Automated floorplan generation in architectural design: A review of methods and applications

Ramon Elias Weber*[a,b], Caitlin Mueller[a], Christoph Reinhart[b]

[a] Digital Structures Group, Department of Architecture and Urban Planning, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA, *reweber@mit.edu

[b] Sustainable Design Lab, Department of Architecture and Urban Planning, Massachusetts Institute of Technology

1. Abstract

Accommodating predicted population growth and urbanization within the UN Climate Goals poses a significant challenge for disciplines that engage with the built environment. High performing buildings of the future should offer spatial quality for their users while utilizing resources as efficiently as possible for both construction and operation. In this review, we survey the value proposition of automatic floorplan layout generation methods and their opportunities for design guidance, feedback, and optimization in the creation of new buildings, in addition to applications for inventory characterization to survey existing housing stock and guide building policy and code. We divide existing methods into three categories: *bottom-up methods*, *top-down methods,* and *referential methods.* We explore advantages and challenges for each approach and propose a hybrid method for future building layout automation that utilizes a new set of metrics to create sustainable buildings of the future.

(141 words)

Keywords: Floorplans, automation, generative design, buildings, sustainability, layout optimization, machine learning, geometry

2. Introduction

The International Energy Agency (IEA) estimates that the global built floor area will increase by some 235 billion m^2 until 2050, to accommodate a growing population and rising standards of living [1]. The environmental, economic, and ethical implications of this prediction are momentous. At a time when humanity has around 580 Gt CO_2e left to burn to keep global warming below 1.5°C, the "plan" to double the size of the building sector in a single generation seems risky. While one may expect significant efficiency gains in terms of area-weighted resource use and construction costs vis-à-vis today's building practices, the crucial question is whether humanity really needs that indoor space?

The construction sector – and especially the high-performance design community – have long embraced computer-based performance analysis methods for embodied and operational energy use associated with construction materials and building use. However, space efficiency evaluations are far less common. Usually, there is a design brief provided to the architect that stipulates a certain amount of program including a set percentage for circulation and space conditioning equipment. The sum of these space uses adds up to an overall building volume that can then be explored via massing studies. The spatial relationship between a massing volume or a floorplan and the distribution of program is, of course, quite complex, ranging from desired adjacencies to minimal width or depth requirements. In terms of future reuse opportunity, the design team would ideally also like to know how amenable a

given floorplan is to adaptive reuse or which walls could be load bearing while supporting good daylighting etc.

This paper reviews previous automated floorplan layout generation methods and assesses their actual and potential application for architectural design, urban planning, and real-estate development. A floorplan layout creation method creates an architectural layout from a series of geometric constraints and/or programmatic requirements. It should already be noted here that thus far only experimental artistic architectural design practices [2,3] and the real-estate sector (Table 1) have been eager to embrace novel generative and artificial intelligence-based layout automation tools, whereas the architectural profession at large has expressed some reservation against algorithms whose perceived ultimate goal might be to replace the profession. The purpose of this paper is to clarify the capabilities and limitations of existing methods and envision how they could contribute towards the design of more elegant, effective, and flexible spaces on a scale ranging from individual floors and buildings to whole cities.

A number of reviews on computational layout automation have been conducted that included industrial facility layouts [4], focused on specific computational methods, such as evolutionary algorithms [5,6] or agent based methods [7]. Furthermore methods have been surveyed with a focus on methods optimizing for energy usage [8,9]. In this manuscript, we initially review which professions have thus far used automated space layout methods and for what purpose. We then introduce a new approach that productively combines these methods with a specific focus on early-stage architectural massing studies and existing building stock characterization.

3. Value proposition

The first automated floorplan generation methods were introduced close to half a century ago. The underlying motivation has changed over time and still varies significantly between different projects and tools today. This section provides an account of significant historic precedents followed by contemporary use cases, in which the authors see exciting, new applications of the underlying technologies.

Past and current approaches

Automatic space layout creation methods were introduced as artistic speculations on the future role of computers and artificial intelligence in architecture in the 1970s. Yona Friedman proposed the *Flatwriter* [10] to generate apartment layouts that would accommodate usage preferences of all neighbors in cooperative housing projects. Automating the layout creation process was seen as an enabler for participatory design. Cedric Price proposed the “Generator”: A reconfigurable voxel based spatial unit that could be reconfigured by visitors into different layouts [11]. Both pre-computational proposals envisioned floorplans designed by a modular kit of parts in a bottom-up process.

During the late 1970s, George Stiny started working on the analysis and reproduction of building layouts via so-called parametric shape grammars [12]. By understand the design and spatial qualities of existing buildings such as Palladian villas [13] or Frank Lloyd Wright’s prairie houses [14], the underlying patterns could be deployed to recreate buildings of the same type. Similarly, Christopher Alexander represented spatial relationships in traditional architectural floorplans via relational graphs and tree

structures [15]. These abstractions were essential first steps necessary for automating the generation of floorplans and continue to play a foundational role in contemporary computational approaches.

Grammar based design methods have been successfully implemented into computational workflows to procedurally compute building volumes with detailed facades. Notably Esri's City Engine [16] which integrates the CGA++ shape grammar language [17] that enables the generation of differentiated building envelopes for visualization purposes of urban design proposals.

As of today, automated building-level layout tools have not made much headway into mainstream architectural practice where, their use is mainly reserved for speculative design exercises or specific niche applications such as office furnishing and electric lighting layouts in interior design [18] or complex programming exercised for hospitals, airports or large scale residential and commercial developments [19–21]. For such applications, automatically generated design options can augment or replace conventional manual design processes by offering not a single optimal solution but a family of directions for further design exploration [22].

In contrast to architectural design, the real estate sector has enthusiastically embraced and supported the creation of a plethora of floorplan and building automation software and practice. Several companies of varying size now focus on the creation of automatic layout tools to assist property developers and decision makers. They promise to maximize the potential buildable area and perform automatic analysis of a site, creating semi-automatic feasibility studies that can inform investment opportunities for land acquisition and maximize rentable area. Whether geared towards the real estate industry or conceived as in-house software tools in architecture and engineering firms, most current approaches tackle layout automation on the scale of a single building massing. They include different apartment (or in the case of hospitality, hotel room) mixes and simplified core placements with single or double loaded corridors. Table 1 provides a snapshot of prominent automated space layout creation method at the time of writing.

