Climate-Adaptive Generative Design of Residential Buildings Using Reinforcement Learning and Parametric Modeling

Mirella Pruschke
Department of Computer Science and Engineering, University of Nevada, Reno, Reno, NV, USA.
hellomirella@unr.edu

Mees Postma
Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO, USA.
contactmees@missouri.edu

Abstract

Residential building design under climate uncertainty requires simultaneous decisions about envelope geometry, glazing, shading, thermal mass, and natural ventilation, yet these decisions are usually made before operational control strategies are known. This paper develops a methodological framework for climate-adaptive generative design in which reinforcement learning learns a policy over parametric residential design variables rather than only over HVAC setpoints. The design problem is formulated as a constrained multi-objective Markov decision process in which states encode partial design configurations and simulation-derived performance, actions modify geometric and envelope parameters, and rewards combine normalized energy, thermal discomfort, life-cycle carbon, and cost terms with feasibility penalties. A parametric modeling layer maps design actions to simulation-ready geometry and building systems, while a simulation layer evaluates performance across climate scenarios and occupancy assumptions. Proximal policy optimization is used to stabilize learning under high-dimensional, non-stationary design rewards, and a robust evaluation rule integrates expected and worst-case climate performance. The framework also specifies Pareto filtering, constraint handling, uncertainty treatment, and sensitivity analysis so that generated designs can be assessed as trade-off candidates rather than as a single optimum. The main contribution is the formal coupling of generative design with a learned sequential decision policy, enabling the production of diverse, climate-conditioned residential design candidates and providing a technically defensible basis for early-stage design exploration. Practical significance lies in linking envelope-level climate adaptation with operational flexibility and in making constraint-aware design search more scalable than repeated independent optimization runs.

Keywords

reinforcement learning; parametric design; climate adaptation; residential buildings; multi-objective optimization; building performance simulation

1. Introduction

Residential buildings are long-lived systems whose energy demand, indoor comfort, and carbon performance depend on decisions made during early design. Building performance

simulation provides a principled way to estimate these consequences before construction, and whole-building energy simulation programs have become established tools for evaluating envelope, system, and control assumptions under dynamic weather and occupancy conditions [1]. However, simulation alone does not resolve the design problem: it evaluates a specified configuration, whereas design requires searching a large and often discontinuous space of geometric, material, and operational alternatives. The resulting gap is particularly acute for residential buildings, where small changes in window-to-wall ratio, orientation, shading depth, roof form, and thermal mass can shift heating, cooling, daylight, and peak demand outcomes in different directions.

Climate adaptation adds a second layer of complexity. Climate-adaptive building envelopes and related technologies can modify heat, light, and air exchange in response to environmental conditions, but their design must anticipate seasonal and diurnal variability rather than a single design-day condition [2]. In residential design, adaptation may be passive, such as adjustable shading, operable windows, or thermally massive elements, or active, such as controlled ventilation and mechanical conditioning. The design question is therefore not simply which envelope minimizes annual energy use; it is which configuration remains comfortable, affordable, and low-carbon across plausible climate realizations and occupancy patterns. This framing makes the design problem multi-objective and stochastic, and it requires a search method that can represent dependencies among geometric form, envelope properties, and control possibilities.

Parametric design offers a suitable representational substrate because it encodes buildings as families of geometries controlled by explicit variables. Yet parametric models are only generative when coupled with a search or learning procedure that proposes configurations. Early work on generative design methods established that rule-based and evolutionary procedures can explore design alternatives beyond manual enumeration, but such methods often require repeated independent evaluations and do not naturally produce a reusable design policy [7]. Simulation-based optimization has since become a common approach for building performance problems, with evolutionary algorithms, direct search, and surrogate-assisted methods applied to envelope and system design [9]. Reviews of computational optimization for sustainable building design show that algorithm choice, objective normalization, constraint representation, and simulation cost strongly affect the usefulness of the resulting solutions [10]. Uncertainty analysis is equally important because weather, occupancy, and material properties can alter the ranking of design alternatives [11].

