Smart Agricultural Technology 14 (2026) 102316

Contents lists available at ScienceDirect

Smart Agricultural Technology

journal homepage: www.journals.elsevier.com/smart-agricultural-technology

FloraForge: Procedural generation of editable and analysis-ready 3D plant geometric models using LLM-assisted template design

Mozhgan Hadadi [a], Talukder Z. Jubery [a], Patrick S. Schnable [a], Arti Singh [a], Bedrich Benes [b], Adarsh Krishnamurthy [a,*], Baskar Ganapathysubramanian [a,*]

[a] Iowa State University, Ames, 50011, Iowa, USA
[b] Purdue University, West Lafayette, 47906, Indiana, USA

ARTICLE INFO

Keywords:
Procedural modeling
Non-uniform rational B-spline surfaces
Large language models
3D plant models
Parametric geometry

ABSTRACT

Accurate 3D plant models are crucial for computational phenotyping and physics-based simulation; however, current approaches face significant limitations. Learning-based reconstruction methods require extensive species-specific training data and lack editability for hypothesis-driven research. Procedural modeling offers parametric control and large model variability but demands specialized expertise in geometric modeling and an in-depth understanding of complex procedural rules, making it inaccessible to domain scientists. We present FloraForge, an LLM-assisted framework that enables domain experts to generate biologically accurate, fully parametric 3D plant models through iterative natural language Plant Refinements (PR) during template creation, minimizing the need for programming expertise. Our co-design workflow leverages LLM-assisted code generation to progressively refine Python scripts that generate parameterized complex plant geometries as Non-Uniform Rational B-Spline (NURBS) surface representations, with botanical constraints. Plant organs are represented as spline surfaces that can be easily tessellated into polygonal meshes with arbitrary precision, ensuring compatibility with functional structural plant analysis workflows such as light simulation, computational fluid dynamics, and finite element analysis. We demonstrate the framework by generating procedural models of multiple maize genotypes, soybean (which shares the procedural generator with mung bean), and mung bean, with one plant tracked across different developmental stages. We fit procedural models to empirical LiDAR and NeRF-derived point cloud data through manual refinement of the Plant Descriptor (PD), a human-readable YAML file originally templated by the LLM, obtaining consistently low mean symmetric Chamfer distances that indicate close agreement between generated models and measured plant geometry. The pipeline generates dual outputs: triangular meshes (represented as STL or OBJ files) for visualization and triangular meshes with additional parametric metadata for quantitative analysis (stored as SMESH files). We further illustrate analysis-ready use by coupling the procedurally generated models to the HELIOS framework to simulate diurnal photosynthetically active radiation interception in virtual maize and mung bean fields across growth stages. Our framework uniquely combines pre-trained LLM-assisted template creation, mathematically continuous representations that support both phenotyping and rendering, and direct parametric control through the PD. The framework makes sophisticated geometric modeling accessible to plant science researchers while maintaining mathematical rigor through biologically interpretable parameterizations; additionally, the iterative PR dialogue produces an explicit record of model properties that is typically absent in conventional procedural modeling pipelines.

1. Introduction

Accurate three-dimensional (3D) plant geometric models are foundational to modern plant science, enabling computational phenotyping, physics-based simulation of agroecosystems, and functional-structural modeling of growth processes [1–4]. However, generating such models from scratch (forward modeling) or finding a 3D model representation for captured data (reconstruction) remains a significant bottleneck: learning-based reconstruction (i.e. AI based) methods require extensive species-specific training datasets and produce representations that resist direct editing, while procedural modeling systems demand specialized expertise in geometric programming and rule-based algorithms that few plant scientists possess. So-called inverse procedural models often represent only a class of objects and fail on detailed geometries [5–7]. This

* Corresponding authors.
E-mail addresses: adarsh@iastate.edu (A. Krishnamurthy), baskarg@iastate.edu (B. Ganapathysubramanian).

https://doi.org/10.1016/j.atech.2026.102316
Received 19 January 2026; Received in revised form 11 May 2026; Accepted 12 June 2026
Available online 15 June 2026

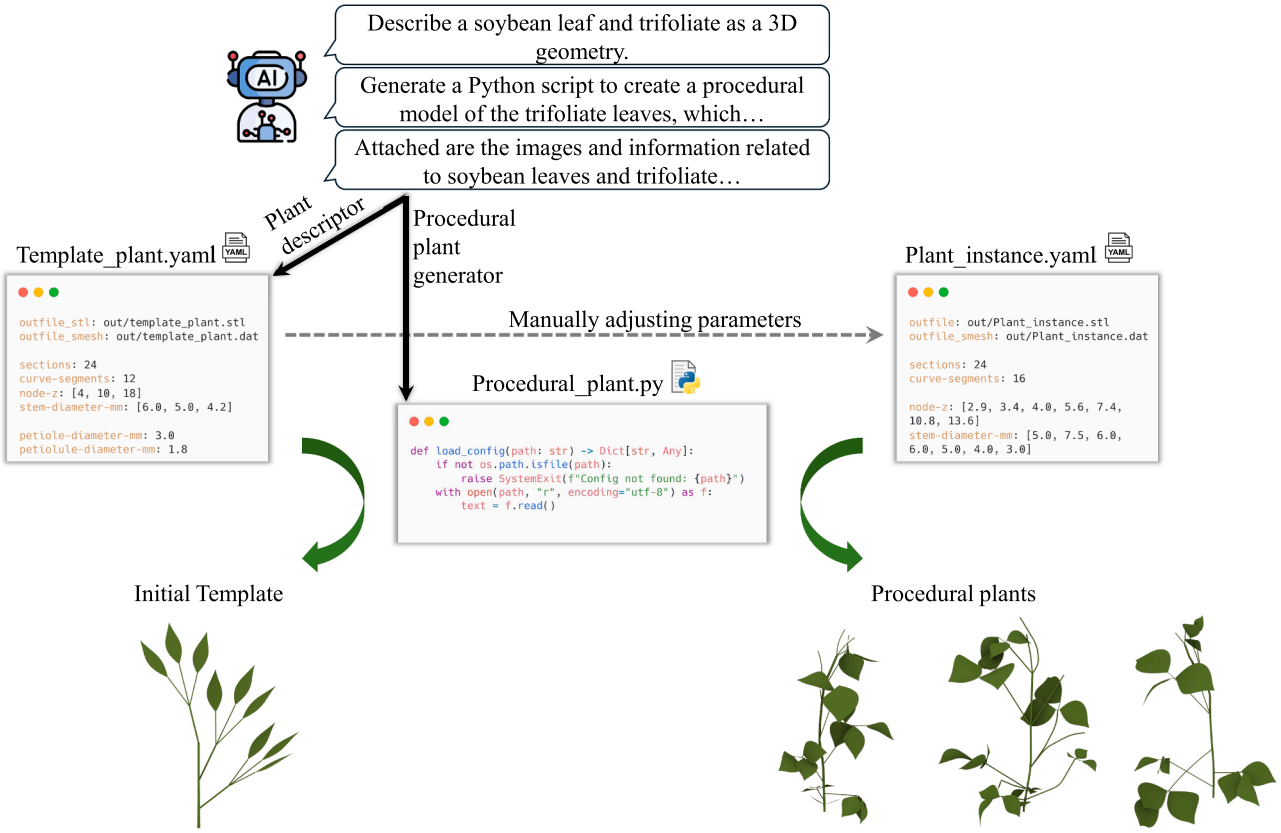


Fig. 1. FloraForge: LLM-assisted procedural modeling workflow. Users provide Plant Refinements (PR) to obtain a Procedural Plant Generator (PPG; Python) and a Plant Descriptor (PD; YAML). Executing PPG(PD) yields an initial 3D template with full parametric control, allowing domain experts without programming expertise to develop custom modeling tools through iterative refinement.

fundamental tension between accessibility and analytical utility has constrained the adoption of sophisticated 3D modeling in plant phenotyping, breeding, and agronomic research.

Current approaches face three fundamental limitations that prevent widespread adoption. First, learning-based reconstructions [8–10] yield mesh or latent representations that lack parametric editability, thereby preventing researchers from modifying biologically meaningful traits, such as leaflet pitch, internode elongation, or organ curvature, for hypothesis-driven studies. Second, procedural modeling frameworks such as L-systems [11,12] and template-based systems [13,14] require deep expertise in programming the grammar rules and geometric modeling, making them inaccessible to domain scientists who lack specialized training in 3D graphics or geometric modeling. Moreover, existing procedural models are complex, non-linear systems that are difficult to control. Third, most existing reconstruction methods are suited for artificial objects, such as CAD models (e.g., [15,16]), and fail on thin and long geometries, which is a typical case for vegetation. These limitations force researchers to choose between accessible but uneditable and approximate reconstructions and powerful yet inaccessible procedural tools, a choice that ultimately hinders scientific progress.

Recent advances in large language models (LLMs) have demonstrated exceptional capabilities in code synthesis, multimodal reasoning, and tool utilization for 3D content creation [17–20]. These models demonstrate remarkable proficiency in generating correct object attributes, parsing textual descriptions, correctly interpreting code functions, and facilitating efficient human-computer interactions through natural language dialog. Despite these advances, existing LLM-driven geometric modeling systems primarily focus on general objects, such as CAD models or architectural applications, without addressing the unique challenges of botanical morphology, including hierarchical organ structures, species-specific features, growth constraints, phyllotactic arrangements, and the need for biologically interpretable parameterizations that directly map to phenotypic traits. Moreover, these systems typically generate assets for visualization rather than scientific analysis, usually lacking the mathematical rigor required for quantitative phenotyping and computational simulation. A critical gap lies at the intersection of accessibility and analytical rigor. Plant scientists need tools that are as accessible as learning-based reconstruction methods yet provide the parametric control and mathematical continuity of procedural modeling, without requiring programming expertise or species-specific training data. Bridging this gap requires three synergistic capabilities: (1) automated generation of procedural templates incorporating botanical domain knowledge, (2) mathematically continuous parametric geometric representations suitable for both quantitative analysis and realistic visualization, and (3) human-readable high-level biologically meaningful parameter interfaces enabling direct editing by domain experts.

We present **FloraForge** (see Fig. 1), an LLM-assisted framework that bridges this gap through three synergistic innovations. First, we leverage LLMs to automate the creation of procedural modeling templates from botanical descriptions, eliminating the need for manual geometric programming. Domain experts progressively refine Python scripts that implement continuous non-uniform B-spline surface representations with botanical constraints through structured iterative dialog, without requiring expertise in geometric modeling. Second, we represent plant organs as continuous B-spline surfaces that provide both the mathematical properties required for quantitative analysis (smoothness, differentiability, and exact curvature computation) and the realism needed for visualization. The Non-Uniform Rational B-spline (NURBS) representations ensure compatibility with radiation simulations, computational fluid

dynamics, and finite element workflows[1] while supporting physically realistic rendering. Third, we expose all morphological parameters through Plant Descriptor (PD) (human-readable YAML files), enabling domain experts to directly edit and version-control plant architectures without programming knowledge. Hierarchical parameter inheritance (global defaults with node-specific and organ-specific overrides) minimizes redundancy while preserving biological realism and flexibility. This combination of LLM-assisted template generation, mathematically rigorous representations, and accessible parametric control uniquely positions our framework to extend sophisticated geometric modeling capabilities to plant scientists without specialized programming backgrounds. FloraForge equips domain scientists to prototype custom modeling tools for the plant forms studied here, supporting the iterative refinement of both procedural algorithms (via natural-language dialogue with the LLM) and individual plant instances (via PD parameter editing). We illustrate on monocot and dicot crop architectures, with natural future extensions to vines, shrubs, and trees.

Our framework addresses the critical limitations of existing approaches through four key differentiators. First, compared to learning-based reconstruction methods such as Demeter [8] (which require 600 + annotated scans) and NeuraLeaf [9] (which require thousands of 2D leaf samples per species), our approach is *training-free*. We utilize existing pre-trained LLMs through iterative natural language dialog to generate explicit, biologically interpretable parameters stored in version-controllable YAML files rather than opaque latent vectors. This eliminates the data acquisition bottleneck and provides the transparency essential for scientific reproducibility. Second, unlike traditional L-systems [11,21,22] that represent plants as discrete branching graphs requiring expert rule design, we provide continuous NURBS surface representations with guaranteed smoothness (C^1 continuity) or higher continuity within knot spans, depending on degree and knot multiplicity for precise morphometric measurements, curvature analysis, and numerical simulations. Third, in contrast to inverse procedural approaches [6,23] that optimize parameters for pre-existing templates, we *automate the template design process itself* through LLM co-design, enabling rapid adaptation to new species without requiring manual geometric specification or optimization convergence. Finally, while recent LLM-based 3D generation systems [17,18] demonstrate impressive capabilities for buildings and general objects, they lack the botanical domain knowledge and phenotyping-oriented parameterizations essential for plant science applications.

Our contributions advance the state of the art in plant geometric modeling with the following innovations:

- **LLM-assisted procedural plant generator:** A framework that automates the creation of botanically-constrained NURBS surface templates through iterative natural language PR, eliminating the need for expertise in geometric modeling or procedural modeling systems.
- **Analysis-ready continuous representations:** Explicit NURBS surface formulations provide mathematical continuity for precise phenotypic measurements, numerical simulation compatibility, and physically realistic rendering within a unified representation. We demonstrate this by coupling the generated models to HELIOS for canopy-level light interception studies.
- **Accessible parametric control:** Human-readable PD exposes morphological parameters (curvature, width profiles, orientation, and deformation) for direct editing by domain experts, with hierarchical parameter inheritance minimizing redundancy while preserving biological realism. While users still require botanical intuition to adjust parameters meaningfully, the framework eliminates the need for programming or geometric modeling expertise.
- **Cross-species validation:** Demonstration of template-based modeling across monocot (maize) and dicot (mung bean) architectures, fitting procedural models to empirical LiDAR point clouds, and showcasing generalization capabilities for diverse plant morphologies and developmental stages.

FloraForge accelerates the research cycle for plant phenotyping, breeding, and precision agriculture applications by enabling domain scientists to develop custom plant modeling tools without specialized computational expertise. Researchers can rapidly prototype species-specific templates, fit models to acquired real-world data through intuitive parameter adjustment, and generate analysis-ready outputs compatible with downstream computational workflows, all without needing to write code or master complex procedural or geometric modeling systems.

Terminology. We use the following terms throughout this paper:

- **Plant Descriptor (PD)**: a YAML file containing all architectural and morphological parameters for one plant instance.
- **Plant Refinements (PR)**: natural-language instructions provided by the user during the co-design loop.
- **Procedural Plant Generator (PPG)**: a Python script that uses a PD to generate a 3D plant geometry.

The following sections position our work within the broader landscape of plant modeling research (Section 2), detail the LLM-assisted development workflow and procedural generation algorithms (Section 3), and demonstrate the framework's capabilities through fitted models spanning multiple crop species (Section 5). We conclude with a discussion of limitations, future directions, and the broader implications of LLM-assisted scientific code development for computational plant science (Section 6).

2. Related work

Procedural plant modeling. Procedural modeling has long been a cornerstone of plant representation in computer graphics and computational biology. Lindenmayer's systems (L-systems), introduced by Lindenmayer [24], provide a formal grammar for modeling plant development through parallel string rewriting. These systems have evolved to incorporate stochastic variations [25], context-sensitive rules [12], and parametric control [26]. Recent implementations, such as Infinigen [14], demonstrate the power of procedural generation in creating diverse, realistic object geometries with adjustable parameters and rule-based systems. Despite their expressiveness, L-systems face several limitations for accessible plant modeling. First, designing appropriate production rules and parameter sets for new species requires substantial expertise in formal grammars and botanical morphology [5,27]; moreover, data-driven extensions that learn L-system parameters from observations require large training corpora. Second, L-systems primarily generate branching structures represented as skeletal graphs, requiring additional algorithms to construct continuous surface meshes [28]. Alternative procedural approaches have explored template-based modeling [13,29], where parametric surface primitives are assembled to construct plant organs. These methods benefit from explicit geometric control but typically require manual specification of component relationships and parameter values. Recent ML-based approaches use dictionaries of templates to reconstruct 3D model of a forest [30]. Our work builds upon this foundation by automating template creation through LLM-assisted design while maintaining the editability advantages of parametric representations.