Table 1: Representative sample of contemporary automatic space layout creation methods in practice.

Typology	Scale	Output	Client	Use case	Company Product	Name	Citation
Residential, Commercial, Mixed use	(L) Multiple buildings	Massing, architectural program	Real estate, architects	Increase speed of design	Software	Archistar	[23]
Res, Com, Mixed	(L) Multiple buildings	Massing, floorplans	Real estate	Feasibility studies, real estate	Software	Testfit	[24]
Residential	(L) Multiple buildings	Massing, architectural Program	Architects, Real estate	N/A	Software	Matterlab	[25]
Res, Com, Mixed	(L) Multiple buildings	Massing, architectural Program	Architects, Real estate	Feasibility studies, design exploration	Software	Spacemaker	[26]
Residential	(L) Multiple buildings	Massing, architectural Program	Construction company	Modular construction	Software	KREO	[27]
Res, Com, Mixed	(L) Multiple buildings	Massing	Architects, Real estate	Sustainable design, feasibility studies	Software	Digital Blue Foam	[28]
Res, Com, Mixed	(L) Multiple buildings	Massing, architectural Program	Architects	Feasibility studies, design exploration	Software	Delve	[29]
Res	(S) Single floor	Architectural layout	Architects	Feasibility	Architecture	finch 3d	[30]
Educational	(M) Single building	Architectural layout from predefined building blocks	Community	Participatory design	Government	Seismic School App	[31]
Residential	(L) Multiple buildings	Massing, architectural Program	Community, architects, real estate	Participatory design, feasibility	Government	Prism App	[32]
Undefined	(S-L)	Building massing, lighting layout, window placement	Architects, engineers	Framework for automated design	Software	Project Refinery	[33]
Residential	(L) Multiple buildings	Massing, architectural program	Architects, government	Design exploration, community engagement	Software	Typenhaus+	[34]
Undefined	(L) Multiple buildings	Massing	In-House design tool	Design exploration	Architecture	Scout	[35]

Undefined	(L) Multiple buildings	Massing, architectural program	In-House design tool	Design exploration	Engineering	Site Solve	[36]
Res, Com, Mixed	(M) Single building	Massing, architectural program	Municipalities, real estate, architects	Design exploration, feasibility	Software	Omrt ostate	[37]
Res, Com, Mixed	(L) Multiple buildings	Massing	Homeowners, real estate	Land acquisition, real estate evaluation	Software	CityBldr	[38]
Hotel	(M) Single building	Massing, architectural program	Hospitality companies	Feasibility, early-stage planning	Software	Parafin	[39]
Res, Com, Mixed	(L) Multiple buildings	Massing, architectural program	In-house	Design exploration, feasibility	Architecture	Gensler Blox	[40]
Residential	(S) Single Apartment	Furnished architectural layout	Architects	Increase speed of design	Software	PlanFinder	[41]

Future applications

We note that there is currently no technical implementation of automatic space layout creation available that can fully replace a skilled human designer. More importantly, given the crucial role that architecture can play to provide more socially equitable and environmentally responsible spaces, we do not believe that the end goal of automated floor-plan generator methods should be to replace the human architect. Instead, we have identified three broad use cases that go beyond efficiency maximization and rather focus on improving the quality of the design process and resulting architecture. These use cases are design feedback, design guidance/optimization, and inventory characterization.

3.1. Use Case 1 - Design Feedback

Building massing decisions can have a significant and hard to predict impact on resulting interior space layouts, which in turn have cascading effects on building occupancy, structural efficiency, and even operational energy use. Fast simulations and generative design tools can help designers develop their own intuition for such relationships. In structural engineering education for architects and engineers, real-time simulations have become a useful tool to visualize problems and help designers build a geometric intuition to create more efficient structures [42,43]. Real-time visualization of the impact of design decisions can be useful to convey information to decision makers. When combined with novel interfaces, non-expert stakeholders or the local community can be engaged and learn about the design process more easily [44].

Plugging in to existing design workflows, automatic layout generation could help to visualize how changes in a building's massing relates to constraints for circulation, program, or structural requirements, as illustrated in Figure 1. Showcasing how building cores must be dimensioned for a given floorplate and the influence of floor-plan typologies on the energy usage of a building [45] can give architects a more intuitive understanding in early design processes. Programmatic changes based on different lighting and thermal requirements can have a direct influence on a building's energy budget: Interactive approaches can give a more intuitive understanding of these requirements, leading to solutions that can negotiate between requirements of different stakeholders. Furthermore, materially integrated design processes can visualize how different construction methods and material systems have different constraints during the design process. A direct comparison and calculation of embodied carbon and achievable spans could help users find new more sustainable design solutions [46].

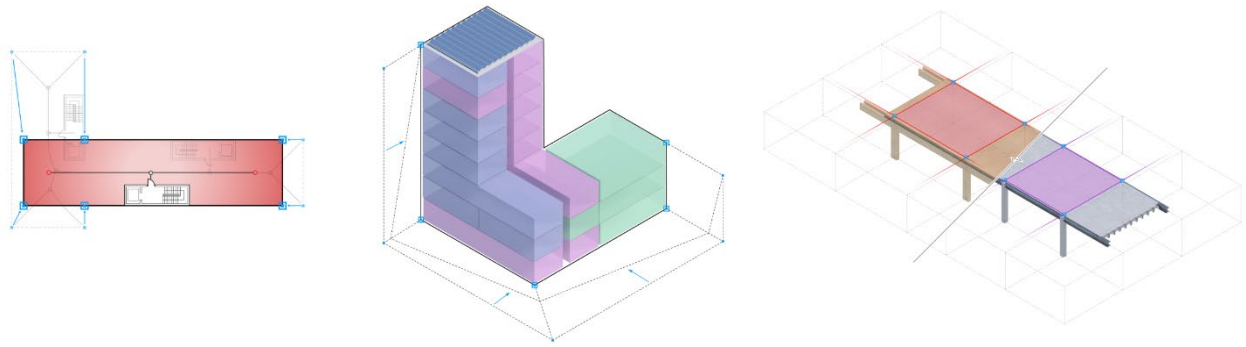

Figure 1: Pedagogical use of generative design tools to help build design intuition for circulation (left), program and energy (middle) and material-based constraints (right).

