Generative Optimisation of Indoor Farm Design with Genetic Algorithms and Building Performance Simulation

Student Name: Vincent Kenny
Supervisor Name: Dr Tom Friedetzky
Submitted as part of the degree of [MEng Computer Science] to the
Board of Examiners in the Department of Computer Sciences, Durham University

Abstract—Vertical farming offers potential solutions to food security challenges but demands significant energy and complex design optimisation. This paper presents GIFDO (Generative Indoor Farm Design Optimiser), a computational framework integrating procedural geometry, EnergyPlus building performance simulation (via the OpenStudio SDK), and a Genetic Algorithm (GA). GIFDO automates the optimisation of indoor farms by simultaneously evolving building geometry, envelope materials, and operational setpoints. It balances potential crop profit against operational energy costs using a fitness function that rewards diminishing returns on profit relative to total energy expenditure. The framework was tested using weather data for Oslo (Norway) and Singapore. The GA effectively converged, generating architecturally distinct designs adapted to each climate; Oslo solutions featured compact, insulated forms, while Singapore designs favoured larger, predominantly glazed structures. GIFDO successfully identified complex design tradeoffs, such as balancing reduced volume against higher lighting intensity in the Oslo design. Predicted energy cost distributions aligned with literature benchmarks, although absolute predicted yield values are likely overestimated due to limitations in the employed crop model. This study validates that integrating genetic algorithms with building energy simulation provides a powerful methodology for exploring the vertical farm design space, identifying efficient solutions adapted to specific environmental contexts, thereby advancing the development of more efficient Controlled Environment Agriculture.

Index Terms—Architecture, Computer-aided design, Constrained optimization, Evolutionary computing and genetic algorithms,

1 Introduction

According to data from the world bank [1] the amount of arable land per person has decreased consistently year on year since 1960. Rising populations, combined with land scarcity caused by soil degradation, urbanisation, and climate change are creating challenges for food security in some parts of the world. Reliable access to fresh water is also a growing concern, with UNICEF predicting some 700 million people could be displaced by intense water scarcity by 2030. Chunyang et al. [2] highlight urban populations as those who will suffer the most, with 1.693 to 2.373 billion people facing water scarcity across 193 to 284 cities by 2050.

This race for resources is led by economic globalisation, and the increasing demand for the conversion of what were previously natural ecosystems into crop and pastoral land. Lambin and Meyfroidt [3] highlight land changes as a major driver of global environmental change, triggering deforestation and reduced global biotic diversity. In this paper the primary tools for slowing the trend of expansion, and thus alleviating resource scarcity, were identified: agricultural intensification, land use zoning and capital investment.

This necessity to adapt creates opportunities for governments and private organisations to innovate in the agricultural sphere, investing into growing methods with improved space efficiency, higher yields and vastly reduced water requirements. Vertical farming emerges as one such promising technique. Vertical farming is an innovative method of agriculture which involves stacking layers of crops on top of each other in closed, climate controlled structures. This approach allows for precise control over temperature, light, humidity, and CO2 levels which can lead to higher yields and faster crop cycles than traditional farming methods whilst protecting from pests, diseases and unpredictable weather patterns.

There are several examples of small commercially viable vertical farming operations running today. A current large-scale UK example is GrowUp Farms. Founded in 2013, the company opened its Pepperness vertical farm in 2022. The site draws renewable electricity and waste heat from an adjacent bio-energy plant and is designed to produce about 1.4 million bags of salad per week. The company sells their leafy greens under the "Unbeleafable" label, and they are distributed through major UK retailers.

There are, however, significant hurdles for vertical farming to overcome before it can become commercially viable or technically feasible on a large scale. Prohibitively large setup and operational costs due to high energy prices [4] can dissuade investors from providing capital, as profit margins can be slim to none depending on efficiency and yield. To add to this, running and managing the irrigation systems, climate control and lighting is an extremely complicated task, requiring sophisticated technology and expertise to design and operate. And here arises an opportunity for innovation: generative design offers a powerful computational approach to tackle these challenges by automating the exploration and evaluation of numerous design alternatives based on performance criteria.

The core design challenge for Controlled Environment Agriculture (CEA), including vertical farms, thus becomes a constrained optimisation problem: maximising valuable crop yield while minimising operational costs, particularly energy usage for lighting and climate control, within architectural and environmental constraints. To address this, this paper presents GIFDO (Generative Indoor Farm Design Optimiser), a computational framework developed specifically to integrate procedural geometry generation, Building Performance Simulation (BPS) using EnergyPlus via the OpenStudio SDK, and Genetic Algorithm (GA) optimisation for performance-driven indoor farm design.

GIFDO builds upon established methods which have been effective in related domains, adapting these techniques for the unique requirements and objectives of vertical farming. The primary aim of this research is to demonstrate the feasibility and effectiveness of this integrated computational approach for automating the design process and identifying high-performance indoor farm solutions tailored to specific environmental contexts. To evaluate the framework's adaptability, optimisation runs were performed using contrasting climates: Oslo (Norway) representing a cold northern environment with significant seasonal daylight variation, and Singapore, representing a hot, humid, equatorial environment with relatively consistent daylight patterns. This allows for assessment of how the optimisation process adapts design solutions (geometry, materials, setpoints) to substantially different external conditions and energy demands.

The remainder of this paper is structured as follows: Section 2 reviews relevant literature in vertical farming, building performance simulation, and generative design. Section 3 details the GIFDO methodology, including candidate representation, genetic operators, simulation setup, and the fitness function. Section 4 presents the results from the Oslo and Singapore optimisation runs. Section 5 evaluates these findings, discusses performance benchmarks, and outlines methodological limitations. Finally, Section 6 concludes the paper, summarising key insights and suggesting directions for future research.

2 Related Work

2.1 Vertical Farming and CEA

Controlled Environment Agriculture, particularly vertical farming, presents promising avenues for enhancing sustainable food production amidst growing pressures on land and water resources. However, realising the full potential of these systems is hindered by significant challenges. Beyond logistical hurdles such as site-specific design adaptation [5], a primary obstacle limiting economic viability and scalability is the substantial operational energy consumption [4]. The intensive energy demands, primarily associated with artificial lighting and precise climate control (HVAC), create a critical need for energy-efficient designs. Effectively addressing this requires robust simulation tools to predict performance and systematic optimisation methods to find designs that minimise energy use while maintaining productive growing conditions, thereby improving the overall feasibility of vertical farming.

2.2 Building Performance Evaluation

Evaluating building designs requires robust simulation tools. Early simulators like TRNSYS [6] used component libraries for dynamic analysis, while the influential DOE-2 program, particularly version 2.1E [7], employed sequential calculations and hourly time steps, becoming an industry benchmark. While foundational, these methods had accuracy limitations for complex interactions. EnergyPlus [8] superseded these tools, offering an integrated simulation approach using a heat balance engine. This allows simultaneous calculation of loads and system responses, potentially at sub hourly steps, improving accuracy. Critically for this research, EnergyPlus includes daylighting simulation capabilities and supports coupling with external tools, making it suitable for automated optimisation.

A significant hurdle when using detailed building performance simulations within optimisation is the computational runtime. Direct simulation can be time consuming, limiting the number of evaluations possible. To address this, some researchers use surrogate models to approximate simulation results. For instance, Magnier et al. [9] trained an Artificial Neural Network (ANN) on data from 450 TRNSYS simulations, each representing a design with 20 input variables. This ANN achieved an average error of just 0.5% for total energy consumption compared to the full TRNSYS simulation. By substituting this fast ANN for direct TRNSYS evaluation within their genetic algorithm, they reduced the estimated runtime for a complete optimisation run from "10 years" down to only 7 minutes. While developing a similar neural network was beyond the scope of this project, which used direct EnergyPlus simulation, this example highlights the potential of surrogate modelling for significantly accelerating future iterations of the GIFDO framework.

2.3 Generative Design in Architecture

The concept of using algorithmic processes for architectural design optimisation has a history tracing back to ideas like John Frazer's "An Evolutionary Architecture" [10], which proposed using computational evolution to generate forms optimised for environmental performance, material efficiency, and specific parameters like lighting and heating. Since then, various computational techniques have been explored within architectural generative design. Rule-based systems, such as shape grammars, have been employed to automatically generate diverse and detailed building elements like facades from a set of primitives [11]. More recently, learning-based approaches have emerged; Chang et al. [12], for example, used a reinforcement learning agent coupled with EnergyPlus to optimise campus-scale building massing for objectives including solar gain and HVAC load, demonstrating the potential for exploring large geometric spaces. However, their focus remained on urban form, omitting detailed envelope design or internal operational setpoints relevant to specialised structures like vertical farms.

The practical application of generative design in architecture for performance optimisation is increasingly validated, as evidenced by this 2024 review [13] from Suphavarophas et al. The paper confirms that architects increasingly use generative design, supported by growing research, to explore design possibilities and achieve

demonstrable efficiency gains compared with conventional methods. Significantly, their review identifies Evolutionary Algorithms (EAs), a broad class of optimisation heuristics inspired by natural selection, as the most frequently utilised class of technique within architectural GD for energy efficiency, labelling them as ”standard” in the field. These EAs are frequently employed alongside parametric modelling to navigate complex multi-objective problems. Within this dominant category of EAs, genetic algorithms represent a particularly widespread and influential approach utilised in architectural performance optimisation. The established success and suitability of GAs for such problems provide strong justification for their application in the context of optimising vertical farm designs.

Genetic algorithms themselves are inspired by natural evolution [14], acting as heuristic search methods that iteratively refine a population of potential solutions through processes like mutation and crossover over successive generations to progress towards an optimum. Each individual solution within the population is typically encoded as a ’chromosome’ upon which these genetic operators act. The inherent flexibility of GAs and their ”ability to work with many parameters simultaneously in a search space of complicated structure” [15] make them effective methods for exploring complex solution spaces, which common in generative design.

Their application in performance-driven architecture is particularly relevant, often involving coupling with building performance simulation engines to evaluate design fitness based on energy or environmental criteria. An early example by Wang et al. [16] successfully used a GA to identify Pareto optimal building designs that reduced life cycle environmental impacts by 65%. Wang notes however, that there is room for further optimisation if the scope was expanded to cover passive solar design strategies, as well as more complex building shapes.

Caldas’s GENE_ARCH platform [17] provides another illustration, combining a GA with the DOE-2.1E simulation engine. In this framework, chromosomes encoded extensive building details, including whole-building geometry, room adjacencies, materials, and shading devices. Fitness evaluation, driven by thousands of DOE-2 simulations, encompassed multiple objectives such as heating, cooling, and lighting energy. A case study applying GENE_ARCH explored over 2000 variants and achieved a life cycle energy reduction of approximately 13% compared to the architect’s baseline design [17]. This work highlighted the GA’s ability to navigate complex architectural decision spaces, a significant challenge for typical rule-based or gradient optimisation methods. These examples confirm the utility of GAs for exploring complex forms and optimising performance based on simulation results, supporting their use in specialised domains, such as indoor farm design.

Within Controlled Environment Agriculture, generative techniques have also been applied, often targeting internal system optimisation. For instance, Jia et al. [18] used parametric modelling and a genetic algorithm to optimise the layout and spacing of internal cultivation frames, aiming to reduce supplemental lighting energy by improving natural light distribution to lower layers. Such studies, while demonstrating valuable energy savings through internal component optimisation, typically do not address potential performance gains from variation of the primary building envelope itself.

2.4 Research Gap

Optimisation techniques, coupling genetic algorithms and building performance simulation, have also been applied effectively in other architectural domains. For instance, Schwartz et al. [19] developed an automated process for optimising residential building designs, targeting reduced Life Cycle Carbon Footprint (LCCF) and Life Cycle Costs (LCC). In their work, a set of building parameters including materials, window position and size, and floor-plan layouts were optimised using the NSGA-II genetic algorithm [20] coupled with EnergyPlus simulations. The research successfully generated a Pareto front of solutions demonstrating significant potential reductions in both LCCF and LCC compared to existing literature. However, a notable limitation of this approach stemmed from the restricted exploration of building geometry. The layouts were limited to combinations of rectangular room blocks generated by their space-allocation engine, yielding only 32 floor-plan variants for the GA to explore. This constraint highlights the challenge of incorporating more extensive geometric exploration, including topological variation, within such optimisation frameworks, a necessary step for potentially uncovering more globally optimal or architecturally diverse solutions.