Inverse procedural modeling. Inverse procedural modeling (IPM) [31] addresses the challenge of inferring procedural parameters from existing geometry, enabling compression and variation generation [6,7,32,33].

[1] A key requirement for performing high fidelity computational analysis is accurate 3D geometric representation. However, this is not the only requirement. One often needs associated material properties to perform these analyses. These could include surface properties (e.g., BRDF) for radiation simulations and mechanical properties (e.g., stiffness) for structural simulations.

For plant modeling, IPM approaches have explored fitting L-system parameters to tree point clouds [5], reconstructing branching structures from images [34], and optimizing parametric models to match observations [23]. CropCraft [23] represents recent progress in inverse procedural modeling for agricultural crops, using Bayesian optimization to fit parametric plant models to observations from neural radiance fields. While effective for reconstruction, these optimization-based approaches require predefined procedural templates and struggle to generalize to new species with different morphological characteristics.

Learning-based plant reconstruction. Recent advances in neural representations have enabled powerful learning-based approaches for 3D plant reconstruction. Methods like Demeter [8] learn parametric morphological models from real-world data, encoding topology, articulation, shape, and deformation into compact latent spaces. TreeStructor [35] builds 3D simulation-ready models of trees from single Google street-view photographs via diffusion. NeuraLeaf [9] introduces neural parametric models that disentangle 2D base shapes from 3D deformations, enabling the reconstruction of individual leaves with natural deformations. Neural hierarchical decomposition [10] decomposes plants into box hierarchies at varying levels of detail, accommodating both houseplants and outdoor trees. While these learning-based methods achieve impressive reconstruction quality, they present several limitations for accessible plant modeling workflows. First, they require substantial species-specific training data. Demeter collected over 600 annotated soybean scans, TreeStructor uses thousands of synthetic tree models, and NeuraLeaf requires 2D leaf datasets with thousands of samples per species [8,9]. Second, the learned representations, while compact, lack the interpretability and direct editability that domain scientists require for hypothesis-driven research. Even if some of the parameters are available, they are challenging for domain experts to comprehend. Third, these methods focus primarily on reconstruction from observations rather than generative modeling for custom species design.

Point clouds are becoming the primary method for capturing 3D geometries because of the proliferation of acquisition sensors. The reconstruction methods [28,36] address the challenge of fitting parametric models to 3D scan data. Recent work combines NURBS surface fitting with optimization strategies [28], achieving high-fidelity reconstruction of maize plants through a two-stage optimization process. However, these approaches assume the availability of dense 3D point clouds and focus on inverse modeling rather than forward template creation.

Parametric geometric representations. NURBS provide a rigorous framework for representing curves and surfaces with high precision and flexibility [37]. NURBS are a de facto industrial standard in CAD modeling, and they offer several advantages for plant modeling: (1) exact representation of both analytic shapes and freeform surfaces, (2) local control through weighted control points, (3) mathematical continuity properties essential for quantitative analysis, and (4) compact representation compared to polygon meshes [38]. Rational B-splines are a subset of NURBS and use piecewise polynomial functions to define smooth curves and surfaces [39]. The local control property, where changes to individual control points and their knots affect only nearby regions, makes B-splines particularly suitable for modeling organic shapes with localized variations. Recent work has explored the use of B-splines [40] and parametric leaf surfaces [41], demonstrating their effectiveness in capturing botanical morphology. Traditional approaches to NURBS-based plant modeling require manual specification of control points, knot vectors, and surface parameterizations [13]. Our work addresses this limitation by leveraging LLMs to automate the design process, generating appropriate NURBS parameterizations from high-level botanical descriptions while maintaining the mathematical properties essential for scientific applications.

LLM-assisted 3d modeling. LLMs have recently been explored for 3D content generation through code synthesis. 3D-GPT [17] employs multi-agent systems to break down procedural modeling tasks, generate Python code, and interface with Blender for scene generation. LL3M [18] presents a multi-agent system that writes interpretable Python code for creating 3D assets, emphasizing modularity and integration with artist workflows. BlenderLLM [19] fine-tunes language models specifically for CAD script generation based on user instructions. These systems demonstrate the potential of LLMs for democratizing 3D content creation. However, they focus primarily on general geometric modeling of manufactured objects (such as chairs, airplanes, cups, and lamps) or architectural applications, lacking the botanical knowledge and domain-specific constraints necessary for plant modeling. Furthermore, existing LLM-based 3D generation systems typically operate at the level of individual objects or scenes, without addressing the hierarchical organ structures, growth patterns, and phenotypic constraints that characterize plant morphology. Recent work on LLM-assisted CAD generation [42,43] explores frameworks for automated geometry generation through function calling and agent workflows. These approaches enable LLMs to interpret design requirements and generate executable code for CAD software. However, they have not been applied to the unique challenges of botanical modeling, where biological priors, morphological constraints, and phenotypic parameters must be incorporated into the generation process.

Our work takes a complementary approach: rather than optimizing parameters for pre-existing templates, we automate the template creation process itself. By leveraging LLMs to design appropriate parametric representations from botanical descriptions, we enable rapid prototyping of modeling tools for diverse plant species without requiring manual template engineering or extensive optimization. FloraForge occupies a unique position at the intersection of procedural modeling, parametric representations, and LLM-assisted design. Unlike learning-based methods that require extensive training data, we exploit the reasoning capabilities of pre-trained LLMs. Unlike traditional procedural systems that demand expert knowledge, we provide an accessible interface through natural language descriptions and human-readable parameter files. Unlike existing LLM-based 3D generation systems, we incorporate botanical knowledge and domain-specific constraints essential for plant phenotyping applications. Our system also does not provide descriptions of low-level geometric parameters; instead, it offers more meaningful biological parameters. The combination of LLM-assisted template design, explicit NURBS representations, and human-editable parameters addresses the critical gap in accessible, editable plant modeling tools for domain scientists. Table 1 summarizes how FloraForge compares with representative approaches from each category discussed above.

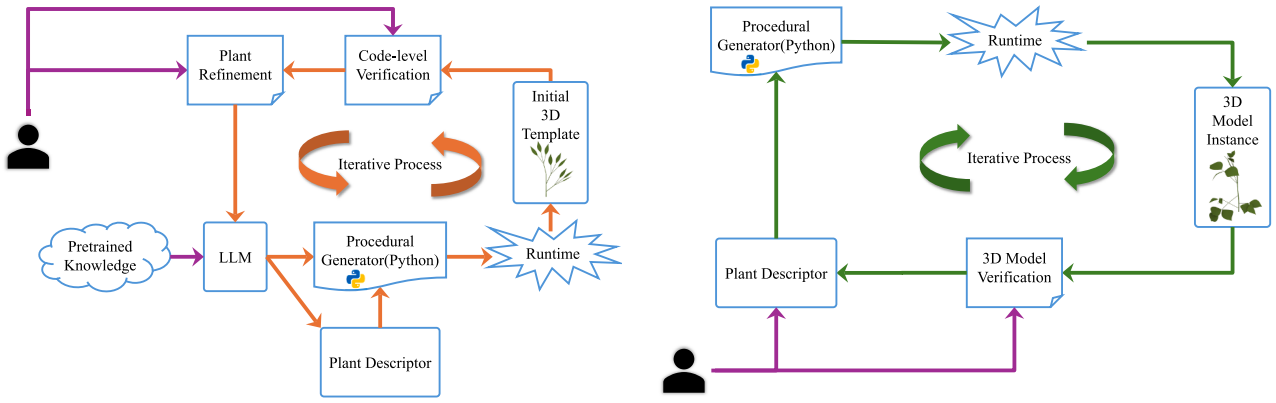
3. Methods

We have developed an editable, analysis-ready procedural modeling pipeline for generating biologically accurate 3D plant architectures. The system integrates LLM-assisted iterative co-design, parametric NURBS surface representations, and hierarchical procedural architectural frameworks to produce geometrically precise, fully parametric plant models. We demonstrate it on both dicotyledonous species (mung bean *Vigna radiata*) and monocotyledonous species (maize *Zea mays*). All procedural parameters are stored in Plant Descriptors (PDs), ensuring complete reproducibility, version control, and editability of generated plant architectures.

The workflow comprises two stages (see Fig. 2). In *Step 1* (LLM-assisted template co-design and Code-level Verification; CLV), the user interacts with the LLM only through Plant Refinements (PR). From these prompts, the LLM synthesizes a Procedural Plant Generator (PPG; Python) with an LLM-seeded Plant Descriptor (PD). Executing the PPG with this PD yields an initial 3D template. CLV consists of running the script and visually inspecting the output to ensure that the code and the produced geometry align with the intended modeling purpose (e.g., correct organ types and hierarchy, plausible scales/units, expected attachment logic); CLV is qualitative and does not involve quantitative

Table 1
Qualitative comparison of FloraForge with representative plant modeling approaches.

Criterion	L-systems [11]	Demeter [8]	CropCraft [23]	Hadadi et al. [28]	LLM-3D [17,18]	FloraForge (ours)
Training data	None (rules)	600+ scans	None (per-instance)	None	None	None
Template design	Manual rules	Learned	Pre-defined	Manual	LLM (general)	LLM (botanical)
Representation	Skeletal graph	Latent/mesh	Procedural mesh	NURBS	Mesh/CSG	NURBS
Parametric edit	Rule params	Latent params	Procedural params	Control points	Code params	YAML params
Surface continuity	Discrete	C^0 mesh	C^0 mesh	C^1+	Varies	C^1+
Programming expertise	Expert	None	None	Expert	Moderate	None (Step 1: LLM)
Fitting expertise	N/A	Automatic	Automatic (BO)	Automatic (PSO+NURBS-Diff)	N/A	Manual (Step 2)
Simulation-ready	No	Yes (HELIOS)	Yes (rad. transfer)	Yes (STL+SMESH)	No	Yes (STL+SMESH)
Output format	Skeleton to mesh	Mesh	Mesh	STL + SMESH	Mesh/CSG	STL + SMESH
Reproducibility	Rules (code)	Learned weights	Optimized params	Optimized ctrl pts	Generated code	YAML (version-ctrl)



(a) Step 1: LLM-Assisted Template Co-Design & Code-Level Verification

(b) Step 2: Manual Instance Fitting & 3D Model Verification

Fig. 2. Two-stage FloraForge workflow. (a) LLM produces the *procedural generator* and an *LLM-seeded descriptor*; executing these yields the *initial 3D template* and passes *code-level verification*. (b) The *procedural generator* is reused while the *plant descriptor* is manually refined; executing these yields the *3D model instance*, which is validated by *3D model verification* (visual inspection + Chamfer distance).

fitting at this stage. In *Step 2* (manual instance fitting and 3D Model Verification, MV), no LLM interaction occurs; the PPG is held fixed while the PD is iteratively edited by the user. This editing requires botanical knowledge and geometric intuition; users adjust parameters such as node heights, leaf angles, curvature values, and hinge positions, but it does not require programming. After each edit, the PPG is executed to render a candidate model that is visually compared against the target point cloud; edits continue until visual agreement is achieved. The final output is a model instance, for which MV additionally quantifies agreement using the symmetric Chamfer distance (CD).

Scope of LLM contribution. To clearly delineate roles, the LLM's contribution is confined to Step 1, where it generates the PPG code and the initial PD template from natural language dialogue. The LLM does not participate in Step 2 (model fitting) or in the generation of the quantitative results reported in Section 5. For the dicot template, the co-design process involved 4 dialogue sessions totaling approximately 70 iterative prompts; for the maize template, 1 session with approximately 14 prompts (representative excerpts are shown in Table A.1). Both templates were developed within approximately two weeks, producing a combined ~2,100 lines of specialized NURBS procedural code. For reference, the authors' prior NURBS-based maize pipeline [28], which addresses a related problem (automatic inverse fitting), required approximately 12 months of development by the same first author without LLM assistance. While these pipelines differ in scope, the comparison illustrates the acceleration that LLM-assisted code generation provides for NURBS-based plant modeling. The dicot PPG was further developed through soybean-focused dialogue but applied without modification to mung bean, demonstrating that the design captured generalizable botanical structure rather than species-specific heuristics. Step 2 replaces programming with the more accessible (though still nontrivial) task of adjusting biologically meaningful PD parameters. Soybean thus serves only as the botanical reference for the Step 1 template co-design; soybean point cloud data were not available, so all quantitative fitting and trait results reported in Section 5 are computed on maize and mung bean instances.

3.1. Hierarchical plant architecture

Plant topology is encoded as a node-based hierarchy reflecting species-specific botanical organization. The plant geometry is generated from the topological graph by providing geometric descriptors at each node. The procedural generation framework accommodates two fundamental architectural patterns:

Dicotyledonous architecture (soybean/mung bean). Dicot models implement a four-level hierarchy:

1. **Main stalk**: Central stem with nodes positioned at absolute heights $\{z_1, z_2, \ldots, z_N\}$ and internodes connecting consecutive nodes. Stem curvature is controlled via per-node curvature parameters.
2. **Petioles**: Branch structures extending from nodes at species-specific azimuth, pitch, and roll angles. Each node can support one or more petioles (typically 1–2 per node in our implementations).
3. **Petiolules**: Secondary branches extending from petiole tips in a trifoliate arrangement (terminal, left lateral, and right lateral), each with independent orientation and curvature parameters.
4. **Leaflets**: NURBS surface leaflets attached to petiolule tips, with per-leaflet control over morphology (length, width, shape, camber, twist, fold) and orientation (yaw, pitch, roll).

Each component exhibits distinct geometric properties: stems, petioles, and petiolules are modeled as a generalized cylinder with the main axis

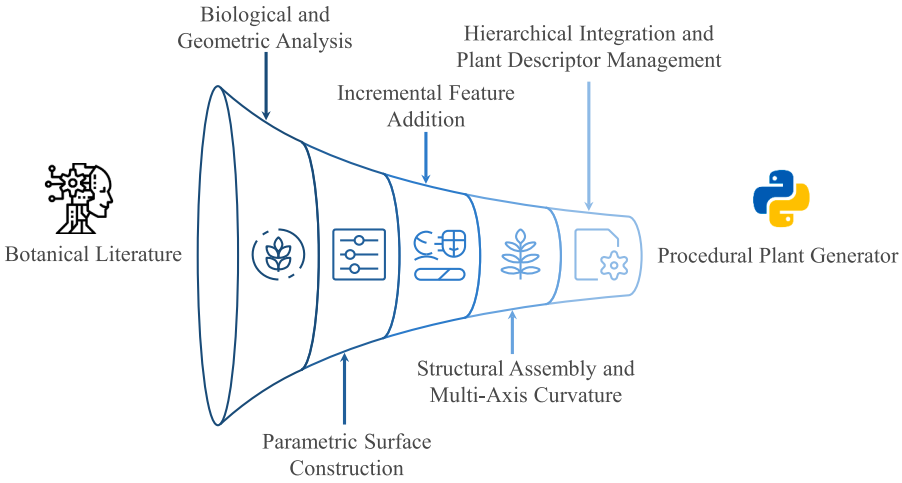


Fig. 3. LLM-assisted iterative development workflow showing the five-stage progression from botanical literature analysis through parametric surface construction, incremental feature addition, and structural assembly to final hierarchical integration and Plant Descriptor (PD) management, resulting in a Procedural Plant Generator (PPG).

being the stem and the sweeping profile formed by a circular cross-section NURBS curve with controllable diameter variation and 3D curvature; leaflets are modeled as tensor product bi-variate NURBS surfaces with explicit parametric control over all morphological attributes.

Monocotyledonous architecture (maize). Monocot models implement a simplified two-level hierarchy:

1. **Culm (main stem)**: Central axis with phyllotactic node arrangement (typically alternating 180° azimuthal positioning). The culm extends beyond the terminal leaf by a configurable apical extension to represent the vegetative apex.
2. **Leaf blades**: Linear-lanceolate leaves attached directly to nodes without intermediate branching structures (petioles/petiolules). Leaf attachment uses hard-graft geometry where the leaf base control points are sampled from the culm surface to ensure seamless integration.