Reinforcement learning has been developed mainly for building operations, where an agent learns control actions such as supply air temperature, fan speed, or setpoint adjustments from interaction with a simulated environment. Deep reinforcement learning has been applied to HVAC control [4], whole-building energy model control [5], and demand response [6]. These studies demonstrate that sequential decision-making under dynamic conditions is a natural fit for building control. They also reveal recurring difficulties: reward design, safety constraints, sample efficiency, and the gap between simulated training and real operation. The present paper argues that these difficulties are not confined to operations. They reappear, in modified form, when reinforcement learning is used for design generation, because design variables define a sequential construction process in which each action changes the geometry and performance context of later actions.

A recent generative optimization of building blocks shows that density, solar exposure, and structural performance can be explored jointly in a multi-objective search [8]. That line of

work confirms the value of generative search for early-stage massing and environmental performance, but it also highlights a limitation: a set of optimized candidates is not the same as a design policy that can be conditioned on climate and sampled repeatedly. The framework proposed here treats reinforcement learning as a generative design engine. The agent does not control HVAC equipment during operation; it constructs residential design configurations by selecting parametric design actions. The central contribution is a constrained multi-objective Markov decision process formulation that couples a parametric building model, a simulation-based performance evaluator, and a policy-gradient learner. The policy is intended to generate diverse climate-adaptive residential design candidates and to expose trade-offs among energy, comfort, carbon, and cost.

2. Problem Formulation and Climate-Adaptive Design Space

2.1 Design variables and parametric representation

The residential design space is represented by a vector of parameters that control geometry, envelope composition, glazing, shading, thermal mass, and passive ventilation features. Let x $\in$ X denote a complete design vector, and let c denote a climate scenario drawn from a set of plausible current and future weather files. The parametric model P maps x to a simulation-ready building description, including wall and roof surfaces, window areas, orientation, internal zoning, and system assumptions. A simulation operator S then evaluates the building description under climate scenario c. The performance vector is written as follows.

$$\min_{x \in X} F(x, c) = \big(\mathrm{E}(x, c), D(x, c), C(x, c), K(x, c)\big)$$

Here E is annual energy use intensity, D is a thermal discomfort indicator such as weighted unmet heating and cooling hours, C is life-cycle carbon, and K is a normalized cost proxy. All four objectives are treated as quantities to be reduced after appropriate normalization. The vector notation is deliberate: no single scalar objective is assumed to represent residential quality. A design with low energy use but excessive overheating, or low carbon but prohibitive glazing cost, should remain visible as a trade-off rather than being collapsed prematurely.

$$y(x, c) = S(\mathrm{P}(x), c)$$

In this expression, y is the simulation output vector from which E, D, C, and K are derived. P is assumed to be deterministic for a given x, while S may include stochastic occupancy schedules or model uncertainty. This separation is useful because it allows the design policy to operate on parameters that designers can interpret, while the simulation layer remains responsible for physically grounded performance estimation. The parametric representation should include explicit feasibility bounds, such as maximum glazing area, minimum daylight provision, structural depth limits, and constructability constraints. These bounds define the legal action space and prevent the learner from generating physically implausible configurations that happen to score well under a narrowly defined reward.

2.2 Performance objectives and constraints

The objective vector must be normalized before it is used in learning. A practical normalization is based on reference values derived from the design space itself, such as the mean and standard deviation of each objective over an initial sampling of feasible designs. Normalization prevents energy, which may be measured in kilowatt-hours per square metre, from dominating discomfort, which may be measured in hours. It also makes reward weights

interpretable as relative priorities rather than as artifacts of unit choice. Constraints are represented separately from objectives. Thermal comfort constraints may require occupied-zone operative temperature to remain within an adaptive comfort band for a specified fraction of occupied hours. Daylight constraints may require a minimum useful daylight illuminance or daylight factor in habitable rooms. Structural and cost constraints may limit cantilever length, glazing area, or envelope thickness.