Although related architectural optimisation faces limitations in geometric flexibility, research specifically targeting CEA and vertical farms frequently restricts its scope in other ways. Many studies optimise internal parameters like lighting conditions [21], cultivation frame spacing [18], or operational schedules within existing or simple building structures, rather than generatively designing the building envelope itself. Consequently, almost no studies tackle the full problem: optimising free-form geometry, envelope materials, and operating set-points in one BPS-driven run that weighs crop revenue against lighting, heating, and cooling costs.

Therefore, the specific research gap addressed by this work is the development and evaluation of an integrated computational framework that simultaneously:

1) Employs a flexible, mesh-based geometric representation capable of evolving complex, non-cuboid building forms.
2) Couples this representation with a GA for evolutionary search.
3) Utilises building simulation software (EnergyPlus) for automated evaluation of energy performance and daylighting.
4) Optimises geometry, envelope materials, and operational setpoints concurrently.
5) Targets objectives directly relevant to vertical farm viability, specifically balancing potential profit against operational energy costs.

The GIFDO system, detailed in the following sections, was created to directly address this gap, providing a tool for exploring performance driven design automation specifically for Controlled Environment Agriculture facilities.

3 SOLUTION

3.1 Architectural Overview

The Generative Indoor Farm Design Optimisation (GIFDO) framework automates the performance-driven design of indoor vertical farms by integrating procedural geometry generation with evolutionary optimisation algorithms. This framework is implemented in Python (version 3.11.9). For building performance evaluation, it employs the OpenStudio SDK (version 3.8.0) to interface with and manage simulations performed by the EnergyPlus [8] engine (version 24.1.0).

At its core, the framework employs a genetic algorithm that iteratively refines a population of candidate indoor farm designs. This iterative process evaluates candidates using EnergyPlus simulations to determine energy performance and the SIMPLE crop model to estimate potential yield, aiming to balance these conflicting objectives. To explore the system's performance under different climatic conditions, two optimisation runs were conducted using weather data for Oslo (Norway) and Singapore. The characteristics and performance of the resulting optimised designs from these runs are presented and analysed in Section 4.

3.2 Candidate Representation

Within the genetic algorithm, each potential solution is encapsulated within a Candidate object, representing a complete blueprint for an individual indoor farm design. Each candidate design must contain all data needed to create a representation of itself in OpenStudio. This means fully describing the geometry of all external walls, roofs, and floor surfaces, as well as the materials used in their construction. The candidate also stores the operational setpoints governing its internal environment.

3.2.1 Geometry

The geometry of the building structure is represented as a mesh, managed internally using ordered arrays for vertices and faces (handled via the Trimesh Python library [22]). Each face is defined by three vertices.

Initialisation begins with a standard cuboid mesh structure, centred at the origin, whose dimensions (width, depth, height) are sampled from uniform random distributions within predefined size bounds. A series of vertex perturbation attempts are subsequently applied to introduce variation, transforming the cuboid into a more complex polyhedron. The suitability of this representation lies in its ability to span a diverse solution space, while validity can be checked computationally with standard operations from a mesh library.

A key aspect of this representation is the consistent indexing of mesh elements. This fixed indexing scheme, maintained across candidates sharing the same subdivision level, ensures that the i-th element (vertex or face) in one mesh corresponds topologically to the i-th element in another. This principle is fundamental, enabling element-wise genetic operations like vertex interpolation during crossover (detailed in Section 3.3.2) and allowing reliable identification of feature indices, such as those defining the floor surface, for applying specific constraints.

Rather than attempting to construct complex constrained geometries algorithmically from scratch, the geometry is generated through an iterative perturbation process, with each modification checked against strict validity criteria. This approach ensures that mesh integrity is preserved throughout both initialisation and genetic operations. Any modification that results in an invalid state is immediately discarded, guaranteeing that all candidates consistently represent feasible building geometries. Specifically, the following five constraints are enforced across the entire mesh:

- There can be no self intersections. These are identified using the PyMeshLab library.
- The height of the mesh must not exceed the minimum or maximum boundaries ($h_{\min}$and$h_{\max}$).
- The total footprint of the mesh, measured by the area of the projection onto the horizontal plane, must not exceed the minimum or maximum boundaries ($A_{\min}$and$A_{\max}$)
- No face in the mesh can have an area which is less than 2% of the mean area of all mesh faces. This is done to prevent numerical instability in the EnergyPlus software. Excessively small surfaces are assigned a correspondingly small thermal mass, which can become superheated by a typical amount of incident sunlight. EnergyPlus is not designed to deal with surfaces this small.
- The mesh must be watertight, forming a closed volume.

A vital aspect of design validity, is its feasibility to be used as a growing space. First and foremost the floor must be level. Upon initialisation of the primitive cube mesh, the vertices and faces comprising the bottom face are identified, and limited to movement in the horizontal plane. The following three constraints apply only to floor vertices:

- There can be no overlapping floor faces. This constraint is separate from the "no self intersections" check in the mesh as a whole, as the PyMeshLab library does not identify coplanar polygons as intersecting. The Shapely library is better suited to identifying overlapping coplanar polygons, hence its use.
- A non-floor vertex may not move to a point below the height of the floor plane in the z dimension.
- The sum of the areas of the floor faces must be greater than 5% of the total surface area of the mesh. Although this research does not explicitly model the feasibility of constructing each design, this rule has the advantage of filtering out obviously infeasible designs, with large overhanging volumes supported by a tiny floor contact area. All thermal transfer between the interior building space and the ground passes through the floor surface, so avoiding excessively small floor areas helps preserve intended behaviour in the EnergyPlus ground heat calculations.

3.2.2 Materials

Materials are handled in a way that is tightly coupled with the mesh representation of building geometry. Each candidate object maintains a list of Material instances, where

TABLE 1: Thermal Properties of Available Opaque Construction Materials

Material Name	Thickness	Conductivity	Density	Specific Heat	Absorptance (α)			Roughness
	(m)	($W\,m^{-1}\,K^{-1}$)	($kg\,m^{-3}$)	($J\,kg^{-1}\,K^{-1}$)	Thermal	Solar	Visible	
Wall Materials								
Insulation Board (50mm)	0.0508	0.03	43	1210	0.90	0.70	0.70	MediumRough
Lightweight Concrete (200mm)	0.2032	0.26	464	880	0.90	0.70	0.70	MediumRough
Heavyweight Concrete (100mm)	0.1016	1.95	2240	900	0.90	0.70	0.70	MediumRough
Floor Materials								
Heavyweight Concrete (150mm)	0.1524	1.95	2240	900	0.90	0.70	0.70	MediumRough
Heavyweight Concrete (300mm)	0.3048	1.95	2240	900	0.90	0.70	0.70	MediumRough

TABLE 2: Optical and Thermal Properties of Available Glazing Construction Materials (Normal Incidence)

Material Name	Thickness	Conductivity	Solar Properties			Visible Properties			Infrared Properties		
	(mm)	($W\,m^{-1}\,K^{-1}$)	T	R_{front}	R_{back}	T	R_{front}	R_{back}	T	E_{front}	E_{back}
Low Iron (3mm)	3.0	0.9	0.899	0.079	0.079	0.913	0.082	0.082	0.0	0.84	0.84
LoE Clear (6mm)	6.0	0.9	0.430	0.300	0.420	0.770	0.070	0.060	0.0	0.84	0.03
Clear Float (6mm)	6.0	0.9	0.775	0.071	0.071	0.881	0.080	0.080	0.0	0.84	0.84

T: Transmittance, R: Reflectance, E: Emissivity.

each entry corresponds directly to a face in the mesh. Each material is an instance of either OpaqueMaterial or GlazingMaterial, which are subclasses of the primary Material class. Materials are categorized into two distinct sets for the assignment process: a 'floor' set, containing only designated opaque floor materials, and a 'wall' set, which includes all glazing materials plus the remaining opaque materials suitable for walls and other non-floor surfaces. During assignment, materials for floor faces are sampled exclusively from the 'floor' set, while materials for all other faces are sampled from the 'wall' set. This allows for the inclusion of materials, such as 300mm heavy concrete, which are strong candidates for floor construction, but would be infeasible to build vertically with. It also allows us to exclude glazing materials from the floor, where they would not contribute any daylighting to the internal volume. The candidate class includes validation checks to ensure these material assignments adhere strictly to the defined wall and floor sets.

The properties for all window materials used in the simulation are taken from the WindowScreenMaterials.idf, which is provided in the EnergyPlus install folder. The opaque materials and their properties are from the ASHRAE 2005 Handbook of Fundamentals [23]. The chosen glazing materials present the optimiser with three distinct options, balancing maximum daylight penetration against reduced thermal transmission. Similarly, the Opaque materials offer an extremely low conductivity option in the insulation board material, and two types of concrete, which offer a larger thermal mass for better temperature stabilisation. Additional material types can be added, however these existing materials are sufficient to explore the capability of our design algorithm to manage tradeoffs, and tailor it's material choices to the environment. Tables 1 and 2 detail the full thermal and optical properties of all the materials used.

3.2.3 Scalar Properties

Along with geometry and materials, each candidate includes two numerical setpoints: a target temperature T_{set} (°C) and a target mean daily Photosynthetically Active Radiation (PAR) Energy I_{set} ($MJ\,m^{-2}\,d^{-1}$) over the growing area. These values are stored as floats, along with a tuple representing the maximum and minimum values that each setpoint can take. These setpoints are used as inputs in our crop model, as well as to calculate the expected energy demands for lighting and heating. For the purposes of this simulation, we assume a constant temperature and PAR setpoint throughout the entire year.

Finally, each candidate maintains a subdivision level property, initialised to 0. This integer represents the level of geometric detail applied to the candidate's mesh structure. A level of 0 corresponds to the base 12-face cuboid mesh, while each subsequent increment signifies a uniform subdivision where the number of mesh faces and vertices increases, allowing for potentially more complex and refined geometries. The specific mechanism and timing for increasing this level are part of the genetic algorithm's progression strategy, detailed in Section 3.3.4.

3.3 Genetic Operators

The evolutionary search algorithm employed in this work utilises three primary genetic operators. Firstly, random initialisation generates the starting population of candidates, providing broad, initial coverage of the solution space. Secondly, mutation introduces random alterations to existing candidates, facilitating local exploration and maintaining population diversity. Finally, crossover combines characteristics from two selected parent candidates to produce offspring, thereby exploiting promising regions of the search space identified by successful parents.

3.3.1 Random Initialisation

Random initialisation of a candidate begins with its geometry. A cuboid mesh is created, the dimensions of which (width, depth, height) are sampled from a uniform random distribution within the permitted size bounds from the configuration file. The mesh is centred at the origin of the coordinate space. Subsequently, a series of N_v vertex perturbation attempts are sequentially applied to introduce variation. Each perturbation attempt proceeds as follows:

A vertex index is first selected uniformly at random from the mesh vertices. The calculation of the displacement vector is based upon the predefined minimum and maximum coordinate bounds of the design space, $\vec{p}_{\text{min}}$ and $\vec{p}_{\text{max}}$, and a maximum displacement factor, f_{max} (where $0 \leq f_{\text{max}} \leq 1$).

The dimensional range is calculated:

$$\vec{R} = \vec{p}_{\text{max}} - \vec{p}_{\text{min}} \tag{1}$$

A random magnitude vector, $\vec{f} = (f_x, f_y, f_z)$, is generated where each component f_k is sampled independently from a uniform distribution:

$$f_k \sim \mathcal{U}(-f_{\text{max}}, f_{\text{max}}) \quad \text{for } k \in \{x, y, z\} \tag{2}$$

The primary displacement vector, $\vec{d}$, is then the element-wise product of the magnitude vector and the range vector:

$$\vec{d} = \vec{f} \odot \vec{R} \tag{3}$$

For vertices belonging to the floor surface, only the horizontal components of this vector are used, in order to maintain a flat base. The applied displacement is $\vec{d}_{\text{floor}}$, defined as:

$$\vec{d}_{\text{floor}} = (d_x, d_y, 0) \tag{4}$$

For all other vertices, the full vector $\vec{d}$ from Equation (3) is used. The resulting displacement vector is applied to the vertex in question. The validity of the resulting mesh is then checked against the set of constraints detailed in Section 3.2.1. If the mesh is valid, the perturbation is accepted, otherwise it is discarded.