Maize leaves are characterized by linear geometry with monotonic width taper, longitudinal venation represented through camber curvature, and localized hinge bending at user-defined positions to capture characteristic blade drooping.

Hierarchical parameter inheritance. PD parameters follow a hierarchical precedence: global defaults specify baseline values for all organs; per-node overrides adjust parameters at specific nodes; and per-organ overrides (e.g., per-leaflet in trifoliates) provide fine-grained control. This inheritance structure minimizes parameter redundancy while preserving biological realism and user flexibility.

Species-agnostic generality. The architectural framework is designed to generalize across monocot and dicot families that share the currently implemented branching topologies (stalk-only for monocots; stalk → petiole → petiolule → leaflet for dicots). During hierarchical construction, the presence or absence of intermediate branching structures (petioles, petiolules) is determined by plant type. In particular:

- Maize (monocot): Leaves attach directly to the stalk without branching structures. The algorithm proceeds from stalk construction directly to leaf attachment at nodes.
- Soybean/Mung Bean (dicot): Compound leaves require petiole branches from nodes, with petiolules branching from petioles and leaflets terminating petiolules.

This flexibility enables extension to other species with similar branching topologies (e.g., other grasses, legumes, or compound-leaved crops) by specifying the appropriate branching topology and morphological parameters in PDs without modifying the underlying procedural algorithms. Extension to fundamentally different growth forms (e.g., vines, rosettes, trees with secondary branching) would require additional algorithmic modules, as discussed in Section 6.

3.2. LLM-assisted code generation process

The modeling workflow originated from structured, iterative human-AI dialogues designed to refine domain-specific Python scripts progressively. Rather than attempting to generate complete solutions in a single step, we employed an iterative refinement strategy. Each conversation focused on solving a single, well-defined sub-problem before advancing to the next component. This approach mirrors how expert modelers incrementally build complex systems and has been shown to improve stability, interpretability, and convergence compared to monolithic generation approaches. Fig. 3 illustrates the five-stage iterative development workflow.

OpenAI's GPT-5 Thinking was systematically guided through the following iterative development stages:

Stage 1: Biological and geometric analysis. The LLM was prompted to analyze botanical literature and generate detailed 3D geometric descriptions of target plant organs. This stage established the morphological vocabulary and geometric primitives required for subsequent modeling steps.

Stage 2: Parametric surface construction. Beginning with the smallest structural unit (the leaflet), the LLM generated initial Python code to construct NURBS surfaces using the `NURBS-Python (geomdl)` library [44]. Early implementations produced geometric errors (e.g., inverted normals, collapsed control points, incorrect knot vectors), which were identified through visual inspection and fed back to the LLM with specific correction instructions. Reference images from botanical atlases, point cloud data from physical plants, and morphological constraints from literature (e.g., typical length-to-width ratios, curvature ranges) were progressively incorporated into the dialogue to constrain the solution space and improve biological realism.

For dicot leaflets, initial surfaces were flat and symmetric. Iterative refinement introduced asymmetric width profiles (controlled by separate left and right beta-function parameters), midrib offset, and localized apex/base shaping.

For monocot blades, the LLM initially implemented leaflets using dicot-style width profiles. Feedback specified the need for monotonic tapering to zero width at the tip and linear geometry, prompting the generation of the power-law width profile and tip-collapse control point arrangement.

Stage 3: Incremental feature addition. Once basic surface generation was validated, additional morphological realism was iteratively introduced through focused refinements, each tested and validated before proceeding:

- **Leaf asymmetry:** Midrib offset, per-side width bias, and lateral skew parameters (see width profiles in Section 4).
- **Global shape modifications:** Camber (cross-sectional bending; Eq. (13)), longitudinal bend (gravitational droop for dicots, upward arch for monocots; Eq. (12)), twist about the midrib (Eq. (14)), and V-fold (dicot-specific folding; Eq. (15)).
- **Per-leaf orientation:** Yaw, pitch, and roll angles for leaflet attachment to petiolule/culm frames (frame construction in Section 4).
- **Localized bending (monocot-specific):** Hinge-style transformations enabling sharp curvature at arbitrary positions along maize blades, with smoothstep-based transition zones to avoid control point discontinuities (Eq. (16)).
- **Hard-graft attachment (monocot-specific):** Replacement of leaf base control point rows with culm surface samples, including azimuthal arc sampling and radial offset control, to eliminate artificial gaps between blade and culm (Eq. (19)).

Each feature was added individually, with the LLM generating modified code, the user executing and visualizing outputs, and iterative corrections addressing geometric artifacts or parameter coupling issues (e.g., hinge bending causing unintended tip drift, which was resolved by rotating about per-column pivot points rather than a single global pivot).

Stage 4: Structural assembly and multi-axis curvature. After establishing leaf modules, the workflow proceeded to construct the stem, petiole, and petiolule using a node hierarchy and curved centerline generation. The LLM generated code implementing:

- **Turtle-based centerline construction:** Inspired by the L-systems, we used the incremental tangent vector updates via rotation matrices, advancing position along curved trajectories.
- **Single-axis curvature:** Per-internode curvature κ (centerline curvature, e.g., in rad/cm; radius R=1/κ; see Eq. (1)) and azimuth ϕ defining the bending plane.
- **Multi-axis piecewise curvature:** Extension to support multiple simultaneous curvature terms with localized spans (e.g., spiral curvature via overlapping orthogonal bending planes), enabling complex 3D trajectories.
- **Parallel transport frames:** Twist-minimizing frame propagation for circular sweep surface construction.
- **Branch attachment frames:** Computation of local coordinate systems using Frenet frames (tangent-normal-binormal) at petiole and petiolule base points using azimuth, pitch, and roll angles.

Errors in coordinate system conventions (e.g., azimuth mapping to incorrect bending directions), parameter broadcasting (e.g., a single curvature value applied per node vs. per petiole), and rotation composition order were identified through trial runs and corrected through iterative dialogue refinement.

Stage 5: Hierarchical integration and PD management. The final integration phase connected leaf modules to branch attachment points (petiolules for dicots, culm nodes for monocots) with per-node, per-petiole, and per-leaf parameter control. The LLM consolidated all command-line arguments into hierarchical PDs. This hierarchical parameter resolution enables complete specification of plant architecture state through structured, version-controllable data files, eliminating hard-coded parameters and enabling reproducibility across research groups.

Pipeline outputs. The procedural pipeline produces three output formats, each optimized for distinct use cases:

- **STL mesh files (.stl):** Tessellated triangular meshes derived from NURBS surface evaluation, suitable for visualization, rendering, and 3D printing. Tessellation density is controlled via surface sampling parameters (`sample_size_u`, `sample_size_v`) to balance geometric fidelity and file size. Note that the NURBS can also be adaptively sampled, generating more and smaller triangles in areas with higher curvature.
- **Analytic SMESH files (.dat):** Parametric surface metadata encoding NURBS control points, knot vectors, and degrees for all plant organs. SMESH format preserves the continuous mathematical representation, enabling direct import into isogeometric analysis frameworks, CAD software, and custom analysis pipelines without polygonal approximation.
- **YAML PD files (.yaml):** Human-readable parameter files recording all architectural and morphological parameters, ensuring complete reproducibility of generated models. PDs serve as version-controllable "plant recipes", enabling collaborative model refinement and systematic parameter space exploration.

4. Mathematical foundations

Plant organs in FloraForge are represented as tensor product NURBS surfaces[2], providing smooth, mathematically continuous geometric primitives essential for analysis-ready modeling. Standard B-spline basis functions, tensor-product surface formulations, and rotation/frame-transport foundations are summarized in Appendix B for completeness; here, we present the formulations specific to our procedural plant-generation framework.

4.1. Stem and branch curvature via piecewise multi-axis bending

Stem and branch curvature is controlled through curvature parameters κ applied incrementally along discretized segments.

$$\kappa(s) = \left\| \frac{d\mathbf{T}}{ds} \right\|, \qquad \mathbf{T}(s) = \frac{d\mathbf{C}}{ds}, \qquad R_\kappa(s) = \frac{1}{\kappa(s)}. \tag{1}$$

For each internode segment of length L, divided into n_{seg} steps with stepsize $ds = L/n_{\text{seg}}$, the tangent direction $\mathbf{T}$ is updated via rotation about a specified axis $\mathbf{a}$:

$$\mathbf{T}_{k+1} = \mathbf{R}(\mathbf{a}, \kappa\, ds)\, \mathbf{T}_k. \tag{2}$$

The next centerline point is then computed as:

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{T}_{k+1}\, ds. \tag{3}$$

For single-axis curvature, the bending plane normal (rotation axis) is defined by the azimuth angle ϕ in the global xy-plane:

$$\mathbf{a}(\phi) = \begin{bmatrix} -\sin(\phi) \\ \cos(\phi) \\ 0 \end{bmatrix}. \tag{4}$$

This parameterization maps azimuth $\phi = 0°$ to the $+y$ axis and $\phi = 90°$ to the $-x$ axis, enabling intuitive directional control of stem bending.

For multi-axis piecewise curvature (implemented via `*-kappa-terms` in PDs), multiple curvature terms

$$\{(\kappa_j, \mathbf{a}_j, [s_0^j, s_1^j])\} \tag{5}$$

[2] All plant organ surfaces in this work use NURBS with uniform unit weights ($w_{i,j} = 1$), as plant organs do not require exact conic section representation. The framework and output formats are fully compatible with non-unit weights if needed for future applications.

can be applied simultaneously within specified fractional spans $[s_0, s_1] \subset [0, 1]$ of an internode. At each step fraction $s = k/n_{\text{seg}}$, all active terms (where $s_0^j \le s \le s_1^j$) contribute rotations sequentially:

$$\mathbf{T}_{k+1} = \prod_{j \in \text{active}(s)} \mathbf{R}(\mathbf{a}_j, \kappa_j \, ds) \, \mathbf{T}_k. \tag{6}$$

This formulation enables localized bending, spiral curvature, and complex 3D stem trajectories through the addition of curvature components.

4.2. Circular sweep surface construction

Stems, petioles, and petiolules are represented as generalized cylinder NURBS surfaces generated by sweeping circular cross-sections along curved centerline polylines. Given a centerline $\{\mathbf{p}_i\}_{i=0}^{M}$, radius profile $\{r_i\}_{i=0}^{M}$, and parallel transport frames $\{(\mathbf{T}_i, \mathbf{N}_i, \mathbf{B}_i)\}_{i=0}^{M}$, the control point grid is constructed by sampling U points around circular rings at each centerline vertex:

$$\mathbf{P}_{u,v} = \mathbf{p}_v + r_v\big(\cos(\theta_u)\,\mathbf{N}_v + \sin(\theta_u)\,\mathbf{B}_v\big) \tag{7}$$

where $\theta_u = 2\pi u/U$ for $u \in \{0, 1, \ldots, U\}$ (with $u = U$ duplicating $u = 0$ to close the surface seam) and $v \in \{0, 1, \ldots, M\}$. The resulting control point grid of size $(U + 1) \times (M + 1)$ defines a NURBS surface with degrees p_u and p_v (typically 2 and 3, respectively) in the circumferential and axial directions.

4.3. Leaf surface geometry: Parametric width profiles and deformations

Leaf blades (both dicot and monocot blades) are constructed as NURBS surfaces defined by the control point matrices $\mathbf{P}_{i,j}$ with dimensions $M_u \times M_v$ (longitudinal × transverse). The base control matrix is generated in a local coordinate system with the x-axis aligned with the leaf midrib and the yz-plane spanning the leaf width.

Width profile formulation. For monocot blades (maize), the geometry is a ruled surface where the leaf width tapers monotonically from base width W_0 to zero at the tip using a power-law profile:

$$W(u) = W_0\,(1 - u^p), \quad u \in [0, 1], \tag{8}$$

where u is the normalized longitudinal coordinate ($u = 0$ at base, $u = 1$ at tip) and p controls the taper rate (default $p \approx 1.2$).

For dicot leaflets (soybean/mung bean), asymmetric width profiles are generated using a beta-function-inspired formulation:

$$W(u; \alpha, \beta) = W_{\max}\, u^{\alpha}\,(1 - u)^{\beta} \tag{9}$$

normalized such that $\max_u W(u) = W_{\max}$. Parameters α and β control the position and sharpness of maximum width, enabling the representation of diverse leaf shapes (ovate, lanceolate, and linear). Additional localized modulation functions sharpen the apex and round the base:

$$W_{\text{final}}(u) = W(u) \cdot \big(1 - 0.35\, a_{\text{apex}}\, f_{\text{tip}}(u)\big) \cdot \big(1 + 0.20\, a_{\text{base}}\, f_{\text{base}}(u)\big), \tag{10}$$

where constants 0.35 and 0.20 are empirically determined shape coefficients, $f_{\text{tip}}(u) = \max(0, (u - 0.8)/0.2)$ and $f_{\text{base}}(u) = \max(0, (0.15 - u)/0.15)$ are ramp functions, and $a_{\text{apex}}, a_{\text{base}}$ are shape-specific coefficients.

Leaf centerline and longitudinal curvature. The leaf midrib centerline $\mathbf{C}(u)$ in local coordinates is constructed with optional lateral skew and longitudinal bending:

$$\mathbf{C}(u) = \begin{bmatrix} L\,u \\ y_{\text{skew}}(u) \\ z_{\text{bend}}(u) \end{bmatrix}, \tag{11}$$

where L is leaf length, $y_{\text{skew}}(u) = \alpha_{\text{skew}} W_0 \sin(\pi u)$ introduces lateral asymmetry, and the longitudinal bending term is

$$z_{\text{bend}}(u) = \beta_{\text{bend}}\, L \sin(\pi u). \tag{12}$$

This creates an upward (or downward) arch. For dicots, the sign convention is negative ($\beta_{\text{bend}} < 0$) to produce gravitational droop; for monocots, positive values produce upward curvature typical of grass leaves.

Cross-sectional camber. Transverse curvature (camber) is applied perpendicular to the midrib using a longitudinally varying amplitude:

$$z_{\text{camber}}(u, v) = \alpha_{\text{camber}} L\,(1 - |v|)^{p_{\text{camber}}}\, \sin(\pi u), \tag{13}$$

where $v \in [-1, 1]$ is the transverse coordinate (negative = left, positive = right), α_{camber} is the camber amplitude coefficient, and p_{camber} controls the decay rate from midrib to edge (higher values produce flatter edges).

Twist and V-fold. Longitudinal twist is introduced by rotating the local transverse frame progressively along the midrib:

$$\theta_{\text{twist}}(u) = \alpha_{\text{twist}}\, u. \tag{14}$$

applied via rotation matrix $\mathbf{R}(\mathbf{T}(u), \theta_{\text{twist}}(u))$ where $\mathbf{T}(u)$ is the local tangent to the midrib.

V-folding along the midrib (common in dicot leaflets) is implemented by rotating control points about the tangent vector $\mathbf{T}(u)$ with an angle proportional to transverse position:

$$\theta_{\text{fold}}(u, v) = \alpha_{\text{fold}}\, \text{sign}(v)\, |v|^{p_{\text{fold}}}\, [\sin(\pi u)]^{q_{\text{fold}}}, \tag{15}$$

where p_{fold} controls the fold sharpness and q_{fold} modulates the envelope along the leaf length.

Localized hinge bending (monocot). Maize leaves often exhibit sharp localized bends at specific positions along the blade. This is implemented via hinge transformations applied to control point rows distal to a specified normalized position u_0:

$$\mathbf{P}_{i,j}^{\text{hinge}} = \mathbf{P}_{i_0,j} + \mathbf{R}(\mathbf{a}_{\text{hinge}}, w_i\, \theta_{\text{hinge}})\,(\mathbf{P}_{i,j} - \mathbf{P}_{i_0,j}), \tag{16}$$

where i_0 corresponds to u_0, $\mathbf{a}_{\text{hinge}}$ is the hinge rotation axis (typically the binormal $\mathbf{B}_{i_0}$), θ_{hinge} is the hinge angle, and w_i is a smooth ramp weight:

$$w_i = \begin{cases} 0 & u_i < u_0 \\ \text{smoothstep}\left(\frac{u_i - u_0}{\sigma}\right) & u_i \ge u_0 \end{cases}, \quad \text{smoothstep}(x) = x^2(3 - 2x) \tag{17}$$

with transition width σ. Multiple hinges can be applied sequentially to create complex sigmoidal curvature.

