PAVER

AIoT for Cascade Hydroponic Greenhouses with Solar Energy Harvesting and Water Regeneration System

D6.1 Survey of advances in digital twin parametrization

Author: Universidad Politécnica de Cartagena

June 2025

This project is part of the PRIMA Programme supported by the European Union.

This work was supported by the grant PID2023-148214OB-C21 funded by MICIU/AEI/10.13039/501100011033 and by FEDER/EU.

Contents

1 Introduction

A digital twin is a dynamic, virtual replica of a physical asset, system, or process that continuously receives data from the real world to mirror its behavior in real time [1]. This concept, which originated in the manufacturing and aerospace sectors, is rapidly expanding into new domains, including agriculture, where it has the potential to revolutionize how complex systems are understood, monitored, and optimized.

In the agricultural sector, and particularly in controlled environments like greenhouses, digital twins provide a powerful framework to integrate multiple sources of data, simulate system behavior, and visualize conditions in a highly intuitive manner. By coupling real-time sensor inputs with 3D models and intelligent data analysis, a digital twin can offer a holistic view of the system that goes far beyond traditional monitoring dashboards [2]. This capability enables farmers, researchers, and decision-makers to detect anomalies, evaluate performance, and make data-driven interventions with greater precision.

Agrivoltaic greenhouses present an especially compelling case for the implementation of digital twins. These hybrid systems combine plant cultivation with photovoltaic energy production, introducing new layers of complexity in terms of shading patterns, microclimate variations, energy yield optimization, and crop-light interactions. Managing such a system requires continuous observation and multi-variable analysis, which digital twins are well-suited to support [3]. Through real-time visualization and analytical feedback, they enable stakeholders to better understand how changes in environmental conditions, energy availability, or control strategies affect both plant growth and energy generation.

Several technological approaches can be employed to build a digital twin in this context. These include 3D modeling and rendering engines (such as Unity or Unreal Engine) for immersive visualization, IoT platforms and edge devices for real-time data acquisition, and machine learning algorithms for detecting patterns or predicting outcomes [4]. Digital twins can also serve as interactive platforms that aggregate data from sensors, weather forecasts, or historical trends, allowing users to simulate scenarios or to monitor key variables such as temperature, humidity, light levels, or energy output.

Despite their advantages, implementing a functional and scalable digital twin is not without challenges. Some of the key difficulties include ensuring accurate data synchronization between the physical and virtual systems, developing robust data pipelines that can handle heterogeneous sources, and maintaining the fidelity of the 3D representation over time [2]. Furthermore, interpreting the results produced by AI models within the digital twin requires domain-specific knowledge and thoughtful design of the user interface to support effective decision-making.

This report presents an overview of the main tools, techniques, and design principles involved in the parametrization and construction of digital twins for agrivoltaic

greenhouses. The focus is placed on their role as platforms for real-time data visualization and integration of artificial intelligence models, with the goal of enhancing system transparency, operability, and overall performance. Additionally, the report includes a proposed workflow for the creation of Digital Twins specifically designed for agrivoltaic greenhouse applications.

The rest of the report is organized as follows: Section 2 presents the methodology followed in this document. The explanation of the concept of Digital Twin and the latest state of the art is presented in Section 3. Section 4 detail the available tools for the creation of Digital Twins. The explanation and evaluations on how to use these tools is presented in Section 7. Section 6 discusses our views on the parametrization of Digital Twins in agrivoltaic greenhouses from the aquired knowledge and propossess a workflow for Digital Twin creation in the agrivoltaics sector. Lastly, the conclusion is presented in Section 7.

2 Methodology

The methodology adopted in this report is as follows. According to data from the Lens portal for patent and scholarly search [5], there has been a steady increase in the number of publications related to digital twins in agriculture in recent years (see Figure 1). Among these publications, approximately 80% are journal articles, 7.8% are datasets, 4.1% are conference proceedings, 3.3% are book chapters, 2.8% are preprints, and the remaining documents fall into other categories. Regarding the origin of these publications, the majority were produced in the United States, with a total of 487 records. China follows closely with 430, followed by India with 262, and the United Kingdom with 213 (see Figure 2). Although numerous related works exist, we considered it necessary to evaluate each tool individually to determine whether the creation of a complete digital twin is feasible. Therefore, this report presents the latest tools available for digital twin development, along with a series of preliminary tests conducted in our lab to identify which tools are best suited for the project. We also discuss the parameters that should potentially be considered in the implemented Digital Twin.

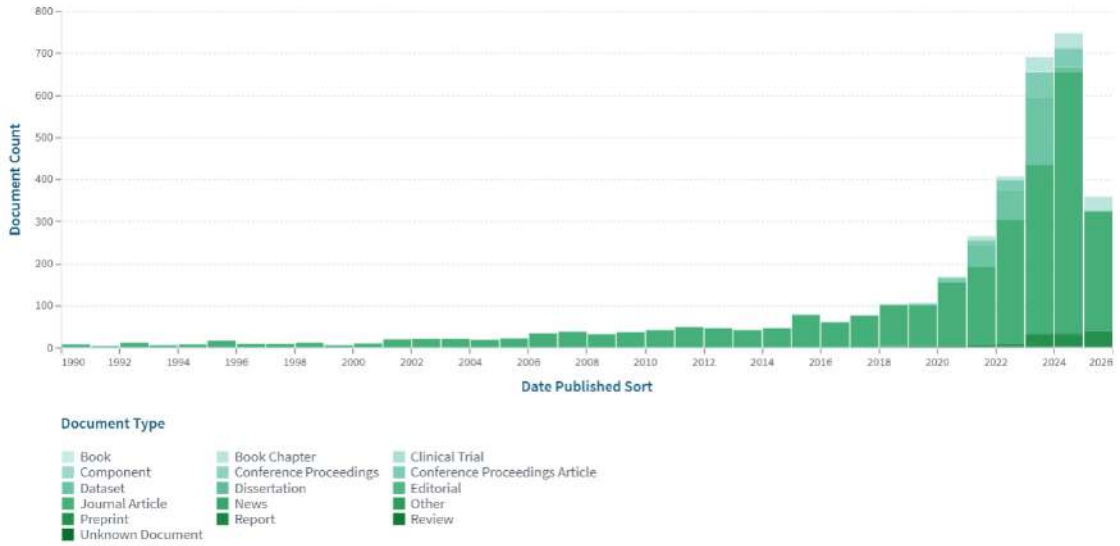


Figure 1: Evolution over time of publish works regading Digital Twins for agriculture.

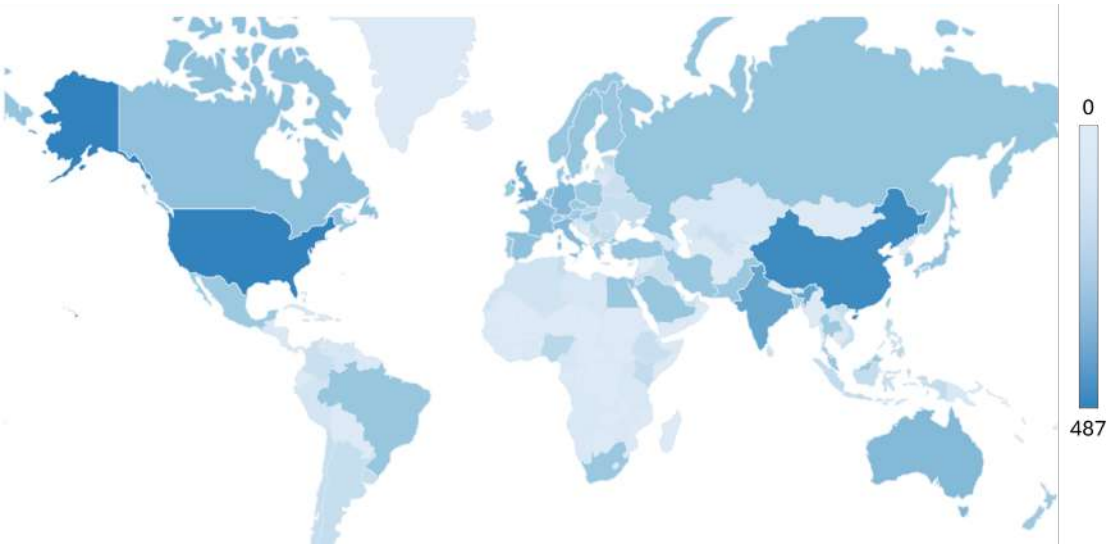


Figure 2: Map of the countries with most works on Digital Twins for agriculture.

3 State of the Art on Digital Twins Applied to Agriculture

Digital Twin (DT) technology, while widely recognized in recent years, has its origins in concepts developed in the early 2000s [6]. It consists of creating a virtual model that accurately represents a physical system, with continuous data exchange linking both

environments. Initially, this approach aimed to improve product lifecycle management and enable better simulation of real-world objects. However, early adoption was limited by technological challenges such as insufficient computing power and connectivity. The concept gained practical relevance when aerospace sectors began using it to simulate and monitor aircraft, allowing for predictive maintenance and extended lifecycle management. Since then, Digital Twins have been adopted across various fields as dynamic, data-driven virtual representations of physical assets.

Since its initial definition, Digital Twin (DT) has been described in various ways depending on the context, often referred to as a virtual model, counterpart, simulation, or representation of a physical entity. Originally focused on aerospace and manufacturing, the concept now applies broadly to systems, products, processes, and even biological entities. A key characteristic across definitions is the bidirectional exchange of real-time and historical data between the physical and digital versions, enabling analysis, simulation, prediction, and control throughout the lifecycle. Unlike static computer models, DTs continuously update based on live data, providing an accurate and dynamic reflection of the physical object. However, confusion remains in the literature due to overlapping terms such as Digital Shadow, Digital Model, Digital Thread, and Product Avatar, which differ in scope and function. To clarify, a Digital Twin can be defined as a dynamic and evolving digital representation of a physical object or process, maintaining synchronization with its physical counterpart through continuous two-way data flow.

Digital Twins (DTs) share several fundamental characteristics despite variations depending on their type. They exhibit high fidelity, meaning they closely replicate their physical counterparts in appearance, function, and detail, which enhances the accuracy of simulations and predictions. DTs are dynamic, continuously updating in real time to reflect changes in their physical twins through constant data exchange. They are also self-evolving, adapting and optimizing themselves over the lifecycle in response to new data, creating a continuous feedback loop. Each DT is uniquely identifiable, maintaining its specific data and models throughout its lifespan. Furthermore, DTs are multiscale and multiphysical, capturing properties ranging from atomic to macroscopic levels and encompassing structural, thermal, and material characteristics. They are multidisciplinary, integrating knowledge from various engineering and IT fields, and hierarchical, composed of interconnected submodels representing different components of the overall system.

From a hierarchical standpoint, Digital Twins can be categorized into three levels based on their scale in manufacturing. The unit level represents the smallest element, such as equipment, materials, or environmental factors, modeled in terms of geometry, function, behavior, and operation of the physical counterpart. The system level combines multiple unit-level Digital Twins to form more complex entities like production lines or factories, enabling better data sharing and resource management. At the highest level, the System of Systems (SoS) integrates several system-level Digital Twins, facilitating collaboration across different departments or enterprises and covering the

entire product lifecycle. Additionally, this hierarchy can be viewed as part/component twins, product/asset twins, system twins, and process twins, where more advanced systems are formed by assembling simpler, lower-level twins.

One of the main challenges in creating a digital twin is accurately recreating a real-world environment in 3D. Some studies rely on manual measurements of the physical space to model it within a virtual environment [7, 8]. This modeling process is often time-consuming and requires the expertise of a skilled 3D modeler to achieve realistic and reliable results. Other studies utilize point clouds to construct three-dimensional digital representations of agricultural farms [9]. Open farms employing this approach typically rely on drones equipped with photogrammetry or LiDAR technologies to capture the visual data, which is subsequently processed. Among the tools tested for postprocessing and generating 3D models of open farms are Agisoft, Pix4Dmapper, OpenDroneMap, DJI Terra, Meshroom, Regard3D, RealityCapture, and 3DF Zephyr. For the specific task of visualizing point clouds acquired by drones, the study in [9] identified OpenDroneMap as the best performing tool. The authors in [10] also discuss the use of augmented, virtual, and mixed reality technologies together with digital twins. However, this report does not go into detail about these technologies, as they are covered in a different report.

In relation to agriculture, there have been works on digital twins for different types of purposes, including living plants or trees, agricultural products, fields, farms, landscapes, buildings, machinery, animals, and the food supply chain and logistics [11]. These digital twin use cases were mostly employed for purposes such as real-time monitoring, optimization, technology integration, system failure analysis and prediction, and for the analysis of energy consumption.The work presented in [2] clearly categorizes research proposals into three groups: digital twin models (7%), partially integrated digital twins (56%) meaning digital twins with limited functionalities, and fully integrated digital twins (37%) which encompass the full set of functionalities expected from digital twins.

Digital twins in agriculture can be applied to open-field orchards. The study in [12] presents a digital twin system for mandarin orchards developed at multiple scales, including regional, inter-orchard, and intra-orchard levels. Data is collected using satellite imagery and visualized through 2D graphs. Machine learning and statistical models are employed to analyze variables such as soil chemical properties and fruit sugar content.

Regarding greenhouses, several research efforts have focused on developing digital twins to monitor and control different environmental and operational parameters. In [13], Simulink is used to construct a digital twin model that tracks temperature, air humidity, soil moisture, and plant health using sensors and cameras placed in a greenhouse scale model. Their architecture also incorporates drivers, relays, and motor actuators to enable environmental control. The system integrates a convolutional neural network (CNN) to predict plant growth. However, their implementation does not include any form of 3D model representation. A separate simulation-based digital twin focused on

monitoring energy consumption in greenhouses, also developed with Simulink, is presented in [14]. The authors propose a five-layer digital twin architecture consisting of physical, virtual, communication, data management, and service layers. The results are visualized using a 2D panel with graphical data representations. The study in [15] introduces a digital twin model that combines monitoring of both greenhouse and underground environments. The system uses Visible Light Communication to transmit data between the two settings. The main objective of this digital twin was to simulate signal propagation losses across various communication technologies. The results were presented through graphical data representations, and no three-dimensional model of the greenhouse was included in the implementation.