This approach allows for tuning of the magnitude of each vertex perturbation, by changing f_{max}. Scaling the displacement by the dimensional range $\vec{R}$ allows this method to generalise to arbitrary bounding box limits. The number of perturbations N_v must be selected carefully, so that significant variation is introduced without excessive distortion. As the number of perturbations increases, the expected volume of the resulting mesh decreases. A large internal volume is a desirable trait for indoor farming, so too many perturbations could skew the global search away from desirable solutions. Figure 1 shows an example of a random candidate mesh generated from this method, with $N_v = 40$ and $f_{\text{max}} = 0.1$ at subdivision level 1.

The process of random material initialisation begins by creating a material array, indexed according to the faces of the candidate mesh. This array is populated by iterating through each face; a material is randomly selected from the designated set of 'floor' materials if the face is a floor, or from the set of 'wall' materials if it is not. The selected material is then stored in the array element corresponding to that face. The indices of each face are constant under all genetic operations, so implicit linking of materials to faces using their indices is sufficient.

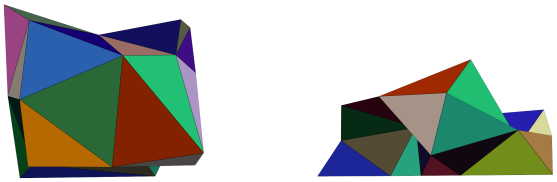


Fig. 1: Mesh Generated by the *Random Initialisation* Operator. (Left) Top View. (Right) Side View.

Temperature and PAR irradiation setpoints are simply uniformly sampled from their respective permitted ranges, which are set at the start of a run. Subdivision level is passed to the candidate constructor, influencing random initialisation, but not taking a random value itself.

3.3.2 Crossover

The crossover operator combines genetic information from two selected parent candidates, denoted P_1 and P_2, to generate a single offspring candidate. This process involves separate blending mechanisms for geometry, materials, and environmental setpoints.

Initially, two blending factors, α and β, are sampled independently from a uniform distribution within the interval [0.2, 0.8]. The factor α controls the interpolation between the parent geometries, while β governs the selection of materials from each parent. Using distinct, randomly sampled factors for each crossover rather than fixed ones is an established practice [24] that introduces an element of exploration into this typically exploitative operator. Given that elite individuals may survive unchanged and be selected for crossover multiple times (potentially with the same partner), this randomness ensures that repeated pairings between the same two parents can still generate diverse offspring, rather than identical copies.

Geometric crossover assumes that P_1 and P_2 have the same mesh topologies, which holds as long as their subdivision level is equal. For each vertex $\vec{v}_i$ in the child mesh, its value is calculated from the corresponding vertices in each parent mesh, $\vec{v}_{1,i}$ and $\vec{v}_{2,i}$, as

$$\vec{v}_i = \alpha \cdot \vec{v}_{1,i} + (1 - \alpha) \cdot \vec{v}_{2,i} \tag{5}$$

All initial meshes are created with the same Trimesh method, which returns a cuboid mesh with identical vertex indexing and face ordering each time. These properties are copied directly from both parents, ensuring the child mesh remains compatible for crossover with other members in the population.

Following the geometric interpolation, the resulting child mesh geometry is subjected to the standard validity checks. If the geometry is found to be invalid, the geometric crossover process is re-attempted by resampling the α parameter, and recalculating the child's vertex coordinates using Equation (5). After a maximum number of failed crossover attempts N_c between the parents P_1 and P_2, the pairing is discarded and a new pair of candidates is selected from the population for crossover.

Successful production of a child mesh is followed by material assignment. For each face s_i in the child mesh,

a corresponding material m_i is assigned probabilistically using β. With probability β the material $m_{1,i}$ is inherited from parent P_1, and with probability $1-\beta$ the material $m_{2,i}$ is inherited from parent P_2. Validity checks are performed to ensure the material assignment was successful, if a check fails then the crossover pairing is discarded and a new pair of parents is drawn from the population.

Bounding the blending factors α and β to the interval [0.2,0.8] prevents candidates effectively cloning themselves, which is a known factor in premature convergence [25]. If α and β were allowed to approach 0 or 1, highly fit parents selected repeatedly for crossover could produce offspring nearly identical to themselves. This would rapidly reduce genetic diversity within the population, increasing the likelihood of the search stagnating in a local optimum.

Finally, the scalar environmental setpoints for temperature and PAR intensity are determined using a similar linear interpolation. A third blending factor, γ, is sampled independently from a uniform distribution over the full interval [0, 1]. Each setpoint in the offspring is then calculated as the weighted average of the corresponding parent setpoints, using γ and $1-\gamma$ as the weights (analogous to the structure of Equation (5)). The full range $[0, 1]$ is permitted for γ, as the simple one-dimensional nature of these setpoints makes them less susceptible to the diversity loss caused by cloning compared to the high-dimensional geometry and material representations.

The subdivision level of each parent candidate P_1 and P_2 must be identical for a successful crossover. As the geometry crossover operation maintains the topological structure from both parents, the resulting child candidate will always have the same subdivision level.

3.3.3 Mutation

Mutation introduces genetic diversity into the population by randomly altering individual candidates after the crossover stage. It operates independently on the candidate's geometry and its operational setpoints, controlled by distinct probabilities.

A child candidate C is subject to geometry mutation with probability p_g (refer to Table 4 for specific value). If triggered, the following sub-mutations are applied, sequentially, each with its own probability.

- Face mutation, which is always applied if geometry mutation is triggered.
- Material assignment mutation with probability $p_{g,m} = 0.5$.
- Scale mutation with probability $p_{g,s} = 0.1$.

The face mutation operator begins with first selecting a face at random from the mesh. Between 1 and 3 of the constituent vertices are then selected, and a perturbation vector is applied (as in Equations (1)–(4)) to shift their coordinates. If a mutation applied to a selected face results in a mesh violating the validity constraints outlined in Section 3.2.1, the geometric changes from that attempt are reverted. Another face is then selected at random for a subsequent mutation attempt. This cycle of selection, mutation, and validation continues until a target number N_r of valid face mutations have been successfully applied within the overall mutation event, where N_r is a random integer chosen from 1 to 3 inclusive.

Material mutation selects up to three faces at random from the mesh. For each face s_i in the mesh of C, the corresponding material m_i is flipped to one of the other materials in the respective material set, floor or wall, depending on the type of face s_i.

With probability $p_{g,m}$, material assignments are altered by randomly selecting a face s_i and replacing its material m_i with a different, randomly chosen material from the appropriate set (wall or floor, Section 3.2.2). This face selection and material replacement is performed N_m times per mutation event, where N_m is a random integer chosen from 1 to 3 inclusive.

Scale mutation applies a uniform scaling factor drawn from the range [0.8, 1.25]. This scaling is performed relative to the origin in the mesh coordinate space. Scaling mutations which result in a mesh which exceeds the minimum or maximum size constraints are reverted. While repeated vertex perturbations can cumulatively alter the overall size of the geometry, the dedicated uniform scaling mutation provides a more direct mechanism for exploring different building volumes. Since face mutation modifies a small number of vertices, achieving significant, proportional changes in overall size might require many mutation steps. Including the uniform scaling operator, albeit with a lower probability $p_{g,s}$, allows the algorithm to more efficiently test different overall scales for the design. This complements the finer-grained shape adjustments provided by vertex perturbation which explore variations in form at a given scale.

Independent of geometry mutation, the candidate's setpoints are mutated with probability $p_s = 0.2$. If triggered, the temperature setpoint T_{set} and light intensity setpoint I_{set} are updated as follows:

$$T'_{\text{set}} = T_{\text{set}} + \mathcal{N}(0, \sigma_T) \tag{6}$$

$$I'_{\text{set}} = I_{\text{set}} + \mathcal{N}(0, \sigma_I) \tag{7}$$

The mutated setpoints T'_{set} and I'_{set} are then clamped to their predefined ranges (Table 4). The standard deviations σ_T and σ_I are subject to decay over time, to promote wide exploration initially, and then make smaller adjustments later into a run. The exact progression strategy is detailed in Section 3.7.

3.3.4 Subdivision

Mesh subdivision increases the geometric resolution of candidates by splitting each triangular face into four smaller triangles. This refinement adds vertices and increases the face and vertex count by a factor of four each time the operation is applied. This allows for the representation and subsequent refinement of finer geometric details.

Critically, subdivision is applied synchronously across all members of the population at a given generation. This synchronicity is mandated by the crossover operator, which requires parent candidates to possess identical mesh topologies to produce valid offspring. Each time a subdivision occurs the subdivision level$s$, starting at 0, is incremented by 1. This property is tracked on a population-wide level.

The subdivision level (s) influences the candidate generation process in two additional ways. Firstly, the number of

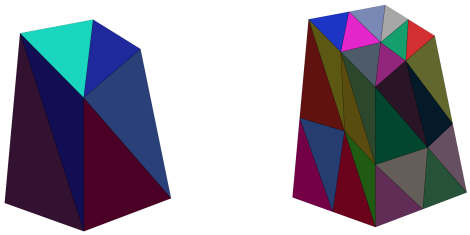


Fig. 2: (Left) Subdivision Level 0. (Right) Subdivision Level 1.

random vertex perturbations applied during the initialisation of a candidate (Section 3.3.1), N_v, scales exponentially with the s to maintain sufficient geometric variation relative to the increased mesh density:

$$N_{v,s} = a \cdot 4^s \tag{8}$$

Where a is the base vertex perturbation count at subdivision level 0.

Secondly, the magnitude of vertex perturbations applied during random initialisation (Section 3.3.1) is inversely scaled with the subdivision level s. The maximum perturbation factor, $f_{\max,s}$, used to determine displacement (Equation (2)) is adjusted according to:

$$f_{\max,s} = \frac{f_{\max,0}}{s+1} \tag{9}$$

Where $f_{\max,0}$ is the base factor at subdivision level 0. Reducing the perturbation magnitude in this way allows for finer geometric adjustments on higher-resolution meshes, facilitating the refinement of promising shapes identified during broader exploration at lower subdivision levels.

3.4 Measuring Yield: SIMPLE Crop Model

TABLE 3: Tomato Parameters Used in the SIMPLE Crop Model (Sourced From [26])

Description	Value
Cumulative temperature to maturity	2800 °C d
Potential harvest index	0.68
Deg-days to 50% leaf cover	520 °C d
Deg-days to 50% senescence	400 °C d
Base temperature for growth	6 °C
Optimal growth temperature	26 °C
Radiation-use efficiency	1.00 g MJ^{-1} m^{-2}
Max daily senescence cut (heat)	100 °C d
Max daily senescence cut (drought)	5 °C d
Heat-stress onset temperature	32 °C
Heat-stress extreme temperature	45 °C
RUE increase per ppm CO_2	0.07 per ppm
Drought sensitivity (ARID index)	2.5
Growing cycle length	120 d

The available growing area inside a candidate building is calculated using the usable growing volume proportion ϕ_{grow} and the vertical growing rack spacing s_{rack} (Table 4). The building mesh is divided into a series of horizontal slices, starting from the floor and then at every s_{rack} meter vertical interval. The area of these slices are then summed over and multiplied by ϕ_{grow}, and the result is taken as the usable growing area A_{grow}.

Crop yield was estimated using a Python implementation [27] of the SIMPLE crop model [26]. Tomato was selected as the representative crop for this study, reflecting its prevalence in vertical farming systems [28]. As simulating water availability was outside the scope of this research, irrigation constraints within the model were disabled. The inputs provided to the SIMPLE model were the PAR irradiation setpoint, the temperature setpoint, and the specific tomato crop parameters detailed in Table 3.