The design problem is therefore a constrained multi-objective optimization problem. A candidate x is feasible only if all constraint functions g_j(x) satisfy their bounds b_j. Feasible candidates are compared using Pareto dominance, while infeasible candidates receive penalties during learning. This arrangement is important because a policy that learns only from a scalarized reward may converge to a single compromise and lose the diversity needed for early-stage design. The Pareto interpretation is specified later, after the learning formulation, because the reinforcement learning agent requires a scalar reward even when the final assessment is multi-objective.

3. Reinforcement Learning Architecture for Generative Design

3.1 Markov decision process formulation

The generative design process is modeled as a finite-horizon Markov decision process. At each step t, the state s_t contains the current partial design configuration, the remaining design budget, and summary statistics from any performance evaluations completed so far. The action a_t selects a parametric modification, such as changing the window-to-wall ratio in a facade zone, adjusting overhang depth, altering roof pitch, modifying thermal mass, or enabling a passive ventilation aperture. The transition function T is deterministic with respect to the parametric model but stochastic with respect to simulation noise if occupancy or weather sampling is used. The discount factor γ controls how strongly early design moves are valued relative to final performance. The tuple is defined as follows.

$$M = (S, A, T, r, \gamma)$$

Here S is the state space, A is the action space, T is the transition kernel, r is the reward function, and $\gamma \in [0,1]$ is the discount factor. The state space is hybrid: it includes continuous parameters, discrete choices such as glazing type, and categorical variables such as orientation sector. The action space is also hybrid, so a Gaussian policy is used for continuous actions and a categorical policy for discrete actions. A complete episode ends when the design budget is exhausted or when no feasible action remains. The final state represents a complete candidate design that can be simulated and assessed.

The policy is a parameterized distribution π_θ(a_t | s_t). For a given policy, the expected discounted return is written as follows.

$$J(\theta) = E_{\pi_\theta}[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)]$$

The expectation is taken over trajectories generated by the policy, the initial state distribution, and any stochastic simulation variables. The objective J(θ) is not the final design objective. It is the expected cumulative reward collected during the construction of a design. This distinction matters because a design action may be locally unattractive but strategically useful: for example, increasing thermal mass may reduce later overheating and therefore improve the final comfort reward. The sequential formulation allows the policy to represent such dependencies, whereas a one-shot optimization method must rediscover them through repeated evaluations.

3.2 Reward shaping and constraint handling

The reward function translates the multi-objective design problem into a learning signal. It combines normalized performance terms with penalties for constraint violation. A general form is as follows.

$$r_t = -\sum_{i=1}^{m} w_i \tilde{f}_i(s_t) - \sum_{j=1}^{k} \lambda_j \max(0, g_j(s_t) - b_j)$$

In this expression, $\tilde{f}_i(s_t)$ is the normalized value of objective i associated with the partial or completed design state, w_i is its weight, $g_j(s_t)$ is constraint j, b_j is its allowable bound, and λ_j is a penalty coefficient. The negative sign converts the minimization problem into a reward maximization problem. The penalty term is active only when a constraint is violated. A common failure mode is that a large penalty makes the reward landscape too steep near the feasible boundary, causing the policy to collapse toward overly conservative designs. A more stable formulation uses a moderate penalty during early learning and increases λ_j as the policy approaches feasibility. The weights w_i should be treated as design-preference parameters, not as universal constants, because different residential projects may prioritize affordability, comfort, or carbon differently.

Reward shaping must also account for simulation cost. If every action triggers a full annual simulation, the learning problem becomes prohibitively expensive. A staged evaluation scheme is therefore preferable. Early in an episode, a reduced-order model or a short simulation period estimates directional effects. Full annual simulation is reserved for completed designs or for checkpoints at which the partial design is physically coherent. The reward can then combine a cheap progress term with a delayed final performance term. This design choice introduces a bias-variance trade-off: cheap models reduce variance in the learning signal but may misrank candidates if their domain of validity is exceeded. The safe approach is to use reduced-order evaluation only for actions whose effects are well approximated by the reduced model and to calibrate the reduced model against full simulation on a representative sample of feasible designs.

