**A**) Greyscale heightmap encoding, where pixel intensity encodes relative elevation to form furrow depressions between crop rows. (**B**) Soil texture applied over the heightmap geometry in Gazebo.



(**A**) (**B**)

Figure 10: Generated agricultural scenarios using the heightmap-based terrain with crop placement. (**A**) Dense planting configuration with the Summit-XL robot navigating between early-stage crop rows. (**B**) Sparse planting configuration demonstrating the parametric control over inter-plant and inter-row spacing.

5.2.3 Resource Consumption

Table 4 and Fig. 11 present resource consumption across four plant densities (P1–P4, 275–2200 plants).

Table 4: Average resource consumption during agricultural scenario simulation (mean ± standard deviation across 3 repetitions).

Configuration	CPU (%)	RAM (GB)	GPU (%)
P1 (1.0 m × 1.0 m)	24.6 ± 6.8	16.6 ± 0.3	25.9 ± 1.5
P2 (2.0 m × 1.0 m)	27.7 ± 10.1	14.9 ± 0.2	28.9 ± 1.7
P3 (2.0 m × 2.0 m)	18.0 ± 6.3	13.8 ± 0.2	22.5 ± 2.6
P4 (3.0 m × 2.0 m)	28.1 ± 9.3	13.4 ± 0.2	21.8 ± 2.0

CPU usage shows high variability with no clear plant density correlation, suggesting efficient spatial culling. RAM consumption correlates strongly with plant count (decreasing from 16.6 to 13.4 GB as plant count decreases, ≈1.45 MB/plant), establishing memory as the primary scalability constraint for large scenarios. GPU utilisation remains moderate (22%–29%) with low variability.

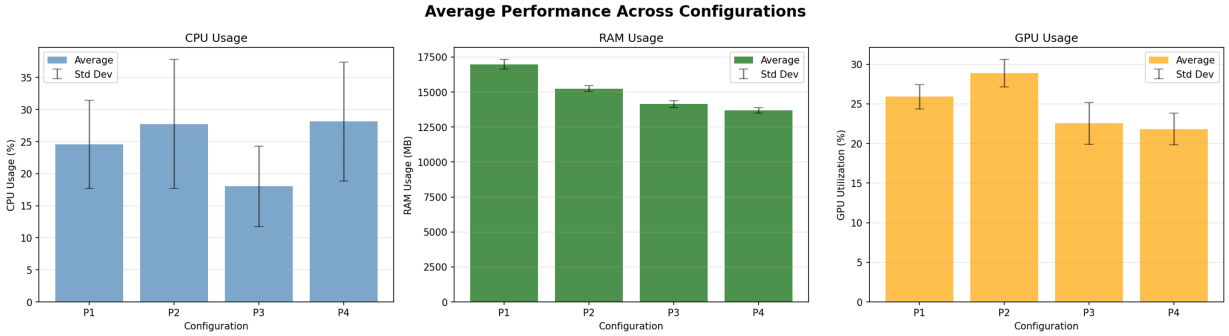


Figure 11: Average resource consumption across agricultural scenario configurations. Error bars represent standard deviation across three independent repetitions.

5.3 Autonomous Navigation Validation

To demonstrate the practical applicability of the generated scenarios, autonomous navigation experiments were conducted using the Nav2 stack [44] and the Summit-XL platform described in Section 4. The robot was tasked with following predefined waypoints along crop furrows in the heightmap-based scenarios, utilising the RealSense D455 depth camera for obstacle detection and the RTK-GPS module for global localisation.

The navigation experiments confirmed that the generated scenarios support closed-loop Nav2 operation, with the robot successfully following waypoint sequences along crop furrows when using early and intermediate growth stage models. However, a strong dependency on crop height relative to camera mounting position was observed. When mature corn models exceeded the camera height, the perception system failed to reliably detect crop row boundaries, leading to degraded costmap generation and navigation failures. This limitation, which mirrors real-world challenges in tall-crop environments, will be addressed in future work through the integration of complementary sensor modalities and crop-height-aware planning strategies.

To provide a quantitative assessment of navigation performance in the generated environments, the Summit-XL platform was deployed in a heightmap-based agricultural scenario and tasked with executing a complete zigzag route spanning two crop rows including headland turn transitions. Furrow detection was performed using an RGB-D (Red-Green-Blue plus Depth) camera with a single-line furrow-following algorithm, while global localisation relied on an RTK-GPS receiver generating `NavSatFix` waypoints. The recorded trajectory was compared against the ideal reference path (the digital twin waypoint sequence generated by the scenario tool) using cross-track error as the evaluation metric. A total of 607 poses were evaluated.