Digital twins are also employed to simulate various scenarios in order to evaluate potential actions in response to hypothetical events that cannot be tested in real life, either due to the potential harm they may cause or because of the high economic or labor-related costs involved. For instance, the study by [7] used their digital twin to simulate the behavior of a harvesting robot, enabling them to conduct experiments before deploying the equipment in a real greenhouse environment. Other studies predict the evolution of environmental parameters inside the greenhouse, such as temperature or humidity [8]. A digital twin of the SILAL strawberry greenhouse in Abu Dhabi was developed within the ROS Gazebo simulation environment to test a deep learning based robotic system [16]. The system integrates the MARTA robot, which uses a telescopic arm, and a YOLOv9 GLEAN deep learning model that enhances strawberry detection using high resolution image features. The model was trained with a combination of real and synthetic images to improve detection robustness. Visual servoing, implemented using ROS MoveIt, allowed for precise motion control during strawberry grasping. The results showed detection precision and recall values above 0.99. The system outperformed other models tested in the simulated greenhouse environment. The environment includes a 3D greenhouse model built in Gazebo, while smaller objects were designed separately using Blender. The research in [17] introduces GreenH, a family of domain-specific languages designed to simplify the design and operation of IoT architectures in smart greenhouses. These languages aim to support the development and monitoring of digital twins by improving the way greenhouse processes are modeled and simulated. The authors propose a methodology based on high-level metamodels and formal grammar definitions. Their evaluation highlights strong performance in terms of language expressiveness, readability, and scalability. Among the most innovative implementations of digital twins in agriculture, particularly in hydroponic greenhouses, is the work presented in [18]. This study introduces a system that enables remote control of a robotic arm using a wearable glove. The authors describe three distinct modes of interaction for carrying out agricultural tasks: real to digital, where data from the physical environment is transferred to the digital twin; digital to digital, where simulations are used to optimize system performance; and digital to real, where actions performed within the digital model are executed in the physical world.

For the particular case of agrivoltaics, the study presented in [19] focuses on the

development of a large-scale digital twin aimed at predicting crop yield under different solar panel shading conditions. The proposed framework integrates IoT technologies through a dense sensor network and sensor fusion techniques, potentially incorporating satellite imagery to collect environmental and crop data from open fields. The ultimate goal is to optimize both energy generation and agricultural productivity using AI-based data analysis. Preliminary experiments are being conducted at a laboratory in Milan, where wooden panels simulate shading patterns typical of northern Italy. These tests support the creation of a detailed digital twin model by providing environmental and crop response data. The main challenges in agrivoltaic greenhouse design according to [20] is reducing the negative effects of shading caused by photovoltaic panels. The authors propose a detailed irradiance simulation model, that can be used in digital twins, based on ray tracing that accounts for the optical properties of the greenhouse covering. Using satellite-derived solar data, the model delivers high-resolution simulations in both time and space. To validate its accuracy, the simulated irradiance levels were compared to real measurements from a previous experimental setup. Sensor positions were optimized using one day of minute-by-minute irradiance data and later applied to simulate conditions over several months. The model showed a small average deviation of 2.88 % in radiation reduction compared to experimental data across a full crop cycle. The authors in [21] presented a digital twin that allows exploring how agrivoltaic systems, particularly those with horizontal single-axis trackers, can optimize the balance between agricultural productivity and solar energy generation. The study introduces a method to dynamically adjust the position of solar panels in response to crop-specific light requirements. Using an apple orchard in Germany as a case study, simulations were carried out to compare conventional shading practices with a new approach based on absolute irradiance targets, measured in watts or watt-hours per square meter. The results, obtained using the APyV simulation tool, show that it is possible to meet 91% of the apples' light needs with a controlled panel strategy, while accepting a 20% reduction in electricity production. The study also highlights the times when light availability falls short, underlining the trade-offs involved in crop-oriented optimization. Lastly, the Python Agrivoltaic Simulation Environment (PASE), an open-source tool designed to evaluate the productivity of agrivoltaic systems in terms of both energy generation and agricultural output, was presented in [22]. It was developed collaboratively to address stakeholders' modeling needs while considering local regulatory conditions. The framework predicts solar irradiance, crop and photovoltaic yields, and calculates land productivity metrics such as the land equivalent ratio. Validation experiments showed high accuracy in solar radiation estimation, with a 1% error under varied sky conditions. Its application was demonstrated using wheat crops in the BIODIV-SOLAR pilot, including a sensitivity analysis to optimize system configuration based on inter-row spacing.

This report focuses on the tools that enable the creation of the 3D model representation of agrivoltaic greenhouses, as opposed to 2D visualization prevalent in many of the existing solutions. We aim to include real-time monitoring, simulation and control

features considering important variables from the plants, the IoT sensor devices, the farmers, and the photovoltaic infrastructure.

4 Tools and Platforms Evaluated for the Creation of a Digital Twin

This section depicts the hardware and software tools that have been evaluated to implement the digital twin of the agrivoltaic greenhouse. This evaluation has been performed by testing the tools using an engine for irrigation, in the case of modeling objects, and our lab, for testing the digitalization of the environment. The aim of these tests is to gain knowledge of the features of each tool and assess their suitability for the implementation of this project's digital twin.

4.1 Software for Digital Twin Implementation

This subsection presents a review of the available tools that are either open source or offer a free-to-use version. The aim is to identify accessible solutions that can support the development and implementation of digital twins without incurring significant licensing costs.

CloudCompare: This open-source software tool is widely recognized in the 3D processing community and is designed for handling and analyzing point clouds and 3D meshes [23]. Its modular structure and efficient algorithms make it well suited for advanced post-processing of scan data. Key features include noise removal algorithms (such as Statistical Outlier Removal and RANSAC), spatial filtering, segmentation based on geometry or color, and registration methods like ICP to align datasets from different sources. It provides strong tools for metric analysis (such as distances, curvature, and normals), point cloud comparison (Cloud-to-Cloud and Cloud-to-Mesh) to assess deviations, and precise geometric transformations. It is especially useful for exporting point clouds to a wide range of 3D file formats, including those not natively supported by proprietary software, which is essential for ensuring compatibility in complex workflows.

Agisoft Metashape: This photogrammetry software is a solution for generating 3D models from a series of overlapping images [24]. Its core engine is based on Structure from Motion (SfM) and Multi-View Stereo (MVS) algorithms. Metashape performs 3D reconstruction by identifying and matching features across the input photographs. The workflow includes image alignment, creation of an initial sparse point cloud, generation of a high-density point cloud, construction of a polygonal 3D mesh, and finally, mesh texturing. Its ability to handle large image datasets (including gigapixel-scale projects) and manage the process semi-automatically makes it a standard tool for producing

photorealistic models. Additionally, it supports the direct import of pre-existing point clouds for further processing, such as mesh or texture generation, or integration with other datasets.

Jawset Postshot: As a plugin and post-processing tool, Jawset Postshot is designed to import, manipulate, and render volumetric point clouds within 3D content creation (DCC) environments such as Houdini or Cinema 4D [25]. Its main advantage lies in its ability to handle large-scale point datasets while maintaining smooth interaction in the viewport and producing high-quality renders. In the context of Gaussian Splatting, Postshot serves as an key component. It supports the import of 3D point-based data enriched with color, intensity, and volumetric density attributes. By integrating with rendering engines, it enables the generation of photorealistic images from these point-based models, accurately managing aspects such as occlusion, transparency, and lighting to simulate how light interacts with each Gaussian "splat", which is a fundamental aspect in volumetric photogrammetry applications.

Houdini: This 3D modeling software from SideFX is recognized for its node-based procedural architecture, which makes it highly suitable for simulations, visual effects (VFX), and the procedural generation of geometry and content [26]. In the context of point clouds, Houdini provides a flexible environment for detailed data manipulation. It supports the import and visualization of large-scale point cloud datasets, enabling operations such as denoising, filtering, downsampling, and geometric transformations. Its node system allows users to build complex processing chains for tasks like advanced segmentation, retopology, or converting point data into polygonal geometry. It is particularly well-suited for experimenting with advanced methods such as Gaussian Splatting, enabling the development and refinement of algorithms that convert point data into volumetric representations, or their integration into production workflows where spatial data processing is a central requirement.

Unreal Engine: Developed by Epic Games, Unreal Engine is a real-time graphics engine widely used in video game development and increasingly adopted in areas such as architectural visualization (ArchViz), virtual production, and simulation [27]. Its scalable architecture and advanced rendering pipeline enable the handling and display of high-density point clouds and 3D meshes generated from reality capture processes. It features a physically based rendering (PBR) system for creating realistic textures and surface details, along with a real-time global illumination system. The engine's ability to incorporate these models into interactive scenes, together with particle systems, physics, and cinematic tools, makes Unreal Engine one of the most popular platforms for creating immersive environments and interactive, photorealistic digital twins.

Twinmotion: Twinmotion, also developed by Epic Games, is a real-time architectural visualization and design tool focused on quickly creating immersive visualizations and experiences [28]. While it does not directly process point clouds, its strength lies in its ease of importing already processed 3D models (e.g., Revit meshes, Blender models) and enhancing them with PBR materials, global lighting systems, dynamic weather

effects, and vegetation and population elements. Its intuitive interface and real-time rendering capabilities make it well suited for producing architectural visualizations, static renders, animations, and interactive walkthroughs, including export to Virtual Reality (VR) experiences.

Unity: Unity is a widely used real-time graphics engine developed by Unity Technologies, applied across a broad range of fields including video games, simulations, interactive visualization, and immersive Virtual Reality (VR) and Augmented Reality (AR) experiences [29]. Its flexibility comes from a component-based architecture and support for C# scripting, enabling a high level of customization and development of specific features. Unity imports 3D models (meshes, objects) created from point clouds and builds virtual environments with realistic lighting, materials, and physics. It has a large ecosystem of tools and plugins, combined with the ability to deploy applications across multiple platforms (desktop, mobile, VRAR), that makes it another of the top choices for developing digital twins and immersive applications.

Revit: Developed by Autodesk, Revit is a Building Information Modeling (BIM) software widely used as a standard in the architecture, engineering, and construction (AEC) industry [30]. Its main function is to create parametric 3D models that represent not only the building's geometry but also detailed technical and semantic information about its construction components (materials, properties, relationships). This ability to integrate accurate data throughout the project lifecycle allows for collaborative design, construction documentation, and multidisciplinary coordination. In the context of digital twins for the built environment, Revit enables the creation of an information-rich "as-built" model, which supports the integration of reality capture data and change management. Its models can be exported to formats compatible with other visualization and simulation platforms, serving as the centralized database for an architectural digital twin.

ReCap Pro: Autodesk ReCap Pro is a reality capture tool designed to process and convert raw data from 3D laser scans and photographs into point clouds [31]. It is used in the AEC industry workflow for creating digital representations of existing environments. ReCap Pro provides features that can be utilized for both automatic and manual scan alignment, laser scan registration with images (photogrammetry), noise and artifact removal, and the merging of multiple scans into a single, unified point cloud. It can handle large point datasets and optimize them for use in other CAD/BIM applications. It also features native integration into platforms like Revit and AutoCAD, serving as a pre-processing tool for creating digital twins.

Blender: Blender is an open-source software for 3D content creation and editing. It includes a set of tools for modeling, sculpting, rendering, animation, simulation, and compositing [32]. In the context of digital twins, Blender is useful in post-processing 3D data obtained through reality capture. It supports the import of point clouds and, more commonly, 3D meshes generated from scanners or photogrammetric reconstructions. It includes geometry editing tools, such as retopology, mesh optimization, and

decimation, as well as capabilities for texturing and shading. These tools are used for cleaning, refining, and preparing these models for different applications, including real-time visualization and photorealistic rendering. It supports a wide range of 3D file formats (.obj, .ply, .fbx, .gltf) and has the ability to extend functionality through Python scripting. Therefore, Blender is a tool that can be utilized for adapting digital representations to the specific requirements of a digital twin.

NVIDIA Omniverse: NVIDIA Omniverse is an open-source platform built on Pixar's Universal Scene Description (USD) standard. It is designed for real-time 3D collaboration and photorealistic simulation [33]. In the context of digital twins, Omniverse functions as a central hub that connects various 3D applications, enabling the creation, simulation, and operation of digital twins. It also allows for linking tools from different domains, such as CAD, DCC, and simulation software, within a unified environment. Omniverse supports the integration of 3D data from multiple sources, provides physical simulation through its physics and AI engines, and offers real-time photorealistic rendering using ray tracing and path tracing techniques. It also includes a modular architecture and support for custom extensions.

4.2 Digitization of Environments

The first step in creating a Digital Twin is virtualizing the target environment to create a 3D representation as close to the original agrivoltaic greenhouse as possible. It is important to distinguish between the digital representation of the environment (a greenhouse in this case) and the digital model of the smaller elements that will include interactive features. The representation of the environment won't be as detailed as the models of each object to avoid performance issues. The approach considered for the creation of the 3D representation of the greenhouse is performing a scan of the target greenhouse. In order to scan the environment, two different techniques have been considered, which are LiDAR (Light Detection and Ranging) and photogrammetry.

4.2.1 Terrestrial laser scanners (LiDAR):

The LiDAR system is a remote sensing technology that enables the capture of precise three-dimensional information about the environment using laser light pulses. Its operating principle is based on emitting laser beams toward a surface and measuring the time it takes for each pulse to reflect back to the sensor. From this time-of-flight measurement, the distance between the sensor and the scanned object is calculated. By repeating this process millions of times per second and from different angles, it is possible to generate a detailed three-dimensional representation of the environment, known as a point cloud.

LiDAR has become an essential tool in various disciplines due to its ability to obtain high-resolution and high-accuracy spatial data. Its most popular applications are cartography and topography, where it is used to generate digital terrain models; civil engineering and architecture, for scanning buildings and infrastructures; and autonomous navigation systems, such as self-driving vehicles, which use LiDAR to detect obstacles and understand their surroundings in real time.

The **FARO Focus Premium 150** [34] is one of FARO's latest-generation 3D laser scanners, designed to offer high precision and efficiency. It has a maximum range of 150 meters, delivering distance accuracy of ± 1 mm, with a scanning speed of up to 2 million points per second.

The scanner is equipped with a laser that reflects off a mirror rotating at high speed, allowing the laser to be directed in all directions. Additionally, it has a rotating base that turns simultaneously to cover a full 360^{o} field of view. The scanner calculates the distance to objects by measuring the time it takes for the laser to return after bouncing off a surface. It is also equipped with a panoramic camera that captures spherical photographs while the scanner rotates, enhancing color resolution and reducing scanning time.