Hard-graft leaf base (monocot). For realistic attachment of maize leaves to the culm, the first n_{graft} rows of the leaf control point grid are replaced with points sampled from the culm surface. At each graft row $g \in \{0, 1, \ldots, n_{\text{graft}} - 1\}$, positioned at height:

$$z_g = z_{\text{node}} + g \cdot \frac{\Delta z_{\text{axial}}}{n_{\text{graft}} - 1}, \tag{18}$$

the control points are sampled from a circular arc on the culm surface centered at azimuth ϕ_{leaf} with arc span $\Delta\phi$:

$$\mathbf{P}_{g,j} = \mathbf{p}_{\text{culm}}(z_g) + (r_{\text{culm}}(z_g) + \delta_r)\big[\cos(\phi_j)\,\mathbf{N}_{\text{culm}}(z_g) + \sin(\phi_j)\,\mathbf{B}_{\text{culm}}(z_g)\big], \tag{19}$$

where $\phi_j = \phi_{\text{leaf}} + (2j/(M_v - 1) - 1)\,\Delta\phi/2$, $r_{\text{culm}}(z_g)$ is the culm radius at height z_g, δ_r is a small radial offset, and $\mathbf{N}_{\text{culm}}, \mathbf{B}_{\text{culm}}$ are the culm normal and binormal vectors. This grafting procedure ensures seamless integration of the leaf surface with the culm geometry, eliminating artificial gaps or penetrations. An optional soft blending zone can be applied to rows immediately beyond the graft region ($g > n_{\text{graft}}$, $u < u_{\text{attach}}$) using linear interpolation:

$$\mathbf{P}_{i,j}^{\text{blend}} = (1 - \alpha_i)\,\mathbf{P}_{i,j}^{\text{leaf}} + \alpha_i\,\mathbf{P}_{i,j}^{\text{culm-ring}}, \quad \alpha_i = 1 - \text{smoothstep}\left(\frac{u_i}{u_{\text{attach}}}\right) \tag{20}$$

ensuring a smooth transition from rigid graft to free blade geometry.

Algorithm 1: Unified procedural plant generation (monocot & dicot).

Input: YAML Plant Descriptor (PD): node-z, stem geometry, leaf parameters
Output: STL mesh and SMESH parametric surface files
`// Stage 1: Build stem/culm centerline via turtle-based integration`
Load YAML: node heights $\{z_i\}$, diameters $\{d_i\}$, curvature $\{\kappa_i\}$;
Initialize: $\mathbf{p} \leftarrow [0,0,0]$, $\mathbf{t} \leftarrow [0,0,1]$;
for *each internode* i **do**
  for *integration steps* **do**
    Apply curvature: $\mathbf{t} \leftarrow \mathbf{R}(\mathbf{a}_i, \kappa_i \cdot ds) \cdot \mathbf{t}$;
    Advance: $\mathbf{p} \leftarrow \mathbf{p} + \mathbf{t} \cdot ds$;
  end
end
if *monocot* **then**
  Extend culm beyond terminal leaf;
end
Compute parallel transport frames $\{(\mathbf{T}_j, \mathbf{N}_j, \mathbf{B}_j)\}$;
Generate stem surface via circular sweep;
`// Stage 2: Branch and leaf construction`
for *each node* i **do**
  if *dicot* **then**
    for *each petiole at node* i **do**
      Build petiole centerline with curvature;
      Generate petiole NURBS surface;
      for *trifoliate sides: Terminal, Left, Right* **do**
        Build petiolule centerline;
        Generate petiolule NURBS surface;
        if *leaflet enabled* **then**
          Generate leaflet control grid (beta-function width, camber, V-fold);
          Orient to petiolule tip frame;
          Create leaflet NURBS surface;
        end
      end
    end
  end
  if *monocot* **then**
    if *leaf enabled at node* i **then**
      Generate blade control grid (power-law taper, camber, bend);
      Apply hinge bends at specified positions;
      `// Hard-graft: replace base rows with culm surface samples`
      for *graft rows* $g = 0, \ldots, n_{graft} - 1$ **do**
        Sample culm surface points at node height;
        Replace control grid base with culm samples;
      end
      Create blade NURBS surface;
    end
  end
end
`// Stage 3: Export`
Tessellate all NURBS surfaces → STL mesh;
Export NURBS metadata → SMESH files;

4.4. Chamfer distance for model verification

To quantify geometric agreement between each model instance and its target point cloud, we use the symmetric Chamfer distance (CD). Given a model point set M and a target point set P, the symmetric CD is

$$\mathrm{CD}(M,P) = \frac{1}{|M|}\sum_{\mathbf{m}\in M}\min_{\mathbf{p}\in P}\|\mathbf{m}-\mathbf{p}\|_2 + \frac{1}{|P|}\sum_{\mathbf{p}\in P}\min_{\mathbf{m}\in M}\|\mathbf{p}-\mathbf{m}\|_2. \tag{21}$$

All chamfer distances reported in this work use this symmetric formulation with Euclidean norms in meters.

4.5. Framework implementation

Our method encapsulates the species-agnostic procedural generation workflow, with conditional branching for species-specific features. Fig. 4 illustrates the computational workflow from PD loading through NURBS surface generation to final export. The complete algorithm (see Algorithm 1) consists of 3 stages. Stage 1 constructs stem or culm centerlines via incremental turtle-based integration with Rodrigues rotation for curvature application, followed by parallel transport frame computation and circular sweep surface generation. Stage 2 branches based on botanical architecture: dicotyledonous species (soybean/mung bean) construct hierarchical compound leaf structures through nested petiole→petiolule→leaflet levels, generating 7 NURBS surfaces per trifoliate set (1 petiole + 3 petiolules + 3 leaflets); monocotyledonous species (maize) implement simplified architecture with leaf blades attaching directly to culm nodes via hard-graft integration, where basal n_{graft} control point rows are sampled from the culm surface to eliminate attachment gaps, generating 1 surface per node. Stage 3 performs species-agnostic tessellation and dual-format export (STL for visualization, SMESH for parametric representation). The framework enables extension to arbitrary plant species through PD without algorithmic modification.

PD structure and parametric editability. The human-readable PD format enables domain experts to specify complex plant architectures without programming knowledge. Fig. A.1 demonstrates the hierarchical parameter structure for two morphologically distinct maize plants, highlighting how architectural variation is achieved through parameter editing.

5. Results

The procedural modeling pipeline successfully generated biologically accurate, analysis-ready plant models for multiple species and developmental stages. We demonstrate this capability on five maize genotypes and three mung bean plants at a single time point, and we additionally model one mung bean plant across five developmental stages using the same procedural template. Across the single-time-point fits, we achieve mean symmetric Chamfer distances of 0.021 m for maize and 0.005 m for mung bean when validated against empirical point cloud data (Sections 5.2.1 and 5.2.2).

5.1. Dataset

The maize data used in this study were selected from the publicly available *MaizeField3D* dataset [45], which provides multi-genotype, field-grown maize point clouds acquired using LiDAR-based scanning at Iowa State University. Five plants (T8, CML69, M162W, CML238, and CI90C) were chosen from this dataset, corresponding to genotypes representative of diverse morphological architectures. Each point cloud dataset was used to manually fit the procedural models described below.

The mung bean data used in this study were collected from experimental fields at Iowa State University's Agricultural Engineering and Agronomy Research Farm. Each genotype was grown in a separate 5-foot, three-row plot, containing approximately 100 plants per plot. Plants were sampled at the R3 growth stage. 3D models of mung bean plants were generated from iPhone video recordings using a Neural Radiance Field (NeRF)-based 3D reconstruction method [46]. For cross-sectional analysis, we used 3D point clouds from three randomly selected plants, each representing a different genotype, for procedural model generation. In addition, to study temporal dynamics, we reconstructed a single mung bean plant at five distinct developmental stages and fit a common procedural template to this longitudinal sequence.

5.2. Procedurally generated plant models

We demonstrate the pipeline's capability to generate editable, analysis-ready model instances by fitting procedural templates to

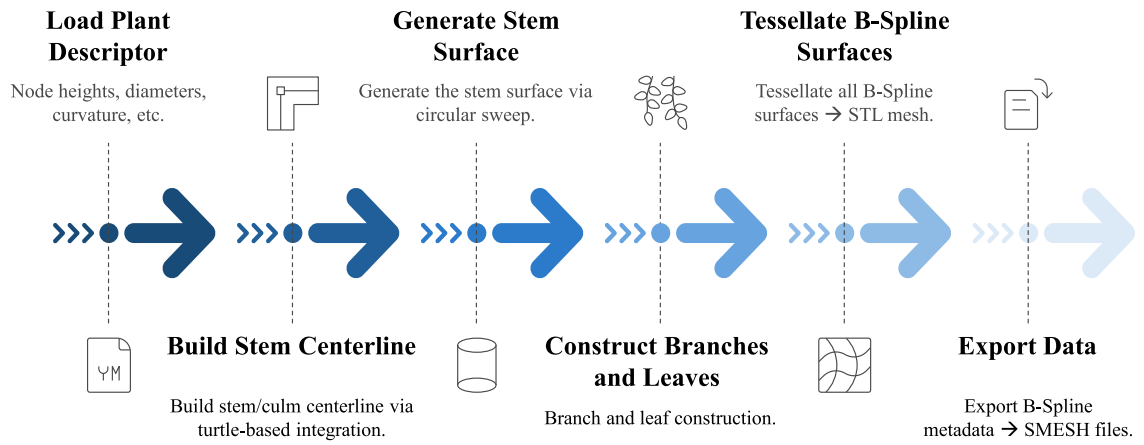
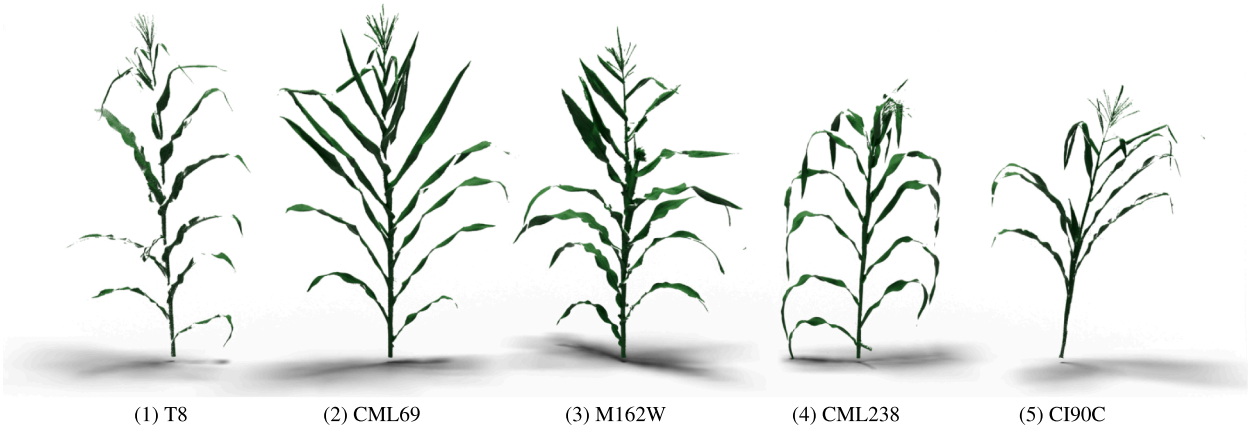
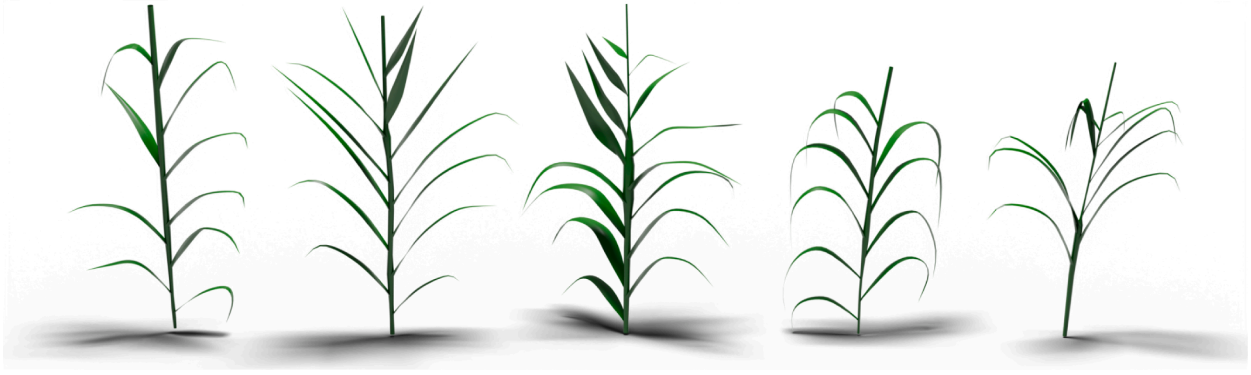


Fig. 4. Procedural generation workflow showing the computational pipeline from the Plant Descriptor (PD) as input through stem centerline construction, branch/leaf assembly, and dual-format export (STL mesh and SMESH parametric surfaces).



(a) Input point cloud data of real plants from LiDAR scanning.



(b) Corresponding procedurally generated models.

Fig. 5. Comparison of empirical LiDAR point cloud data (a) and fitted procedural models (b) for five maize genotypes. The procedural models accurately reproduce plant-specific architectural features, including culm trajectories, leaf orientations, and blade morphology, while maintaining continuous NURBS surface representations suitable for downstream computational analysis.

empirical point cloud data for multiple plant species and developmental stages.

5.2.1. Maize models

Five maize plants representing diverse genotypes were generated by iteratively adjusting PD to achieve geometric correspondence with LiDAR point cloud reconstructions. Fig. 5 compares the scanned point clouds with the LLM-assisted generated 3D model instances, demonstrating the accurate capture of key vegetative morphological features: culm curvature and diameter variation, distichous phyllotactic leaf arrangement, hard-graft leaf attachment geometry, and complex blade morphology, including localized hinge bending and monotonic width tapering. Tassels are present in the point clouds but were not recreated, since the current modeling effort focuses on stalk-and-leaf architecture. Each model required approximately 2–6 hours of parameter adjustment to achieve geometric correspondence with the point cloud data. The PDs for these five plants demonstrate the range of morphological variation achievable through parameter editing.

Fig. 6. Top: LLM-seeded *initial template* generated automatically from (i) approximate node z-positions and (ii) the observed number of leaves for each plant; leaves are placed using a default distichous phyllotaxy (0°/180° alternation) and no per-plant curvature or hinge tuning. Bottom: corresponding manually tuned procedural model instances.

We quantify the geometric fidelity of the fitting using the symmetric Chamfer distance (CD). Across five maize genotypes, manual tuning of the LLM-seeded template reduces CD from **0.1 m** on average to **0.02 m** (mean reduction ≈**79%**; Fig. 7). Per-genotype reductions range from **72%** to **85%** (3.6–6.8× lower CD). This indicates that while the LLM-seeded template provides a plausible initial geometry, targeted adjustments (node heights, leaf length/width, pitch, and localized hinge bends) substantially improve agreement with the point cloud.

To make the starting point explicit, we visualize an LLM-seeded *initial template* behind each fitted maize model in Fig. 6. For every genotype, the initial template PD is drafted by the LLM from the approximate node heights and the observed number of leaves, with leaves placed using a default 0°/180° distichous phyllotaxy and no manual tuning (e.g., no hinge bends, no node-specific curvature overrides). The foreground shows the manually tuned instance. The visual gap between the two aligns with the CD reductions reported in Fig. 7. The figure additionally reports the symmetric CD achieved by the authors' prior two-step optimization pipeline [28], which performs fully automatic PSO and gradient-based NURBS fitting on the same five maize plant point clouds. The primary contribution of FloraForge is the procedural generator pipeline (Step 1), not the fitting process (Step 2): the fact that manual parameter adjustment alone achieves a mean CD of 0.021 m, within 0.01 m of a dedicated automatic optimization pipeline (0.011 m), validates that the LLM-assisted procedural generator produces geometry capable of closely representing real plant architectures. The remaining accuracy gap highlights the value of automating the fitting process.