3.2. Use Case 2 - Design Guidance or Optimization

Generative layout tools can be used to augment different stages of existing digital design workflows. Parametric design spaces can be explored for optimization within predefined constraints [47], and grammar- and aggregation-based automated methods have been used to create new types of modular structures [48]. As speculative and early-stage design tools, automated approaches offer the opportunity to test ideas at scale and generate design options iteratively that would be difficult to achieve with manual workflows (Figure 2).

With highly specific programmatic requirements in specialty typologies, such as hospitals or airports, automated layout methods can help designers to optimize floorplans with adjacency, pathfinding, energetic or daylight heuristics [8,19], or structural system efficiency [49]. Multi-objective optimization and objective functions that are highly specific to the specified architectural problem can be used to negotiate between different (competing or diverging) goals. A series of experimental hybrid semi-automated methods have been deployed in such design processes where physics-based simulations can be steered by a user to inform programmatic distribution of layouts [50].

Referential automated methods can be deployed in later stages of building design. Leveraging architectural catalogues and previously generated designs, methods of automation can reuse and adapt established design solutions for new problems. This has been successfully demonstrated on a material scale where algorithmic workflows can identify closest fit solutions in existing material catalogues [51] as well as for adaption of existing floorplan layouts into new building massing [52].

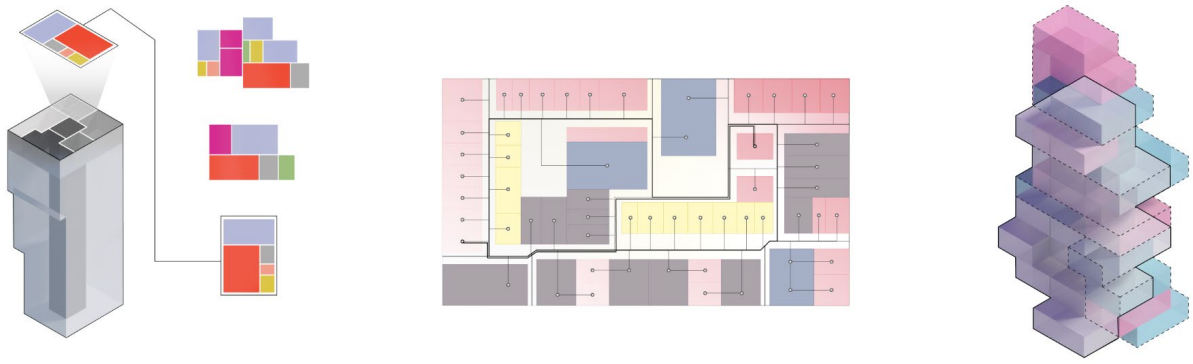

Figure 2: Use of automatic methods as design tool for automatically populating building massing (left), optimizing existing building layouts (middle) and the creation of novel design options (right)

3.3. Use Case 3 – Inventory Characterization

Automatic layout design tools not only offer opportunities for new buildings, but could be used to characterize and redevelop existing urban environments. Making use of widely available geometric massing GIS datasets, existing building stock could be modeled on a building level when combined with automatic floorplan layouts. This could lead better and more detailed understanding of existing housing stock and its embodied material quantities [46].

A better and more visual understanding of future developments enabled by current zoning could lead to more informed decisions for housing policies and building laws and allow for non-experts from the broader public to engage with planning processes. A clear understanding of desired goals could lead to outcome or performance based zoning that can take metrics such as urban comfort, mobility, and daylighting in to account [53]. Identifying possibilities of reuse or densification on a large scale could empower lawmakers to better guide their cities development as depicted in Figure 3.

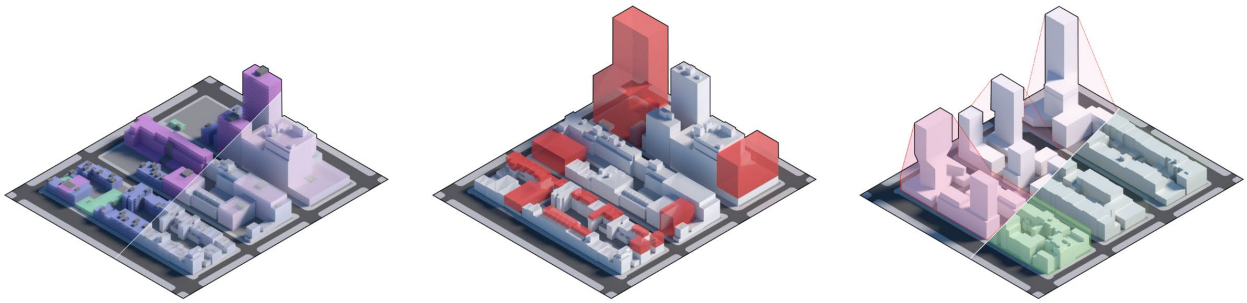

Figure 3: Opportunities of automatic space layout tools to be utilized for the survey of existing housing stock (left), to explore opportunities of densification (middle) and to identify implications of changes in zoning and building codes (right).

4. Methods

Following a description of possible use cases, this section reviews the computational methods underlying previously suggested approaches for automatic space layout generators. With origins in various engineering and computer science disciplines, many of the methods have been developed for

different use cases and have been adapted for building design workflows. This opens the field to new ideas and approaches for spatial design, but can also lead to a mismatch, where methodologies have inherent shortcomings that are difficult to adapt to the requirements of the architecture and planning disciplines. We divide existing approaches into three categories: *bottom-up methods*, *top-down methods* and *referential methods*. We outline the strengths and weaknesses of these categories vis-à-vis previously mentioned use cases.