It is important to note that this system has a main drawback. Since measurements are performed using a laser, certain surfaces do not reflect the light properly, such as window glass. This requires the use of a special spray that eliminates reflections while preserving the original colors as much as possible.

For small, specific, and highly detailed objects, the **FARO Freestyle 2 Handheld Scanner** [35] is a high-precision portable scanner designed to capture this type of 3D data. With a precision of up to 0.5 mm and a resolution of up to 0.2 mm, this device uses structured light technology combined with CMOS sensors to acquire three-dimensional shapes and capture textures and colors with high fidelity. Its scanning capability covers a range from 0.4 meters to 5 meters, with a wide field of view of 45° x 56°, ensuring the detailed capture of data even in large or complex spaces.

It has an aproximate weight of 1 kg and USB connectivity, allowing smooth data transfer. Additionally, the inclusion of an integrated touchscreen provides control and the ability to visualize data in real time. It is also compatible with FARO's SCENE software. Furthermore, it features a snap-in function, which allows it to be connected to a FARO Focus project to complement its scanning capability. This is especially useful for reaching hard-to-access areas where the Focus may not reach, enabling the addition of a higher level of detail to the overall scan. In this way, the Freestyle 2 becomes an useful tool for a variety of applications, providing detailed and accurate data that fulfill highly demanding requirements.

The first step in creating a digital twin from scratch is to gather as much information as possible to build its representation. This can include photographs, plans of the object to be represented, or any type of digital resource. The more information available, the

higher the accuracy of the digital twin. In our case, since we have access to two FARO scanners, digitizing the asset can be quite straightforward, and the results are very precise.

Before starting the scanning process, the scanner must be calibrated. This involves performing a white balance and a series of movements while simultaneously aiming at the calibration plate included with the scanner. This calibration process is performed according the the manufacturer's instructions.

When scanning an object with the handheld scanner, care must be taken to avoid sudden movements. It is important to maintain an appropriate distance from the object, as well as a stable speed and consistent angles. The scanner provides physical and visual feedback to alert the user if any of these conditions occur. If the reference is completely lost, the user must return to a previous position, which will be displayed on the screen. To help prevent this from happening, the scanner uses numbered targets that are placed in the scanning area to aid recognition. In some occasions, there may be some blank spaces in the scan (see Figure 3) that can be filled in by resuming the scaning process until completion (see Figure 4). The previsualization of the project after performing the scans of the application of the handheld scanner is shown in Figure 5.

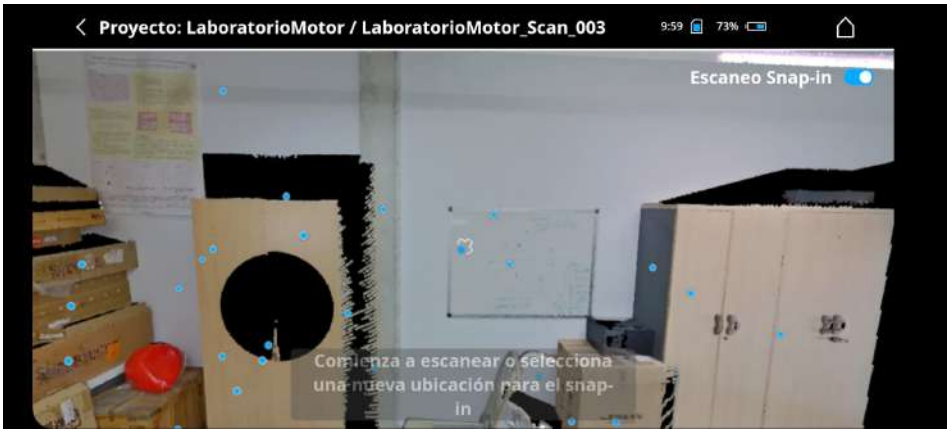


Figure 3: Method to fill in blanck spaces with *FARO Focus*

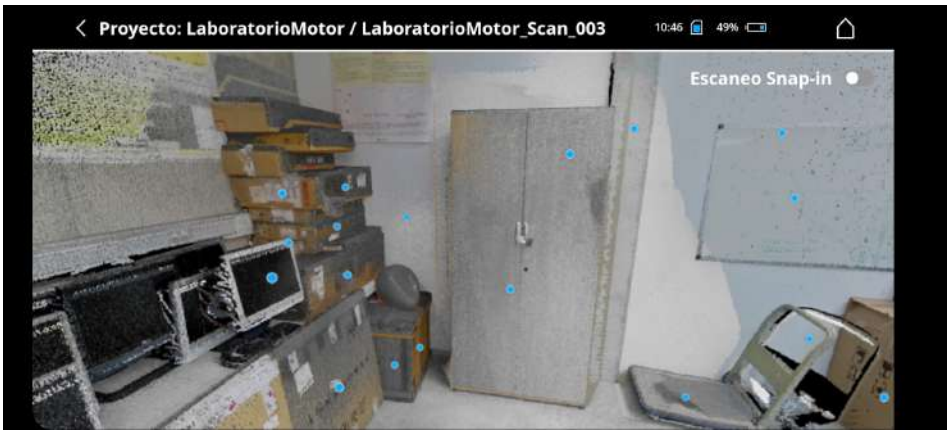


Figure 4: Complete scan done with *FARO Freestyle 2*

Figure 5: Previsualización del proyecto en la aplicación del escáner

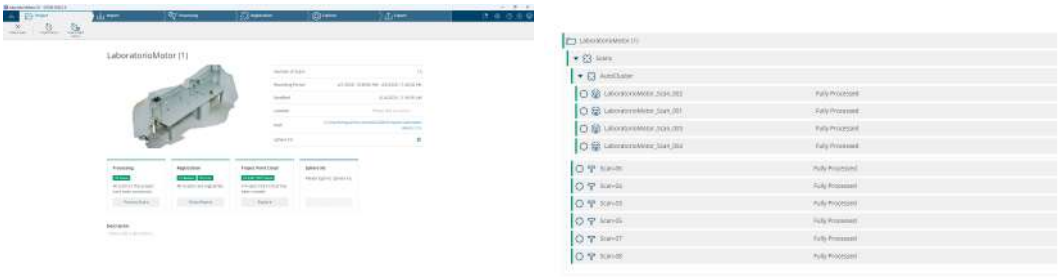
Once the scanning phase is completed, a series of .FLS files are obtained, each corresponding to a different scan performed. Each individual scan provides distinct information that, on its own, is not very useful, but when "stitched" together, they form the structure of the 3D point cloud. This is made possible by the FARO SCENE software.

FARO SCENE is developed by FARO [36] and it is the main tool used to process and register point clouds captured by its 3D laser scanners. Its main purpose is to align and combine multiple individual scans into a single, consistent, and geo-referenced point cloud, using feature-based matching and the Iterative Closest Point (ICP) method. It can handle large amounts of raw scan data, including filtering out noise, reducing point density, and isolating specific objects. It also includes tools to add color to the point cloud using images taken by the scanner, and basic options for 3D viewing and measurement. This software is an important first step in the data workflow, helping ensure the point cloud is accurate and reliable.

First, we must import the scans we consider necessary, if it is a selection of them, or the entire project, which we have previously copied to a high-speed USB 3.0 drive from the Freestyle 2 processing unit. Once we have the files, we proceed with data processing, the duration of which varies depending on the equipment characteristics and the size of the files.

Next comes the registration phase, where the "stitching" process mentioned earlier takes place. The software includes an automatic registration mode, which relies on targets and spheres. Targets are calibrated and numbered plates placed on various planes to help the scanner position itself in space. Spheres serve the same purpose, but their advantage over targets is that, due to their spherical geometry, they are easier for the scanner to detect from any angle.

If manual registration is chosen, the procedure can become somewhat tedious depending on the volume of scans, but the software offers tools such as automatic plane detection, which speeds up the identification of any of the three Cartesian directions. Additionally, it is possible to create clusters—groups of scans—allowing registration to be performed in several steps instead of all scans at once. The project preview and processed scans view in the FARO Scene software is showcased in Figure 6.



(a) Preview (b) Processed scans

Figure 6: Fase de procesamiento

Once the registration phase is complete, the point cloud can be created inan automatic process, and then the scanned replica can be visualized (see Figure 7. Additionally, clipping boxes can be created to section the scan, meshes can be generated for subsequent model export, and finally, a virtual scan can be created to upload to FARO's cloud storage system called FARO Sphere.

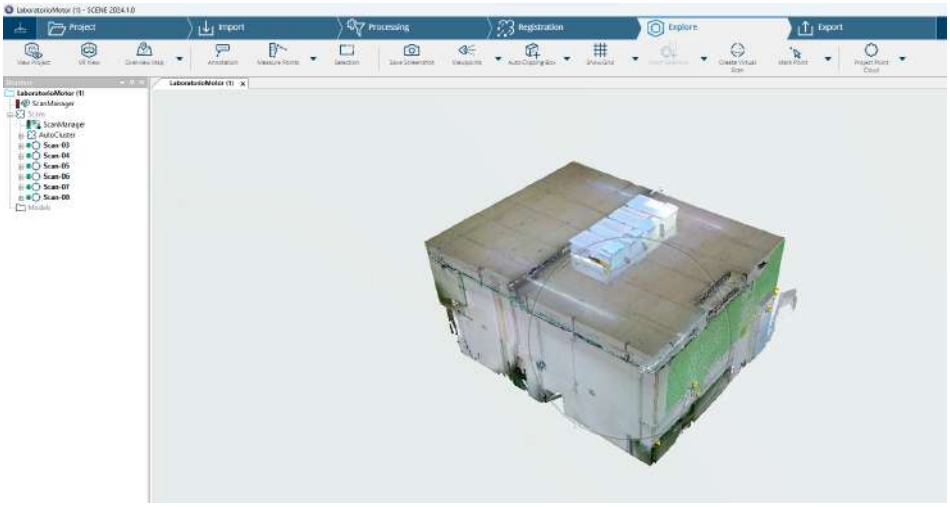


Figure 7: Complete pointcloud.

4.2.2 Photogrammetry

Photogrammetry is a technique that enables precise measurements and three-dimensional reconstructions of real objects and spaces from photographic images. This discipline is based on geometric and optical principles, especially stereoscopy, which allows depth to be inferred by observing the same point from different perspectives. Just as the human eye uses binocular vision to perceive three-dimensionality, photogrammetry employs multiple, usually overlapping, images to calculate the shape and position of the photographed elements.

Through complex mathematical algorithms, which require knowledge of the camera's internal characteristics and its relative position at the time of each capture, triangulation is performed to digitally reconstruct surfaces and structures with high accuracy. This ability to generate three-dimensional models without direct contact makes photogrammetry a useful tool in fields where physical intervention may be limited or delicate.

The applications of this technique are very diverse. In cartography and geodesy, it is used for creating maps and digital terrain models. In architecture, archaeology, and heritage conservation, it allows detailed documentation and analysis of buildings or sites. It has also become essential in civil engineering, mining, environmental management, and more recently in industries such as entertainment, virtual reality, and digital twin generation. This versatility has been driven by the development of specialized software and the use of drones, digital cameras, and powerful processors that automate much of the workflow.

Its advantages include its non-invasive nature, high accuracy, applicability to environments of vastly different scales, and relatively low cost compared to other technologies such as laser scanning. However, it also has certain limitations, such as the need for adequate lighting conditions, dependence on image overlap, and sensitivity to reflective or low-contrast surfaces. Additionally, the analysis and reconstruction processes can be demanding in terms of computational resources.

In order to test the photogrametry tools, a set of videos were filmed at the lab to be later procesed to create a point cloud.

COLMAP

COLMAP is an open-source 3D reconstruction tool widely used in academic settings. It enables complete photogrammetric reconstructions and offers both a graphical user interface and a command-line interface. The are some prerequisites to ensure the tool is used correctly:

- The images need to be captured with sufficient overlap and varied angles to obtain a good reconstruction of the 3D space.
- COLMAP needs to be adequately installed. As an option, CUDA support could be included.
- Some basic knowledge on basic camera parameters and photogrammetric workflows is advised.

The workflow that needs to be used to work with COLMAP is the following:

1. **Project creation:** `File → New Project`. Definition of working folder and image import.
2. **Feature extraction:** `Feature Extraction` using SIFT.
3. **Feature matching:**
   - `Exhaustive Matching`: small datasets.
   - `Spatial/Sequential Matching`: if there are metadata or sequence.
4. **Sparse reconstruction:** `Start Mapper`. Generation of camera poses and initial point cloud.
5. **Alignment verification:** visual inspection of the model and export from `sparse/0`.
6. **Dense reconstruction (optional):**

- `Run Stereo`: disparity map computation.
- `Run Fusion`: depth fusion for dense cloud.

7. **Exporting results:** `File → Export Model` in `.ply`, `.obj`, `.txt`.

Object Digitization

1. Scanning
   Handheld or mobile scanners: FARO Freestyle 2, which allow for a more dynamic and flexible capture, especially useful for hard-to-reach areas or to complete zones not covered by static scans.

2. Photogrammetry
   The previously mentioned tools (COLMAP, Metashape, RealityCapture) are also used to digitize objects, provided that a sufficient number of well-lit images with good coverage are available.

3. Adding interactivity to the objects.

Unity

In **Unity**, interactivity is implemented through **C# scripts**, supported by physical components and the event system. The basic workflow includes:

- **Colliders and Rigidbodies**: Objects must have *Colliders* to detect clicks or collisions. Optionally, a *Rigidbody* is added if physical response is desired.
- **Interaction scripts**: Functions like `OnMouseDown()`, `OnTriggerEnter()`, or *raycasts* from the camera are programmed to detect interactions.
- **Event system and UI**: Using the *Inspector*, events are connected to public functions. Buttons, sliders, and other UI elements can also be used.
- **XR and AR**: Using the *XR Interaction Toolkit*, interactions in virtual or augmented reality are enabled, such as grabbing, releasing, or rotating objects.

5 Export of the point cloud to specialized software

Once the data capture through 3D scanning is completed, the next step is to export the resulting point cloud to specialized software that enables detailed processing and analysis. This data transfer is crucial for performing tasks such as noise removal,

segmentation, alignment of multiple scans, and the generation of more complex three-dimensional models. This section describes the file formats used for the export process, as well as the tools employed to ensure efficient and accurate integration of the point cloud into the selected software environment.

5.1 Tool workflow

This section outlines the workflow of the tools evaluated.