Ablation study. To isolate the contributions of individual modules, we evaluate geometric fidelity as features are progressively removed. All such comparisons start from the same LLM-generated Step 1 template, so they quantify the contribution of Step 2 manual refinement and of specific procedural features rather than the contribution of the LLM-generated template itself. First, the comparison between LLM-seeded templates and manually tuned instances (Fig. 7) shows that Step 2 refinement reduces the mean CD from 10.2 cm to 2.0 cm, confirming that the LLM initialization provides a plausible starting geometry that targeted parameter adjustment substantially improves. Second, Table 2 reports the effect of disabling hinge bending: mean CD increases from 2.0 cm to 3.46 cm (+73%), with genotype-dependent effects ranging from +3% (CML69, straight leaves) to +177% (CML238, strongly drooping leaves), confirming that localized blade curvature is essential for capturing characteristic maize leaf morphology. Third, removing hard-graft attachment had minimal impact on CD (+4%), as the graft region constitutes a small fraction of total plant surface; its primary contribution is visual fidelity at the leaf-stem junction rather than global geometric accuracy.

Table 2
Ablation study: effect of hinge bending on symmetric Chamfer distance (cm) across five maize genotypes.

Plant	Full model	Without hinge bending	Improvement
T8	2.22	3.08	39%
CML69	2.43	2.50	3%
M162W	1.47	3.88	164%
CML238	1.96	5.42	177%
CI90C	1.91	2.41	26%
Mean	**2.00**	**3.46**	**73%**

Phenotypic trait validation. To complement the symmetric Chamfer distance, we compared phenotypic traits extracted from input point clouds and from FloraForge-fitted surfaces using consistent measurement

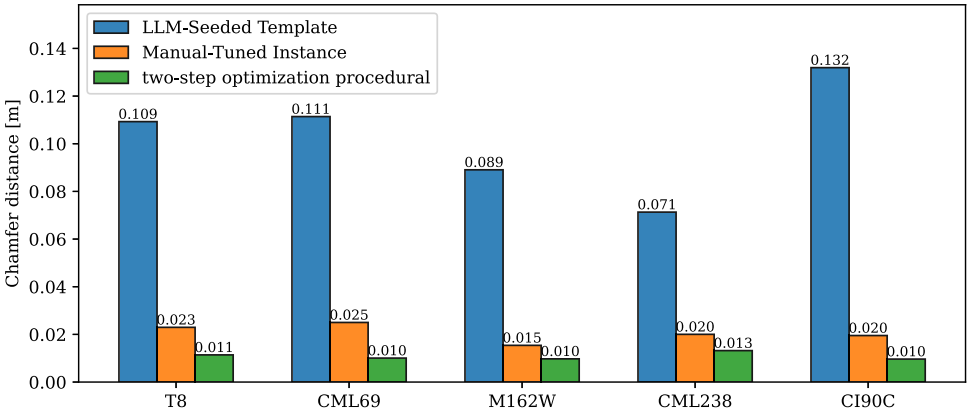


Fig. 7. Symmetric Chamfer Distance (CD) for maize: LLM-seeded initial template vs. FloraForge manual-tuned instance vs. two-step automatic optimization pipeline [28]. Manual tuning reduces the mean CD from 0.102 m to 0.021 m; the automatic optimization achieves 0.011 m on the same point clouds.

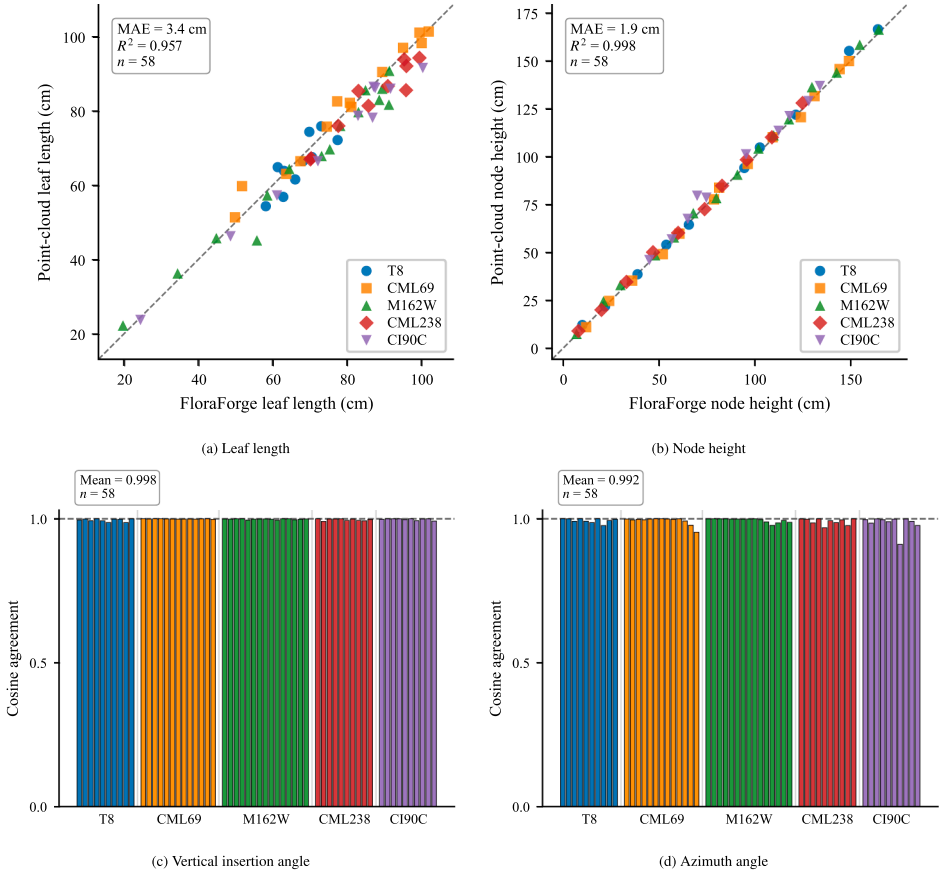


Fig. 8. Phenotypic trait agreement between measurements extracted from empirical point clouds and fitted FloraForge models for five maize genotypes (58 leaves). Top row: scatter plots for leaf length and node height with 1:1 reference lines. Bottom row: per-genotype cosine agreement for vertical insertion angle and azimuth, where values near 1.0 indicate near-perfect directional correspondence.

protocols (Fig. 8). Leaf length was measured as the curvilinear midrib path from the leaf base to the tip, node height as the vertical position of the leaf attachment point on the culm, vertical insertion angle as the angle between the initial linear midrib segment and the local stalk direction, and azimuth as the horizontal compass direction from the leaf base to the tip. Across 58 leaves from five maize genotypes, the mean absolute errors were 3.4 cm for leaf length ($R^2 = 0.957$) and 1.9 cm for node height ($R^2 = 0.998$). For angular traits, the mean per-genotype cosine agreement was 0.998 for vertical insertion angle (MAE = 3.2°) and 0.992 for azimuth (MAE = 5.1°), indicating near-perfect directional correspondence. At the whole-plant level, plant height agreed within 7.0 cm. These trait-level comparisons show that the fitted procedural models preserve biologically meaningful architectural traits in addition to achieving geometric agreement with the source point clouds. Per-leaf azimuth scatter plots are provided in Fig. A.2 for additional detail.

5.2.2. *Mung bean models*

Three mung bean plants were modeled to demonstrate the pipeline's capability to represent dicotyledonous compound leaf architectures. Fig. 9 compares empirical point cloud data with fitted procedural models, illustrating the framework's capacity to capture hierarchical branching structures characteristic of trifoliate legumes.

(a) Original mung bean point cloud data

(b) Corresponding procedurally generated models

Fig. 9. Comparison of original point cloud data (a) and fitted procedural model instances (b) for three mung bean plants. The hierarchical compound leaf architecture (stalk→petiole→petiolule→leaflet) is accurately reproduced, demonstrating the framework's capability to represent complex multi-level branching structures characteristic of dicotyledonous species.

Each mung bean instance was fitted to its point cloud by iteratively editing PD parameters and visually verifying architectural traits. A per-leaf trait protocol analogous to the maize analysis (Fig. 8) does not transfer cleanly to a trifoliate compound leaf, since the petiole → petiolule → leaflet structure does not provide a single midrib path per leaf or an unambiguous vertical insertion-angle reference; we therefore did not report a directly comparable per-leaf trait analysis. During fitting, the operator visually verified main-stalk node placement, stalk curvature, and stem diameter; per-node petiole length, azimuth, pitch, and diameter; trifoliate-set arrangement (terminal, left-lateral, and right-lateral petiolules) together with per-petiolule length, pitch, and diameter; per-leaflet length and aspect ratio, and leaflet azimuth and pitch relative to the supporting petiolule; localized leaflet shape modifiers (bending, V-fold, and midrib offset) where needed; and overall plant height and canopy envelope. Fitting was concluded once these traits agreed visually with the point cloud.

After visual convergence, we quantified the geometric fidelity for mung bean plants using the symmetric CD between each fitted model instance and its point cloud (Fig. 10). CD values were 0.005, 0.006, and 0.004 m for MB_1-MB_3, respectively (mean ≈ 0.005 m; SD ≈ 0.001 m), indicating sub-centimeter agreement between the procedurally generated models and the reconstructions.

5.2.3. *Multi-timepoint modeling: Mung bean growth stages*

To demonstrate the framework's capability for longitudinal phenotyping, in addition to the three single-time-point mung bean plants, we modeled a single mung bean plant across five developmental stages. Fig. 11 compares point cloud reconstructions with fitted procedural models at each time point. By reusing the same PPG and updating only growth-related parameters in the PD (node heights, internode lengths, leaf dimensions, and petiole/petiolule lengths), we achieve a consistent architectural representation across developmental stages while capturing morphological changes during vegetative growth.

5.3. *Virtual light simulation with HELIOS*

To demonstrate the analysis-ready nature of our procedurally generated models and their utility for functional-structural plant modeling, we performed virtual light interception simulations using the HELIOS 3D modeling framework [47]. We evaluated canopy-level photosynthetically active radiation (PAR) interception for both a mature maize field and mung bean plants across five developmental stages, demonstrating the framework's applicability to diverse crop architectures and phenological contexts.

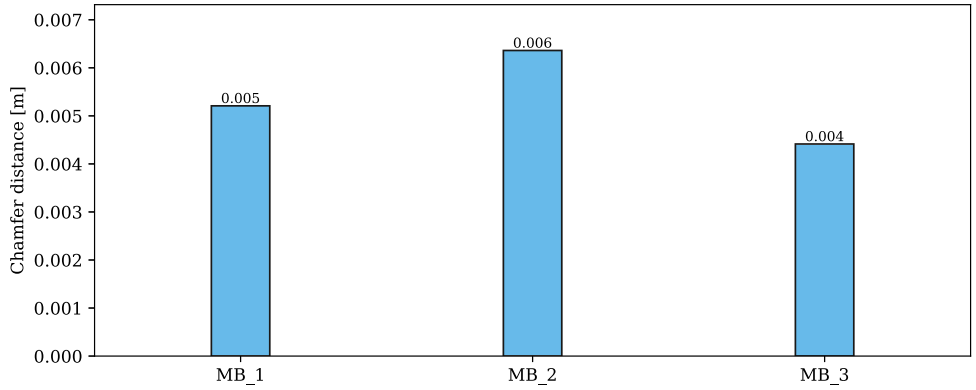


Fig. 10. Symmetric Chamfer Distance (CD) between each fitted model instance and its point cloud for three mung bean plants (MB_1-MB_3).

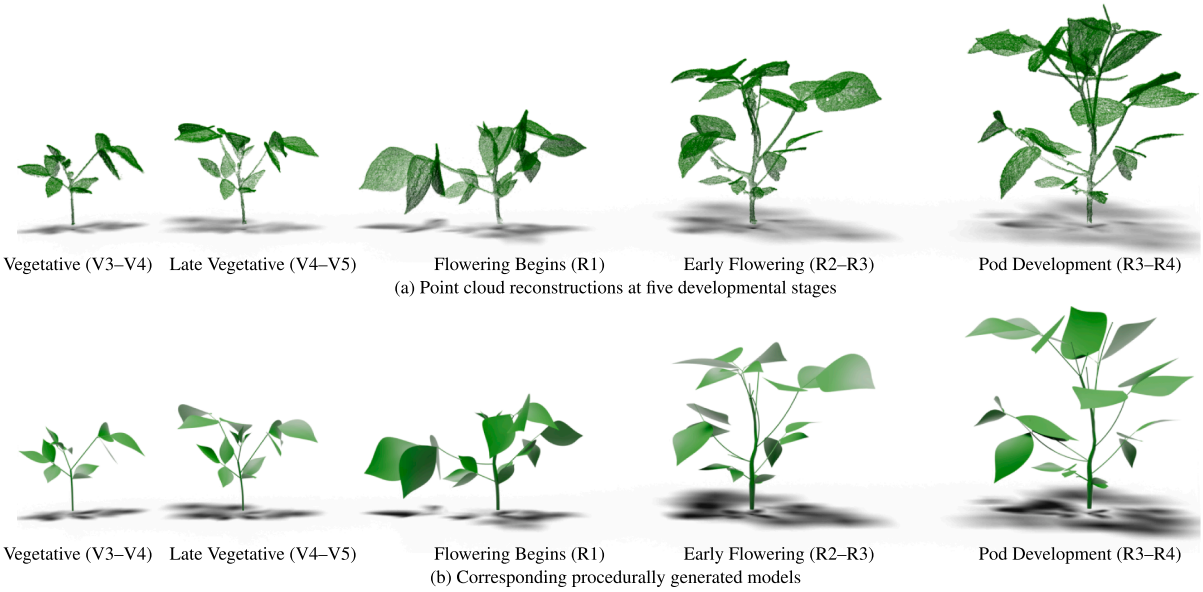


Fig. 11. Multi-timepoint modeling of a single mung bean plant across vegetative growth stages. (a) NeRF-reconstructed point clouds from different time points. (b) Fitted procedural models using consistent template architecture with timepoint-specific growth parameters. The framework captures progressive stem elongation, leaf expansion, and emergence of secondary trifoliate sets, demonstrating capability for longitudinal phenotyping and growth trajectory analysis.

Virtual field construction. For each simulation, we constructed virtual field plots from the STL mesh outputs generated by our procedural pipeline (Section 3). Individual plant STL files were replicated with randomized azimuthal rotations ($\pm 20^\circ$ about the vertical axis) to simulate natural morphological variability, then assembled into field configurations and converted to PLY format for compatibility with HELIOS. This workflow demonstrates the direct utility of procedural outputs for downstream simulation applications without requiring manual mesh editing or intermediate processing steps.

Maize field: We constructed a 3-row by 15-plant grid following standard maize agronomic spacing (30 in. or 0.762 m between rows, 6 in. or 0.152 m between plants), yielding 45 plants per field. We selected one representative genotype from the five fitted maize models (Section 5.2.1) and generated 45 structural variants by applying randomized azimuthal rotations ($\pm 20^\circ$ about the vertical axis) to create the virtual plot. This approach preserves genotype-level architectural features while introducing realistic intra-genotype diversity.

Mung bean fields: For each of the five growth stages from Section 5.2.3, we constructed a 3-row by 24-plant grid following mung bean agronomic spacing (15 in./0.381 m between rows, 5.5 in./0.140 m within-row spacing), yielding 72 plants per field. For each growth stage, we generated 72 structural variants through randomized azimuthal rotations ($\pm 20^\circ$) of the base procedural model.