From these inputs, the model returns the expected total annual yield per square meter of growing area, Y_{tomato} (kg m^{-2}). The value per kilogram of tomato, V_{tomato}, is assigned a value of 4 USD kg^{-1} based on current UK market pricing. The total estimated annual value of the crop yield, $V_{\text{total,tomato}}$ (USD), is then calculated as:

$$V_{\text{total,tomato}} = Y_{\text{tomato}} \cdot A_{\text{grow}} \cdot V_{\text{tomato}} \tag{10}$$

3.5 Simulation Tools: OpenStudio and EnergyPlus

3.5.1 Overview

For a complete assessment of the suitability of a candidate, the lighting and thermal properties must be evaluated. This process is handled by the EnergyPlus application. EnergyPlus [8] is a full building simulation engine, developed by the US Department of Energy. It uses a suite of physics-based modelling techniques to calculate thermal loads, ventilation requirements, and daylight interaction, as well as a host of other building lifecycle measures. It is widely utilised for algorithmic building design optimisation because it offers the combined benefits of being open-source, an industry standard, and possessing extensive simulation capabilities. OpenStudio is an open-source API built on top of EnergyPlus, and provides a wide array of language bindings to interact with the EnergyPlus engine.

Interaction between the candidates in the genetic algorithm and EnergyPlus happens in seven main stages:

- Conversion of Material instances into OpenStudio constructions.
- Surface-by-surface recreation of the candidate building mesh within OpenStudio, using the defined geometry and constructions.
- Addition of temperature control objects (thermostat and HVAC).
- Addition of lighting objects (illuminance maps).
- Setting the weather file.
- Translation from OpenStudio Model object into EnergyPlus idf file, and passing to the EnergyPlus engine for simulation.
- Parsing of EnergyPlus outputs, and collection of data back into the main Python application.

EnergyPlus' lighting and temperature control simulations are done on the CPU. This simulation step dominates the total runtime per generation in our application, with lighting calculations taking 12 seconds per candidate at subdivision level 0, and 30 seconds per candidate at subdivision

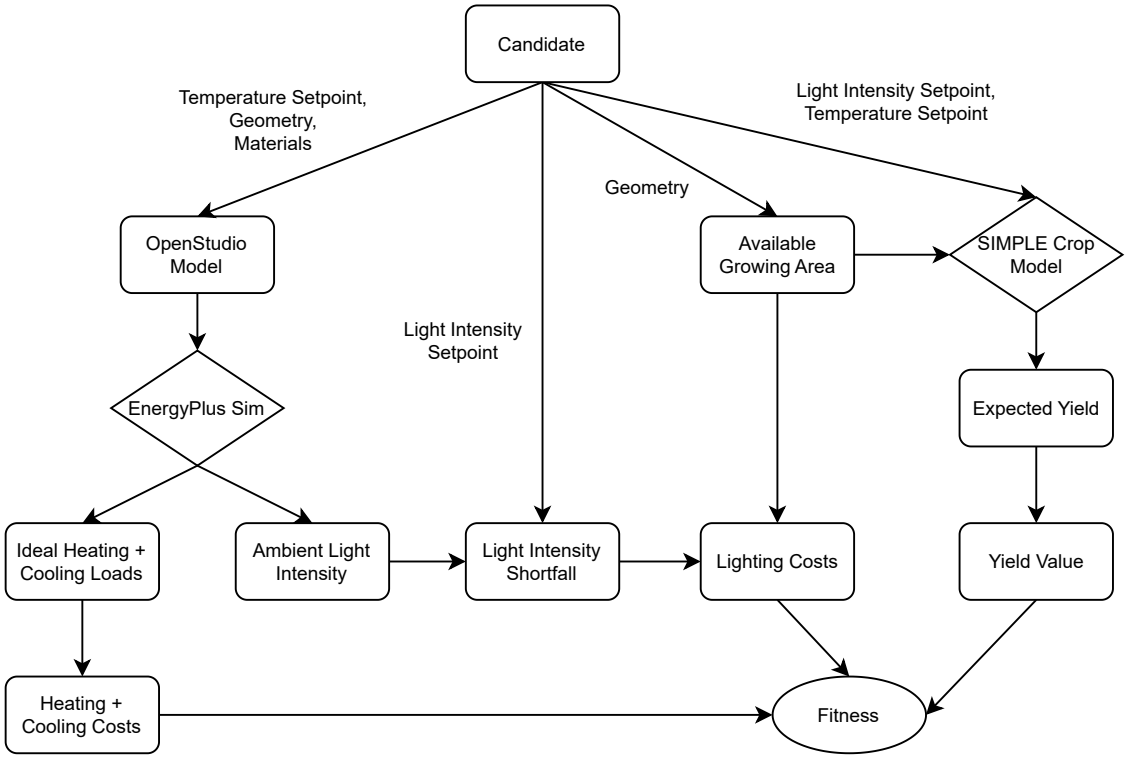


Fig. 3: Candidate Fitness Evaluation Pipeline: the Parallelised Portion of GIFDO

level 1. GIFDO parallelises this process, distributing candidates to a group of threads. Each thread is allocated a unique temporary working directory in the local file system. Within this directory a single thread is responsible for constructing the OpenStudio model, running it through EnergyPlus, and parsing the results files, which EnergyPlus outputs to this directory, updating the fitness of the candidate using the formula described in Section 3.6. Figure 3 shows the parallelised candidate evaluation pipeline.

3.5.2 Heating and Cooling Demands

Annual heating and cooling demands were calculated using EnergyPlus's 'ZoneHVAC:IdealLoadsAirSystem'. This object acts as a 100% efficient, and infinitely scalable system which calculates the exact heating and cooling load required to perfectly meet the temperature setpoint at each time step throughout the year. EnergyPlus accounts for solar gains, as well as building envelope heat transfer: both through external surfaces to the outside air, and through the floor into the ground.

Whilst this object is useful for getting raw ideal energy loads ($E_{\text{ideal,heat}}$ and $E_{\text{ideal,cool}}$), it is a significantly inaccurate representative of the practical real-world energy requirements of a building. EnergyPlus offers several objects, which can be added to a thermal zone to model actual boilers, chillers, air conditioning systems etc. These objects model realistic capacities, efficiency ranges, and optimal operating temperatures. However, the specific method of temperature control varies massively for different indoor farms, and capacity requirements based on ambient temperatures and indoor space dimensions are near impossible to generalise. The Coefficient Of Performance (COP_{cool}) for cooling, and heating efficiency parameter (η_{heat}) serve as catch-all parameters to encapsulate expected average efficiency of a heating and cooling system over a year period. The values for these parameters were selected based off minimum performance standards set by the Singapore National Environment Agency [29].

Total annual heating costs $C_{\text{total,heat}}$ (USD) are therefore calculated using:

$$C_{\text{total,heat}} = \left(\frac{E_{\text{ideal,heat}}}{\eta_{\text{heat}} \times 3.6 \times 10^6} \right) \cdot C_{\text{heat}} \tag{11}$$

And total annual cooling costs $C_{\text{total,cool}}$ (USD) using:

$$C_{\text{total,cool}} = \left(\frac{E_{\text{ideal,cool}}}{\text{COP}_{\text{cool}} \times 3.6 \times 10^6} \right) \cdot C_{\text{cool}} \tag{12}$$

The factor 3.6×10^6 converts the energy input from Joules to kilowatt-hours. C_{heat} and C_{elec} are the prices in USD per kilowatt-hour for heating and electricity (used for cooling) respectively. See Table 4 for values used in the simulations in Section 4.

3.5.3 Natural Light

There are two main options for building daylight simulation in OpenStudio and EnergyPlus. OpenStudio offers integration with Radiance [30], a physics-based ray-tracing engine used for accurate daylighting analysis. Radiance can be used via an OpenStudio measure. Using ray-tracing allows Radiance to yield accurate illuminance values, that take into account all building geometry, windows, interior surfaces and weather conditions. However, the primary drawback of Radiance is its significant computational runtime; Li et al. [31] found that, for a basic box shaped room with windows,

and using the ”accurate” simulation preset, Radiance takes approximately 9 minutes on a modern CPU (Intel i7-8650U). This would be prohibitive for the thousands of simulations required by GIFDO's optimisation process, which involves more complex building geometries. Furthermore, Radiance employs stochastic Monte Carlo ray-tracing techniques, which can introduce run-to-run variance in results.

EnergyPlus' internal daylighting calculations use split-flux [32] formulas to determine illuminance with a reduced accuracy. The significantly shorter runtime however, along with fully deterministic results, justifies its use in this application. With access to more compute power, further development could yield a more accurate fitness function by replacing the integrated EnergyPlus daylighting calculations with a Radiance measure.

Split-flux lighting calculations are triggered by addition of an illuminance map object to the model. An illuminance map is a grid of points distributed at regular intervals across a horizontal plane, and is used to determine average hourly illuminance (lx) at a given height. Two horizontal illuminance maps are defined: one near the lowest anticipated crop level and another at the building's geometric centroid height.

For each map and each daylight hour, an hourly average illuminance is calculated. This average considers only those sensor points which lie within the building's volume, a check performed using intersections between the map points and the projected mesh geometry (Calculated using the Shapely Python package [33]). This hourly average illuminance is then multiplied by the horizontal cross-sectional area of the building volume at the map's height. Summing these hourly area-weighted values over 24 hours yields an estimate for the total daily luminous energy (in lumen $\cdot$ h) received by each representative slice. Let these totals be E_{floor} and E_{centroid}. The mean of (E_{floor} and E_{centroid}) is taken as the representative mean daily total luminous energy E_{full} (lm $\cdot$ h) from natural light available within the volume.

E_{full} is converted to the total daily PAR energy, E_{PAR} (J), using standard conversion factors. The key factors are the conversion from luminous flux to PAR photon flux ($k_p \approx 0.0185\,\mu\text{mol}\,\text{s}^{-1}\,\text{lm}^{-1}$ for daylight [34]) and the average energy content per PAR photon ($e_{\text{photon}} \approx 0.217\,\text{J}\,\mu\text{mol}^{-1}$ [35]). The conversion is performed as follows:

$$E_{\text{PAR}} = E_{\text{full}} \cdot 3600 \cdot k_p \cdot e_{\text{photon}} \tag{13}$$

This total daily PAR energy is then expressed as daily PAR irradiation H_{PAR} (MJ m^{-2}) by dividing by the usable growing area, A_{grow} (m^2), and scaling from Joules to Megajoules:

$$H_{\text{PAR}} = \frac{E_{\text{PAR}}}{A_{\text{grow}} \times 1 \times 10^6} \tag{14}$$

Now that the units match, the difference between H_{PAR} and the PAR irradiation setpoint is the supplementary lighting required from artificial lights to meet the demand. This value is multiplied back over the growing area A_{grow}, and then by 365 days for an annual run for a total annual energy load from lighting $E_{\text{total,light}}$ (MJ).

The photosynthetic efficacy of the supplementary lighting system is denoted by η_{LED} (μmol J^{-1}). The total annual cost of supplementary lighting, $C_{\text{total,light}}$ (USD), is then calculated from the total annual PAR energy shortfall, $E_{\text{total,light}}$ (MJ), as follows:

$$C_{\text{total,light}} = \frac{E_{\text{total,light}}}{e_{\text{photon}} \times \eta_{\text{LED}} \times 3.6} \times C_{\text{elec}} \tag{15}$$

Where the factor 3.6 converts the required electrical energy from Megajoules to kilowatt-hours.

3.6 Fitness function

$$\text{Fitness}_1 = V_{\text{total,tomato}} - C_{\text{total,heat}} - C_{\text{total,cool}} - C_{\text{total,light}} \tag{16}$$

Initially, fitness was calculated purely by total profit, subtracting total energy costs from total crop value using Equation 16. This function should reward designs which produce high yields, and at the same time incentivise optimisation to minimise energy usage. However, in practice, once a setpoint configuration was found to be profitable, the designs would grow endlessly until they filled the entire bounding volume. This occurred because the fitness function lacked any penalty for size; if a given unit of farm volume yielded a net profit, simply adding more identical units (increasing the overall size) would linearly increase the total profit, providing no incentive to stop growing.

$$\text{Fitness}_2 = \frac{V_{\text{total,tomato}}}{C_{\text{total,light}} + C_{\text{total,heat}} + C_{\text{total,cool}}} \tag{17}$$

Shifting to the efficiency measure shown in Equation 17 yielded similar issues, providing an avenue of exploitation for candidates. Designs which turned off all their lighting and heating could shrink the denominator to near zero, achieving enormous efficiency scores.