5.1.1 CloudCompare:

CloudCompare has been primarily used as an intermediate tool for converting point clouds between different formats and optimizing their size through subsampling techniques. This process is crucial when working with high-density point clouds, as it significantly reduces the number of points without compromising the overall geometry of the scene, thereby facilitating subsequent use in software with performance or loading capacity limitations.

Within the *Edit* > *Subsample* menu, CloudCompare offers three main point cloud reduction methods:

- **Random**: this method selects an exact number of randomly chosen points from the entire cloud. It is useful when a representative sample is needed regardless of the spatial distribution of the points. However, it may result in uneven density regions, as the selection does not consider point positions.
- **Random %**: instead of specifying a fixed number of points, this method allows defining a percentage of the total. For example, choosing 10% will retain one tenth of the original points, randomly distributed. It offers similar advantages to the previous method but provides a more direct proportional estimate.
- **Spatial**: this option divides the 3D space into a regular cubic grid (defined by a point spacing parameter) and selects a single representative point per cell. The result is a more evenly distributed and spaced cloud, ideal for preserving the overall shape of the object or environment without overpopulated areas. It is the most effective method when seeking a balanced reduction that remains consistent with the spatial structure of the data.

The general workflow begins with the import of the point cloud in its original format [8] (such as `.xyz`, `.e57`, or `.las`). Once loaded, the cloud can be visualized, inspected, and cleaned if necessary by removing noise or isolated points. Next, the most appropriate subsampling method is applied according to the objective and the size of the cloud.

Finally, the reduced cloud is exported in the desired format compatible with other tools in the workflow, such as `.ply`, `.obj`, or `.txt` [9]. This functionality is essential when working with software that does not directly support the formats generated by the scanner or that requires lighter files to function properly.

Figure 8: Visualization of the point cloud in CloudCompare.

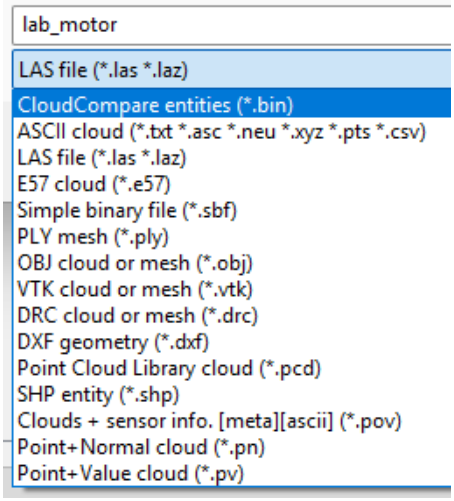


Figure 9: Export formats of the point cloud.

5.1.2 Agisoft Metashape:

With the aim of exploring automatic methods for generating 3D models from point clouds, an evaluation process was carried out using **Agisoft Metashape** [24] in its *Standard* and *Professional* versions, both in free trial mode.

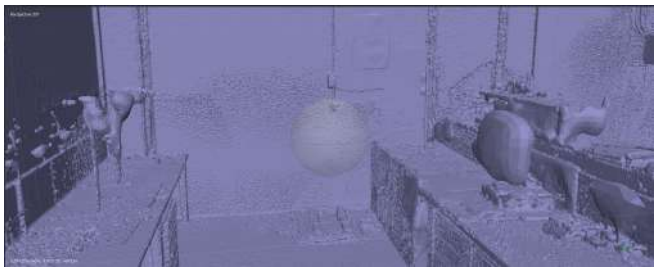
Agisoft Metashape is specialized photogrammetry and 3D reconstruction software that, besides generating models from sets of images, offers capabilities for importing dense point clouds and subsequent semi-automatic creation of 3D meshes.

The tests followed the following procedure:

1. **Point cloud import**: The point cloud previously generated by 3D scanning was directly imported in compatible formats such as `.ply` and `.xyz`.
2. **Processing parameter configuration**: Within Metashape, settings related to cloud density, point classification thresholds, and target resolution for mesh reconstruction were adjusted.
3. **3D model generation**: The program's **"Build Model"** option was used, which allows creating a three-dimensional model directly from the imported point cloud.
4. **Export**: An attempt was made to export the generated models in standard formats (`.obj`, `.fbx`, among others) for subsequent integration into graphics engines and virtual reality environments.

During the tests, the following issues were observed:

- **Program instability**: Due to the large size and high density of the point cloud (tens of millions of points), Metashape experienced recurrent *crashes* during the mesh generation stages, even when attempting to reduce the processing quality.
- **Inconsistent results**: On occasions when the mesh reconstruction was successfully completed, the generated model showed poor quality, with overly simplified geometries or, conversely, incoherent structures due to the program's difficulty in handling such a large volume of input data. However, in several attempts, a higher-quality meshed model was obtained, although significant inconsistencies persisted, such as incomplete surfaces, disproportionate volumes in some laboratory objects, and visible deformations on flat surfaces like walls and floors, which exhibited bulges and roughness uncharacteristic of their nature. These contrasts in the results can be seen in Figures 10a and 10b, which show an example of a poor-quality mesh and another with a notable, though still imperfect, improvement, respectively.



(a) Model with poor reconstruction

(b) Model with improved quality, although with persistent errors.

Figure 10: Comparison between two mesh reconstruction results.

- **Hardware limitations and optimization**: Despite having optimization options, the need to handle large files exceeded the practical capabilities of the trial version used, which directly impacted the sustainability of the workflow.

Consequently, although Agisoft Metashape proved to be a powerful tool in contexts with smaller data volumes, its use was **discarded** for the development of this project due to its inability to handle the full required point cloud in a stable and efficient manner.

5.1.3 Unreal Engine

In **Unreal Engine**, interactivity is primarily based on the visual **Blueprints** system, although it can also be developed using **C++**. The fundamental steps are:

- **Collision and detection**: Object collision is enabled using components such as *Box Collision* or *Sphere Collision*, and the object is marked as interactive.
- **Events in Blueprints**: Events like `OnBeginCursorOver`, `OnClicked`, or `OnComponentBeginOverl` are used to define interaction logic.
- **Input configuration**: Actions or axes are defined in the *Input Mapping* system to bind keys, mouse, or VR controllers to interactive functions.

- **UI and advanced interaction**: Using **Widgets** created in *UMG*, graphical interfaces can be displayed when interacting with objects.
- **Custom logic**: Conditions, variables, animations, and visual effects are combined to define dynamic behaviors.

With the aim of creating an immersive environment that combines real data obtained through 3D scanning with visualization in virtual reality, a complete workflow has been developed in **Unreal Engine**. This environment includes both the laboratory point cloud and an individual meshed model of the engine, which is monitored via sensors. The following steps were followed.

Unreal Engine 5 was used, configuring the project to support virtual reality experiences. To this end, the following essential *plugins* were enabled:

- **LiDAR Point Cloud Plugin**: allows the import, visualization, and manipulation of point clouds directly within the engine.
- **Meta XR Plugin**: provides compatibility with virtual reality devices such as Meta Quest 2 and Quest Pro, enabling full immersion in the 3D environment.

The activation of the *LiDAR Point Cloud Plugin* is done through Unreal Engine's plugin management menu, as shown in Figure 11.

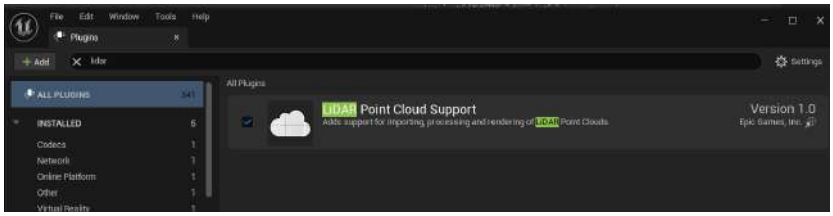


Figure 11: Activation of the LiDAR Point Cloud plugin before starting the project.

Once the plugins are enabled, the engine is restarted to apply the changes correctly.

The point cloud generated from the laboratory scan is imported through the *LiDAR Point Cloud Plugin*. This allows adjusting visualization parameters such as point size, intensity, and custom colors.

Several visualizations have been generated, both in light tones (Figure 12) and in dark tones (Figure 13), in order to study spatial perception and contrast under different lighting and background conditions.

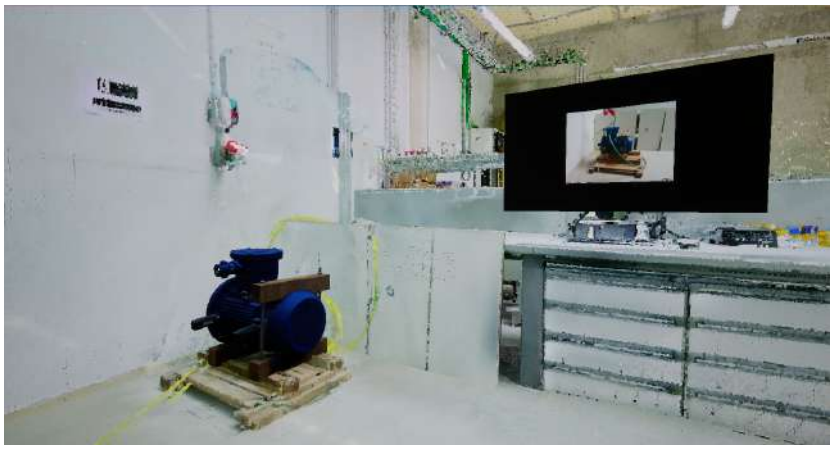


Figure 12: Visualization of the point cloud in light tones.

Figure 13: Visualization of the point cloud in dark tones.

In addition to visualization, the functionality of the *LiDAR Point Cloud Plugin* has been used to generate a 3D mesh directly from the point cloud. The procedure consists of selecting the cloud, right-clicking on it, and choosing the **"Create Static Mesh"** option, as shown in Figure 14.

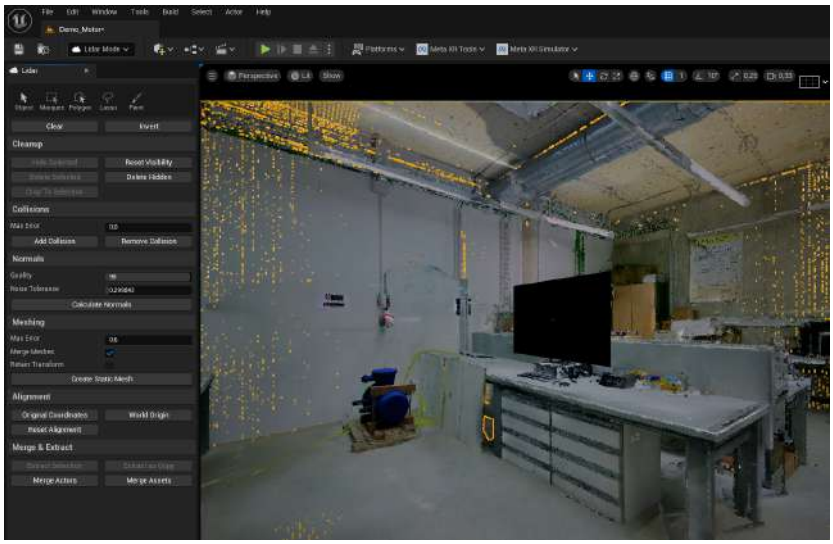


Figure 14: LiDAR plugin menu where the "Create Static Mesh" option is selected.

In this operation, a maximum allowable error value is set, configured in this case to 0.6 millimeters, to maintain a balance between geometric detail and efficiency. The result is a meshed 3D model that can be used as a physical surface, structural reference, or to apply collisions in the environment, as shown in Figure 14.

The meshed model shown in the figure presents both positive features and relevant limitations. On the one hand, it is a high-quality mesh, especially considering that it was generated using an *Unreal Engine* plugin, making it fully functional within this environment. However, several drawbacks can also be observed. One of the most evident is the appearance of bands or lines running across the mesh, a direct consequence of the scanner's movement during the acquisition of the point cloud.

To obtain the meshing shown in Figure 15, it was necessary to use a point cloud with the highest amount of information possible, which in this case reached approximately 213 million points. This high density caused multiple program crashes before achieving the final reconstruction. Although the mesh quality improves proportionally with the cloud density, this poses an additional problem: the resulting model is so large that it cannot be exported out of *Unreal Engine*, making external processing impossible to, for example, remove the bands generated by the scanner.

Additional tests have been carried out by rasterizing the point cloud in order to reduce its size and make export feasible. However, this process entails a significant loss of mesh quality, which limits its usefulness beyond the Unreal environment.

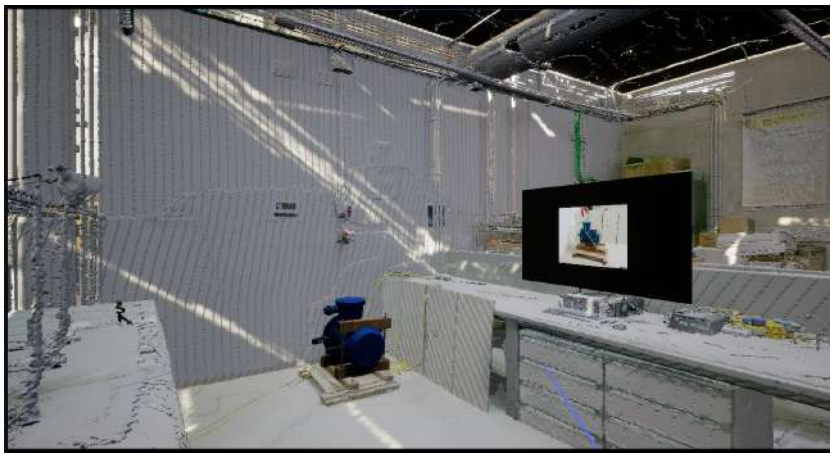


Figure 15: Automatically generated mesh from the point cloud in Unreal Engine.

The individually pre-generated 3D meshed model of the engine is imported and positioned accordingly within the point cloud or on the generated mesh. This model is linked to an external sensor system, whose data is transmitted in real time to Unreal Engine via a connection based on protocols such as `MQTT` or through serial communication using *Blueprints* or `C++` code.

The values captured by the sensors, such as temperature, vibrations, or rotation speed, are displayed through informational panels, color indicators, and overlaid graphs within the virtual reality environment.

Finally, after integrating all the above elements, the environment is configured for exploration in virtual reality using the **Meta XR Plugin**. Free navigation is enabled, interaction with floating panels, and focus on critical areas of the engine.

5.1.4 Houdini y GSOPs:

With the aim of exploring alternatives for generating Gaussian Splatting representations from point clouds, the use of **Houdini** in combination with the *GSOPs* repository developed by *David Rhodes* [37] has been evaluated.