4.1. Indexing and Search

Four main databases were used to retrieve research articles for this review paper: Web of Science [54], Google Scholar [55], Journals indexed in the Architecture and Civil Engineering disciplines from Scopus [56], as well as CumInCAD [57], a database of conferences and journals in the architectural computational design disciplines. In a second step we analyzed the references mentioned in the review and methods articles as well as tracking novel work that cited the relevant articles. A fully automated search and bibliometric analysis was not possible as floorplan and layout automation keywords are used in different disciplines for applications in electric circuit and factory layout planning and design. We identified 49 different methods with geometric architectural outputs of which 14 are bottom-up, 27 are top-down, and 8 are referential. Methods with architectural intent that did not result in a floorplan layout (e.g. only studied adjacency graphs in floorplans, or building massings) were excluded.

4.2. Bottom-up Methods

Architectural design briefs often have highly prescriptive spatial requirements. Because of heavily specified room sizes or adjacency constraints, layout designs often lend themselves well to be generated via bottom-up design processes. Bottom-up generator methods are therefore conceptually related to traditional design methods such as mind mapping of spatial relationships, bubble diagrams, and physical modelmaking strategies. When using modular construction logics that make use of prefabricated systems in concrete or timber [58] bottom-up aggregation logics enable the exploration of different design variations and part-whole relationships.

In a final structure, a series of predefined building blocks are aggregated into a larger assembly. As computational methods for the design of floorplans and building layouts, these aggregations can either be static (with predetermined architectural spaces) or adaptive (changing during computation) and can be coupled with heuristics to achieve a desired global outcome. Transformations during the aggregation process occur on the individual parts themselves.

During the bottom-up aggregation process, additional layers of information can be superimposed on the digital model to either change, swap out existing units or guide further aggregations. These heuristic methods can include analytic metrics such as spatial relationships (proximity requirements), environmental performance (daylight access, energy usage), structural efficiency metrics, or geometric details (proportions). A schematic of a bottom-up automatic design process for a series of rooms is described in Figure 4. Table 2 compares different approaches and implementations of bottom-up methods.

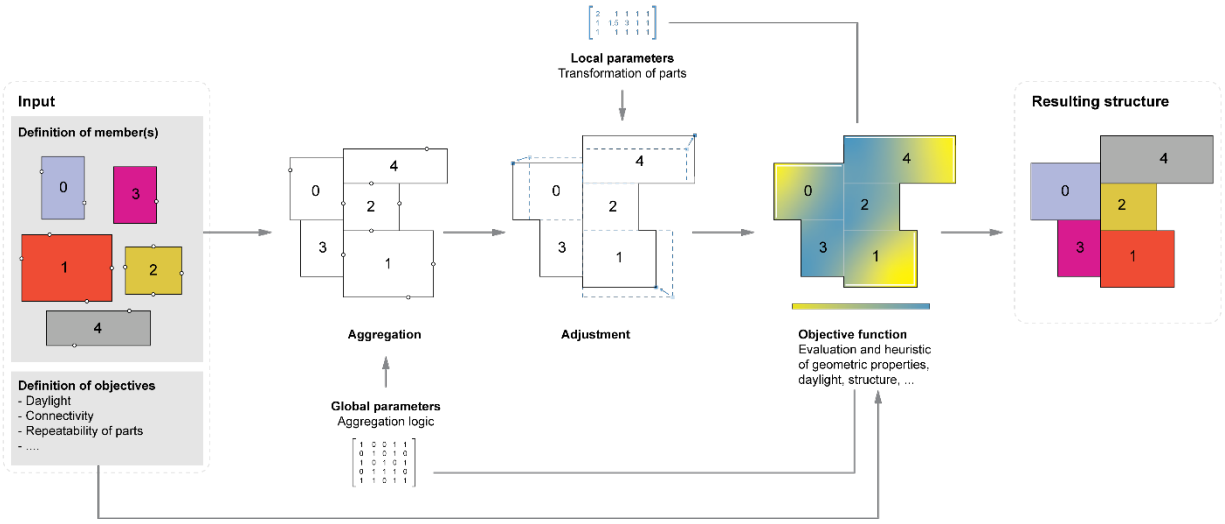

Figure 4: Schematic of bottom-up automatic space layout design methods. Starting with a definition of members and objectives, an initial aggregation and adjustment transforms the individual parts themselves. An objective function evaluates the outcome and drives local and global parameters to adjust the outcome.

Table 2: Comparison of bottom-up automatic space layout creation methods in literature. Genetic Algorithms (GA), Mathematical programming (MP)

Typology	Scale	Objective Function	Optimizer	Inputs	Output	Speed	Architectural Quality	Citation
N/A	(S) Single floor	Minimal wall length	GA	Number and areas of rooms	Floorplan, based on grid	N/A	Low	[59]
Residential	(S) Single floor	Maximize cross ventilation, (perimeter to area ratio) and minimize weighted sum of distances (closeness of rooms)	GA	Tree representation of program	Floorplan, differentiated rooms connected	N/A	Low	[60]
Public	(S) Single floor	Alignment, adjacency, orientation, proportion (of single rooms)	Physically Based	Area, adjacency	Modeling architectural design objectives in physically based space planning	N/A	Low	[61]
Residential	(S) Single floor	Minimize gap space.	Evolutionary algorithm	Area, location preference	Assigned program on existing layout, differentiated boundaries	N/A	Low	[62]
Residential	(M) Multiple floors	Connectivity, adjacency, envelope containment, convexity	1.Baysian network for Program generation, 2. Metropolis algorithm	Area, foootprint, aspect ratio, adjacency, adjacency type	Program layout	~ Seconds to 7 min	High	[63]
Residential, Office	(M) Multiple floors	Shading of neighboring building, occupied area, courtyard size	Quadratic programming, simulated annealing	Boundary, total floor area, # courtyards	Massing with specified floor area	~16 min	Medium	[64]
Office	(M) Multiple floors	Spatial configuration: semi-automatic methods for layout generation in practice	Physically Based	Area, adjacency	Program layout	N/A	Med	[50]
Residential	(M) Multiple floors	Daylight, predicted mean vote, shading	Simulated annealing	Programmatic units	Aggregation of modular programmatic units	4 min	Low	[65]
Residential	(M) Multiple floors	Maximize area in boundary, proximity and connectivity of program	Rectangular Voronoi Subdivision, Genetic Algorithm	Area, weighted adjacency matrix	Volumetric Arrangement of layout	12 min	Low	[66]
Residential	(S) Single floor	Adjacency, size	MP	600x400pixel raster image or vector graphic, area, adjacency	Layout on input image or vector graphic.	1.3-45.6 s	Medium	[67]