Radiation simulation protocol. We configured HELIOS to simulate diurnal PAR exposure over a full day (7:00–20:00) using geographical coordinates for Ames, Iowa (42.03°N, 93.63°W; UTC-5). To ensure realistic atmospheric conditions, simulations were configured with date-specific meteorological parameters corresponding to actual field data collection dates (Table 3). For maize, we used representative anthesis-stage conditions (August 7, 2020, 60 days after planting). For mung bean growth stages, temperature and relative humidity were obtained from the Iowa Environmental Mesonet archive [48,49] for the respective measurement days. Atmospheric pressure was set to 101.3 kPa for all simulations. We assumed clear-sky conditions and set the Ångström aerosol turbidity coefficient to $\beta = 0.05$, a typical value for clean mid-latitude continental atmospheres and the default example used by the REST2 clear-sky model in HELIOS [50].

The simulation computed absorbed PAR on all exposed plant surfaces at hourly intervals using area-weighted radiation fluxes. Solar position was calculated using the built-in solar position model in HELIOS, which accounts for atmospheric refraction and the observer's latitude. Direct solar radiation was modeled as a collimated source with 1000 rays per band to ensure accurate shadow casting and penetration into the canopy. Diffuse radiation was implicitly handled through the REST2 solar flux calculations. Leaf optical properties in the PAR band were set to representative values for C4 (maize) and C3 (mung bean) foliage:

Table 3
Date-specific atmospheric conditions used in HELIOS simulations. Temperature and relative humidity values for mung bean were retrieved from Iowa Environmental Mesonet historical weather data for Ames, Iowa. Maize conditions represent typical anthesis-stage values.

Crop	Growth Stage	Date	Temp. (K)	RH	Turbidity
Maize	Anthesis	Aug 7, 2020	300.00	0.500	0.05
Mung bean	Vegetative (V3–V4)	July 23, 2025	300.65	0.780	0.05
	Late Vegetative (V4–V5)	July 27, 2025	299.82	0.864	0.05
	Flowering Begins (R1)	Aug 1, 2025	291.48	0.682	0.05
	Early Flowering (R2–R3)	Aug 8, 2025	300.37	0.800	0.05
	Pod Development (R3–R4)	Aug 12, 2025	296.48	0.826	0.05

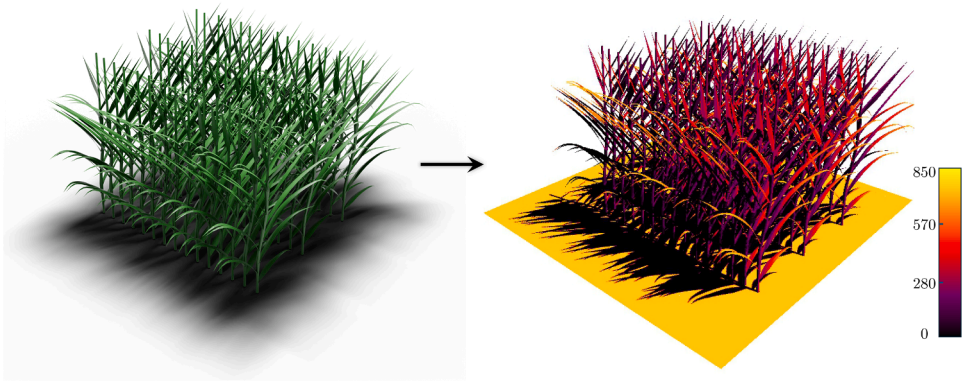


Fig. 12. HELIOS radiation simulation for a procedurally generated maize field. Left: Virtual field plot (3 rows × 15 plants, 45 plants total) constructed from a fitted maize model with 45 structural variants generated through randomized azimuthal orientations (±20°). Right: Radiation simulation output showing spatially varying absorbed PAR mapped across canopy surfaces using color gradients.

reflectivity $\rho = 0.05$, transmissivity $\tau = 0.05$, and emissivity $\varepsilon = 0.90$. Scattering depth was set to 5 to account for multiple reflections within dense canopies. Total daily intercepted PAR was obtained by integrating hourly absorbed flux values across the simulation period.

Fig. 12 illustrates the HELIOS simulation setup and output for the maize field. The left panel shows the procedurally generated virtual field, demonstrating the characteristic vertical architecture and distichous leaf arrangement captured by our models. The right panel displays the radiation simulation result, where absorbed PAR is spatially mapped across the canopy using color gradients.

Fig. 13 presents the simulation setup and outputs for all five mung bean growth stages. The left column shows the procedurally generated virtual fields, demonstrating the progressive increase in canopy density and structural complexity from early vegetative (V3–V4) to pod development (R3–R4) stages. The right column displays the corresponding radiation simulation results. In early vegetative stages, sparse canopy coverage results in limited light interception, with most radiation reaching the ground surface. As development progresses, increasing leaf area and canopy closure lead to enhanced radiation capture, with the most developed stages (R2–R3 and R3–R4) showing near-complete interception of incoming PAR. These spatial patterns highlight the functional consequences of architectural traits captured by our procedural models.

Fig. 14 presents the diurnal PAR interception profiles for both crops. For maize at anthesis (Fig. 14a), the simulation yielded a total daily intercepted PAR of 46.1 kWh/m^2, with peak absorption occurring at 14:00. The diurnal curve exhibits a characteristic bell shape with a gradual increase during morning hours, sustained high interception during midday, and decline during afternoon. The relatively flat peak between 12:00–15:00 reflects the mature canopy structure with high leaf area index, enabling efficient light capture across a broad range of solar elevations.

For mung bean (Fig. 14b and Table 4), total daily intercepted PAR increased progressively from early vegetative stages to peak during pod development, reflecting the expansion of photosynthetic surface area through stem elongation, node addition, and trifoliate emergence captured by our temporal fitting process. Vegetative stages (V3–V4) exhibited the lowest PAR interception (1.36 kWh/m^2/day), consistent with limited canopy development and small leaf area. Late vegetative (V4–V5) and flowering initiation (R1) stages showed moderate increases (2.73 and 4.33 kWh/m^2/day, respectively) as additional trifoliate sets expanded and canopy closure began. Early flowering (R2–R3) and pod development (R3–R4) stages achieved peak interception values (8.81 and 10.77 kWh/m^2/day), corresponding to maximal leaf area index and complete canopy closure observed in the data.

Table 4
Total daily intercepted PAR for virtual crop fields. Maize values represent a mature canopy at anthesis. Mung bean values show progressive increase across developmental stages. All values represent area-weighted absorbed radiation integrated over the simulation period (7:00–20:00).

Crop	Growth Stage	Total PAR (kWh/m^2)
Maize	Anthesis	46.1
Mung bean	Vegetative (V3–V4)	1.36
	Late Vegetative (V4–V5)	2.73
	Flowering Begins (R1)	4.33
	Early Flowering (R2–R3)	8.81
	Pod Development (R3–R4)	10.77

Implications for phenotyping and crop modeling. This analysis demonstrates three key capabilities enabled by our framework. First, the seamless integration of procedural outputs with physics-based simulation tools (HELIOS) confirms that our surface representations meet the geometric fidelity requirements for quantitative agroecosystem modeling. The use of date-specific atmospheric conditions and site-accurate solar geometry ensures that simulated PAR values reflect realistic growing

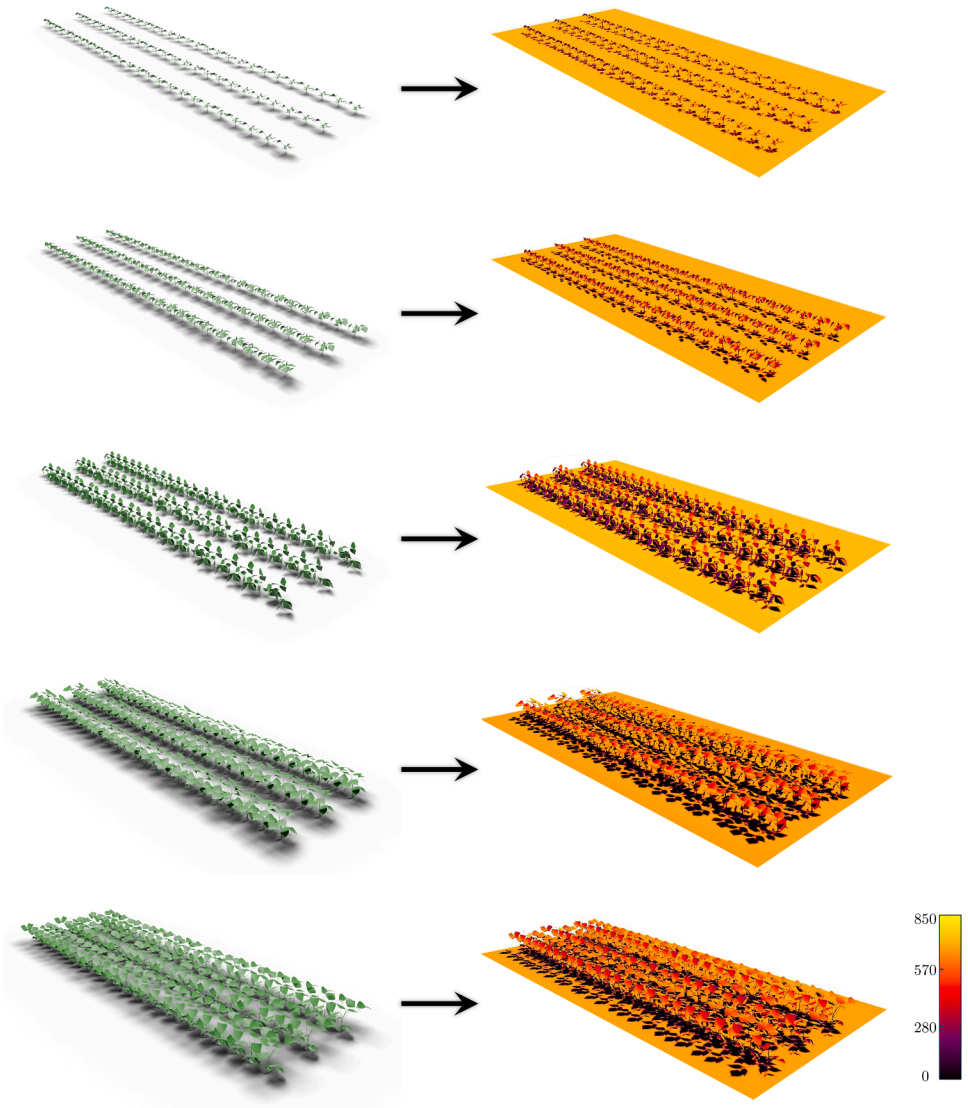
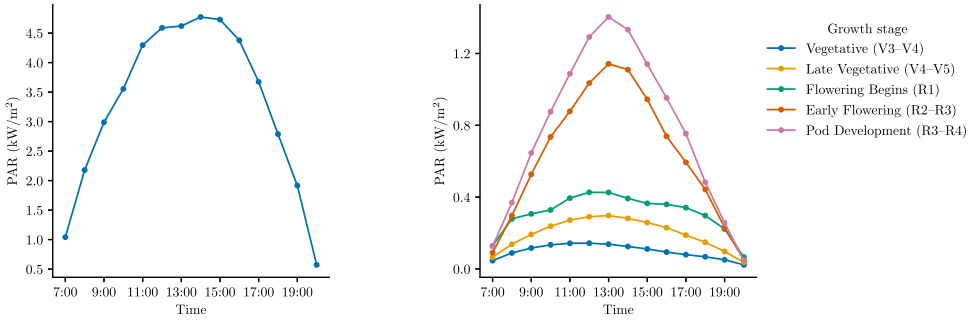


Fig. 13. HELIOS radiation simulation setup and outputs for mung bean across five growth stages. Left column: Procedurally generated virtual fields, each containing 72 plants (3 rows × 24 plants) generated as 72 structural variants with randomized azimuthal orientations (±20°). Right column: Corresponding radiation simulation outputs showing spatially varying absorbed PAR mapped across canopy surfaces. The progression from sparse early vegetative canopy (top) to dense pod development canopy (bottom) demonstrates how architectural changes captured by procedural modeling translate to functional differences in radiation interception patterns.



(a) Maize field (anthesis)

(b) Mung bean growth stages

Fig. 14. Diurnal PAR interception profiles. (a) Maize field at anthesis stage showing peak interception during midday with sustained high values reflecting mature canopy structure. (b) Mung bean plants across five developmental stages, demonstrating progressive increase in light capture from early vegetative (V3–V4, blue) to pod development (R3–R4, pink). Peak interception occurs during midday hours (12:00–14:00) across all stages. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 5
Morphological parameter ranges across fitted plant models.

Parameter	T8	CML69	M162W	CML238	CI90C	Unit
Maize Plant Architecture						
First node height	10	12	7	8	45	cm
Total height	201	196	197	163	171	cm
Number of leaves	10	13	15	10	10	–
Max diameter	42	52	70	40	40	mm
Max curvature (κ)	0.007	0.010	0.005	0.005	0.011	rad/cm
Leaf length range	60–80	50–102	20–85	50–78	25–102	cm
Pitch angle range	15–55	32–70	15–45	0–30	20–73	deg
Hinges per leaf	0–2	0–1	0–5	2–3	1–3	–
Mung Bean Architecture						
	MB_1	**MB_2**	**MB_3**			
Total leaflets	21	20	15			–
Petiole length range	3.0–16.0	11.0–20.5	4.5–19.8			cm
Leaflet length range	4.5–11.0	6.5–11.5	5.7–12.0			cm
V-fold angle range	5–20	10–20	5–20			deg

conditions, making the framework suitable for precision agriculture applications where environmental context is critical.

Second, the ability to perform virtual light experiments on both single mature plants (maize) and temporally consistent developmental series (mung bean) enables hypothesis testing about growth-environment interactions and architectural trade-offs without requiring destructive sampling or extensive field instrumentation. For example, the temporal mung bean series reveals how progressive canopy closure drives exponential increases in light capture efficiency during critical reproductive stages, providing quantitative insights into the relationship between architectural development and resource acquisition.

Third, the parametric nature of our models allows systematic exploration of architectural trait effects on light capture efficiency. For example, by modifying leaf angles, internode lengths, or branching patterns in the Plant Descriptors and re-running simulations to quantify functional consequences. Such virtual experimentation capabilities are increasingly valuable for crop ideotype design and breeding program optimization in precision agriculture contexts, where understanding the relationship between plant architecture and resource capture efficiency can inform selection strategies for improved yield and resource use efficiency under varying environmental conditions and management practices.

We note that the HELIOS PAR results demonstrate geometric compatibility with physics-based simulation tools rather than agronomically calibrated predictions; field validation of simulated radiation values is planned for future work. Additionally, the current virtual fields use a single fitted model per genotype with randomized azimuthal rotations, which does not capture true intra-genotype morphological diversity.

5.4. Morphological diversity

Analysis across fitted models quantifies morphological diversity and validates biological realism. Table 5 summarizes parameter ranges. Plants demonstrate substantial diversity across architectural scales (6.4-fold range in first node height), geometric parameters (curvature spanning 10-10^3cm bending radii), and morphological features (leaf lengths 4.5–102 cm, orientation covering the full spherical range). All parameter ranges fall within biologically plausible values, validating morphological fidelity. Maize models extensively use hinge bending (10–35 hinge points per plant), while dicot models achieve complexity through hierarchical branching and leaflet orientation. Overall, these ranges confirm that the fitted PDs span meaningful inter-genotype and inter-species variation while remaining within biologically plausible bounds.

5.5. Limitations and future work

Fitting automation. The current Step 2 workflow requires 2–6 hours of manual parameter adjustment per plant, compared to seconds-to-minutes for automatic methods such as Demeter and CropCraft once trained. This reflects a deliberate design trade-off: FloraForge prioritizes editability, transparency, and simulation-readiness over fitting speed, while requiring no training data, no species-specific annotation, and no segmented input—preparatory steps that can themselves require significant time and expertise (e.g., Demeter's 600+ annotated scans). The CD comparison with the authors' prior automatic pipeline [28] (Fig. 7) confirms that the procedural generator produces geometry with a mean CD within 0.01 m of a dedicated optimization method, validating the pipeline, while highlighting the value of automating Step 2. We are developing a VLM-based pipeline that automatically initializes PD parameters from multi-view images, reducing manual effort to leaf-level fine-tuning. As an initial feasibility example, Appendix A.3 shows VLM-predicted culm node placements from front-view maize renderings, which can provide approximate `node-z` initialization before manual refinement.