This repository provides a set of operators (*SOPs*) specifically designed to work with point data in Houdini, allowing the conversion of standard point clouds into optimized representations for splatting techniques, generating files compatible with advanced volumetric rendering methods.

The workflow with GSOPs in Houdini is as follows:

1. **Point cloud import**: Standard format files such as `.ply` or `.xyz` were loaded into Houdini using geometry reading nodes (*File* SOP).

2. **Conversion using GSOPs**: Based on the imported geometry, nodes provided by the *GSOPs* repository were used to transform the point cloud into three-dimensional Gaussian representations. This process involves creating attributes such as position, orientation, and scale for each Gaussian.

3. **Export of the generated model**: Finally, the data was exported in compatible formats for subsequent rendering or visualization processes, such as `.json` or `.bin`.

During this workflow, manual adjustments were made to the parameters of density, point size, and spread to study their influence on the final result.

Import and processing tests were conducted using various point cloud samples:

- **Complete point clouds of the laboratory**: Attempts were made to process high-density clouds generated by the full 3D scanning of the environment, with files containing more than 50 million points.
- **Point clouds of individual objects**: Tests were also conducted with smaller `.ply` files corresponding to specific objects such as the engine or laboratory furniture, ranging between 1 and 8 million points.

However, a *critical limitation* was found in the free version of Houdini (Houdini Apprentice): a maximum of *8 million points* could be processed per scene. This restriction prevented working with the complete point cloud of the laboratory, as it was necessary to drastically reduce the resolution or fragment the data, compromising the quality and continuity of the digital twin.

Even after successful tests with smaller objects within the allowed limit, it was concluded that the tool was not viable for the full development of the project due to this scalability limitation.

Therefore, after evaluating these initial tests, the use of Houdini and GSOPs was *discarded* as a solution for the final conversion and visualization of the laboratory's point clouds, redirecting efforts towards more scalable alternatives specific to large data volumes.

5.1.5 TwinMotion:

Twinmotion is software developed by Epic Games focused on real-time architectural and urban visualization. Its ease of use, library of pre-designed objects, and capability to generate virtual walkthroughs make it an effective solution for visually appealing representations of digital twins. Although it does not allow connectivity with sensors or data analysis, it is highly effective as a passive visualization tool, especially useful in communication, design, and presentation phases of environments generated from point clouds or BIM models.

1. **Importing the point cloud and setting up the environment**
   One of the most notable features of Twinmotion is its ability to directly import point clouds in formats such as .las or .e57, without the need to previously convert them into meshes. These clouds are integrated as visual reference objects, static but with high geometric fidelity [16], allowing for contextualization of scanned spaces such as laboratories, industrial warehouses, or urban environments. Once the point cloud is incorporated into the environment, its scale, orientation, and position are adjusted. This can be done manually or with reference to additional models. Subsequently, the overall project environment is configured by adding elements such as lighting, weather, HDRI skies, or predefined landscapes, which enhance the visual readability of the model.

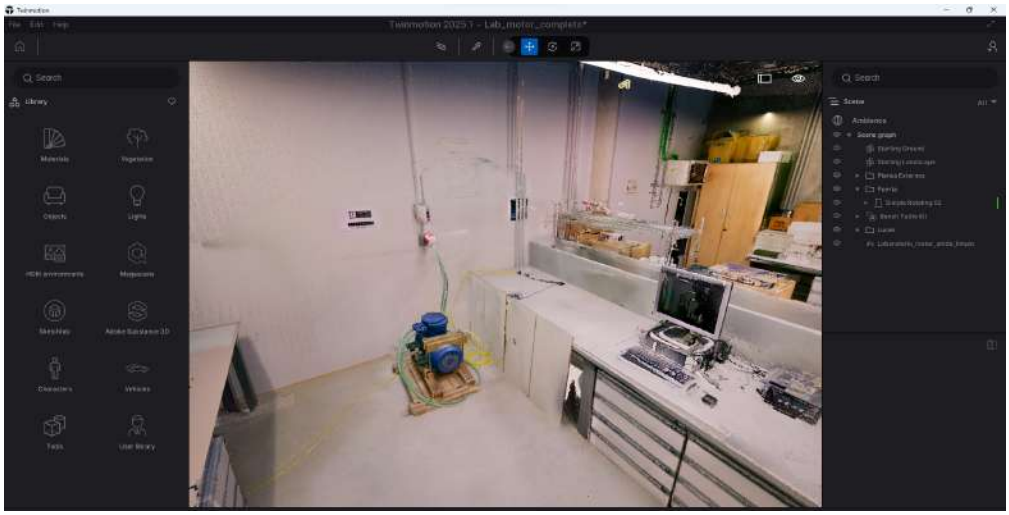


Figure 16: Vista en TwinMotion de la nube de puntos.

2. **Insertion of objects and scene staging**
   Twinmotion features an extensive library of objects including furniture, animated people, vegetation, vehicles, and machinery that can be freely placed in the environment. Although these elements lack functional logic or connectivity, their incorporation helps enrich the scene, facilitate spatial understanding, and provide narrative to the digital twin. Cameras with predefined paths can also be inserted to generate visual sequences or static views that simulate real walkthroughs of the virtual environment. This type of staging is especially useful in educational or outreach settings, where the goal is to bring the digital twin closer to non-technical audiences without requiring the use of complex tools.

3. **Production of visualizations and interactive presentations**
   One of Twinmotion's strengths is its ability to quickly and easily generate multimedia content. The user can export:

   - High-quality rendered still images.

- Video animations with camera transitions, weather changes, or time-lapse sequences.
- Interactive 360° panoramic tours, navigable from web browsers.
- Executable presentations (Twinmotion Presenter) that allow exploring the environment without installing the full software.

These features are key for presenting the digital twin in educational settings, conferences, or decision-making processes, where visual comprehension of the space takes precedence over technical analysis (see Figure 17).

(a) Point cloud visualization without the engine.

(b) Point cloud visualization with the engine's meshed model.

Figure 17: Different visualizations of the point cloud in TwinMotion.

4. **Export to Unreal Engine for interactive environments**
   Although Twinmotion is essentially a passive visualization tool, it allows the complete export of the project to Unreal Engine using the `.udatasmith` format.

This transfer preserves geometries, materials, lighting, and cameras, enabling further development in Unreal to incorporate interactive features, programmed logic, or real-time data integration. However, this export has a significant limitation: only those elements created directly within the Twinmotion environment—either through the software's integrated library or from tangible files in formats such as `.obj`—are correctly transferred. In the tests performed, it was not possible to export the point cloud to Unreal Engine, which limits the ability to directly utilize scanned data at this stage.

5. **Integration with other BIM and CAD platforms**
   Another notable point is its direct integration with tools such as Revit, Archicad, SketchUp, or Rhino through synchronization plugins (Direct Link). This connectivity allows automatic updating of changes made in the modeling software within the Twinmotion environment, improving efficiency in collaborative workflows. In BIM projects, this feature facilitates a seamless transition between technical modeling and final visualization, without the need for manual exports or format conversions.

In this context, Twinmotion stands out as a particularly effective tool for passive visualization, thanks to its intuitive interface, fast response times, and visually appealing results. However, when the project's goal includes integrating monitorable elements, real-time data interaction, or programmed logic, it is more convenient to develop directly in Unreal Engine. Its greater customization capability, along with native support for a wide range of formats and systems, makes it a more suitable platform for building functional and dynamic digital twins from the early stages of the project.

5.1.6 Autodesk tools

Revit:

Autodesk Revit is a fundamental tool in BIM (Building Information Modeling) environments and allows precise modeling of infrastructures based on information captured in the field, such as point clouds. Below is a workflow oriented towards the generation of digital twins from this type of data:

1. **Importing the point cloud**
   The first step is to import the point cloud into the Revit environment. It is recommended to use Autodesk ReCap beforehand to clean, register, and export the data in compatible formats such as .rcp or .rcs. Once prepared, these files can be inserted into the Revit model as a visual reference for modeling.

2. **Alignment and coordinate system definition**
   It is essential to ensure the point cloud is correctly positioned and oriented relative

to the project's coordinate system. Levels, grids, and base points are defined, facilitating the structuring of the modeling and integration with other models or disciplines.

3. **BIM modeling based on the point cloud**
   Using the point cloud as a visual guide, the manual reconstruction of architectural, structural, or installation elements (walls, floors, doors, ducts, etc.) is carried out, as shown in Figure (18). This phase is time-intensive and requires experience to correctly interpret the scanned geometries. When necessary, custom families are created to represent non-standard elements.

4. **Enriching the model with information**
   Once the geometric elements are modeled, relevant technical information is incorporated: materials, physical properties, lifespan, regulatory codes, among others. This transforms the model into a true digital twin by containing both geometry and associated data.

5. **Export and interoperability**
   The generated model can be exported in various formats depending on the final objective: .IFC for BIM interoperability, .FBX or .OBJ for visualization, or through specific plugins (such as Unity Reflect or Datasmith) for game engines or simulation platforms.

6. **Integration with IoT systems and monitoring platforms**
   Revit can be linked to platforms like Autodesk Forge, Azure Digital Twins, or WillowTwin for sensor integration and real-time monitoring of the built environment. These connections enable obtaining a functional digital twin that reflects the operational status of the building or infrastructure.

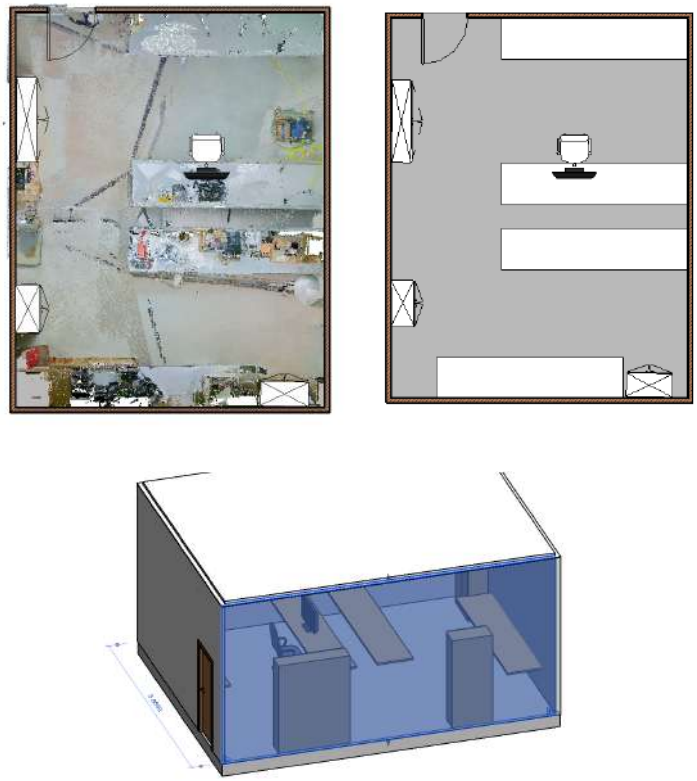


Figure 18: Plans in Revit

Advantages and disadvantages of using Revit for digital twins

- Ventajas:
  - Highly detailed and structured model.
  - Compatibility with BIM standards (IFC).
  - Integration with IoT platforms and immersive visualization.
  - Extensive support and Autodesk tools ecosystem.
- Disadvantages:
  - Manual modeling from point clouds is laborious and requires specialized training.
  - High resource consumption with dense point clouds.
  - High license costs.
  - Steep learning curve for new users.

– Limited visual fidelity of the environment due to the absence of elements such as furniture and lack of realistic textures.

Autodesk ReCap Pro:

Autodesk ReCap Pro holds a strategic position in workflows involving the conversion of scanned data into interpretable three-dimensional geometries. This software has established itself as an effective tool for automatically transforming point clouds into mesh models, reducing the technical complexity of the process and facilitating interoperability with other digital environments, especially within the Autodesk application ecosystem.

ReCap Pro supports the direct import of point clouds in widely used standard formats such as `.e57`, `.rcs`, `.xyz`, and `.ptx`. This compatibility enables an efficient transition from the acquisition phase to the digital working environment without the need for intermediate conversions. Once the data is imported, the user can intuitively visualize and manage the cloud, define areas of interest, perform spatial cuts, and apply basic cleaning or segmentation filters.

One of the most relevant features of ReCap Pro is *Scan to Mesh*, which allows generating a three-dimensional mesh from a point cloud with minimal user intervention. This process is carried out through proprietary algorithms that reconstruct the surface from the captured data, producing a closed, textured, and optimized mesh ready for export. Processing can be done locally or via cloud services, which is especially useful for projects requiring rapid iteration or when working with limited-capacity workstations [38].

The automatic mesh generation allows configuring quality and resolution levels according to project needs, balancing geometric detail with computational performance (see Figure 19). This feature is particularly important in digital twin projects, where spatial fidelity must be combined with the efficiency of the final model.

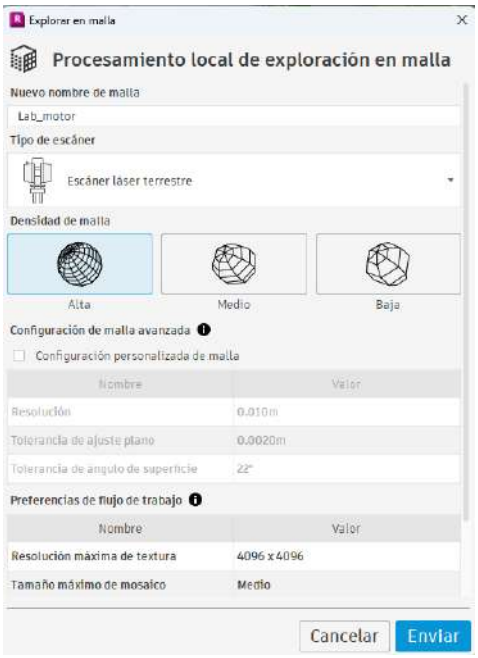


Figure 19: Configuration for mesh creation in ReCap Pro

Once generated, the mesh can be exported in various standard formats such as `.OBJ`, `.FBX`, and `.RCS`, ensuring compatibility with applications like Revit, 3ds Max, AutoCAD, Twinmotion, Unity, or Unreal Engine. This versatility makes ReCap Pro a bridging tool between data acquisition and modeling, visualization, and simulation environments. In this project, the export was done in `.OBJ` format, prioritizing compatibility with visualization platforms such as Twinmotion and GSOPs [39].