The results, illustrated in Fig. 12, show that the robot maintained a global cross-track RMSE of 0.177 m (MAE = 0.109 m). Segment-level analysis yielded an in-furrow RMSE of 0.200 m and a headland RMSE of 0.170 m, with a maximum deviation of 0.472 m occurring at headland turns where the DWB local controller executes higher-curvature manoeuvres. The colour map confirms that deviations are concentrated at turning points while the straight furrow segments are tracked with sub-0.15 m error. These results demonstrate that the generated scenarios provide a functional and quantitatively characterisable benchmark for autonomous navigation algorithm development, enabling the complete Nav2 stack to be parameterised and evaluated entirely within simulation prior to physical field deployment.

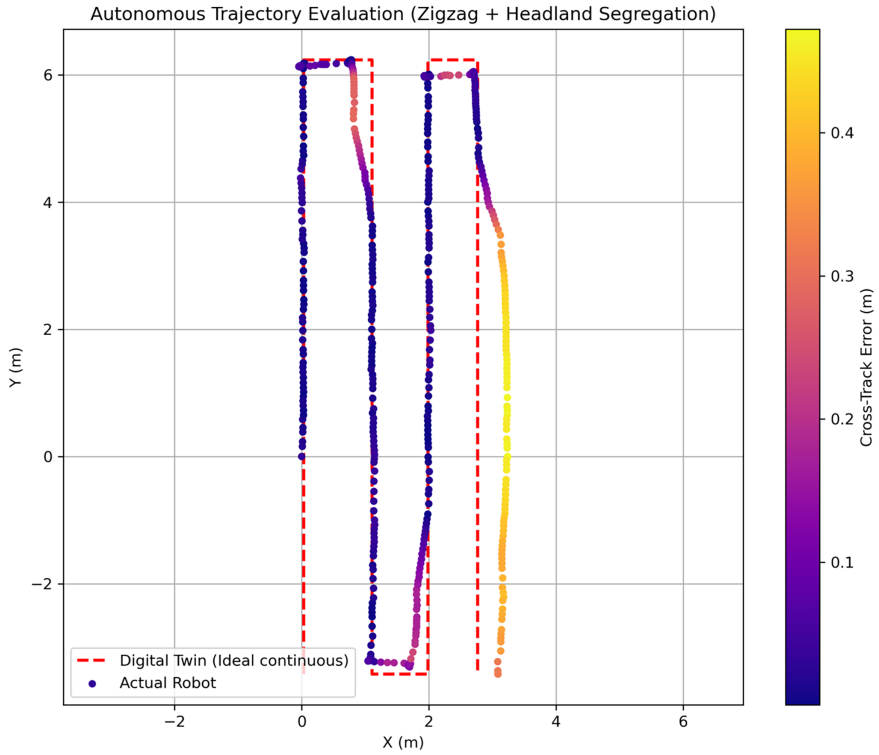


Figure 12: Autonomous trajectory evaluation in the generated agricultural simulation environment. The dashed red line represents the ideal reference path (digital twin); coloured dots show the recorded robot poses colour-coded by instantaneous cross-track error.

6 Discussion

Experimental results reveal critical trade-offs between fidelity and efficiency. As noted in Section 5.1, terrain generation is an offline, one-time cost that does not affect simulation runtime. Nevertheless, the exponential growth of generation time with LoD level (22.5 s at LoD 20 vs. 3139 s at LoD 22 for large scenarios), driven by exponentially increasing tile counts, is relevant for practical workflow planning. LoD 19–20 provides optimal balance for real-time robotic simulation, offering sufficient geometric detail for sensor simulation while maintaining responsive performance on standard hardware.

CPU emerges as the primary bottleneck during active simulation (245.7% utilisation at LoD 20), while the GPU remains underutilised (38%), indicating that rendering capacity is not yet fully exploited and leaving room for enhanced visual quality without additional CPU overhead. RAM consumption, on the other hand, scales linearly with agricultural element density (1.45 MB/plant), making memory the primary constraint for large-scale scenarios. Taken together, these findings suggest that plant density configurations P2–P3 (550–1100 plants) paired with LoD 19 terrain represent the practical optimum, maximising scenario realism within the limits of a standard 32 GB workstation.