3.3 Policy optimization

Proximal policy optimization is adopted because it provides stable updates in continuous and hybrid action spaces while avoiding the severe policy collapse that can occur with unconstrained policy-gradient updates. The clipped surrogate objective is written as follows.

$$L^{CLIP}(\theta) = \hat{E}_t[\min(\rho_t(\theta)\hat{A}_t, \mathrm{clip}(\rho_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)]$$

Here $\rho_t(\theta) = \pi_\theta(a_t \mid s_t) / \pi_{\theta_{old}}(a_t \mid s_t)$ is the probability ratio between the current and old policies, $\hat{A}_t$ is an advantage estimate, and ε is the clipping range. The clipped term limits the incentive for large policy changes when the advantage is positive or negative. This is useful in design generation because the reward landscape is non-stationary: early actions change the meaning of later actions, and a large update can push the policy into a region where most actions violate geometric or cost constraints. The advantage estimate can be computed with generalized advantage estimation using a learned value function. The value function shares a representation with the policy but is trained with a separate regression loss.

Alternative reinforcement learning algorithms are credible. Soft actor-critic offers better sample efficiency in continuous control and encourages exploration through entropy regularization. However, its off-policy replay buffer may be dominated by early infeasible designs if the initial policy explores outside the feasible region. Trust-region policy optimization is also suitable but computationally heavier. Proximal policy optimization

provides a practical compromise for a design agent whose action space is mixed and whose simulation budget is the main constraint. Real-world reinforcement learning studies emphasize that sample efficiency, safety, and distribution shift are persistent challenges [14]; these concerns apply directly to design generation, where the "environment" is a simulation model that may not represent every construction or operational condition.

4. Coupled Simulation and Parametric Modeling Workflow

4.1 Climate and occupancy boundary conditions

Climate adaptation requires more than a single weather file. The design policy should be trained and evaluated across a set of climate scenarios that represent current conditions, plausible warming, and extreme events relevant to the project location. Each scenario c_k has an associated probability p_k or a weight reflecting decision-maker beliefs. The policy state can include summary climate features, such as heating and cooling degree days, solar exposure, and humidity, so that the learned policy can condition its actions on climate rather than memorizing a single site. Occupancy schedules should also be treated as uncertain. Residential occupancy is not identical to office occupancy: weekday and weekend patterns, occupant density, window-opening behavior, and internal gains vary widely. A design that performs well under one occupancy schedule may fail under another, particularly when natural ventilation or shading depends on occupant interaction.

A robust evaluation rule can be defined as a weighted combination of expected and worst-case performance. One form is as follows.

$$f_rob(x) = (1-\alpha)\sum_{k=1}^{K} p_k f(x, c_k) + \alpha \max_k f(x, c_k)$$

Here f_rob(x) is the robust performance value, K is the number of climate scenarios, p_k is the probability or weight of scenario k, and $\alpha \in [0,1]$ is a risk-aversion parameter. When α is zero, the rule reduces to expected performance; when α is one, it reduces to worst-case performance. This formulation does not eliminate uncertainty, but it makes the treatment of uncertainty explicit and adjustable. For residential design, a moderate α is often defensible because extreme overheating events and peak cooling demand can impose health and cost consequences that are not captured by annual averages.

4.2 Generative geometry and performance evaluation

The parametric model must produce geometry that is both design-relevant and simulation-ready. This requires a consistent naming and zoning convention: each facade should have a defined orientation, each window should belong to a thermal zone, and each shading device should be associated with the glazing it protects. The model should also expose variables at the level of design intent rather than at the level of mesh vertices. Examples include window-to-wall ratio per orientation, overhang depth, fin spacing, roof albedo, insulation thickness, thermal mass class, and operable window area. This level of abstraction reduces the dimensionality of the action space and keeps the learned policy interpretable to architects and engineers.