Within the scope of this work, ReCap Pro was used to transform the point cloud obtained with FARO Focus and FARO Freestyle 2 scanners into a three-dimensional mesh representing the laboratory under study. The selection was limited to areas relevant for the digital twin simulation, applying a high resolution level to achieve the best possible quality.

The integration of this mesh into visualization platforms was performed without structural adjustments, demonstrating the effectiveness of the automatic conversion process and the suitability of the generated format (see Figure 20).

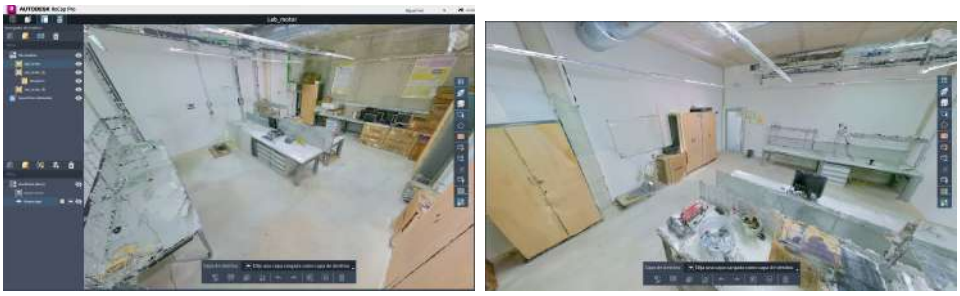


Figure 20: Visualization of the generated mesh

5.1.7 Blender

The process of editing three-dimensional meshes in Blender becomes essential when working with models obtained through laser scanning or photogrammetric reconstruction. These meshes, generally exported in formats such as `.ply`, `.obj`, or `.fbx`, often contain imperfections, high polygon density, or unnecessary fragments that need to be corrected before being integrated into a digital twin.

The workflow begins with importing the model into a new Blender scene via the *File ¿ Import* menu, selecting the appropriate format (see Figure 21). Once loaded, it is important to check the scale and orientation of the object, as it is common for them to be imported with incorrect transformations relative to the global coordinate system.

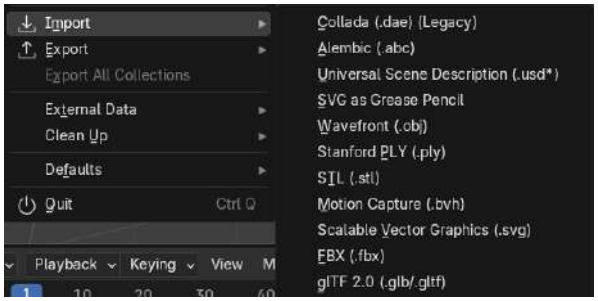


Figure 21: Blender Import Menu

After importing, enter *Edit Mode* to begin structural cleanup. Blender allows you to remove duplicate or overlapping vertices using the *Merge by Distance* tool. It is also advisable to use the loose elements selection to identify and delete floating or disconnected geometry.

Once the geometry is cleaned, the **Decimate** modifier can be applied (see Figure 22), which allows progressive reduction of the polygon count while maintaining the overall shape of the object. This step is essential when optimizing the model for real-time use.

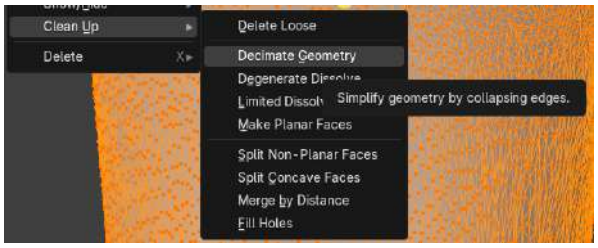


Figure 22: Decimate Modifier

The next step involves fine topology repair. Blender offers the Sculpt Mode, which includes a wide range of tools (see Figure 23) such as Smooth to soften surfaces and Grab to correct localized deformations.

Figure 23: Sculpt Mode Tools

If the mesh does not contain valid UV coordinates, an *unwrap* is performed from the *UV Editing* workspace. For complex meshes without logical topology, it is common to use the *Smart UV Project* option, which performs an optimized automatic projection.

Once the UVs are projected, materials are created and textures are assigned from the *Shader Editor*, linking nodes such as *Image Texture* to the *Base Color* channel of the main material.

Finally, the model is exported from *File* > *Export*, selecting the most suitable format for the final environment (for example, `.glb` for Twinmotion or `.fbx` for Unreal Engine) as shown in Figure 24.

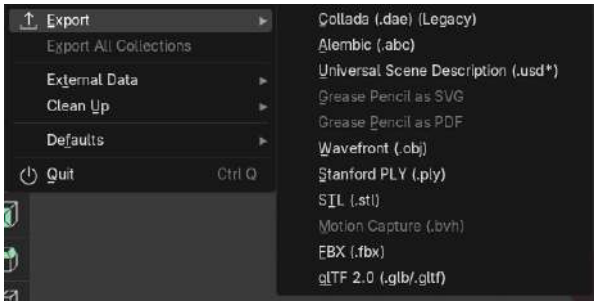


Figure 24: Available export formats

This workflow allows transforming a raw mesh obtained from a scan or reconstruction into an optimized, textured model prepared for use in interactive environments,

simulations, or immersive visualizations, meeting the technical and aesthetic requirements demanded in digital twin projects.

5.1.8 NVIDIA Omniverse USD Composer

The process in Omniverse USD Composer begins by importing the point cloud into the workspace. This cloud is used as a reference to recreate the laboratory, allowing precise visualization of the structure and furniture locations [40]. It is crucial to thoroughly analyze the point cloud, as any imperfections or architectural details of the laboratory can only be accurately replicated if the spatial information provided by the cloud is correctly interpreted.

After establishing the structure, the next step is to add the laboratory's fixed furniture and equipment. For this, we use the Omniverse USD Composer *Marketplace*, which offers a wide variety of elements, both generic and specific (see Figure 25). It is important to select these resources based on how physically similar they are and whether their dimensions match the actual objects in the laboratory. Often, the imported elements require adjustments in scale, rotation, and position to fit perfectly into the scene. If any furniture or equipment is not available in the Marketplace or has very particular features, we will need to model it in an external software compatible with USD and then import and adjust it within Omniverse.

(a) NVIDIA assets.

(b) Assets Marketplace

Figure 25: Available content

At this stage, we focus on ensuring a true-to-life appearance of the digital environment by applying materials and textures to all surfaces. Thanks to Omniverse, we can use physical materials (MDL). This allows us to achieve a highly realistic presentation of the decor and furniture in the architecture laboratory (see Figure 26).

Lighting is another important aspect in building a digital environment. It is necessary to restore the position, type, and intensity of artificial light sources in the labora-

tory, as well as to model the natural light entering through windows.

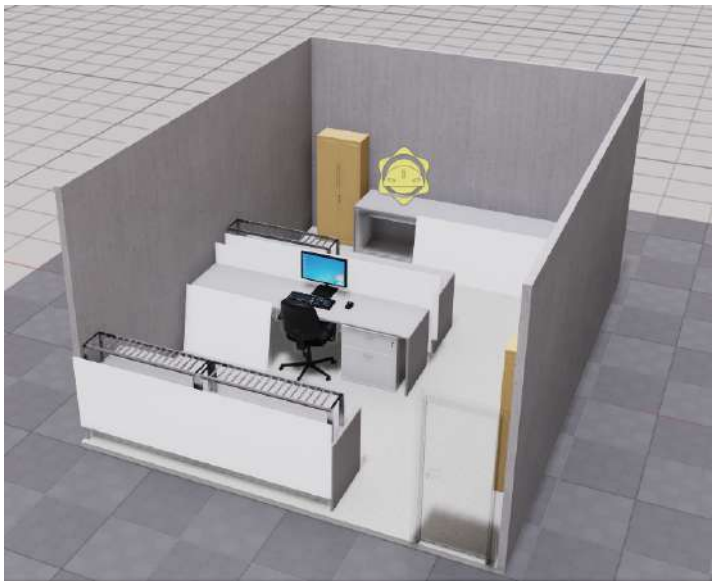


Figure 26: Visualización del proyecto en Omniverse

To organize the model in Omniverse USD Composer, it is ideal to structure it into logical and hierarchical groups, leveraging the capabilities of USD (Universal Scene Description) to manage layers and references. This facilitates collaborative work, version control, and future expansions or modifications of the digital environment. In fact, Omniverse USD Composer is designed for teamwork, allowing multiple users to work simultaneously on different parts of the model and see changes in real time.

Finally, Omniverse's interoperability with other applications through connectors (e.g., for Revit, 3ds Max, Blender, Rhino, Siemens NX, among others) makes it possible to import existing architectural or equipment models and export the digital environment for use in other contexts, such as advanced visualization, virtual reality, or physical simulations. This workflow ensures that the laboratory's digital environment is an accurate, visually coherent replica, ready for integration into collaborative and multidisciplinary processes [41].

5.2 Gaussian splatting for photogrammetry result management

Gaussian splatting is a modern 3D visualization technique that enables the representation of scenes from point clouds without the need to reconstruct traditional surfaces or meshes. Instead of working with polygonal geometry, this technique associates a small three dimensional Gaussian to each point, essentially an ellipsoidal "blob" that can adapt to the shape of the real world environment.

These Gaussians not only have a position in space, but can also contain information such as color, transparency, and orientation. When projected onto the final image, they produce a continuous, smooth, and visually realistic representation, even when the data is sparse or contains noise.

One of the main advantages of *Gaussian Splatting* is that it does not require a 3D mesh reconstruction, which saves significant time and simplifies the process. This makes it especially useful when working with scanned real world environments, where generating a clean and coherent mesh can be both complex and time consuming. Moreover, since it conforms well to the original scanner data, it preserves a high level of visual fidelity, which is crucial for applications such as digital twins or virtual walkthroughs.

In this project, the technique is used as the final step of the workflow, with the aim of creating an immersive, high quality visual experience that faithfully represents the scanned environment.

5.2.1 Technical Functioning of Gaussian Splatting

Once the conceptual basis of Gaussian Splatting is understood, it is important to delve into its technical functioning in order to understand why it is so effective at representing complex scenes from point clouds.

The process begins with a three-dimensional point cloud, typically obtained through 3D scanning or photogrammetry. Each of these points is transformed into an *anisotropic Gaussian*, meaning they are not simple symmetrical spheres but ellipsoids that can vary in shape, orientation, and size. This is achieved by assigning each Gaussian a covariance matrix that defines its spatial spread.

In addition to their shape, each Gaussian contains visual attributes such as color (RGB), opacity (*alpha*), and a contribution weight. These elements enable a rich representation filled with information and smooth transitions between points, which is essential when the cloud density is not uniform.

The final representation is generated through the *projection of the Gaussians* onto the image plane of a virtual camera. This process takes into account the intrinsic and extrinsic parameters of the camera to correctly position each Gaussian in the image. A visual composition is then performed in depth order using techniques such as *alpha blending*, which allows proper handling of transparency and occlusion.

A fundamental part of Gaussian Splatting is the *optimization* phase. Based on a set of real images, the parameters of each Gaussian such as position, shape, color and transparency are adjusted to minimize the difference between the original images and the rendered ones. This adjustment is carried out using numerical optimization techniques, often supported by machine learning methods, although it does not require complex neural networks.

Thanks to this optimization, the final result is a visual representation that closely matches the real scene, preserving the most relevant details without the need to build a polygonal mesh. This not only improves performance but also reduces intermediate steps in the workflow, making it a particularly useful tool for the creation of digital twins or immersive experiences based on real-world data.

5.2.2 Application of Gaussian Splatting with Jawset Postshot

To carry out the final representation using Gaussian Splatting, the tool **Jawset Postshot** was used. This application specializes in visualizing three-dimensional scenes from real data such as point clouds and reconstructed cameras. It allows direct import of files generated by COLMAP or other reconstruction platforms and applies Gaussian Splatting efficiently and with visually appealing results.

The use of Jawset Postshot offers several key advantages in the context of this project:

- **Ease of use:** Unlike other more technical tools, Postshot features a user-friendly graphical interface that allows adjusting visual parameters and exporting results without the need for programming or advanced configuration.
- **Compatibility with COLMAP:** It enables direct import of the files `cameras.txt`, `images.txt`, and `points3D.txt`, facilitating integration with existing workflows based on reconstruction with COLMAP.
- **Real-time visualization:** The tool allows real-time preview of Gaussian Splatting effects and interactive adjustment of parameters such as Gaussian size, density, or transparency level, greatly speeding up the creative process.
- **Flexible export:** Results can be exported in various formats, enabling integration with graphics engines like Unreal Engine or visualization tools like Twinmotion for immersive experiences.
- **Automatic optimization:** It includes an efficient optimization system that improves the visual quality of the Gaussians without the need for external training or computationally intensive processes.

Below, in Figures 27,28, and29, the process carried out in Jawset Postshot is shown, from loading the reconstructed data to configuring the Gaussians:

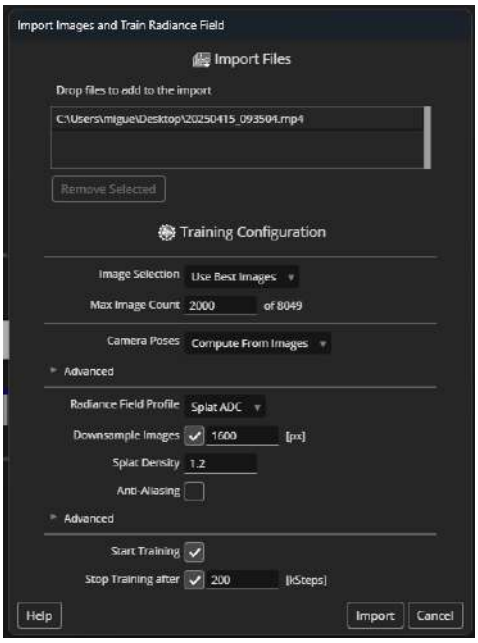


Figure 27: Loading COLMAP files in the Jawset Postshot interface.