The purpose of GIFDO is to inform design of real farming operations, and telling operators to ”build as big as you can” is not useful or actionable in any real sense. Similarly, instructing operators to turn all lights and heating off to get the most energy efficient crop yields is not sensible.

A better approach was to explore the most efficient design for a given yield, allowing the search algorithm to explore the tradeoff between yield and energy cost. Therefore, a different approach was explored, defining a yield value target y. For yields up to this target ($V_{\text{total,tomato}} \leq y$), the fitness was calculated using the direct profit-based method outlined in Equation (16). However, for any yield exceeding the target ($V_{\text{total,tomato}} > y$), the base value contribution was capped at y, and only the square root of the *excess* yield ($\sqrt{V_{\text{total,tomato}} - y}$) was added before subtracting the total costs. This applied a diminishing return specifically to yield generated beyond the target y.

This proved too harsh a cutoff, as any design returning less than y yield value was instantly uncompetitive, and likewise any design significantly over the limit was getting almost no additional reward for the extra yield. It also resulted in significant fitness clustering around y, as yield value was much larger than the cost penalties in most cases.

$$\text{Fitness}_4 = k_{\text{value}}\sqrt{V_{\text{total,tomato}}} - (C_{\text{total,light}} + C_{\text{total,heat}} + C_{\text{total,cool}}) \tag{18}$$

Taking square root of the yield value, and then multiplying by a constant k_{value} to move the optimal yield point

higher gave the diminishing returns required for exploring constrained shape and material optimisation, without the harsh cutoff associated with the target-based approach. The specific constant k_{value} used is solely to put the optimum yield tradeoff in the approximate range to yield sensible designs for the given size constraints. $k_{\text{value}} = 350$ was found to be appropriate for a 50x50x25m bounding volume, so it was with this value that Equation 18 was selected as the fitness function.

3.7 Genetic Algorithm Run

TABLE 4: Summary of Key GIFDO Parameters: Values Used to Generate Results in Section 4

Parameter Description	Symbol	Value
Genetic Algorithm Core Parameters		
Population Size	N_{pop}	50
Number of Generations	N_{gen}	1200
Worker Threads	N_{thread}	12
Subdivision Generation	G_{subdiv}	701
Survivor Rate (Elitism)	r_{surv}	Decay: $0.4 \rightarrow 0.16$ Generation: $0 \rightarrow 300$
Crossover Rate	r_{cross}	Increase: $0 \rightarrow 0.76$ Generation: $0 \rightarrow 400$
Mutation Parameters		
Setpoint Mutation Prob.	p_s	0.2
Geometry Mutation Prob.	p_g	Decay: $1 \rightarrow 0.6$ Generation: $0 \rightarrow 400$
Temp Mut. Std Dev	σ_T	Decay: $1 \rightarrow 0.1$ (°C) Generation: $0 \rightarrow 1000$
PAR Mut. Std Dev	σ_I	Decay: $1 \rightarrow 0.1$ Generation: $0 \rightarrow 1000$ ($\text{MJ/m}^2\text{/d}$)
Candidate Bounds		
Min Building Height	$h_{\min}$	5 m
Max Building Height	$h_{\max}$	25 m
Min Building Footprint	$A_{\min}$	$25\,\text{m}^2$
Max Building Footprint	$A_{\max}$	$2500\,\text{m}^2$
Min Temperature Setpoint	$T_{\min}$	18 °C
Max Temperature Setpoint	$T_{\max}$	30 °C
Min PAR Setpoint	$I_{\min}$	$0\,\text{MJ/m}^2\text{/d}$
Max PAR Setpoint	$I_{\max}$	$12\,\text{MJ/m}^2\text{/d}$
Farm Parameters		
Usable Growing Vol. Proportion	ϕ_{grow}	0.5
Vertical Rack Spacing	s_{rack}	1.5 m
LED Efficacy	η_{LED}	$2.5\,\mu\text{mol J}^{-1}$
Electricity Price	C_{elec}	$0.23\,\text{USD kW}^{-1}\,\text{h}$
Heating Energy Price	C_{heat}	$0.18\,\text{USD kW}^{-1}\,\text{h}$
Heating System Efficiency	η_{heat}	0.9
Cooling System COP	COP_{cool}	3.8
CO_2 Concentration	$[CO_2]$	1000 ppm
Sqrt Yield Value Multiplier	k_{value}	350

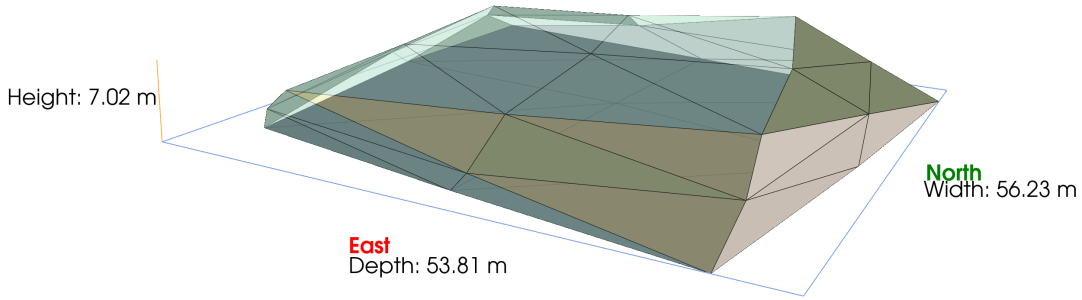
The parameters used to generate the solutions in Section 4 are detailed in Table 4. CO2 concentration was set at 1000 ppm, as recommended by Doddrell et al. [36] for optimal tomato growth. Electricity and gas prices in Singapore are on a regulated tariff. Electricity is priced at 0.3065 Singapore Dollars (SGD) [37], and gas at 0.2337 SGD [38] using bulk tariff A. These prices correspond to 0.23 USD and 0.18 USD respectively at the current exchange rate, so these values were used for C_{elec} and C_{heat} respectively.

At the end of each generation $N_{\text{pop}} \cdot r_{\text{surv}}$ elite candidates are first selected to survive to the next generation based on their fitness. Next, $N_{\text{pop}} \cdot r_{\text{cross}}$ child candidates are created from crossover, and added to the next generation (Section 3.3.2). Crossover is performed using a rank-based roulette wheel selection, with a selection pressure of 1.8. A portion of these child candidates are subject to mutation as detailed in Section 3.3.3. The remaining space in the next generation is populated with randomly initialised candidates, as detailed in Section 3.3.1.

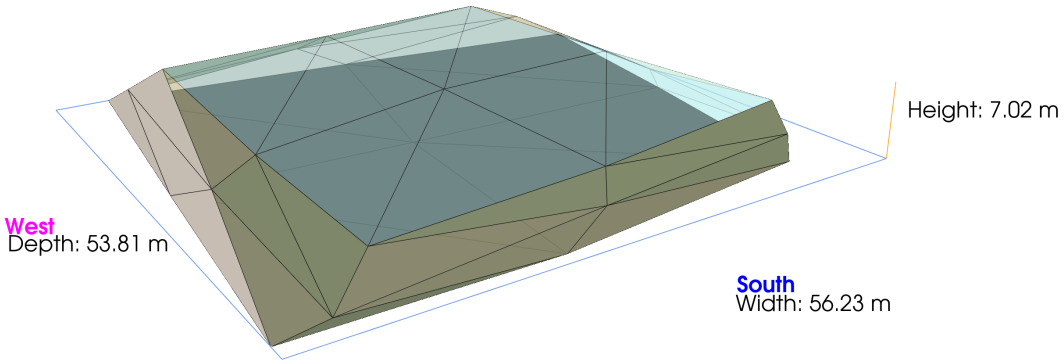
Crossover rate r_{cross} increases from 0 to 0.76, and geometry mutation rate p_g decreases from 1 to 0.6 over the first 400 generations. This is based on research from Hassanat et al. [39], who demonstrate Dynamic Decreasing of high mutation ratio/dynamic increasing of low crossover ratio (DHM/ILC) to outperform static crossover and mutation rates in the travelling salesman problem.

Subdivision is triggered after 700 generations. This provides enough time for the candidates at the base subdivision level (0) to converge to a near optimal solution, whilst still maintaining some genetic diversity. The simulation concludes after 1200 generations have passed, as fitness was observed to have plateaued by this point and further subdivision was not computationally feasible with the hardware available.

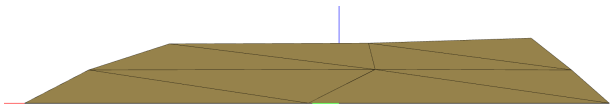
4 Results



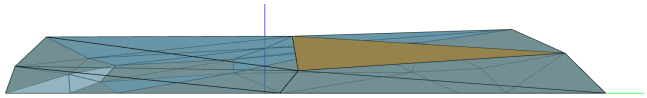
(a) Oslo design render, viewed from the South West. Faces colour coded according to their material.



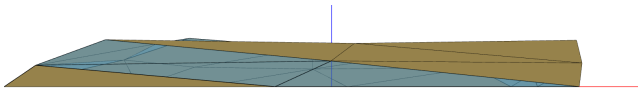
(b) Oslo design render, viewed from the North East. Faces colour coded according to their material.



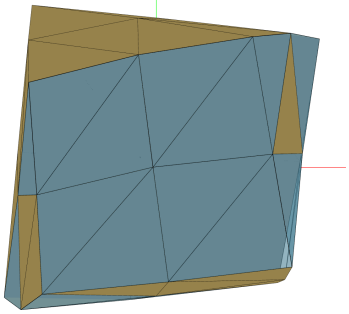
(c) North side view



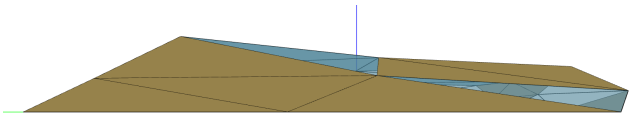
(d) East side view



(e) South side view



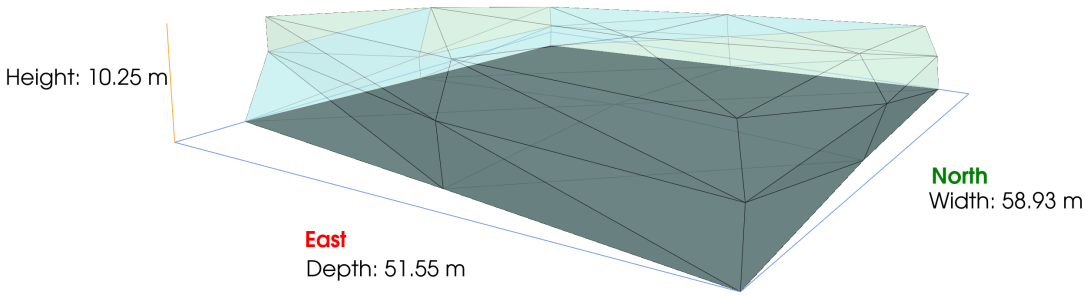
(f) Plan view (north side top)



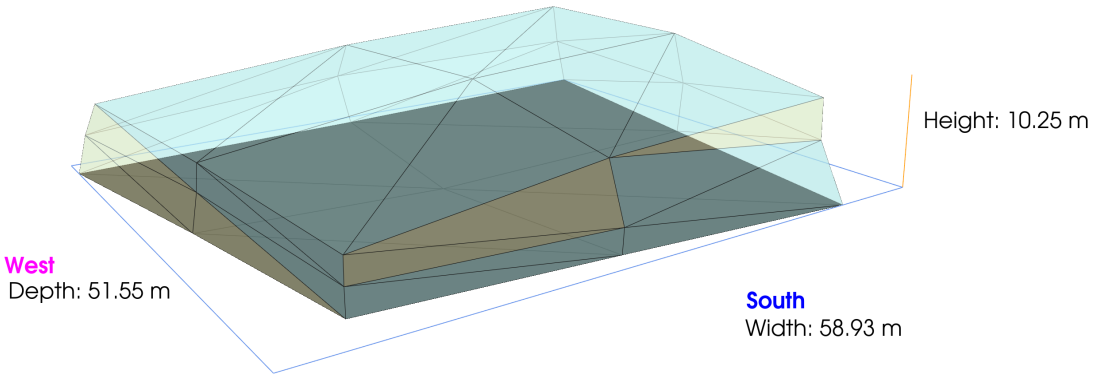
(g) West side view

Fig. 4: Figures (a) and (b) show the material assignments on each face for the generated Oslo design. Figures (c) - (g) show the distribution of opaque and glazing materials.