Single-operator fitting. All Step 2 manual fitting reported in this work was performed by a single expert user (the first author), so inter-rater variability in final model quality (the spread in achievable Chamfer distance, fitting time, and trait-level agreement when different users fit the same point cloud) is not characterized here. This is relevant because the visual-comparison component of Model Verification, while supplemented by the quantitative Chamfer distance, retains a subjective element. A multi-rater study, in which several users with comparable botanical training independently fit the same plants, would quantify this variability and is a natural direction for future evaluation.

Generalization boundaries. The framework currently supports two branching topologies: stalk-only for monocots and stalk to petiole to petiolule to leaflet for dicots. Extension to fundamentally different growth forms (vines, rosettes, trees with secondary branching) would require additional procedural modules developed through further LLM co-design sessions, though the existing PPG scripts and PD structures serve as documented starting points that can accelerate template development for new species. Code-level Verification in Step 1 relies on visual assessment against a bounded set of criteria, including organ presence, correct attachment hierarchy, plausible scales, and absence of geometric artifacts, supplemented by quantitative Chamfer distance in Step 2.

6. Conclusions

We presented FloraForge, an LLM-assisted procedural modeling framework that enables domain scientists to generate editable, analysis-ready 3D plant models without requiring expertise in geometric modeling. Users progressively refine PPG that implements NURBS surface representations with botanical constraints through iterative PR. The framework was validated across 8 plants spanning monocotyledonous (maize)

and dicotyledonous (mung bean) architectures, demonstrating successful adaptation to fundamentally different morphological organizations through human-readable PDs. Across these fitted instances, we obtained mean symmetric Chamfer distances of approximately 0.02 m for maize and 0.005 m for mung bean relative to LiDAR/NeRF point clouds. We demonstrated analysis-ready compatibility by driving HELIOS simulations of diurnal photosynthetically active radiation interception in virtual maize and mung bean fields. The continuous NURBS representation uniquely combines three essential properties: explicit parametric control for intuitive morphological editing, mathematical continuity ensuring smooth surfaces suitable for quantitative analysis, and direct compatibility with isogeometric analysis and computational simulation workflows. The dual-output format (STL meshes for visualization, SMESH parametric surfaces for analysis) addresses both graphics-oriented and computation-oriented applications.

Several future extensions would enhance the utility and automation of the framework. *Species generalization* to additional crops (wheat, rice, sorghum) and growth forms (rosette plants, vining species, trees) would demonstrate the breadth of the LLM-assisted co-design approach. Each new species requires a single template creation phase via LLM PR, after which multiple instances are generated through PD parameter editing. *Automated parameter extraction* from point clouds would eliminate the current manual fitting workflow (currently approximately 2 to 6 hours of expert time per plant), enabling high-throughput phenotyping. Potential approaches include geometric feature extraction with direct parameter mapping, gradient-based inverse procedural optimization, or learning-based methods where neural networks predict PD parameters from 3D scan input. *Temporal modeling* via parameter interpolation across developmental stages would enable growth simulation and prediction, with time-series point cloud data constraining continuous growth trajectories. Finally, *integration with simulation workflows*, including computational fluid dynamics for microclimate modeling, finite element analysis for lodging prediction, and global illumination for photosynthesis simulation, would demonstrate the analysis-ready nature of the NURBS representation for quantitative phenotyping and physics-based modeling applications.

FloraForge lowers the barrier to sophisticated 3D modeling capabilities for plant phenotyping, breeding, and precision agriculture research by bridging the expertise gap between botanical domain knowledge and geometric programming. The LLM-assisted co-design workflow transforms procedural template creation from a specialized programming skill into an accessible capability, enabling researchers to develop custom modeling tools tailored to specific scientific objectives. The remaining manual fitting step (Step 2) still requires botanical and geometric intuition but operates at a level of abstraction, adjusting biologically meaningful YAML parameters that is accessible to domain scientists without programming training. More broadly, the workflow demonstrated here, pairing a domain expert's natural-language description of a structured object with an LLM that synthesizes the corresponding geometric code and exposing the result through human-readable parametric files, may prove useful in other scientific settings where specialized geometric modeling is a bottleneck for non-programmer experts.

CRediT authorship contribution statement

Mozhgan Hadadi: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation; **Talukder Z. Jubery:** Writing – review & editing, Validation, Supervision, Project administration, Data curation, Conceptualization; **Patrick S. Schnable:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Funding acquisition; **Arti Singh:** Writing – review & editing, Supervision, Data curation; **Bedrich Benes:** Writing – review & editing, Supervision, Methodology; **Adarsh Krishnamurthy:** Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization; **Baskar Ganapathysubramanian:** Writing – review & editing, Writing – original draft, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization.

Ethical statement

All authors agree that:

This research presents an accurate account of the work performed, all data presented are accurate and methodologies detailed enough to permit others to replicate the work.

This manuscript represents entirely original works or if work or words of others have been used, that this has been appropriately cited or quoted and permission has been obtained where necessary.

This material has not been published in whole or in part elsewhere.

The manuscript is not currently being considered for publication in another journal.

That generative AI and AI-assisted technologies have not been utilized in the writing process or if used, disclosed in the manuscript the use of AI and AI-assisted technologies and a statement will appear in the published work.

That generative AI and AI-assisted technologies have not been used to create or alter images unless specifically used as part of the research design where such use must be described in a reproducible manner in the methods section.

All authors have been personally and actively involved in substantive work leading to the manuscript and will hold themselves jointly and individually responsible for its content.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the AI Institute for Resilient Agriculture (USDA-NIFA 2021-67021-35329), NSF BTT-EAGER IOS-1842097, NSF 2412929/2412928 and Iowa State University's Plant Science Institute. It was also supported by NSF 2506783, 2417510, 2412928, and USDA-NIFA 1032382 and 1032672 to Purdue. We thank Ian Ostermann for help with Blender rendering and visualization used in preparing the figures.

Data availability

The code (PPG scripts) used to generate the procedural models, YAML Plant Descriptors for all fitted models, and representative Plant Refinement (PR) prompt transcripts from the LLM co-design sessions in this work are available at the FloraForge GitHub repository (https://github.com/baskargroup/FloraForge).

Appendix A. Supplementary material

Fig. A.1 compares representative Plant Descriptors for two maize genotypes.

Plant (4) CML238

```
# Plant structure
outfile: out/maize_CML238.stl
seed: 2025

# Culm geometry (cm)
node-z: [8, 20, 34, 47, 60, 74,
         83, 96, 109, 127]

# Diameter profile (mm)
stem-diameter-mm: [15, 15, 15, 15,
                   24, 40, 33, 25,
                   25, 25]

# Culm curvature
stalk-kappa: [0.00, 0.00, 0.005,
              0.0, 0.00, 0.00,
              0.00, 0.00, 0.0,
              0.005]
stalk-bend-az-deg: [0, 0, 318, 0,
                    0, 0, 0, 0,
                    0, 318]

# Phyllotaxy (degrees)
leaf-az-deg: [116, 315, 115, 320,
              127, 320, 110, 330,
              115, 110]
leaf-pitch-deg: [0, 15, 15, 25, 30,
                 25, 30, 25, 30, 30]

# Leaf morphology defaults
leaves:
  globals:
    ctrl_u: 9
    ctrl_v: 5
    camber: 0.018
    camber_pow: 1.4
    width_pow: 1.20
```

Plant (5) CI90C

```
# Plant structure
outfile: out/maize_CI90C.stl
seed: 2025

# Culm geometry (cm)
node-z: [45, 57, 65, 70, 75, 96,
         113, 119, 129, 135]

# Diameter profile (mm)
stem-diameter-mm: [25, 35, 38, 40,
                   38, 10, 8, 15,
                   9, 13]

# Culm curvature
stalk-kappa: [0.002, 0.001, 0.002,
              0.0, 0.011, 0.002,
              0.00, 0.00, 0.0,
              0.00]
stalk-bend-az-deg: [280, 280, 180,
                    0, 288, 200,
                    0, 0, 0, 0]

# Phyllotaxy (degrees)
leaf-az-deg: [136, 272, 118, 140,
              273, 274, 80, 272,
              98, 288]
leaf-pitch-deg: [35, 40, 50, 40, 48,
                 45, 73, 35, 40, 20]

# Leaf morphology defaults
leaves:
  globals:
    ctrl_u: 9
    ctrl_v: 5
    camber: 0.018
    camber_pow: 1.4
    width_pow: 1.20
```

Per-leaf parameter overrides with localized bending

```
# CML238 - Leaf 3
nodes:
  3:
    length: 70
    width: 9.5
    bend: 0.35
    leaf-bend-hinges:
      - {u0: 0.1, angle-deg: 30,
         axis: B, smooth: 0.02}
      - {u0: 0.35, angle-deg: 20,
         axis: B, smooth: 0.02}
      - {u0: 0.85, angle-deg: 20,
         axis: B, smooth: 0.02}
```

```
# CI90C - Leaf 3
nodes:
  3:
    length: 91
    width: 8.5
    bend: 0.15
    leaf-bend-hinges:
      - {u0: 0.82, angle-deg: 90,
         axis: B, smooth: 0.03}
```

Fig. A.1. Comparison of plant descriptors for two morphologically distinct maize plants from Fig. 5. **Top:** Plant-level structural parameters. CML238 exhibits gradual internode spacing (8–127 cm) with minimal curvature, while CI90C shows elevated base height (first node at 45 cm) and pronounced culm bending ($\kappa = 0.011$ rad/cm). **Bottom:** Per-leaf parameter specifications. CML238 leaf 3 contains three localized hinges for complex sigmoidal curvature, whereas CI90C leaf 3 uses a single dramatic 90° bend at $u = 0.82$. The hierarchical structure (global defaults + node-specific overrides) reduces parameter space while maintaining biological realism. All parameters are human-readable and directly editable without programming knowledge.

Table A.1 shows representative Plant Refinement excerpts from the co-design sessions for both dicot (soybean, mung bean) and monocot (maize) templates.

Fig. A.2 reports the azimuth angle comparison referenced in Section 5.2.1.

Fig. A.3 illustrates the preliminary VLM-based node initialization referenced in Section 5.5.

Table A.1
Representative Plant Refinement (PR) prompts from the LLM co-design sessions (GPT-5 Thinking), organized by development stage. Prompts are lightly edited for brevity; full transcripts will be released.

Stage	Species	Prompt excerpt
1: Biological & geometric analysis	Soybean	"Describe the soybean leaf and trifoliate as a 3D geometry and what parameters does it have?"
	Maize	"Describe the maize plant as a 3D geometry."
2: Parametric surface construction	Soybean	"I want to use the geomdl library and create cylinders based on NURBS surfaces. The cross section can be periodic control points in a circular shape to create a closed NURBS curve and then sweep to create stalk, petiole, and petiolule. Export both STL and SMESH (.dat)."
	Maize	"Instead of length-to-width ratio, I want to assign width and length directly. The shape of the maize leaf starts with a specific width and at the tip the width becomes 0 (control points collapse to the same location)."
3: Error correction	Soybean	"The main stalk is towards $-z$, not $+z$. Why are the petioles pointing downward, not upward?"
	Maize	"Why does assigning `leaf-bend-hinges` change the tip control point placement? They are no longer at the same location after I assign hinges."
4: Feature addition	Soybean	"Right and left sides of the leaves are not exactly mirroring each other. Sometimes the midrib is a little towards the right or left side. Also, consider curvature of the leaves."
	Maize	"How can I have curvature in any part of the length of each leaf? Sometimes close to the tip I want a sharp curvature, or sometimes in the middle of the leaf."
5: Structural assembly	Soybean	"What if I want to bend them in various directions, for example bending at 0° and 90° at the same time? What if I want to bend any internode with 2 or more different curvatures?"
	Maize	"The stalk should continue after the last leaf. The control points on the part of the leaves connecting to the stalk should be in the form of the stalk at that internode, like a hard graft."
6: Integration & PD management	Soybean	"I want to control each set of trifoliate on the petiolules separately. How can I have more than 1 petiole at the same node?"
	Maize	"Add config file support. Put all arrays in a YAML instead of a long CLI."

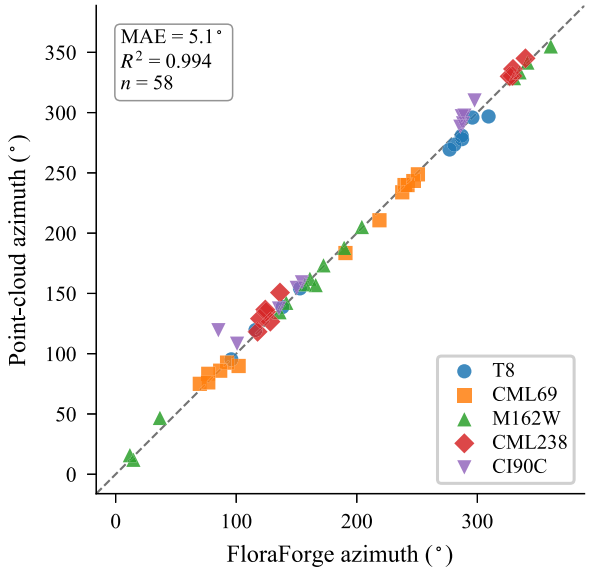


Fig. A.2. Trait-level comparison for leaf azimuth between empirical point clouds and fitted FloraForge models across 58 maize leaves. The dashed line indicates the 1:1 relationship. Azimuth is defined as the horizontal direction from the leaf base to the tip.

Fig. A.3. Preliminary feasibility examples for VLM-assisted node initialization. Given front-view maize renderings, an image-capable vision-language model places candidate node locations along the culm (magenta points), which can be used to initialize the plant descriptor parameter `node-z` before manual fine-tuning. These examples are intended to illustrate a path toward reducing Step 2 fitting time rather than a fully integrated or quantitatively evaluated component of the current FloraForge pipeline.

Appendix B. Standard mathematical background

This appendix provides the standard B-spline and rotation foundations used throughout the paper. These results are well-established in the computer-aided geometric design literature [37,38].

B.1. B-spline basis functions

At the core of B-splines are basis functions defined over a knot vector. A B-spline curve of degree p is defined by a set of $n+1$ control points $\{\mathbf{P}_i\}_{i=0}^{n}$ and a knot vector:

$$\mathbf{U} = \{u_0, u_1, \ldots, u_{n+p+1}\}, \tag{B.1}$$

containing non-decreasing real numbers that partition the parametric space. The B-spline basis functions $N_{i,p}(u)$ are defined recursively using the Cox-de Boor formula. Starting with piecewise constant functions for $p = 0$:

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1} \\ 0 & \text{otherwise.} \end{cases} \tag{B.2}$$

For higher degrees $p = 1, 2, 3, \ldots$, the basis functions are computed recursively:

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u). \tag{B.3}$$

The curve is then expressed parametrically as a weighted sum of control points:

$$\mathbf{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\,\mathbf{P}_i, \quad u \in [u_p, u_{n+1}]. \tag{B.4}$$

These basis functions possess several key properties that make them suitable for plant modeling: *local support* (each $N_{i,p}(u)$ is non-zero only over $p+1$ knot spans, enabling local shape control), *partition of unity* ($\sum_{i=0}^{n} N_{i,p}(u) = 1$ for all u, ensuring affine invariance), and *non-negativity* ($N_{i,p}(u) \ge 0$), ensuring smooth interpolation and intuitive geometric control.