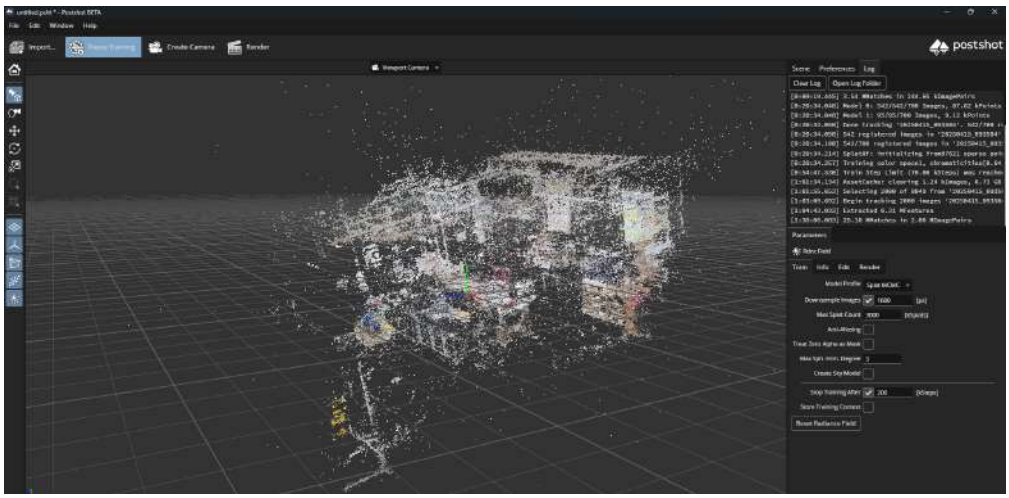


Figure 28: Point cloud generation in Postshot.

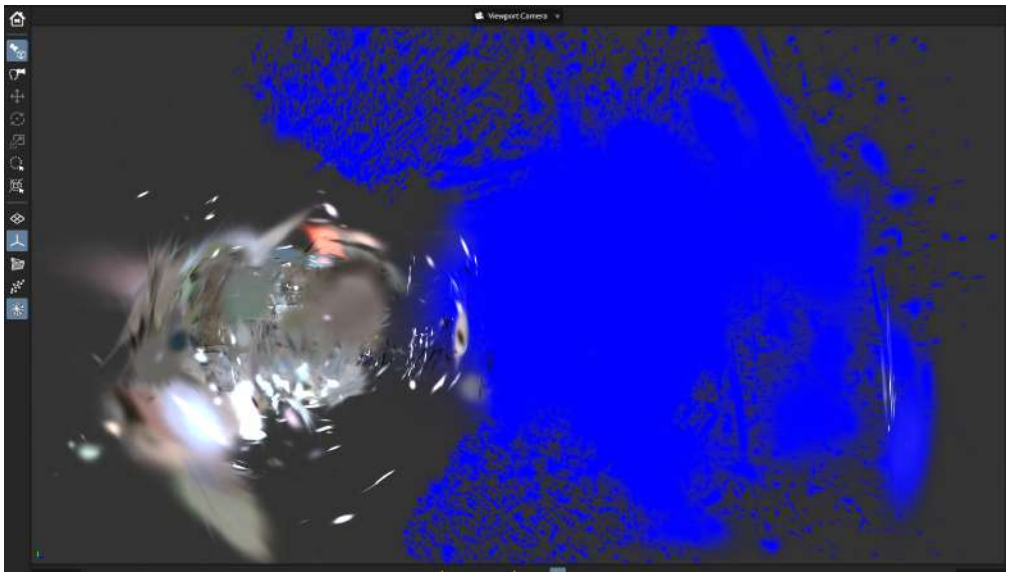


Figure 29: Cleaning of splats.

Thanks to this tool, high-quality visualizations have been obtained that allow for an immersive and smooth exploration of the environment, as shown in Figure 30.

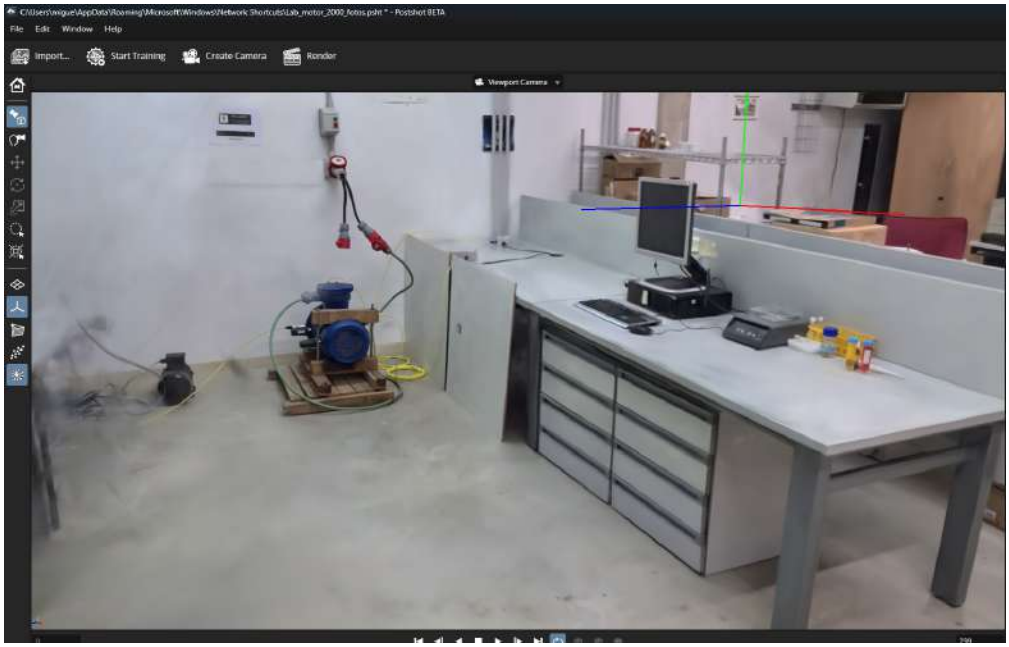


Figure 30: Final representation using Gaussian Splatting with Postshot.

Despite the numerous advantages offered by the *Gaussian Splatting* technique and its implementation through the *Jawset PostShot* tool, it is important to point out

certain limitations that should be considered when incorporating them into a workflow aimed at creating digital twins. These limitations can affect both system performance and the visual quality of the results as well as their compatibility with other platforms.

- **High resource consumption:** The optimization process of thousands of gaussians can be very demanding from a computational standpoint. It requires a large amount of GPU memory and processing power, especially when working with dense point clouds. This can be an obstacle for systems with limited hardware or without access to specialized workstations.
- **Initial learning curve:** Although PostShot's interface is visually accessible, mastering its operation can be complex for users without experience in 3D graphics or digital reconstruction. Concepts such as anisotropic dispersion, virtual camera setup, or managing visual parameters require some technical background.
- **Insufficient documentation:** As the tool is still in active development, the official documentation available is scarce. From my experience, I had to rely on forums, conduct numerous tests on my own, and consult specialized communities to properly understand and take full advantage of certain features. The lack of official documentation led me to adopt a more experimental and self-taught approach during the process.
- **Limitations in detailed editing:** Once the gaussians are rendered, it is difficult to make fine adjustments on specific points of the cloud. Modifications such as removing isolated noise or correcting specific areas are not fully accessible from the PostShot interface, which may limit artistic or technical control over the scene.
- **Dependence on the COLMAP format:** Although PostShot integrates well with files generated by COLMAP, this advantage also creates a dependency. If the input files (cameras, images, and 3D points) are not properly structured, loading errors or inconsistencies in visualization may occur.
- **Absence of polygonal geometry:** *Gaussian Splatting* does not generate traditional meshes or surfaces. While this simplifies the visualization process, it can be a limitation when the digital twin must be integrated into platforms that require explicit geometry, such as CAD applications or reverse engineering environments.
- **Limited integration with BIM or IoT environments:** Currently, PostShot does not offer native integration mechanisms with BIM (*Building Information Modeling*) systems or IoT platforms. This lack may reduce its applicability in more complex industrial projects or environments requiring interoperability with other digital systems.
- **Issues exporting to real-time engines:** Although it is possible to export results for use in engines such as *Unreal Engine*, sometimes part of the visual

fidelity obtained within PostShot is lost. This is due to differences in how each environment manages and renders the gaussians.

These limitations do not diminish the potential of PostShot or the *Gaussian Splatting* technique, but they do highlight the need to carefully evaluate their suitability based on the project objectives. Proper planning and a critical analysis of the tools used are key to making the most of their capabilities and minimizing their drawbacks.

Jawset PostShot has proven to be a versatile and effective tool for applying Gaussian Splatting using real data. Its ease of integration with COLMAP and its visual power make it an ideal solution for generating immersive digital twins, like those developed in this project. However, its use is not without limitations. Despite its ability to produce detailed visual representations, processing large volumes of data can be slow and demanding in terms of computational resources. Additionally, the visualization quality can be affected by the resolution of the point cloud or the inherent limitations of the Gaussian Splatting technique, which may result in certain artifacts or inconsistencies in the final digital twin representation.

5.3 Data Management and Visualization

Databases that can be used and how to access data from the tools/programs employed: To extract data from the InfluxDB database in Unreal Engine, the `HTTP Blueprint` plugin [42] is used, which allows sending HTTP requests to an external URL. A POST request is configured targeting the InfluxDB server on port 8086, including HTTP headers such as `Authorization` (with a security token) and `Content-Type` set as `application/vnd.flux`, indicating that a query in the Flux language will be sent. The request body contains the specific query to obtain acceleration and vibration data from the engine sensors. Once sent, the application processes the server response and converts the data into usable variables within Unreal Engine, thus enabling the real-time connection of the digital twin with the sensor system.

In Unity, to access Influx data you must use the `UnityEngine.Networking` library, which allows making web requests. First, the URL of the server to which the request will be sent must be set. Then, the query must be constructed to select exactly which data to extract from the database and subsequently saved as a byte array, which is the format used by the predefined function for making requests. Additionally, it is necessary to add several parameters to the request, such as the `content-type`, which must be `"vnd.flux"` to indicate the type of language used in the Influx query — in this case, `"flux"`, a language native to InfluxDB — and the header `"Accept"` to indicate the expected response type from the server. Additionally, the `Authorization` header must be added to specify the security token for database access; in our case, this was not incorporated since we use a reverse proxy on the server to hide all that sensitive information. Once the request is made, the server returns a CSV-formatted response

from which the data must be extracted and formatted in the most convenient way for representation.

Tools for Data Visualization

When it comes to visualizing real-time data within an application or simulation, there are two main approaches: creating visual dashboards directly within the program itself or using external tools such as web dashboards. Each option has its advantages, and the choice largely depends on the type of project and the experience intended for the user.

- *Internally developed dashboards within the program:* On one hand, dashboards developed inside the program allow displaying data directly in the visual environment of a simulation, as seen in engines like Unity or Unreal Engine. This is especially useful when working with digital twins or interactive 3D simulations, since data can be placed exactly where needed: above a virtual machine, inside a screen within the environment, or floating over an object. In Unity, for example, the Canvas UI system can be used along with components like TextMeshPro and C# scripts to update information in real time. In Unreal Engine, tools like UMG are used to create visual widgets with Blueprints or code. This type of integration results in a very rich visual experience but also requires more technical and design work, especially if an attractive and dynamic visualization is sought.

- *Web dashboards (Grafana/Flutter):* On the other hand, there are web dashboards, which rely on external platforms like Grafana or Flutter. These tools are ideal when you want to display data quickly and professionally without having to program a whole interface from scratch. Grafana, for example, connects directly to databases like InfluxDB and allows visualizing data with charts, indicators, and other visual elements simply by configuring some parameters. Additionally, since everything runs in the browser, these dashboards can be accessed from any device, making them perfect for remote monitoring. Flutter also enables creating very attractive and customized visual interfaces, especially if you want to integrate data visualization into a mobile or web app.

Both approaches are not mutually exclusive. In fact, they can be combined: for example, integrating a browser inside the game engine to display a Grafana dashboard directly on a virtual screen within the simulation. This allows leveraging the best of both worlds: the visual power of the 3D environment and the flexibility of web tools.

6 Discussion on the parameterization of Digital Twins in agrivoltaic greenhouses

The surveys cited at the beginning of this report do not include a specific category detailing which parameters are monitored in the digital twins they analyzed. However, any parameter that can be captured using sensors, cameras, or even manual input from farmers can potentially be visualized, analyzed, simulated, and adjusted within a digital twin environment. Therefore, this section identifies the key parameters or variables that can be integrated into our digital twin or into any digital twin developed for similar purposes, extending beyond the default configurations intrinsic to the software tools used to create the system.

Considering the knowledge we have gained from literature and from performing preliminary tests with the tools described in this report, we propose the a workflow for the creation of Digital Twins intended for agrivoltaic greenhouses (see Figure 31). Firstly, it is necessary to scan the target environment (buildings, big structures, etc.) and the objects users will interact with through the Digital Twin (Sensor devcies, actuators, etc.). Then, the resulting scans need to be pre-processed to build the point cloud from the performed scans. Then, the resulting point cloud needs to be exported in the desired format to process it further. During this step, additional processing may be done to eliminate unwanted scanned areas or to remove points in wrong placements. Once the processed point cloud is exported, a mesh model can be rendered from it. This step also requires adjustments to ensure the best quality visual representation is achieved. Lastly, the models can be imported into the Digital Twin engine, that allows implementing the desired functionalities including real-time data integration, data visualization, data analysis, and remote control of elements in the greenhouse. To do this, the Digital Twin needs to be fed with relevant data.

There are five distinct sources of information that can be incorporated into the digital twin of the hydroponic agrivoltaic greenhouse envisioned for this project to ensure an accurate and detailed representation. Each category includes a set of variables that can be monitored, offering complementary data that improves the overall understanding of the system's condition and behavior. Table 1 presents a list of variables that can potentially be tracked and integrated into the digital twin.

IoT sensors deliver continuous, real-time measurements of environmental and operational parameters, which are necessary for monitoring conditions inside the greenhouse. The solar panel infrastructure provides data on energy generation and system performance, which impacts the overall energy management and sustainability of the installation. Hydroponic system data offers insights into nutrient levels, water quality, and plant growth conditions, directly affecting crop health and productivity. Information on greenhouse infrastructure ensures the physical characteristics and configurations are accurately represented, allowing the digital twin to reflect the actual environment. Finally, direct input from the farmer adds contextual knowledge, manual adjustments,

and observations that sensors alone might not capture. All these information sources allows for accurate visualization, simulation, and decision-making processes, ultimately enhancing the effectiveness of the Digital Twin as a tool for monitoring, control, and optimization.