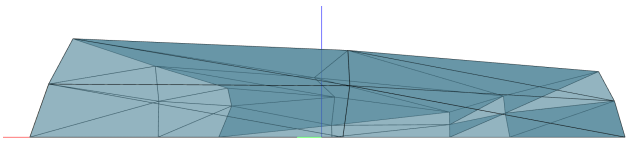
The optimal Oslo design shown in Fig. 4 uses both opaque and glazing materials. The two opaque wall materials used are 50mm insulation board (olive green), and 200mm lightweight concrete (beige). The two glazing materials used are low iron 3mm glass (blue translucent) and clear 6mm (yellow translucent). All floor surfaces are 300mm heavyweight concrete. This design is shorter than the design for Singapore, but with a similar footprint, resulting in a volume of 10 500 m^3. The roof slopes downwards towards the southern side in both the Oslo design and the Singapore design (Figure 5). This design achieved a final fitness score of 1,441,262.



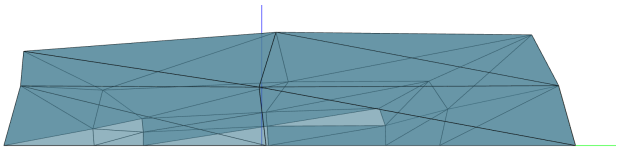
(a) Singapore design render, viewed from the South West. Faces colour coded according to their material.



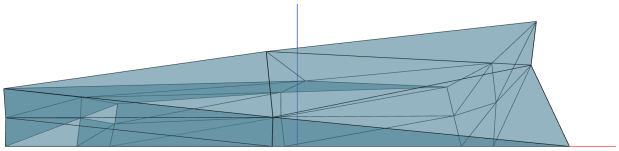
(b) Singapore design render, viewed from the North East. Faces colour coded according to their material.



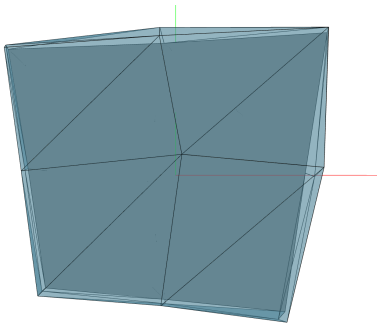
(c) North side view



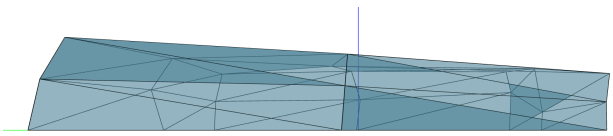
(d) East side view



(e) South side view



(f) Plan view (north side top)



(g) West side view

Fig. 5: Figures (a) and (b) show the material assignments on each face for the generated Singapore design. Figures (c) - (g) show the distribution of opaque and glazing materials.

The optimal Singapore design shown in Fig. 5 is dramatically different to the Oslo design. Glazing materials were selected for every surface, other than the floor where they are specifically restricted. The two glazing materials used are low iron 3mm glass (blue translucent) and clear 6mm (yellow translucent). All floor surfaces are 300mm heavyweight concrete. The internal volume is 19000 m^3, making it significantly larger than the Oslo design yet achieving a comparable yield. This design achieved a fitness score of 2,037,569, perhaps suggesting an easier climate than Oslo.

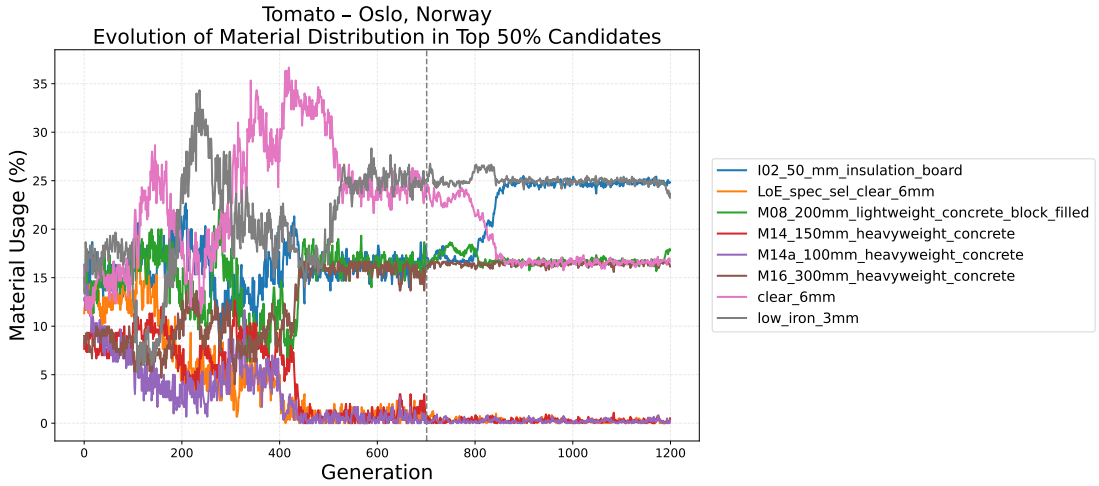


Fig. 6: Evolution of material usage percentages for the top 50% candidates in the Oslo optimisation run.

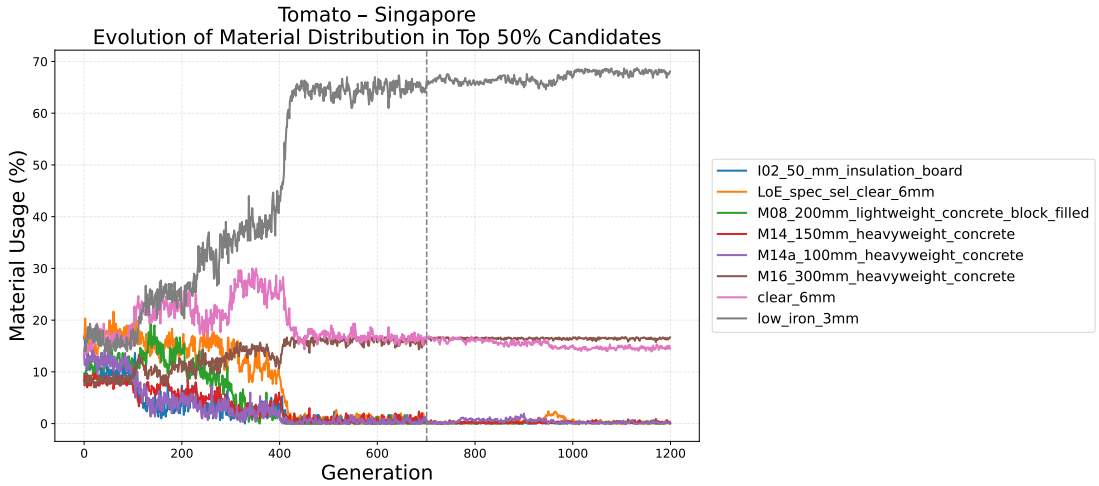


Fig. 7: Evolution of material usage percentages for the top 50% candidates in the Singapore optimisation run.

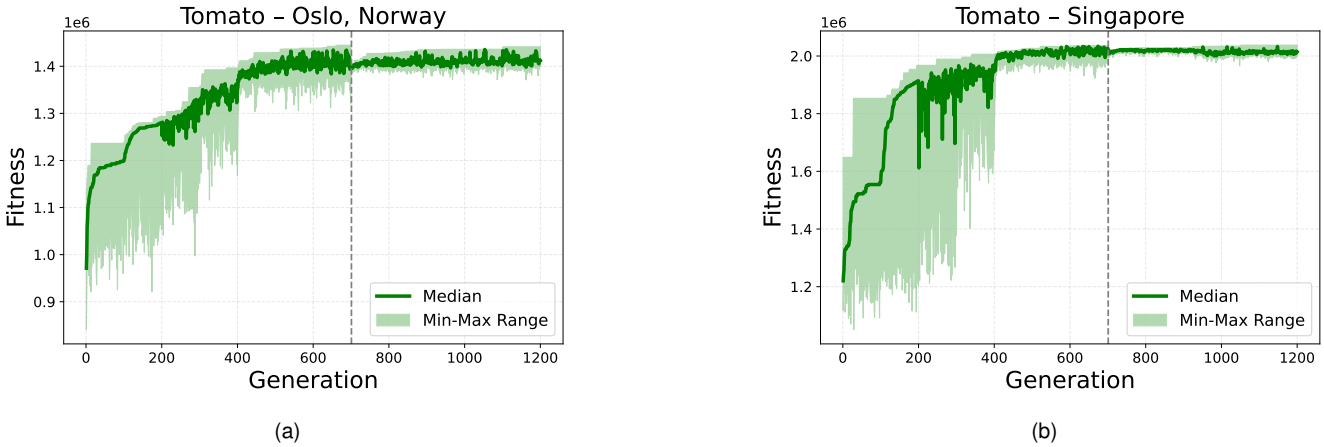
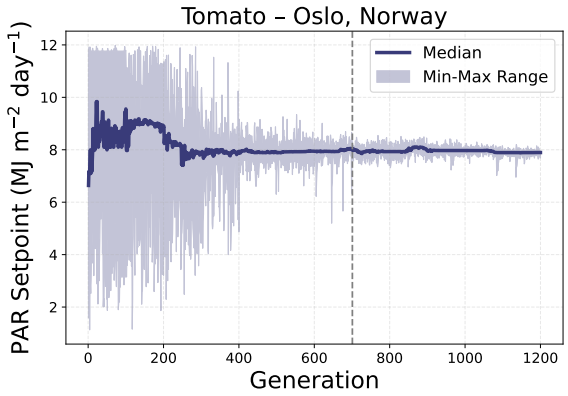


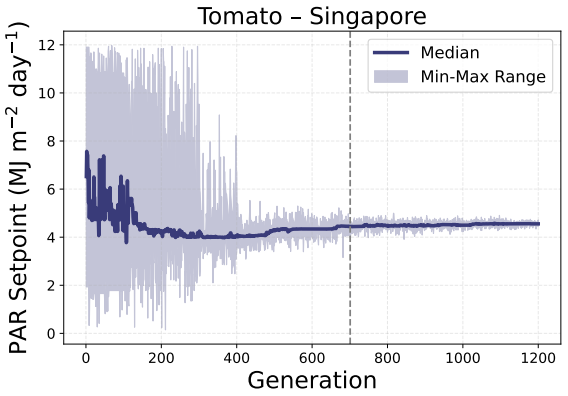
Fig. 8: Distribution of the fitness of the top 50% of candidates over each run for Oslo (a) and Singapore (b)

Figures 6-7 show that the optimal material distribution is mostly achieved before subdivision occurs, except for clear 6mm glass and 50mm insulation board in Fig. 6. The generation where subdivision occurs is shown using the dotted line in Figs. 6-8. Detailed population-wide and candidate specific parameter values can be found in data/run_summaries in the supplementary code repository.

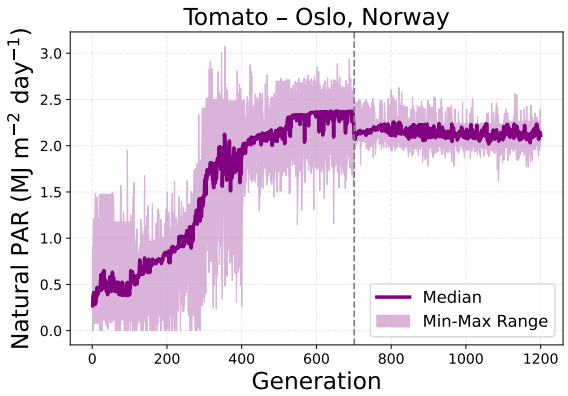
The small fitness drop post-subdivision is caused by EnergyPlus requiring an opaque frame material around glazing surfaces. Subdivision increases the total frame area at the new joins between window surfaces, slightly reducing incoming natural light and consequently lowering fitness. Figures 8 and 9 compare progression of several key variables across the Oslo and Singapore optimisation runs.