Residential	(S) Single floor	Connecting different room graphs to whole buildings	N/A	Programm graphs, layouts	Aggregation of multiple layouts	N/A	Medium	[68]
Residential	(M) Multiple floors	topology, room dimension and aspect ratio, building shape	Agent based	Program graph, area of rooms	Generated layout assigned to Grid voxel	~ Seconds	Low	[69]
Residential	(S) Single floor	Compactness, site boundaries, topology, user rating, circulation, privacy	GA	Areas, adjacency, window door or entrance requirement.	Program layout	6s – 7.3 h	Medium	[70]
Residential	(S) Single floor	None - Exploratory	Graph theory	Dimensional constraints, adjacency	Program layout	~ 1.5-2min	Medium	[71]

Bottom-up processes for exploratory and speculative design have been embraced by the design community to create discrete building systems that reintegrate design thinking with computational methods of design and means of production [72]. Applied to a building scale, they are particularly useful when designing for specific typologies that allow for modular construction and design logics in their realization. Members of a structure, the so-called discrete parts, can be aggregated to respond to specific architectural and spatial constraints or construction requirements, creating opportunities for robotic fabrication and reconfigurable structures [73].

As a response to the large search spaces of bottom up design processes, the Model Synthesis algorithm creates a set of custom constraints that guide the aggregation of user defined modules into complex 3D shapes [74]. Taking the adjacencies of parts of an existing 3D shape as an input, the method can generate new variations of larger dimensions that satisfy the original constraints. The method, also referred to as Wave Function Collapse (WFC), has since gained traction for creating 2D textures (using two dimensional pixel adjacencies) and 3D models for procedural level creation in computer games [75,76] as well as for modular design in the architectural domain [77]. To further guide the search towards solutions with controlled spatial qualities, machine learning (ML) guided heuristics have been proposed [78].

Methods of aggregation with large geometric freedom often create large search spaces that need clever heuristics to guide the exploration and output of good results. Additionally, stochastic methods do not necessarily find a solution, based on the problem settings. Furthermore, the bottom-up methods make it difficult to embed and control layers of hierarchy that are prevalent in architectural design such as different levels of circulation or structural load transfer that require differentiated building components or adjustments.

4.3. Top-Down Methods

Real-world architectural design is often highly constrained by predefined building massing that stems from urban scale considerations, building code, or regulations. This can result in highly prescriptive volumes that define the boundaries of a building that architects want to be fully occupied. When designing a building with such strong constraints on the envelope, defined through contextual requirements, site boundaries, or the reorganization of an existing structure, top-down design methods can be of interest. Methods for subdivision, fitting, shape packing, and iterative agent based methods have been applied across architectural scales to automate design problems (Figure 5), ranging from the material scale with optimal placements and dimensioning of shell components [79] to the layout and partitioning of geographical district scale [80].

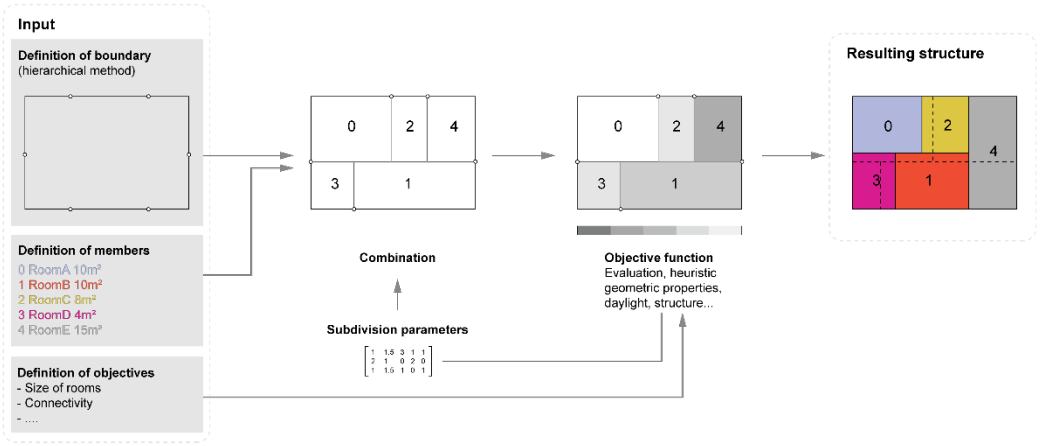

Figure 5: Schematic of top-down automatic space layout creation methods. Starting with a definition of boundary conditions, members and objective functions, an initial subdivision is evaluated by the objective function. Adjustment of the subdivision parameters results in a final structure.

Two promising technological inspirations and very active areas of research originate from the VSLI layout design and the Facility Layout Problem (FLP). Working with hierarchical systems that have interconnected rectangular modules, while integrating material constraints [81], the automation of VSLI circuits design has parallels to spatial layouts. As an optimization problem from the engineering community for arranging program in a given floor space, FLP is applied when machines in a factory hall for have to be laid out for a production line [82]. There has been significant interest in trying to transfer FLP methods to the architectural domain to optimize the placement of room layouts. However, current methods for solving FLP problems make use of highly abstracted mathematical models that are difficult to be transferred to real-world architectural environments and their implementation in available software tools on the market has been limited [83].