The evaluation layer should return not only annual energy but also peak demand, overheating hours, daylight availability, and cost. If the simulation engine does not directly provide a required metric, it should be derived through a documented post-processing rule. For example, discomfort can be computed from zone operative temperature and occupancy

schedules; carbon can be computed by multiplying electricity and fuel use by scenario-specific emission factors. Cost should be treated as a relative proxy unless project-specific cost data are available. The policy should not be trained on a metric that is invisible to the final decision process. Each reward term must map to a design objective that a decision-maker can inspect.

4.3 Training, exploration, and design diversity

Training proceeds in episodes. At the start of an episode, the environment samples a climate scenario, an occupancy profile, and an initial design state. The policy selects actions until the design is complete or the step budget is exhausted. Completed designs are simulated, and the reward is computed. The policy is updated after a batch of episodes. Exploration is controlled by the entropy of the action distribution and by the clipping range. Too little exploration produces repetitive designs; too much exploration produces many infeasible candidates. A balanced schedule reduces the action-space variance over training while maintaining enough stochasticity to discover non-obvious envelope strategies.

Design diversity should be measured explicitly. A policy that always generates the same high-performing configuration is not a generative design tool; it is a single-solution optimizer. Diversity can be encouraged by maintaining an archive of feasible designs and penalizing near-duplicates, or by conditioning the policy on preference weights that vary across episodes. Preference-conditioned training allows the same policy to generate designs for different priorities, such as lower first cost versus lower overheating risk. This is more efficient than retraining a separate policy for every weighting of objectives. The archive also supports post-training Pareto analysis, because the final candidate set can be filtered for non-dominated designs and inspected by designers.

5. Technical Analysis and Design Trade-offs

5.1 Pareto assessment and objective conflicts

The output of the learned policy is a distribution over design candidates, not a single answer. The candidates should be filtered using Pareto dominance. For minimization, dominance is defined as follows.

$$x \succ x' \Leftrightarrow \forall i\, f_i(x) \leq f_i(x') \wedge \exists j\, f_j(x) < f_j(x')$$

A design x dominates x′ if it is no worse on every objective and strictly better on at least one. The non-dominated set forms an approximation to the Pareto front. This assessment is essential because climate-adaptive residential design contains structural conflicts. Increased glazing may improve daylight and winter solar gain but increase summer overheating and cooling demand. Added thermal mass may reduce temperature swings but increase embodied carbon and structural cost. External shading may reduce cooling loads but obstruct useful daylight and winter solar gain. A policy trained with a single reward weighting will sample designs along a particular preference direction; Pareto filtering across multiple preference weights recovers a broader set of trade-offs.

The Pareto front should be interpreted with constraints. A design that is non-dominated but violates a comfort or structural constraint is not a valid recommendation. Constraint-aware Pareto filtering therefore separates feasible from infeasible candidates before dominance testing. This is consistent with the design formulation in Section 2 and prevents the learner from presenting a visually attractive but physically unacceptable configuration as a solution. The final candidate set should include both low-energy and low-discomfort extremes

when such extremes exist, because these help designers understand the cost of improving one objective at the expense of another.

5.2 Sensitivity and failure modes

Sensitivity analysis is necessary because the learned policy may exploit modeling artifacts. A design that performs well because a simulation model ignores a relevant physical mechanism is not a robust result. Sensitivity can be assessed by perturbing design parameters, weather variables, and occupancy schedules around a candidate and observing changes in performance. Parameters with large influence should be examined for physical plausibility. If a small change in overhang depth produces a large discontinuous change in daylight, the cause may be a geometry or sensor-placement artifact rather than a genuine design effect. If a small change in insulation thickness produces an implausibly large energy change, the simulation assumptions should be checked.