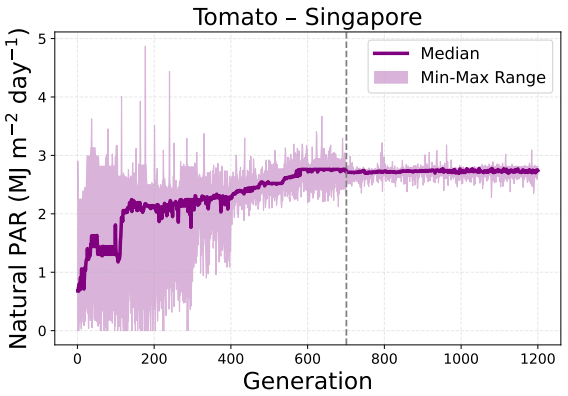
(a) Evolution of PAR setpoint, top 50% candidates (Oslo).



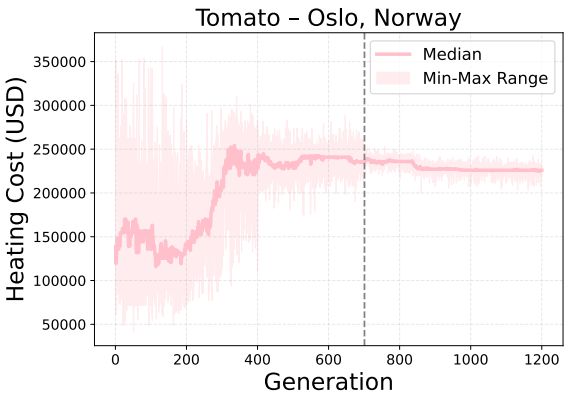
(b) Evolution of PAR setpoint, top 50% candidates (Singapore).



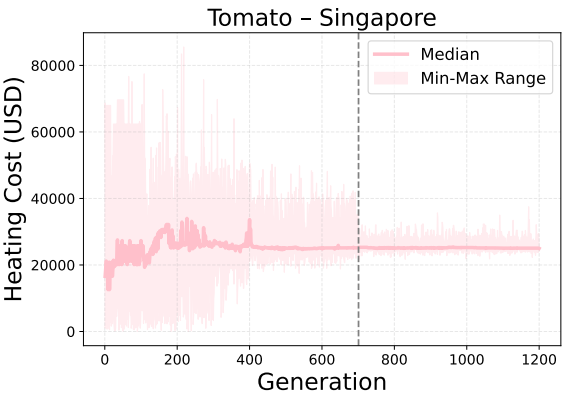
(c) Evolution of natural PAR, top 50% candidates (Oslo).



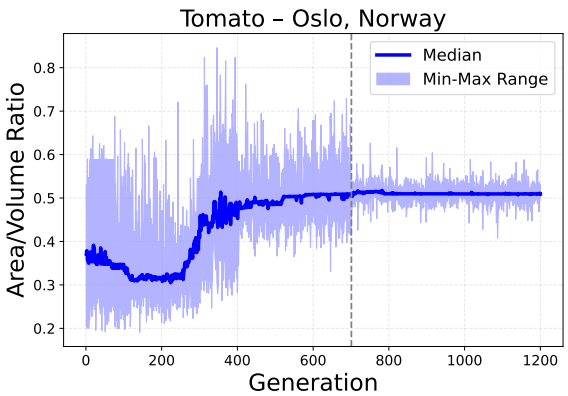
(d) Evolution of natural PAR, top 50% candidates (Singapore).



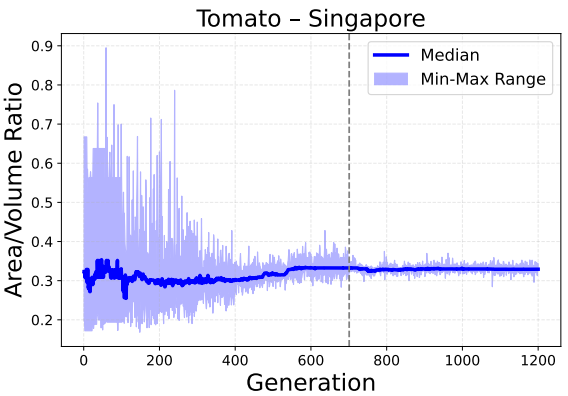
(e) Evolution of heating cost, top 50% candidates (Oslo).



(f) Evolution of heating cost, top 50% candidates (Singapore).



(g) Surface area /volume ratio, top 50% candidates (Oslo).



(h) Surface area /volume ratio, top 50% candidates (Singapore).

Fig. 9: Progression of parameters in the top 50% of candidates by fitness. Left Oslo, right Singapore.

5 Evaluation

5.1 Design Analysis

In both simulations the genetic algorithm proved effective in achieving convergence towards an optimal design. Figures 8 and 9 show significant genetic variety in the initial population, indicating wide exploration, before narrowing down to a more exploitative set of designs by the later generations. This suggests that the crossover, mutation, and elite survivor rates were appropriate for navigating this complex design problem. The top five candidates in each simulation run had nearly identical setpoints, geometry, and materials within their group: this can be seen in the supplementary code and data repository provided with this paper. As both runs were done with the same input parameters (other than the weather file), the considerable difference in the final designs shows that GIFDO is capable of adapting to a range of climates.

Several design trends emerged consistently across both climate scenarios. Designs converged to an identical temperature setpoint of 26 °C, aligning with the documented optimum for the SIMPLE tomato crop model used [26]. Across both simulations, the thicker 300mm heavyweight concrete was consistently chosen for all floor surfaces; this is likely a result of its larger thermal mass helping to mitigate internal temperature variations. A larger thermal mass can both absorb more thermal energy when the external temperature is too high, and dissipate heat for longer once the external temperature drops below the setpoint. Additionally, in nearly every case, the designs expanded horizontally to the maximum allowed footprint ($A_{\max}$ = 2500 m^2), varying height to achieve the desired volume. Maximising the footprint appears driven by the importance of capturing natural light to offset supplemental lighting costs, which dominate the fitness penalty.

Despite these commonalities, the considerable difference in the final designs highlights GIFDO's capability to adapt to specific climates. The Singapore design (Fig. 5) resulted in a large, relatively simple, predominantly glazed structure, though with some variation like a gently sloping roof and mixed glazing on South/West facades. To contrast this, the design generated for Oslo (Fig. 4) is visually more complex and significantly smaller in volume, likely due to the massively increased heating demand required to maintain the 26 °C setpoint. This design features sloping sides apparently suited to capturing sunlight, and a fully opaque North wall where incident light is minimal. Counterintuitively, the optimal Oslo design exhibited a larger surface-area-to-volume ratio than in Singapore; the increased potential heating demand suggested by this ratio seems offset by the algorithm's selection of more opaque surfaces with better insulating properties, such as 50mm insulation board.

This difference in form corresponds to differing operational strategies. To make up for the reduced growing space in the smaller Oslo volume, the PAR intensity setpoint was higher, at 7.89 $\mathrm{MJ\,m^{-2}\,d^{-1}}$, compared to 4.60 $\mathrm{MJ\,m^{-2}\,d^{-1}}$ in the Singapore design. This highlights a key potential advantage of generative design in this domain, where intricate tradeoffs between factors like heating load, lighting intensity, and available space can be balanced in pursuit of efficient yield.

5.2 Performance Context and Model Evaluation

Examining the energy costs provides further context. Both optimisation runs prioritised increasing incident natural light very highly, with lighting costs far exceeding heating and cooling in both runs, although by a smaller margin in Oslo. The total annual cost of lighting in the Oslo design was 933,674 USD, whereas heating totalled 227,432 USD and cooling just 5,163 USD. This results in lighting comprising approximately 80.06% of total operational energy expenditure. This proportion aligns with findings by Eaton et al. [40], who modelled resource consumption in commercial indoor farms in New York and found lighting accounted for 58% to 88% of total facility energy consumption, placing the GIFDO result within a feasible range.

However, comparing specific energy consumption (SEC) reveals a significant discrepancy. Blom et al. [41] report an SEC of 5.3 kWh/kg yield for lettuce in a vertical farm with an 80/20 lighting-to-heating energy split, comparable to the Oslo design's split. Yet, the calculated SEC for the optimised Oslo tomato design was only 0.39 kWh/kg, an order of magnitude lower. The primary reason appears to be the crop yield predictions. The SIMPLE model estimated approximately 690 $\mathrm{kg\,m^{-2}}$ for the Oslo design. This figure is significantly higher than typically reported values; for comparison, Blom et al. found yields of 79 $\mathrm{kg\,m^{-2}}$ for lettuce. While some difference is expected due to crop type, this large disparity suggests the SIMPLE model overestimates yields in this context, potentially due to assumptions about planting density or the exclusion of factors like irrigation limitations.

Because of this yield overestimation, absolute values for expected yield and derived metrics like SEC from this study should be interpreted with caution. However, for the task of design optimisation, the absolute accuracy of the yield figure used in the fitness function is arguably less critical than its ability to differentiate the relative performance between competing designs. Since the yield overestimation is consistent across all candidates (being inherent in the crop model used), it likely still provided a useful proxy for relative productivity, enabling the genetic algorithm to effectively optimise towards designs favouring conditions conducive to higher yields.

5.3 Limitations

Several simplifications within the simulation model warrant discussion. Firstly, the building envelope representation was constrained to only single material layers, omitting the complex multi-layer composition (including insulation, structure, reflective layers, etc.) common in real-world facilities. Exploring multi-layer constructions or allowing mixed materials on faces could yield more realistic and potentially higher-performing designs. Secondly, the daylighting analysis relied on EnergyPlus's internal split-flux method. While computationally efficient, this method has known accuracy limitations compared to detailed ray-tracing approaches like Radiance, particularly in complex geometries with significant inter-reflections [42], potentially impacting the predicted natural PAR availability. Lastly, the conversion of daylight to usable PAR implicitly assumed all daylight falls within the crop photoperiod, which may underestimate

supplemental lighting needs in high-latitude locations with extended summer daylight hours.

The scope of the fitness evaluation was focused primarily on maximising yield value and minimising only operational energy costs. Several significant cost factors were omitted, including the embodied energy and cost of construction materials, the operational costs of irrigation systems, and land costs, which could alter the relative ranking of solutions. Furthermore, the use of a single, weighted-sum objective function (Equation 18) provides a single 'optimal' solution based on the chosen weighting (k_{value}), rather than exploring the full tradeoff landscape. Implementing a multi-objective optimisation approach, such as NSGA-II [20], would yield a Pareto front of non-dominated solutions, offering decision-makers a richer set of designs representing different balances between profit and cost.

Aspects of the genetic algorithm implementation itself offer avenues for future refinement. The similarity of the final top 5 candidates in both the Oslo and the Singapore simulations suggests a large decrease in population diversity towards the end of runs. Incorporating an explicit mechanism for diversity preservation could enhance exploration. The geometric crossover operator, based on linear interpolation of corresponding vertex coordinates, could also be refined. As all candidates share an initial topology, this averaging approach may inadvertently dampen extreme geometric variations; vertices that have undergone significant displacement in one parent risk being pulled back towards the population mean position during crossover. This could potentially limit the exploration of highly diverse building geometry. A better approach which avoids this issue might be the 'shrinkwrap' method, outlined in [43], where vertices from one parent mesh are projected towards the nearest surface points on the other parent. This could avoid the averaging effect and might facilitate the discovery of more varied geometric solutions. Furthermore, the search strategy itself could be adapted; instead of the current single-stage approach tackling all variables simultaneously, exploring multi-stage strategies could allow for optimising additional growth parameters, such as $[CO_2]$ concentration, or colour of growing light used.

6 Conclusion

This research developed and evaluated GIFDO, a computational framework integrating procedural geometry generation, building performance simulation via EnergyPlus, and genetic algorithm optimisation specifically for indoor vertical farm design. The primary aim was the automation of design exploration to balance the competing objectives of potential yield value against operational energy costs within Controlled Environment Agriculture.