Top-down methods take a massing or boundary as an input, as well as a series of entities as fillers or targets for insertion. The input design is subdivided based on geometric constraints to assign spaces. Compared to the bottom-up method, the transformations are done on the global boundary conditions directly, resulting in a solution that will always conform to the initial boundary condition. An overview of different top-down methods is given in Table 3.

Heuristics can be used to assess the current state of subdivision and can inform next steps in the case of iterative optimization processes. This can be computed using mathematical programming, such as Mixed Integer Linear Programming [84], Squarified Treemap algorithms [85] or more geometry based approaches [22,86].

The top-down methods work best when used with fixed boundary constraints. Applied to building design in urban environments, the massing of a building is often predetermined (or highly constrained) by local building codes. In a first step, top-down approaches can be used to evaluate whether a certain boundary condition or building massing can be filled with a desired program or functional unit. To implement hierarchies, recursive subdivision methods that iterate over the resulting subspaces or programmatic clusters. Working on the end of a hierarchical system, the top-down methods are only able to cover a small, previously defined design space; in architectural practice that would mean that stand alone they are less useful for exploratory design stages where the boundary conditions (e.g. building massing) are not yet defined.

288 *Table 3: Comparison of top-down automatic design methods in literature. Abbreviations: Genetic Algorithms (GA).*
289 *Mathematical programming (MP)*

Typology	Scale	Objective Function	Optimizer	Inputs	Output	Speed	Architectural Quality	Citation
Office	(S) Single floor	Adjacency, minimize travel distance	Quadratic assignment	Areas, adjacency	Assigned program on existing layout	High	Very Low	[87]
Office	(M) Multiple floors	Adjacency	GA	Areas, adjacency	Assigned program on existing layout	High	Very Low	[88]
Office	(M) Multiple floors	Adjacency	GA	Areas, adjacency	Assigned program on existing layout	High	Very Low	[89]
Office	(M) Multiple floors	Adjacency	GA	Areas, adjacency	Assigned program on existing layout	High	Very Low	[90]
Hospital	(M) Multiple floors	Adjacency	GA	Areas, adjacency	Assigned program on existing layout	N/A	Low	[91]
Residential	(S) Single floor	Adjacency, room size	MP	Adjacency, area, min width/depth,	Assigned program on existing layout	N/A	Medium	[92]
Residential	(S) Single floor	Adjacency, room size	GA, MP	Areas, adjacency	Design topology (with adjacencies) (tree) and assigned program on existing layout	N/A	Medium	[93]
Residential	(S) Single floor	Adjacency, heating cost, lighting cost, spatial efficiency	GA	Program description (with min and max size), bounding box	Layout in bounding box	188 s	Medium	[94]
Residential	(S) Single floor	Custom fitness	GA	Areas, adjacency, proportions, building perimeter	Assigned program on existing layout (multiples of square foot units)	600s	Low	[95]
Residential	(M) Multiple floors	Adjacency	Stochastic Search	Adjacency, perimeter	Assigned program on existing layout	N/A	Low	[96]
Residential	(M) Multiple floors	Adjacency	GA	Connectivity, area, ratio	Assigned program on existing layout	N/A	Low	[97]
Residential	(S) Single floor	Practicality, originality, user input	GA, NSGA-II	Areas	Assigned program on existing layout	N/A	Low	[98]
Residential	(S) Single floor	Aspect Ratio, area	GA	Area	Assigned program on existing layout	N/A	Low	[99]
Residential	(S) Single floor	Areas	Squarified Treemap KD Tree	Areas, connectivity	Assigned program on existing layout	N/A	High	[85]
Residential	(S) Single floor	Areas, connectivity	GA	Connectivity	Assigned program on existing layout	N/A	Low	[100]
Residential	(S) Single floor	Areas, connectivity	GA	Connectivity, hierarchy	Assigned program on existing layout	N/A	Low	[101]
Residential	(M) Multiple floors	Areas, Connectivity, Material constraints	Non-linear least squares	Connectivity, Areas, Wall fabrication specification	Rooms inside boundary, precast concrete walls	2-3.5 s Seconds	Med	[102]
Residential	(S) Single floor	Areas, connectivity	GA	Connectivity, hierarchy	Assigned program on existing layout	N/A	Low	[103]
Residential	(M) Multiple floors	Adjacency, thermal performance	MP	Areas, connectivity	Assigned program on existing layout	N/A	High	[104]
Office	(M) Multiple floors	Gap spaces	MP	Room templates	Rooms tiles in existing grid	Minutes ~80s	Med	[105]
Trade Fair	(S) Single floor	Mobility, accessibility and coziness of agent-based crowd simulation	Stochastic, Simulated annealing	Agent behavior	Rooms inside boundary	2 - 7 Minutes	High	[106]
Office	(S) Single floor	Compactness	GA	Program description, ~47 geometric properties	Room tiles in existing grid	~520s Minutes	Low	[107]
Hospitals	(S) Single floor	Fitted program, view, travel	K-D Tree, Human evaluation	Program description,	Rooms inside boundary	N/A	Med	[19]

		distance, proportion						
Trade Fair	(S) Single floor	Congestion, exposure	GA, NSGA-II	Boundary, program description	Program distributed in boundary	5 days 20s per iteration	Med	[108]
Residential, office	(S) Single floor	Gap area	MIQP	Site boundary, program description,	Rooms inside boundary	~15 s Seconds	High	[84]
Generic	(S) Single floor	Orientation, adjacency, user selected subdivision grammar	Optimizer (N/A) + Reinforce-ment learning	Site boundary, program description	Rooms inside boundary	N/A	High	[109]
Generic	(S) Single floor	Visibility, Tree Depth, Entropy	Covariance Matrix Adaptation	Parametrized geometric model	Optimized wall layout	2.25 - 7.41 s Seconds	Med	[110]

290

291

4.4. Referential Methods

Learning from precedent has a rich tradition in architectural education and practice. Distributing design culture through “peer reviewed” publications of magazines and monographs (a publication describing the body of work of a single architect or architecture office) or through historical or topic specific anthologies and catalogues has analogies to the scientific community. Standardized reference works outlining basic architectural design strategies [111–114] are used for teaching the design of building layouts. In both professional and educational settings, they are used as reference books for dimensioning of standardized building elements, such as stairwells, circulation, escalators, or bathroom layouts.