7 Conclusions

This paper has presented an integrated two-stage system for semi-automated generation of GPS-accurate agricultural simulation environments from publicly available geospatial data. The approach integrates high-fidelity terrain reconstruction with flexible crop configuration, reducing scenario creation time. Unlike previous systems that focus exclusively on procedural generation on flat terrain or unpopulated topographic geometry, the novelty of this work resides in the integration of both real-world geospatial

topology and specific agricultural assets into a unified pipeline. By combining geospatially accurate terrain generation with interactive agricultural scenario tools, the system addresses a critical gap in agricultural robotics simulation.

Systematic performance characterisation identified LoD 19–20 as the optimal configuration, balancing geometric fidelity against real-time responsiveness on standard hardware. CPU load, rather than GPU rendering, constitutes the primary runtime bottleneck, while RAM scales at approximately 1.45 MB per plant instance, constraining scenario density for large fields.

The agricultural scenario tool successfully demonstrated the capability to generate realistic crop distributions through a rotated grid algorithm with configurable parameters. Height interpolation using k-nearest neighbour averaging ensures accurate vertical positioning on irregular terrain. Complete integration with ROS 2 and Gazebo, including automated launch file generation, enables seamless deployment for robotic navigation research. Validation experiments using a Summit-XL platform equipped with a RealSense depth camera and RTK-GPS confirmed the system's practical applicability for autonomous navigation algorithm development and testing in generated agricultural environments. To support reproducibility and community adoption, the complete system is publicly available as open-source software, with the terrain generation pipeline hosted at a public repository[5] and the robot configuration, plant models, and experimental infrastructure at a second public repository[6].

Despite these capabilities, the current implementation exhibits several limitations:

- **Computational cost at high LoD.** Levels 21–22 are computationally prohibitive for real-time simulation in standard workstations, with frame rates dropping below acceptable thresholds.
- **RAM scalability.** Memory consumption scales linearly at approximately 1.45 MB per plant instance owing to the per-instance scene graph maintained by the OGRE (Object-Oriented Graphics Rendering Engine) renderer. Empirical benchmarking of six optimisation strategies (primitive collision geometries, texture downscaling) confirmed that replacing trimesh collisions with cylinder primitives reduces per-instance cost by only 5%–8%, while removing collision entirely yields ~16% savings but is not viable for navigation tasks.
- **Crop-height dependency.** Navigation performance degrades when crop models exceed the camera mounting height, causing perception failures in the detection of row boundaries.
- **Rigid-body physics only.** Physical interactions are restricted to the standard rigid-body framework of Gazebo, which suffices for sensor-based navigation but limits applicability to contact-intensive tasks such as harvesting.
- **Limited to a single simulation environment.** The proposed system is constrained to operate within Gazebo, as it is tightly coupled to the underlying hardware stack. This strong dependency prevents portability to other simulation environments without substantial re-engineering, making the approach inherently bound to this specific simulator.

Future research will address these constraints through several complementary directions. On the navigation side, integrating additional sensor modalities and developing crop-height-aware planning strategies remain the most immediate priorities, as the current dependency on camera-mounted perception limits reliable operation to early and intermediate crop growth stages. GPU optimisation could also be exploited to improve visual rendering quality without a proportional increase in CPU load. Regarding physical realism, incorporating flexible Cosserat rod models [38] would enable high-fidelity evaluation of contact-based tasks such as harvesting, extending the system beyond its current navigation-focused scope. Physics-based terrain

[5] https://github.com/ssancd03/Geo2Gazebo
[6] https://github.com/Innovations-in-Smart-and-Secure-Systems/summit_agriculture

deformation capturing wheel–soil interactions is a further direction, though its applicability is presently constrained to small-plot studies given the per-contact computational cost at field scale. On the scalability front, migrating to a Gazebo version with OGRE 2 support would unlock per-instance runtime Level of Detail, directly addressing the 1.45 MB per-plant RAM overhead and enabling large-scale scenarios on standard hardware. Domain adaptation techniques for sim-to-real transfer in agricultural perception would complement these efforts by reducing the gap between simulated and real sensor data. Finally, evolving the web interface towards a prompt-based pipeline driven by Large Language Models [37] would allow users to generate complete agricultural scenarios from natural language descriptions, substantially lowering the configuration barrier. These improvements will further enhance the system's utility for developing and validating autonomous navigation algorithms in realistic agricultural simulation environments.

Acknowledgement: This project has been developed with the support of ISDEFE and the Robotics Group of the University of León within the framework of the ISDEFE-ULE Chair.