Several failure modes are specific to reinforcement learning design generation. Reward hacking occurs when the policy finds a configuration that scores well under the proxy reward but violates design intent, such as a facade with almost no glazing that minimizes energy at the expense of daylight. Constraint collapse occurs when penalties are so strong that the policy avoids all actions near feasible boundaries, producing conservative designs that miss climate-adaptive opportunities. Distribution shift occurs when the policy is trained on a narrow set of climate files and then evaluated on a different climate. Simulation noise occurs when stochastic occupancy or weather sampling makes the reward ranking unstable. Each failure mode has a technical response: richer reward diagnostics, constrained policy updates, climate-conditioned training, and variance reduction through common random numbers or deterministic evaluation seeds.

5.3 Computational cost and scalability

The main computational cost is simulation. A policy-gradient update requires many episodes, and each completed design may require annual simulation across multiple climate scenarios. The cost can be reduced by parallel simulation, surrogate modeling, and staged evaluation. Surrogate models can be trained to approximate simulation outputs for candidate designs, but they introduce approximation error. A safe workflow uses surrogate models for candidate screening and full simulation for final assessment. The policy should be trained on a mixture of surrogate and full-simulation rewards if the surrogate is well calibrated. If the surrogate is not calibrated, it should be used only as an exploration aid and not as the basis for final Pareto claims.

Scalability also depends on the state representation. A state that contains the entire geometry mesh is unnecessarily large; a state that contains design parameters, zone-level summaries, and performance history is more compact and more transferable across projects. The policy can then be pretrained on a family of residential typologies and fine-tuned for a specific site. This does not guarantee transfer, because climate and construction constraints differ, but it reduces the simulation burden of learning from scratch. The framework therefore treats reinforcement learning as a reusable design prior rather than as a site-specific optimizer.

6. Discussion: Implementation Conditions and Limitations

The proposed framework is a methodological contribution rather than a validated deployment. Its usefulness depends on several implementation conditions. First, the parametric model must be clean enough that every action produces a valid simulation input. If

the model breaks under feasible actions, the policy learns from simulation errors rather than from design performance. Second, the simulation layer must be calibrated for the relevant residential typology. Default assumptions for occupancy, internal gains, and HVAC efficiency can dominate envelope effects if they are inappropriate. Third, the reward weights and constraint bounds must be set with stakeholder input. A policy trained on arbitrary weights will generate designs that reflect those arbitrary weights, even if the learning algorithm is sound.

The approach also has limitations relative to conventional optimization. Evolutionary algorithms and Bayesian optimization are well understood for building performance problems and can be more sample-efficient for low-dimensional design spaces [18]. Reinforcement learning is most advantageous when the design problem has sequential structure, multiple interacting decisions, and a need for a reusable policy that can generate diverse candidates. In small residential projects with few variables, a conventional simulation-based optimization may be sufficient. The framework should therefore be applied where the design space is genuinely combinatorial and where climate adaptation requires coordinated decisions across envelope, shading, mass, and ventilation.

A further limitation is that the learned policy is only as good as the simulation model and the reward specification. It cannot replace engineering judgment, and it cannot certify compliance with building codes. The generated candidates must be reviewed by architects and engineers, and final designs must be verified with project-specific data. The value of the method lies in earlier exploration and better-structured trade-off information, not in automated final decision-making.

7. Conclusion

This paper has developed a climate-adaptive generative design framework for residential buildings that combines parametric modeling, building performance simulation, and reinforcement learning. The central formulation treats design as a constrained multi-objective Markov decision process in which an agent sequentially modifies parametric building variables and receives rewards based on normalized energy, discomfort, carbon, and cost, with penalties for constraint violation. Proximal policy optimization provides stable learning in hybrid action spaces, while robust evaluation across climate scenarios and occupancy assumptions addresses the uncertainty inherent in climate adaptation. Pareto filtering and sensitivity analysis convert the policy's output into a candidate set that designers can inspect and compare.