With technological advances in computation and ML, there has been a renewed interest in referential automatic layout methods. A high-level overview of the referential method is given in Figure 6 and a comparison of different methods in literature in Table 4.

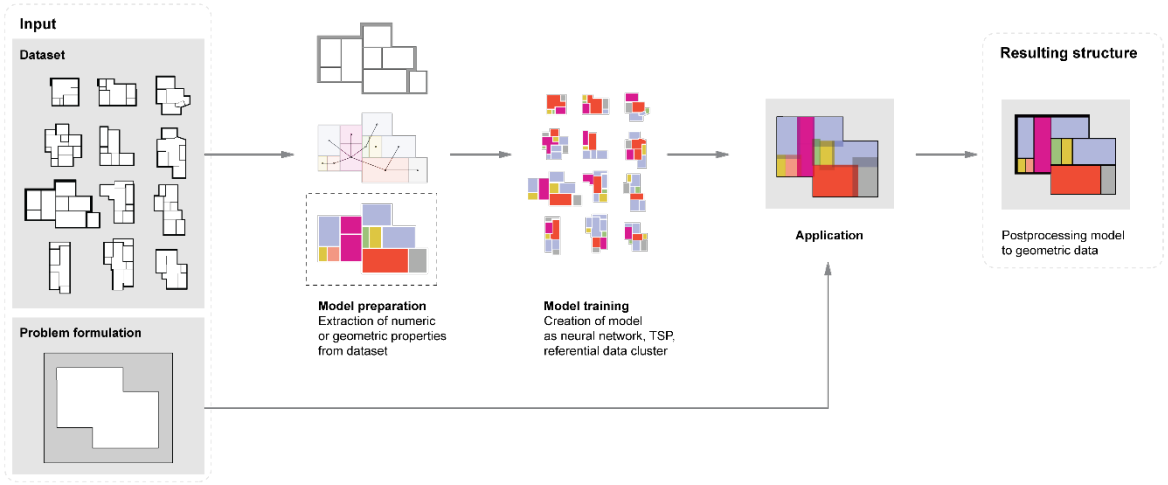

Figure 6: Schematic of referential automatic space layout creation methods. Starting with a dataset (catalogue) of existing structures. A desired property is extracted from the dataset and a model prepared and trained as Neural Net, TSP or referential data cluster. The model is applied to a user specified problem formulation and postprocessed into geometric data resulting in a final structure.

Table 4: Comparison of referential automatic layout design methods

Typology	Scale	Database	Reference Source	Matching	Inputs	Output (D) Direct (P) Post Processed	Speed	Architectural Quality	Citation
Residential	(S) Single Floor	101 single story houses [115]	256 × 256 pixel image, Color Coded	pix2pix NN via RunwayML	Boundary	Rooms color coded in boundary (D), manual tracing for vectors (P)	N/A	Low	[116]
Residential	(S) Single Floor	RPLAN [117]	256 × 256 pixel image, Color Coded	1. CNN for location of program 2. CNN for placement of walls	Entrance, apartment boundary	Wall map (D), vector representation of layout (P)	4 s (Seconds) (Generation) 7 Days (ML Training)	Medium	[117]
Commercial Residential Industrial	(S) Single Floor	700 annotated plans (Boston, USA, collected by author)	? x ? px Image, color coded	pix2pix NN	Boundary of building	Rooms color coded in boundary (D), manual tracing for vectors (P)	N/A	Low	[118]
Residential	(S) Single Floor	500 floorplans, undisclosed	Program graph	Bayesian model, scored adjacency graph	Apartment type	Program graph	N/A	(No architectural representation)	[119]
Residential	(S) Single Floor	RPLAN [117]	128 x 128 pixel image, color coded	1. GNN, CNN program distribution, 2. CNN floorplan image	Entrance, Apartment Boundary, Number/Type of rooms	128 × 128 floorplan image (D), vectorized floorplan (P)	0.4s (Seconds) (Generation)	Medium	[120]
Residential	(S) Single Floor	117,587 Layouts from Lifull [121]	256 x 256 pixel image, program graph	Conv-MPN	Bubble diagram, (Program graph)	Room masks (D), fitted rectangles as rooms (P)	N/A	Medium	[122]
Office	(M) Multiple Floors	120,000 volumetric designs (by authors)	Voxel graph, program graph	1. GNNs for the pro- gram graph 2. GNN voxel graph,	Program Graph, User input during generation.	Volumetric pixel grid representation of program	N/A (Generation) 20 minutes (with user interaction)	Low	[123]
Residential	(S) Single Floor	RPLAN [117]	RPlan images parsed as program graph	1. Relational GAN, 2. Conv-MPN	Program graph	Vector representation of layout	<0.4s ~Realtime (Generation)	High	[124]

Several algorithmic methods for referential design have been used, the most prominent are ML-algorithms with deep neural networks such as generative adversarial networks (GANs), as well as mathematical programming methods to find closest matches.

A series of databases have been ported to be used for generative or transfer purposes and converted to annotated images or graph structures. For the creation of functional relationships between programs [63] 120 commercial real estate plans for single family houses were encoded as graphs [125]. A Japanese real-estate image databases from the with 5.3 Million images [121] was ported for the use with ML algorithms [122]. To more effectively train neural networks, a series of residential floorplan datasets were manually collected and annotated by researchers resulting in RPLAN with 80,000 floorplans [117], Rent 3D with 215 floorplans [126] and CubiCasa5K with 5,000 floorplans from Finnish real-estate marketing material [127]. In industry, floorplan databases enable algorithmic lookup and reuse of floorplan drawings from previous work in the development of new layouts [41,52].