The genetic algorithm converged within the generation limit in each run. Under Oslo weather it produced a compact, well-insulated form; under Singapore's it preferred a taller, mostly glazed cuboid design. Distinct architectural features emerged from this process: Oslo solutions featured smaller volumes and insulated facades to manage heating loads, contrasting with Singapore's preference for larger glazed structures. Alongside these variations, consistent design trends appeared in both scenarios, including the selection of high thermal mass floors and maximised building footprints, highlighting the identified importance of natural light harvesting and internal thermal stability. GIFDO successfully identified nonintuitive tradeoffs, exemplified by the Oslo design's higher PAR target compensating for a smaller growing area compared to the Singapore design. This demonstrates the optimisation's capability to uncover efficient strategies tailored to each context.

Predicted energy distributions, particularly the dominance of lighting costs, showed alignment with established CEA literature benchmarks. However, the calculated Specific Energy Consumption (SEC) of 0.39 kWh/kg was significantly lower than these benchmarks, an outcome primarily attributed to overestimated yield predictions from the SIMPLE crop model. While absolute yield figures generated by this implementation should be viewed in this context, the consistent application of the model across all candidates indicates it likely still served as a valid proxy for relative performance comparison during optimisation. The main outcome is the validation of coupling genetic algorithms with building energy simulation as an effective methodology for exploring performance driven indoor farm designs, proving capable of generating diverse and climate adapted solutions.

The limitations identified suggest several avenues for future work. Enhancing simulation fidelity requires incorporating realistic multi-layer wall constructions and detailed HVAC system models for improved energy prediction accuracy. Using more detailed daylighting simulation tools (e.g., Radiance) and expanding the fitness evaluation criteria to include the energy associated with materials and construction, as well as operational factors like irrigation, would permit more comprehensive design assessments. Furthermore, adopting multiobjective optimisation algorithms would reveal the Pareto front of design tradeoffs, providing deeper insights than the single objective function used here.

Further refinements pertaining to the genetic algorithm include exploring explicit diversity preservation mechanisms and alternative geometric crossover operators (such as 'shrinkwrap' [43]) to potentially enhance solution space exploration and overcome limitations associated with vertex interpolation. Investigating multi-stage optimisation strategies and integrating a crop model specifically validated for indoor farming conditions remain crucial future steps for increasing both the scope and predictive accuracy of the GIFDO framework.

This work demonstrates the clear potential of integrating genetic search algorithms and building performance simulation for the automated design optimisation of vertical farms. GIFDO provides a functional foundation for subsequent research focused on truly 'from scratch' explorative design processes to improve the efficiency, sustainability, and economic feasibility of commercial indoor farming. Refining these computational design approaches is a crucial step for realising the potential of indoor agriculture to address global challenges in food security and resource management.

REFERENCES

[1] World Bank, "Arable land (hectares per person) – world," https://data.worldbank.org/indicator/AG.LND.ARBL.HA.PC?lo\\cations=1W, 2023, accessed: 2024-04-12.

[2] C. He, Z. Liu, J. Wu, X. Pan, Z. Fang, J. Li, and B. A. Bryan, "Future global urban water scarcity and potential solutions," *Nature Communications*, vol. 12, no. 1, p. 4667, 2021.

[3] E. F. Lambin and P. Meyfroidt, "Global land use change, economic globalization, and the looming land scarcity," *Proceedings of the national academy of sciences*, vol. 108, no. 9, pp. 3465–3472, 2011.

[4] A. Arcasi, A. Mauro, G. Napoli, F. Tariello, and G. Vanoli, "Energy and cost analysis for a crop production in a vertical farm," *Applied Thermal Engineering*, vol. 239, p. 122129, 2024.

[5] F. A. Lubna, D. C. Lewus, T. J. Shelford, and A.-J. Both, "What you may not realize about vertical farming," *Horticulturae*, vol. 8, no. 4, p. 322, 2022.

[6] S. A. Klein, "Trnsys-a transient system simulation program." *University of Wisconsin-Madison, Engineering Experiment Station Report*, pp. 38–12, 1988.

[7] F. Winkelmann, B. Birdsall, W. Buhl, K. Ellington, A. Erdem, J. Hirsch, and S. Gates, "Doe-2 supplement: version 2.1 e," Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), Tech. Rep., 1993.

[8] D. B. Crawley, L. K. Lawrie, F. C. Winkelmann, W. F. Buhl, Y. J. Huang, C. O. Pedersen, R. K. Strand, R. J. Liesen, D. E. Fisher, M. J. Witte *et al.*, "Energyplus: creating a new-generation building energy simulation program," *Energy and buildings*, vol. 33, no. 4, pp. 319–331, 2001.

[9] L. Magnier and F. Haghighat, "Multiobjective optimization of building design using trnsys simulations, genetic algorithm, and artificial neural network," *Building and Environment*, vol. 45, no. 3, pp. 739–746, 2010.

[10] J. Frazer, *An evolutionary architecture*. Architectural Association, 1995.

[11] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool, "Procedural modeling of buildings," in *ACM SIGGRAPH 2006 Papers*, 2006, pp. 614–623.

[12] S. Chang, N. Saha, D. Castro-Lacouture, and P. P.-J. Yang, "Generative design and performance modeling for relationships between urban built forms, sky opening, solar radiation and energy," *Energy Procedia*, vol. 158, pp. 3994–4002, 2019.

[13] P. Suphavarophas, R. Wongmahasiri, N. Keonil, and S. Bunyarittikit, "A systematic review of applications of generative design methods for energy efficiency in buildings," *Buildings*, vol. 14, no. 5, p. 1311, 2024.

[14] J. H. Holland, "Genetic algorithms," *Scientific american*, vol. 267, no. 1, pp. 66–73, 1992.

[15] G. Renner and A. Ekárt, "Genetic algorithms in computer aided design," *Computer-aided design*, vol. 35, no. 8, pp. 709–726, 2003.

[16] W. Wang, R. Zmeureanu, and H. Rivard, "Applying multi-objective genetic algorithms in green building design optimization," *Building and environment*, vol. 40, no. 11, pp. 1512–1525, 2005.

[17] L. Caldas, "Generation of energy-efficient architecture solutions applying gene_arch: An evolution-based generative design system," *Advanced Engineering Informatics*, vol. 22, no. 1, pp. 59–70, 2008.

[18] D. Jia, W. Zheng, X. Wei, W. Guo, Q. Zhao, and G. Gao, "Parametric design and genetic algorithm optimization of a natural light stereoscopic cultivation frame," *Agriculture*, vol. 14, no. 1, p. 84, 2023.

[19] Y. Schwartz, R. Raslan, I. Korolija, and D. Mumovic, "A decision support tool for building design: An integrated generative design, optimisation and life cycle performance approach," *International Journal of Architectural Computing*, vol. 19, no. 3, pp. 401–430, 2021.

[20] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[21] T. Verma and R. Shirota Filho, "Plnet: Light recipe design for indoor farming through generative deep learning," in *2024 IEEE Conference on Artificial Intelligence (CAI)*. IEEE, 2024, pp. 1092–1097.

[22] M. Dawson-Haggerty and contributors, "Trimesh," 2019. [Online]. Available: https://github.com/mikedh/trimesh

[23] R. American Society of Heating and A.-C. Engineers, *2005 ASHRAE Handbook: Fundamentals*, ser. ASHRAE Handbook: Fundamentals - SI Edition. ASHRAE, 2005. [Online]. Available: https://books.google.co.uk/books?id=XzO2AAAACAAJ

[24] L. J. Eshelman and J. D. Schaffer, "Real-coded genetic algorithms and interval-schemata," in *Foundations of genetic algorithms*. Elsevier, 1993, vol. 2, pp. 187–202.

[25] G. R. Raidl and J. Gottlieb, "On the importance of phenotypic duplicate elimination in decoder-based evolutionary algorithms," in *Late Breaking Papers at the 1999 Genetic and Evolutionary Computation Conference*. Citeseer, 1999, pp. 204–211.

[26] C. Zhao, B. Liu, L. Xiao, G. Hoogenboom, K. J. Boote, B. T. Kassie, W. Pavan, V. Shelia, K. S. Kim, I. M. Hernandez-Ochoa *et al.*, "A simple crop model," *European Journal of Agronomy*, vol. 104, pp. 97–106, 2019.

[27] Intelligent Systems Center, "croprl: A reinforcement-learning environment for crop models," GitHub repository, https://github.com/iscoe/croprl, 2021, accessed: 2025-04-30.

[28] E. Dohlman, K. Maguire, W. V. Davis, M. Husby, J. Bovay, C. Weber, and Y. Lee, "Trends, insights, and future prospects for production in controlled environment agriculture and agrivoltaics systems," U.S. Department of Agriculture, Economic Research Service, Economic Information Bulletin 264, 2024, accessed 30 Apr 2025. [Online]. Available: https://www.ers.usda.gov/publications/pub-details/?pubid=108221

[29] S. N. E. Agency. (2022) Minimum energy performance standards (meps): Regulated air-conditioners. Accessed 24 April 2025. [Online]. Available: https://www.nea.gov.sg/our-services/climate-change-energy-efficiency/energy-efficiency/household-sector/minimum-energy-performance-standards

[30] G. J. Ward, "The radiance lighting simulation and rendering system," in *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 1994, pp. 459–472.

[31] Q. Li and J. Haberl, "Prediction of annual daylighting performance using inverse models," *Sustainability*, vol. 15, no. 15, p. 11938, 2023.

[32] S. Oh and J. S. Haberl, "Origins of analysis methods used to design high-performance commercial buildings: Whole-building energy simulation," *Science and Technology for the Built Environment*, vol. 22, no. 1, pp. 118–137, 2016.

[33] S. Gillies, C. van der Wel, J. Van den Bossche, M. Taves, J. Arnott, B. Ward *et al.*, "Shapely (version 2.1.0)," https://github.com/shapely/shapely, 2025.

[34] Apogee Instruments, Inc., "Conversion – PPFD to Lux," https://www.apogeeinstruments.com/conversion-ppfd-to-lux/, n.d., accessed: 2025-04-30.

[35] Gigahertz Optik GmbH, "Measurement of PAR (Photosynthetically Active Radiation)," https://www.gigahertz-optik.com/en-us/service-and-support/knowledge-base/measurement-of-par/, n.d., accessed: 2025-04-30.

[36] N. H. Doddrell, T. Lawson, C. A. Raines, C. Wagstaff, and A. J. Simkin, "Feeding the world: impacts of elevated [co2] on nutrient content of greenhouse grown fruit crops and options for future yield gains," *Horticulture research*, vol. 10, no. 4, p. uhad026, 2023.

[37] S. E. M. Authority. (2025, Mar.) Buying electricity at the regulated tariff (1 april – 30 june 2025). Accessed 24 April 2025. [Online]. Available: https://www.ema.gov.sg/consumer-information/electricity/buying-electricity/buying-at-regulated-tariff

[38] ——. (2025, Mar.) Buying gas at the regulated tariff (1 april – 30 june 2025). Accessed 24 April 2025. [Online]. Available: https://www.cityenergy.com.sg/gas-tariffs/

[39] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri, and V. S. Prasath, "Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach," *Information*, vol. 10, no. 12, p. 390, 2019.

[40] M. Eaton, T. Shelford, M. Cole, and N. Mattson, "Modeling resource consumption and carbon emissions associated with lettuce production in plant factories," *Journal of Cleaner Production*, vol. 384, p. 135569, 2023.

[41] T. Blom, A. Jenkins, and A. van den Dobbelsteen, "Synergetic integration of vertical farms and buildings: reducing the use of energy, water, and nutrients," *Frontiers in Sustainable Food Systems*, vol. 7, p. 1227672, 2023.

[42] G. Ramos and E. Ghisi, "Analysis of daylight calculated using the energyplus programme," *Renewable and Sustainable Energy Reviews*, vol. 14, no. 7, pp. 1948–1958, 2010.

[43] F. Claus, B. Hamann, and H. Hagen, "A finite-element based mesh morphing approach for surface meshes," *Computer-Aided Design*, vol. 146, p. 103232, 2022.