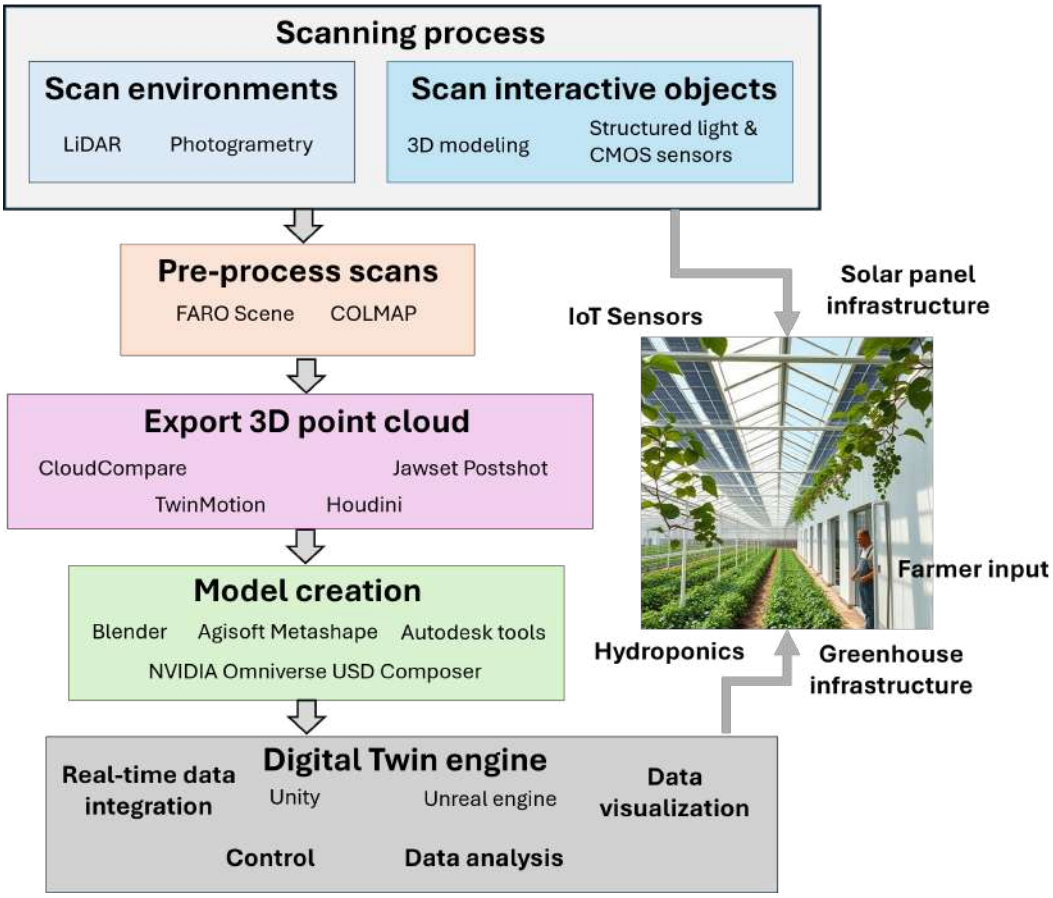


Figure 31: Workflow for the creation of a Digital Twin for an agrivoltaic greenhouse.

To develop a functional and realistic digital twin, it is important to model the system at different levels of detail: object, building, and field. Modeling at the object level allows the representation and tracking of individual components such as sensors, actuators, or control devices, which are used for capturing fine-grained behavior and interactions within the greenhouse. The building level provides context for these objects by situating them within larger structures like greenhouses or control rooms, enabling aspects such as spatial reasoning, infrastructure planning, and environmental interactions within the greenhouse. Finally, the field level accounts for broader areas such as hydroponic cultivation zones or solar panel arrays, which are utilized for evaluating large-scale performance.

Information source	**Variables**
IoT Sensors	Air temperature Humidity CO_2 concentration Light intensity (PAR) Leaf temperature Nutrient solution temperature Electrical conductivity (EC) pH level Water flow rate Tank levels Wind speed
Solar panel infrastructure	Panel temperature Energy production Panel orientation Incident irradiance Efficiency Inverter status Battery charge level (if applicable) Shading percentage
Hydroponics	Nutrient concentrations (e.g., N, P, K) Dissolved oxygen Flow rate of nutrient solution Irrigation frequency and duration Root zone temperature Type of substrate Water consumption System pH
Greenhouse infrastructure	Ventilation status Heating system status Cooling system status Artificial lighting status Shading system position Screen position Door status Structural temperature CO_2 injection system activity
Farmer input	Crop type and variety Planting and harvesting dates Pruning schedule Pest and disease observations Manual overrides of system settings Yield estimations Maintenance logs Operational preferences

Table 1: Examples of variables that can be monitored and integrated into a digital twin for a hydroponic agrivoltaic greenhouse.

In addition to the 3D model, other features of the digital twin play an important role. Visualization of real-time data helps operators quickly assess system conditions and detect anomalies. Data analysis and simulations support informed decision-making by predicting future states based on historical trends and sensor inputs. Finally, actuator control connects digital insights with physical actions, enabling automated responses to changing conditions, such as adjusting irrigation or ventilation—based on current data and predictive models. Together, these components form a complete digital twin that not only reflects the physical greenhouse environment but also enhances its management through real-time data visualization, automation, and predictive capabilities. Table 2 shows how the input data for the digital twin of the hydroponic agrivoltaic greenhouse is used in different ways. The uses are grouped into four main areas: 3D model representation, visualization, simulations, and actuators.

Digital twin feature	**Description**	
3D model representation	Objects	Sensor devices, AI vision cameras, irrigation motors, communication gateways, actuators
	Buildings	Greenhouse structures, storage facilities, control rooms
	Fields	Hydroponic trays, planting zones, solar panel arrays
Visualization	Real-time representation of monitored parameters such as temperature, humidity, solar irradiance, nutrient levels, CO_2 concentration, etc.	
Data analysis and simulations	AI-based data analysis and prediction models for crop growth, disease detection, energy usage optimization, environmental control forecasting	
Actuators	Activation and deactivation of irrigation systems, ventilation fans, shading mechanisms, nutrient dosing pumps, lighting systems based on sensor inputs and simulation outcomes	

Table 2: Digital twin features and their descriptions with examples.

7 Conclusion

Digital twins are an increasingly adopted technology across various sectors, including agriculture. They offer an effective means to remotely monitor, analyze, and control complex systems through interactive 3D interfaces. This report presents an evaluation of the software tools and parameter configurations required to develop a digital twin for an agrivoltaic greenhouse. The proposed tools were tested using 3D scans of an engine used for irrigation and a laboratory environment, obtained through LiDAR and photogrammetry techniques. Finally, the report proposes a workflow for building

Digital Twins intended for agrivoltaic greenhouses and a provides a discussion on the parametrization of digital twins, focusing on variables that can be monitored via IoT sensor networks as well as those defined through direct farmer input.

References

[1] Natasja Ariesen-Verschuur, Cor Verdouw, and Bedir Tekinerdogan. Digital twins in greenhouse horticulture: A review. *Computers and Electronics in Agriculture*, 199:107183, 2022.

[2] Warren Purcell and Thomas Neubauer. Digital twins in agriculture: A state-of-the-art review. *Smart Agricultural Technology*, 3:100094, 2023.

[3] Tarek I Zohdi. A digital-twin and machine-learning framework for the design of multiobjective agrophotovoltaic solar farms. *Computational Mechanics*, 68(2):357–370, 2021.

[4] Cor Verdouw, Bedir Tekinerdogan, Adrie Beulens, and Sjaak Wolfert. Digital twins in smart farming. *Agricultural Systems*, 189:103046, 2021.

[5] Lens portal for patent and scholarly search. `https://www.lens.org/`. [Accessed 07-05-2025].

[6] Maulshree Singh, Evert Fuenmayor, Eoin P Hinchy, Yuansong Qiao, Niall Murray, and Declan Devine. Digital twin: Origin to future. *Applied System Innovation*, 4(2):36, 2021.

[7] Yining Lang, Yanqi Zhang, Tan Sun, Xiujuan Chai, and Ning Zhang. Digital twin-driven system for efficient tomato harvesting in greenhouses. *Computers and Electronics in Agriculture*, 236:110451, 2025.

[8] Hameedur Rahman, Uzair Muzamil Shah, Syed Morsleen Riaz, Kashif Kifayat, Syed Atif Moqurrab, and Joon Yoo. Digital twin framework for smart greenhouse management using next-gen mobile networks and machine learning. *Future Generation Computer Systems*, 156:285–300, 2024.

[9] Paula Catala-Roman, Enrique A Navarro, Jaume Segura-Garcia, and Miguel Garcia-Pineda. Harnessing digital twins for agriculture 5.0: a comparative analysis of 3d point cloud tools. *Applied Sciences*, 14(5):1709, 2024.

[10] Nikolaos Peladarinos, Dimitrios Piromalis, Vasileios Cheimaras, Efthymios Tserepas, Radu Adrian Munteanu, and Panagiotis Papageorgas. Enhancing smart agriculture by implementing digital twins: A comprehensive review. *Sensors*, 23(16):7128, 2023.

[11] Christos Pylianidis, Sjoukje Osinga, and Ioannis N Athanasiadis. Introducing digital twins to agriculture. *Computers and Electronics in Agriculture*, 184:105942, 2021.

[12] Steven Kim and Seong Heo. An agricultural digital twin for mandarins demonstrates the potential for individualized agriculture. *Nature Communications*, 15(1):1561, 2024.

[13] Jehangir Arshad, Ch Ahsan Abbas Sheheryar, Mohammad Khalid Imam Rahmani, Abdul Qayyum, Roumaisa Nasir, Sohaib Tahir Chauhdary, and Khalid Jaber Almalki. Simulink-driven digital twin implementation for smart greenhouse environmental control. *Egyptian Informatics Journal*, 30:100679, 2025.

[14] Dam Xuan Dong, Nguyen Thi Thu Huong, Nguyen Ngoc Bach, Nguyen Quang Ninh, Bui Thi Duyen, and Dam Xuan Dinh. Application of the digital twin for condition monitoring and energy management of electrical loads in greenhouses. *Engineering, Technology & Applied Science Research*, 15(3):24005–24012, 2025.

[15] Pablo Palacios Játiva, Ismael Soto, Cesar A Azurdia-Meza, Iván Sánchez, Riu Wang, and Werther Kern. Hybrid digital twin model for greenhouse and underground environments. *IEEE Access*, 2024.

[16] Rajmeet Singh, Lakmal Seneviratne, and Irfan Hussain. A deep learning-based approach to strawberry grasping using a telescopic-link differential drive mobile robot in ros-gazebo for greenhouse digital twin environments. *IEEE Access*, 2024.

[17] Ahmad F Subahi. Advancing sustainable cyber-physical system development with a digital twins and language engineering approach: Smart greenhouse applications. *Technologies*, 12(9):147, 2024.

[18] Cristian Bua, Luca Borgianni, Davide Adami, and Stefano Giordano. Digital twin for remote control of robotic arm via wearable glove in smart agriculture. In *2025 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6. IEEE, 2025.

[19] Jiawei Chen, Nicola Paciolla, Stefano Mariani, and Chiara Corbari. Agrivoltaics: A digital twin to learn the effect of solar panel coverage on crop growth. *Engineering Proceedings*, 82(1):5, 2024.

[20] Anna Kujawa, Natalie Hanrieder, Stefan Wilbert, Álvaro Fernández Solas, Sergio González Rodríguez, María del Carmen Alonso-García, Jesús Polo, José Antonio Carballo, Guadalupe López-Díaz, Cristina Cornaro, et al. A ray-tracing-based irradiance model for agrivoltaic greenhouses: Development and application. *Agronomy*, 15(3):665, 2025.

[21] Maddalena Bruno, Leonhard J Gfüllner, and Matthew F Berwind. Enhancing agrivoltaic synergies through optimized tracking strategies. *Journal of Photonics for Energy*, 15(3):032703–032703, 2025.

[22] Roxane Bruhwyler, Nicolas De Cock, Pascal Brunet, Jonathan Leloux, Pierre Souquet, Etienne Perez, Etienne Drahi, Sebastian Dittmann, and Frédéric Lebeau. Modelling light-sharing in agrivoltaics: the open-source python agrivoltaic simulation environment (pase 1.0). *Agroforestry Systems*, pages 1–18, 2024.

[23] Cloudcompare. https://www.danielgm.net/cc/.

[24] Agisoft metashape. https://www.agisoft.com/.

[25] Jawset postshot: Training configuration. https://www.jawset.com/docs/d/Postshot+User+Guide/Interface/Training+Configuration.

[26] Houdini. https://www.sidefx.com/products/houdini.

[27] Unreal engine. https://www.unrealengine.com/es-ES.

[28] Twinmotion. https://www.twinmotion.com/en-US.

[29] Unity. https://unity.com/es.

[30] Revit. https://www.autodesk.com/products/revit/overview.

[31] Recap pro. https://www.autodesk.com/products/recap/overview.

[32] Blender. https://www.blender.org/.

[33] NVIDIA Omniverse. https://www.nvidia.com/es-es/omniverse/.

[34] Technical Specification Sheet for the Focus Laser Scanner — knowledge.faro.com. https://knowledge.faro.com/Hardware/Focus/Focus/Technical_Specification_Sheet_for_the_Focus_Laser_Scanner. [Accessed 03-03-2025].

[35] FARO® Freestyle 2 Handheld Scanner. https://www.atecorp.com/products/faro/freestyle-2?srsltid=AfmBOooCEkYtD_B87mil2e6fJoA8A7h529SObSKKTXn5PhvApd7HHNkG. [Accessed 03-03-2025].

[36] Faro scene. https://www.faro.com/es-MX/Products/Software/SCENE-Software.

[37] Repositorio de github gsops. https://github.com/david-rhodes/GSOPs.

[38] Autodesk. Overview of the scan to mesh feature in autodesk recap pro. https://www.autodesk.com/support/technical/article/caas/sfdcarticles/sfdcarticles/ESP/Scan-to-Mesh-feature-in-ReCap-Pro-Overview.html, 2024. [Accessed 07-05-2025].

[39] Autodesk. Export options in recap pro. https://www.autodesk.com/products/recap/compare/export-options, 2024.

[40] NVIDIA Corporation. Nvidia omniverse documentation, 2023.

[41] NVIDIA. USD Composer Overview – Omniverse USD Composer. https://docs.omniverse.nvidia.com/composer/latest/index.html.

[42] Restful plugin. https://www.fab.com/listings/9c3c21b8-6965-4005-b13a-7fbc8ad54fe8.