Funding Statement: This research was funded by Grants PID2024-162298OB-I00 (EXPLICIT sElf-eXPLaInable Cyber-physIcal sysTems) and PID2024-161761OB-C21 (AURORAS - Advanced aUtonomy for RObots in the primary Sector) funded by MICIU/AEI/10.13039/501100011033 and by the European Union.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Miguel Á González-Santamarta, Vicente Matellán-Olivera and Ángel Manuel Guerrero-Higueras; methodology, Miguel Á González-Santamarta, Vicente Matellán-Olivera and Ángel Manuel Guerrero-Higueras; software, Sergio Sánchez de la Fuente, Luis Prieto-López and Miguel Á González-Santamarta; validation, Sergio Sánchez de la Fuente, Luis Prieto-López and Miguel Á González-Santamarta; formal analysis, Sergio Sánchez de la Fuente; investigation, Sergio Sánchez de la Fuente and Luis Prieto-López; resources, Sergio Sánchez de la Fuente, Luis Prieto-López and Miguel Á González-Santamarta; data curation, Sergio Sánchez de la Fuente; writing—original draft preparation, Sergio Sánchez de la Fuente; writing—review and editing, Sergio Sánchez de la Fuente, Luis Prieto-López, Miguel Á González-Santamarta, Vicente Matellán-Olivera and Ángel Manuel Guerrero-Higueras; visualisation, Sergio Sánchez de la Fuente; supervision, Miguel Á González-Santamarta, Vicente Matellán-Olivera and Ángel Manuel Guerrero-Higueras; project administration, Vicente Matellán-Olivera and Ángel Manuel Guerrero-Higueras; funding acquisition, Vicente Matellán-Olivera and Ángel Manuel Guerrero-Higueras. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data and code supporting the findings of this study are openly available in GitHub at https://github.com/ssancd03/Geo2Gazebo (terrain generation pipeline and crop scenario tool) and at https://github.com/Innovations-in-Smart-and-Secure-Systems/summit_agriculture (plant models, Summit-XL robot configuration, ROS 2 navigation stack, and experimental launch infrastructure).

Ethics Approval Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Shamshiri RR, Weltzien C, Hameed IA, Yule IJ, Grift TE, Balasundram SK, et al. Research and development in agricultural robotics: a perspective of digital farming. Int J Agric Biol Eng. 2018;11(4):1–14. doi:10.25165/j.ijabe.20181104.4278.
2. Bechar A, Vigneault C. Agricultural robots for field operations: concepts and components. Biosyst Eng. 2016;149:94–111. doi:10.1016/j.biosystemseng.2016.06.014.
3. Fountas S, Mylonas N, Malounas I, Rodias E, Hellmann Santos C, Pekkeriet E. Agricultural robotics for field operations. Sensors. 2020;20(9):2672. doi:10.3390/s20092672.
4. Collins J, Chand S, Vanderkop A, Howard D. A review of physics simulators for robotic applications. IEEE Access. 2021;9:51416–31. doi:10.1109/ACCESS.2021.3068769.