The contribution is not a claim of empirical superiority over existing optimization methods. It is a technically specified alternative for design problems in which sequential dependencies, multi-objective trade-offs, and climate-conditioned generation matter. The framework clarifies how reward design, constraint handling, simulation cost, and uncertainty should be treated so that reinforcement learning can serve generative design rather than only operational control. Future implementation should focus on calibrating the simulation layer, testing the policy against established optimization baselines, and evaluating whether the generated candidates remain diverse and feasible under realistic project constraints.

References

1. Crawley, D. B., Lawrie, L. K., Winkelmann, F. C., Buhl, W. F., Huang, Y. J., Pedersen, C. O., Strand, R. K., Liesen, R. J., Fisher, D. E., Witte, M. J., & Glazer, J. (2001).

EnergyPlus: creating a new-generation building energy simulation program. Energy and Buildings, 33(4), 319–331.

2. Loonen, R. C. G. M., Trčka, M., Cóstola, D., & Hensen, J. L. M. (2013). Climate adaptive building shells: State-of-the-art and future challenges. Renewable and Sustainable Energy Reviews, 25, 483–493.

3. Attia, S., Hamdy, M., O'Brien, W., & Carlucci, S. (2013). Assessing gaps and needs for integrating building performance optimization tools in net zero energy buildings design. Energy and Buildings, 60, 110–124.

4. Wei, T., Wang, Y., & Zhu, Q. (2017). Deep reinforcement learning for building HVAC control. In Proceedings of the 54th Annual Design Automation Conference (pp. 1–6). ACM.

5. Zhang, Z., Chong, A., Pan, Y., Zhang, C., & Lam, K. P. (2019). Whole building energy model for HVAC optimal control: A practical framework based on deep reinforcement learning. Energy and Buildings, 199, 472–490.

6. Azuatalam, D., Lee, W. L., de Nijs, F., & Liebman, A. (2020). Reinforcement learning for whole-building HVAC control and demand response. Energy and AI, 2, 100020.

7. Krish, S. (2011). A practical generative design method. Computer-Aided Design, 43(1), 88–100.

8. Tabasi, S. F., Luo, D., Rafizadeh, H. R., Le Nguyen, K., & Banihashemi, S. (2025). Generative optimization of building blocks for density, solar and structural performance. Journal of Building Engineering, 111, 113307.

9. Evins, R. (2013). A review of computational optimisation methods applied to sustainable building design. Renewable and Sustainable Energy Reviews, 22, 230–245.

10. Machairas, V., Tsangrassoulis, A., & Axarli, K. (2014). Algorithms for optimization of building design: A review. Renewable and Sustainable Energy Reviews, 31, 101–112.

11. Tian, W., Heo, Y., de Wilde, P., Li, Z., Yan, D., Park, C. S., & Augenbroe, G. (2018). A review of uncertainty analysis in building energy assessment. Renewable and Sustainable Energy Reviews, 93, 285–301.

12. Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). MIT Press.

13. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv. https://arxiv.org/abs/1707.06347

14. Dulac-Arnold, G., Mankowitz, D., & Hester, T. (2019). Challenges of real-world reinforcement learning. arXiv. https://arxiv.org/abs/1904.12901

15. Negendahl, K. (2015). Building performance simulation in the early design stage: An introduction to integrated dynamic models. Automation in Construction, 54, 39–53.

16. Ostergård, T., Jensen, R. L., & Maagaard, S. E. (2016). Building simulations supporting decision making in early design – A review. Renewable and Sustainable Energy Reviews, 61, 187–201.

17. Reinhart, C. F., & Cerezo, C. (2016). Urban building energy modeling – A review of a nascent field. Building and Environment, 97, 196–202.

18. Nguyen, A. T., Reiter, S., & Rigo, P. (2014). A review on simulation-based optimization methods applied to building performance analysis. Applied Energy, 113, 1043–1058.