B.2. B-spline surface formulation

B-spline surfaces extend the concept of B-spline curves into two dimensions using tensor products. Given two knot vectors $\mathbf{U} = \{u_0, \ldots, u_{n+p+1}\}$ and $\mathbf{V} = \{v_0, \ldots, v_{m+q+1}\}$, along with an $(n+1) \times (m+1)$ grid of control points $\mathbf{P}_{i,j}$, the B-spline surface is defined as:

$$\mathbf{S}(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,p}(u)\, N_{j,q}(v)\, \mathbf{P}_{i,j}, \quad 0 \le u, v \le 1 \tag{B.5}$$

where $N_{i,p}(u)$ and $N_{j,q}(v)$ are one-dimensional B-spline basis functions (Eq. (B.3)) corresponding to the respective knot vectors, with degrees p and q in the u and v parametric directions, respectively. This tensor product structure enables independent control of surface smoothness and shape in both parametric directions.

NURBS extend B-splines by introducing weights $w_{i,j}$ for each control point, enabling exact representation of conic sections (circles, ellipses) and improved control over surface shape. However, in this work, all plant organ surfaces are represented as non-rational B-splines with uniform weights ($w_{i,j} = 1$ for all control points). This choice is motivated by three considerations. First, plant organs (leaves, stems, and branches) do not require exact conic section representation; the polynomial basis of non-uniform B-splines provides sufficient geometric complexity to capture organic shapes with high biological fidelity. Second, non-rational formulations eliminate weight normalization computations, reducing computational overhead during surface evaluation and tessellation while maintaining numerical stability. Third, without weight parameters, the relationship between control point positions and surface geometry is more intuitive, simplifying manual parameter adjustment and automated fitting procedures. The resulting non-uniform B-spline representation provides smooth, mathematically continuous surfaces with explicit parametric control, enabling direct editing through PD parameters and compatibility with isogeometric analysis workflows.

B.3. Rotation and frame transport

Procedural construction of curved stems and branches requires smooth propagation of local coordinate frames (tangent $\mathbf{T}$, normal $\mathbf{N}$, binormal $\mathbf{B}$) along centerline curves. We use a right-handed frame where the binormal is

$$\mathbf{B}_i = \mathbf{T}_i \times \mathbf{N}_i, \tag{B.6}$$

with $\times$ denoting the vector cross product. Rotation about an arbitrary axis $\mathbf{a}$ by angle θ is computed using Rodrigues' rotation formula:

$$\mathbf{R}(\mathbf{a}, \theta) = \mathbf{I} + \sin(\theta)\,[\mathbf{a}]_\times + (1 - \cos(\theta))\,[\mathbf{a}]_\times^2, \tag{B.7}$$

where $[\mathbf{a}]_\times$ is the skew-symmetric cross-product matrix of the unit axis vector $\mathbf{a}$:

$$[\mathbf{a}]_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}. \tag{B.8}$$

Expanding Rodrigues' formula explicitly:

$$\mathbf{R}(\mathbf{a},\theta) = \begin{bmatrix} a_x^2 C + c & a_x a_y C - a_z s & a_x a_z C + a_y s \\ a_y a_x C + a_z s & a_y^2 C + c & a_y a_z C - a_x s \\ a_z a_x C - a_y s & a_z a_y C + a_x s & a_z^2 C + c \end{bmatrix}, \tag{B.9}$$

where $c = \cos(\theta)$, $s = \sin(\theta)$, and $C = 1 - \cos(\theta)$.

Parallel transport frames are computed along the stem centerline polyline $\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_M\}$ to minimize frame twist. At each vertex i, the tangent vector $\mathbf{T}_i$ is computed from adjacent points, and the normal $\mathbf{N}_i$ is propagated by rotating the previous normal $\mathbf{N}_{i-1}$ into the plane perpendicular to $\mathbf{T}_i$:

$$\mathbf{N}_i = \mathbf{R}(\mathbf{a}_i, \alpha_i)\,\mathbf{N}_{i-1}, \quad \mathbf{a}_i = \frac{\mathbf{T}_{i-1} \times \mathbf{T}_i}{\|\mathbf{T}_{i-1} \times \mathbf{T}_i\|}, \quad \alpha_i = \arccos(\mathbf{T}_{i-1} \cdot \mathbf{T}_i) \tag{B.10}$$

The binormal $\mathbf{B}_i$ is then computed using Eq. (B.6). This parallel transport procedure ensures smooth, twist-free orientation frames for the generalized cylinder construction.

References

[1] S. Paulus, Measuring crops in 3D: using geometry for plant phenotyping, Plant Methods 15 (2019) 103. https://doi.org/10.1186/s13007-019-0490-0

[2] J.A. Gibbs, M. Pound, A.P. French, D.M. Wells, E. Murchie, T. Pridmore, Approaches to three-dimensional reconstruction of plant shoot topology and geometry, Funct. Plant Biol. 44 (2016) 62–75. https://doi.org/10.1071/FP16167.

[3] J. Vos, J.B. Evers, G.H. Buck-Sorlin, B. Andrieu, M. Chelle, P.H.D. Visser, Functional–structural plant modelling: a new versatile tool in crop science, J. Exp. Bot. 61 (2010) 2101–2115. https://doi.org/10.1093/jxb/erp345.

[4] G. Louarn, Y. Song, Two decades of functional–structural plant modelling: now addressing complexity, Ann. Bot. 126 (2020) 501–509. https://doi.org/10.1093/aob/mcaa143.

[5] J. Guo, H. Jiang, B. Benes, O. Deussen, X. Zhang, D. Lischinski, H. Huang, Inverse procedural modeling of branching structures by inferring L-systems, ACM Trans. Graph. 39 (2020) 1–13. https://doi.org/10.1145/3394105.

[6] O. Stava, S. Pirk, J. Kratt, B. Chen, R. Měch, O. Deussen, B. Benes, Inverse procedural modelling of trees, Comput. Graph. Forum 33 (2014) 118–131. https://doi.org/10.1111/cgf.12282.

[7] O. Šťava, B. Benes, R. Měch, D.G. Aliaga, P. Krištof, Inverse procedural modeling by automatic generation of L-systems, Comput. Graph. Forum 29 (3) (2010) 665–674. https://doi.org/10.1111/j.1467-8659.2009.01636.x.

[8] T. Cheng, A.J. Zhai, E.Z. Chen, R. Zhou, Y. Deng, Z. Li, K. Zhao, J. Shiu, Q. Zhao, Y. Xu, et al., Demeter: a parametric model of crop plant morphology from the real world, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2025, pp. 28740–28751.

[9] Y. Yang, D. Mao, H. Santo, Y. Matsushita, F. Okura, NeuraLeaf: neural parametric leaf models with shape and deformation disentanglement, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2025, pp. 28167–28176.

[10] Z. Liu, Z. Cheng, N. Yokoya, Neural hierarchical decomposition for single image plant modeling, in: Proceedings of the Computer Vision and Pattern Recognition Conference, 2025, pp. 733–742.

[11] P. Prusinkiewicz, A. Lindenmayer, The Algorithmic Beauty of Plants, Springer Science & Business Media, 2012.

[12] P. Prusinkiewicz, J. Hanan, R. Měch, An L-system-based plant modeling language, in: International Workshop on Applications of Graph Transformations with Industrial Relevance, Springer, 1999, pp. 395–410. https://doi.org/10.1007/3-540-45104-8_31.

[13] P. Beardsley, G. Chaurasia, Editable parametric dense foliage from 3D capture, in: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE, 2017, pp. 5315–5324. https://doi.org/10.1109/ICCV.2017.567.

[14] A. Raistrick, L. Lipson, Z. Ma, L. Mei, M. Wang, Y. Zuo, K. Kayan, H. Wen, B. Han, Y. Wang, et al., Infinite photorealistic worlds using procedural generation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 12630–12641.

[15] P. Li, J. Guo, X. Zhang, D.-M. Yan, SECAD-Net: self-supervised CAD reconstruction by learning sketch-extrude operations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 16816–16826.

[16] P. Li, J. Guo, H. Li, B. Benes, D.-M. Yan, SfmCAD: unsupervised CAD reconstruction by learning sketch-based feature modeling operations, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, pp. 4671–4680.

[17] C. Sun, J. Han, W. Deng, X. Wang, Z. Qin, S. Gould, 3D-GPT: procedural 3D modeling with large language models, in: 2025 International Conference on 3D Vision (3DV), IEEE, 2025, pp. 1253–1263. https://doi.org/10.1109/3DV66043.2025.00119.

[18] S. Lu, G. Chen, N.A. Dinh, I. Lang, A. Holtzman, R. Hanocka, LL3M: large language 3d modelers, 2025, https://doi.org/10.48550/arXiv.2508.08228.

[19] Y. Du, S. Chen, W. Zan, P. Li, M. Wang, D. Song, B. Li, Y. Hu, B. Wang, BlenderLLM: training large language models for computer-aided design with self-improvement, 2024, https://doi.org/10.48550/arXiv.2412.14203.

[20] A. Jignasu, K. Marshall, B. Ganapathysubramanian, A. Balu, C. Hegde, A. Krishnamurthy, Evaluating large language models for G-code debugging, manipulation, and comprehension, in: 2024 IEEE LLM Aided Design Workshop (LAD), IEEE, 2024, pp. 1–5.

[21] O. Deussen, P. Hanrahan, B. Lintermann, R. Měch, M. Pharr, P. Prusinkiewicz, Realistic modeling and rendering of plant ecosystems, in: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, 1998, pp. 275–286. https://doi.org/10.1145/280814.280898.

[22] R. Měch, P. Prusinkiewicz, Visual models of plants interacting with their environment, in: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, 1996, pp. 397–410. https://doi.org/10.1145/237170.237279.

[23] A.J. Zhai, X. Wang, K. Li, Z. Jiang, J. Zhou, S. Wang, Z. Jin, K. Guan, S. Wang, CropCraft: inverse procedural modeling for 3d reconstruction of crop plants, 2024, arXiv preprint arXiv:2411.09693 [cs.CV]

[24] A. Lindenmayer, Mathematical models for cellular interactions in development I. Filaments with one-sided inputs, J. Theor. Biol. 18 (1968) 280–299. https://doi.org/10.1016/0022-5193(68)90079-9.

[25] P. Prusinkiewicz, M. James, R. Měch, Synthetic topiary, in: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, 1994, pp. 351–358. https://doi.org/10.1145/192161.192254.

[26] P. Prusinkiewicz, Graphical applications of L-systems, in: Proceedings of Graphics Interface, 86, 1986, pp. 247–253. https://doi.org/10.20380/GI1986.44.

[27] J.J. Lee, B. Li, B. Benes, Latent L-systems: transformer-based tree generator, ACM Trans. Graph. 43 (2024). https://doi.org/10.1145/3627101.

[28] M. Hadadi, M. Saraeian, J. Godbersen, T.Z. Jubery, Y. Li, L. Attigala, A. Balu, S. Sarkar, P.S. Schnable, A. Krishnamurthy, et al., Procedural generation of 3D maize plant architecture from LiDAR data, Comput. Electron. Agric. 236 (2025) 110382. https://doi.org/10.1016/j.compag.2025.110382.

[29] M. Makowski, T. Hädrich, J. Scheffczyk, D.L. Michels, S. Pirk, W. Pał ubicki, Synthetic silviculture: multi-scale modeling of plant ecosystems, ACM Trans. Graph. 38 (2019) 1–14.

[30] X. Zhou, B. Li, B. Benes, A. Habib, S. Fei, J. Shao, S. Pirk, TreeStructor: forest reconstruction with neural ranking, IEEE Trans. Geosci. Remote Sens. 63 (2025) 1–19. https://doi.org/10.1109/TGRS.2025.3558312.

[31] D.G. Aliaga, İ. Demir, B. Benes, M. Wand, Inverse procedural modeling of 3D models for virtual worlds, in: ACM SIGGRAPH 2016 Courses, SIGGRAPH '16, ACM, New York, NY, USA, 2016, pp. 16:1–16:316. https://doi.org/10.1145/2897826.2927323.

[32] I. McQuillan, J. Bernard, P. Prusinkiewicz, Algorithms for inferring context-sensitive L-systems, in: International Conference on Unconventional Computation and Natural Computation, Springer, 2018, pp. 117–130.

[33] A. Lotfi, I. McQuillan, Optimal L-systems for stochastic l-system inference problems, 2024, arXiv:2409.02259

[34] B. Li, J. Kał użny, J. Klein, D.L. Michels, W. Pał ubicki, B. Benes, S. Pirk, Learning to reconstruct botanical trees from single images, ACM Trans. Graph. 40 (1) (2021) 1–15. https://doi.org/10.1145/3478513.3480525.

[35] J.J. Lee, B. Li, S. Beery, J. Huang, S. Fei, R.A. Yeh, B. Benes, Tree-D fusion: simulation-ready tree dataset from single images with diffusion priors, in: Computer Vision - ECCV 2024, Springer Nature Switzerland, 2025, pp. 439–460.

[36] R. Ando, Y. Ozasa, W. Guo, Robust surface reconstruction of plant leaves from 3D point clouds, in: Plant Phenomics, 2021. https://doi.org/10.34133/2021/3184185.

[37] G. Farin, Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide, Elsevier, 2014.

[38] L. Piegl, W. Tiller, The NURBS Book, Springer Science & Business Media, 2012.

[39] E. Catmull, R. Rom, A class of local interpolating splines, in: Comput. Aided Geom. Des. Elsevier, Elsevier, 1974, pp. 317–326. https://doi.org/10.1016/B978-0-12-079050-0.50020-5.

[40] X. Yang, Curve and surface construction with moving B-splines, 2023, https://doi.org/10.48550/arXiv.2307.14677.

[41] D. Bradley, D. Nowrouzezahrai, P. Beardsley, Image-based reconstruction and synthesis of dense foliage, ACM Trans. Graph. 32 (2013) 1–10. https://doi.org/10.1145/2461912.2461952.

[42] S. Wu, A. Khasahmadi, M. Katz, P.K. Jayaraman, Y. Pu, K. Willis, B. Liu, CAD-LLM: large language model for CAD generation, in: Proceedings of the Neural Information Processing Systems Workshop on Machine Learning for Creativity and Design, 2023, pp. 1–3.

[43] A. Schüpbach, R. San Miguel, J. Ferchow, M. Meboldt, From text to design: a framework to leverage LLM agents for automated CAD generation, 5, 2025, pp. 1893–1902. https://doi.org/10.1017/pds.2025.10203.

[44] O.R. Bingol, A. Krishnamurthy, NURBS-Python: an open-source object-oriented NURBS modeling framework in python, SoftwareX 9 (2019) 85–94. https://doi.org/10.1016/j.softx.2018.12.005.

[45] E. Kimara, M. Hadadi, J. Godbersen, A. Balu, T. Jubery, Y. Li, A. Krishnamurthy, P.S. Schnable, B. Ganapathysubramanian, MaizeField3D: a curated 3D point cloud and procedural model dataset of field-grown maize from a diversity panel, Plant Phenomics (2025) 100108. https://doi.org/10.1016/j.plaphe.2025.100108.

[46] B. Mildenhall, P.P. Srinivasan, M. Tancik, J.T. Barron, R. Ramamoorthi, R. Ng, NeRF: representing scenes as neural radiance fields for view synthesis, Commun. ACM 65 (2021) 99–106. https://doi.org/10.1145/3503250.

[47] B.N. Bailey, Helios: a scalable 3D plant and environmental biophysical modeling framework, Front. Plant Sci. 10 (2019) 1185. https://doi.org/10.3389/fpls.2019.01185.

[48] R. Mahmood, R. Boyles, K. Brinson, C. Fiebrich, S. Foster, K. Hubbard, D. Robinson, J. Andresen, D. Leathers, MesoNets: mesoscale weather and climate observations for the United States, Bull. Am. Meteorol. Soc. 98 (2017) 1349–1361. https://doi.org/10.1175/BAMS-D-15-00258.1.

[49] D. Herzmann, J. Wolt, Iowa state university iowa environmental mesonet, 2024, (Accessed on November 30 (2020)).

[50] C.A. Gueymard, REST2: high-performance solar radiation model for cloudless-sky irradiance, illuminance, and photosynthetically active radiation–validation with a benchmark dataset, Sol. Energy 82 (2008) 272–285. https://doi.org/10.1016/j.solener.2007.04.008.