5. Farr TG, Rosen PA, Caro E, Crippen R, Duren R, Hensley S, et al. The shuttle radar topography mission. Rev Geophys. 2007;45(2):RG2004. doi:10.1029/2005RG000183.
6. Koenig N, Howard A. Design and use paradigms for Gazebo, an open-source multi-robot simulator. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); 2004 Sep 28–Oct 2; Sendai, Japan. p. 2149–54. doi:10.1109/IROS.2004.1389727.
7. Macenski S, Foote T, Gerkey B, Lalancette C, Woodall W. Robot operating system 2: design, architecture, and uses in the wild. Sci Robot. 2022;7(66):eabm6074. doi:10.1126/scirobotics.abm6074.
8. Gebbers R, Adamchuk VI. Precision agriculture and food security. Science. 2010;327(5967):828–31. doi:10.1126/science.1183899.
9. Bac CW, van Henten EJ, Hemming J, Edan Y. Harvesting robots for high-value crops: state-of-the-art review and challenges ahead. J Field Robot. 2014;31(6):888–911. doi:10.1002/rob.21525.
10. Slaughter DC, Giles DK, Downey D. Autonomous robotic weed control systems: a review. Comput Electron Agric. 2008;61(1):63–78. doi:10.1016/j.compag.2007.05.008.
11. Weiss U, Biber P. Plant detection and mapping for agricultural robots using a 3D LIDAR sensor. Robot Auton Syst. 2011;59(5):265–73. doi:10.1016/j.robot.2011.02.011.
12. Purcell W, Neubauer T. Digital twins in agriculture: a state-of-the-art review. Smart Agric Technol. 2023;3(1):100094. doi:10.1016/j.atech.2022.100094.
13. Verdouw C, Tekinerdogan B, Beulens A, Wolfert S. Digital twins in smart farming. Agric Syst. 2021;189:103046. doi:10.1016/j.agsy.2020.103046.
14. Dosovitskiy A, Ros G, Codevilla F, López A, Koltun V. CARLA: an open urban driving simulator. In: Proceedings of the 1st Conference on Robot Learning (CoRL); 2017 Nov 13–15; Mountain View, CA, USA. p. 1–16.
15. Shah S, Dey D, Lovett C, Kapoor A. AirSim: high-fidelity visual and physical simulation for autonomous vehicles. In: Hutter M, Siegwart R, editors. Field and service robotics. Springer proceedings in advanced robotics. Vol. 5. Cham, Switzerland: Springer; 2018. p. 621–35.
16. Nie J, Wang Y, Li Y, Chao X. Artificial intelligence and digital twins in sustainable agriculture and forestry: a survey. Turk J Agric For. 2022;46(5):642–61. doi:10.55730/1300-011X.3033.
17. Nasirahmadi A, Hensel O. Toward the next generation of digitalization in agriculture based on digital twin paradigm. Sensors. 2022;22(2):498. doi:10.3390/s22020498.
18. Abbyasov B, Lavrenov R, Zakiev A, Yakovlev K, Svinin M, Magid E. Automatic tool for Gazebo world construction: from a grayscale image to a 3D solid model. In: Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA); 2020 May 31–Aug 3; Paris, France. p. 7226–32. doi:10.1109/ICRA40945.2020.9196621.
19. Scholz D. Tile-based procedural terrain generation [bachelor's thesis]. Vienna, Austria: TU Wien; 2019.
20. Tobin J, Fong R, Ray A, Schneider J, Zaremba W, Abbeel P. Domain randomization for transferring deep neural networks from simulation to the real world. In: Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); 2017 Sep 24–28; Vancouver, BC, Canada. p. 23–30. doi:10.1109/IROS.2017.8202133.
21. Peng XB, Andrychowicz M, Zaremba W, Abbeel P. Sim-to-real transfer of robotic control with dynamics randomization. In: Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA); 2018 May 21–25; Brisbane, Australia. p. 3803–10. doi:10.1109/ICRA.2018.8460528.
22. Sadeghi F, Levine S. Real single-image flight without a single real image. In: Proceedings of Robotics: Science and Systems XIII (RSS); 2017 Jul 12–16; Cambridge, MA, USA. doi:10.15607/RSS.2017.XIII.034.
23. Martini M, Ambrosio M, Navone A, Tuberga B, Chiaberge M. Enhancing visual autonomous navigation in row-based crops with effective synthetic data generation. Precis Agric. 2024;25(6):2881–902. doi:10.1007/s11119-024-10157-6.
24. Parish YIH, Müller P. Procedural modeling of cities. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01); 2001 Aug 12–17; Los Angeles, CA, USA. p. 301–8. doi:10.1145/383259.383292.

25. Deussen O, Hanrahan P, Lintermann B, Měch R, Pharr M, Prusinkiewicz P. Realistic modeling and rendering of plant ecosystems. In: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98); 1998 Jul 19–24; Orlando, FL, USA. p. 275–86. doi:10.1145/280814.280898.
26. Prusinkiewicz P, Lindenmayer A. The algorithmic beauty of plants. New York, NY, USA: Springer-Verlag; 1990. doi:10.1007/978-1-4613-8476-2.
27. Zhou H, Sun J, Turk G, Rehg JM. Terrain synthesis from digital elevation models. IEEE Trans Vis Comput Graph. 2007;13(4):834–48. doi:10.1109/TVCG.2007.1027.
28. Kim BSS, Safarzadeh Ramhormozi R, Wang X. Quality assessment of the OpenStreetMap road network in Calgary, Alberta. In: Proceedings of the 31st International Cartographic Conference (ICC 2023); 2023 Aug 13–18; Cape Town, South Africa. 124 p. doi:10.5194/ica-abs-6-124-2023.
29. Snyder JP. Map projections a working manual. U.S. geological survey professional paper 1395. Washington, DC, USA: United States Government Printing Office; 1987. doi:10.3133/pp1395.
30. Bochtis DD, Sørensen CG. The vehicle routing problem in field logistics part I. Biosyst Eng. 2009;104(4):447–57. doi:10.1016/j.biosystemseng.2009.09.003.
31. Tsolakis N, Bechtsis D, Bochtis D. AgROS: a robot operating system based emulation tool for agricultural robotics. Agronomy. 2019;9(7):403. doi:10.3390/agronomy9070403.
32. Quigley M, Conley K, Gerkey B, Faust J, Foote T, Leibs J, et al. ROS: an open-source robot operating system. In: Proceedings of the ICRA Workshop on Open Source Software; 2009 May 12–17; Kobe, Japan. p. 1–6.
33. de Silva R, Cielniak G, Wang G, Gao J. Deep learning-based crop row detection for infield navigation of agri-robots. J Field Robot. 2024;41(7):2299–321. doi:10.1002/rob.22238.
34. Bourr K. Forest3D: generate populated outdoor environments for gazebo [Internet]; 2025 [cited 2026 Jan 9]. Available from: https://github.com/unitsSpaceLab/Forest3D.
35. FieldRobotEvent. Virtual maize field: a ROS package for procedural crop generation in Gazebo [Internet]. GitHub Repository; 2023 [cited 2026 Jan 9]. Available from: https://github.com/FieldRobotEvent/virtual_maize_field.
36. Afzal A, Le Goues C, Timperley CS. GzScenic: automatic scene generation for gazebo simulator. arXiv:2104.08625. 2021. doi:10.48550/arxiv.2104.08625.
37. Yang Y, Sun FY, Weihs L, Vanderbilt E, Herrasti A, Han W, et al. Holodeck: language guided generation of 3D embodied AI environments. In: Proceedings of the 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2024 Jun 16–22; Seattle, WA, USA. doi:10.1109/CVPR52733.2024.01536.
38. Deng J, Pałubicki W, Marri S, Pirk S, Michels DL, Klein J, et al. Gazebo plants: simulating plant-robot interaction with cosserat rods. arXiv:2402.02570. 2024. doi:10.48550/arXiv.2402.02570.
39. Guo N, Xiong W, Wu Q, Jing N. An efficient tile-pyramids building method for fast visualization of massive geospatial raster datasets. Adv Electr Comput Eng. 2016;16(4):3–8. doi:10.4316/AECE.2016.04001.
40. Garland M, Heckbert PS. Surface simplification using quadric error metrics. In: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97); 1997 Aug 3–8; Los Angeles, CA, USA. p. 209–16. doi:10.1145/258734.258849.
41. Shepard D. A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM National Conference; 1968; New York, NY, USA. p. 517–24. doi:10.1145/800186.810616.
42. Li J, Heap AD. Spatial interpolation methods applied in the environmental sciences: a review. Environ Model Softw. 2014;53(9):173–89. doi:10.1016/j.envsoft.2013.12.008.
43. Bentley JL. Multidimensional binary search trees used for associative searching. Commun ACM. 1975;18(9):509–17. doi:10.1145/361002.361007.
44. Macenski S, Martín F, White R, Clavero JG. The Marathon 2: a navigation system. In: Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); 2020 Oct 25–29; Las Vegas, NV, USA. p. 2718–25. doi:10.1109/IROS45743.2020.9341207.
45. Robotnik Automation. Company profile and mobile robot solutions [Internet]. Valencia, Spain: Robotnik Automation S.L.L.; 2023 [cited 2026 Jan 12]. Available from: https://robotnik.eu/.

46. Robotnik Automation. SUMMIT-XL mobile robot platform: technical specifications [Internet]. Valencia, Spain: Robotnik Automation S.L.L.; 2023 [cited 2026 Jan 12]. Available from: https://robotnik.eu/products/mobile-robots/summit-xl-en/.
47. Intel Corporation. Intel RealSense D400 series product family datasheet [Internet]. Santa Clara, CA, USA: Intel Corporation; 2023 [cited 2026 Jan 12]. Available from: https://www.intel.la/content/www/xl/es/products/sku/205847/intel-realsense-depth-camera-d455/specifications.html.
48. u-blox AG. ZED-F9P high precision GNSS module: data sheet [Internet]. Thalwil, Switzerland: u-blox AG; 2023 [cited 2026 Jan 12]. Available from: https://www.u-blox.com/en/product/zed-f9p-module.
49. Movella Technologies. MTi user manual [Internet]. Henderson, NV, USA: Movella Technologies B.V.; 2023 [cited 2026 Jan 12]. Available from: https://www.xsens.com/hubfs/Downloads/usermanual/MTi_usermanual.pdf.