The combination of large image libraries of floorplans with GANs enabled the creation of programmatic infills into arbitrary floorplan shapes for apartment layouts [116] and allowed for the transfer of different historical architectural styles to apartment floorplans [118]. Recognizing the importance of hierarchies, strategies such as sequential infills (starting with the living room as high importance) [84] or additional graph networks that inform the generation [120] or training data [122,124] highly improve the plausibility of generated floorplans. Featuring online web interfaces, users can manipulate programmatic graphs while seeing a corresponding architectural layout in real time [120,123,124]. However, an emphasis is laid on connectivity of rooms and their sizes or boundary conditions could not be influenced.

The image-based machine learning methods, however, only work on very constrained boundaries and small scales, as all information has to be encoded in a 256x256 pixel image. Even though they can be very accurate inside of a specific domain and create diverse solutions, because of scale limitations, they have only been applied to single story residential apartment layouts so far. Furthermore, the fuzzy

outputs of image-based ML algorithms require significant postprocessing to recreate usable geometries, while using significant computational power and greatly varying in speed.

The strong dependence of the qualities of the outputs on good datasets, makes the lack of involvement of a diverse representation of the design community highly problematic. The large-scale datasets used so far in research are based on availability and have not been peer-reviewed or curated appropriately for architectural, spatial, or cultural qualities or environmental impact, creating unpredictable outputs.

5. Discussion

The previous sections summarize the substantial effort that has already gone into automated space layout generation with existing methods borrowing heavily from advanced computational design and machine learning approaches. It seems obvious that the real estate sector would embrace methods that can provide vital statistics on the marketability of a given massing, such as the number of housing units that can fit or the ratio of rentable to circulation areas. Given that an automated floorplan algorithm combined with a structural sizing tool can deliver a set of drawings that can, in principle, go through permitting and be constructed, it seems equally intuitive that many architects eye such methods with suspicion. The level of detail that such methods provide can create an impression of finality that one traditionally only encounters during later design stages. There is perceived real risk that architects further lose control of the design process at a time when only 2% of US homes are designed by licensed architects. Will that number fall even lower?

We find such thinking somewhat defensive. Rather than hanging on to the *last* 2%, should the profession not focus on the *lost* 98% by creating the best possible design in the most efficient manner? How can the disciplinary knowledge inform design automation to provide better quality and more resource efficient spaces and housing?

As generative methods can produce an infinite range of different solutions a variety of heuristics are used to classify promising solutions or guide optimization processes. This creates an opportunity to include building performance as a driver for design generation, extending the purely geometric objectives such as adjacencies, position, or aspect ratio. Validated methods for building energy simulation and natural ventilation with EnergyPlus [128], and daylight simulation using Radiance [129] have been integrated into layout automation workflows [8,104,130]. Metrics further expanded to include views [110] and agent based simulated of human behavior for both characterization and generation of new floorplans [106,131–133].

A big challenge in the creation of coherent layouts is the problem of scale. As programmatic requirements get more complex it becomes more difficult to coherent layouts that can integrate layers of hierarchy. This requires either a multi-step approach where programmatic units are clustered together and subdivided individually [84] or smaller units (such as a single apartment) are created on their own and then assembled as units into a larger buildings [130]. Hierarchical approaches have also been successfully implemented to inform ML models, where placing the living room first in the creation of apartment floorplans increased the quality of solutions [117].

To support the creation of new hybrid methods, it is important that spatial, environmental, and structural considerations can work in parallel and inform one another. We propose to expand the list of existing metrics to create workflows that can enable floorplan layouts supporting the creation of sustainable and high performing buildings (Table 5). These metrics can be tested at various scales from individual room to apartment, floor, or whole buildings for performance testing and optimization.

Table 5: A new set of holistic metrics to guide automated building layouts.

Spatial	Environmental	Structural
Modularity *Minimal change of the floorplan necessary to create different configurations while retaining same overall layout.*	**Daylight** *Provide access to daylight throughout the building, while minimizing direct solar radiation and glare.*	**Spans** Building layouts that work with minimal spans to reduce amounts of structural materials needed.
Compactness *Reduction of circulation to fit more in a building, while minimizing unused space.*	**Ventilation** *Layouts that promote natural ventilation (cross ventilation).*	**Continuity** *Layouts that stack loadbearing walls and enable optimal placement of shear walls to enable continuous carrying of loads.*
Adaptability *Creating layouts that enable flexibility of use by the inhabitants, creating rooms that can be used for different functions or layouts that enable different uses at the same time e.g. through shielding of noise.*	**Energy** *Minimization of building energy use by positioning and layering of less conditioned zones such as circulation to act as buffers to the conditioned spaces.*	**Material Integration** *Enable layouts that promote structural material systems with low embodied carbon and integrate fabrication constraints such as prefabricated timber modules.*

In addition to combining traditional floorplan generators with the above-mentioned performance workflows, we see three specific use cases where automated floorplan methods can enrich the current design process.

First, for typical urban infills, arguably the most sustainable and urgently needed building typology to accommodate a growing population, top-down methods provide a natural starting point since many massing parameters have already been set through zoning and setback requirements as well as clients' desire to maximize buildable area. There, a hybrid approach seems very promising, combining both top-down and bottom-up methods to negotiate between programmatic requirements and the urban context [130,134]. Referential methods be used to augment currently prevalent metrics to evaluate layout designs, such as daylight access, aspect ratios, or material quantities, verifying the design quality or offer alternative spatial layouts.

In the case of greenfield developments, bottom-up methods can be useful for quick design exploration by creating topologically different iterations. Material and construction constraints such as bay sizes, desired spans or prefabricated small-scale units can be integrated into the members to ensure solutions are feasible. Varying in resolution, the members of a bottom-up method do not have to be defined as single rooms but could be larger units or building parts, that can be refined or filled using top-down or referential approaches.

A third use case relates to building stock analysis. By applying floorplan generator to whole neighborhood massing models, existing urban analysis methods from daylighting to operational and embodied energy can be significantly refined since a floorplan help quantify the amount of material in a building, the likely number of occupants, and the location of internal walls that block daylight.

6. Conclusions

In this review, we have surveyed existing automatic floorplan layout creation methods in architectural design and have introduced a categorization into three methodologies. The bottom-up method proposes to work with a set of parts, such as rooms or preassembled units and to aggregate them into a larger structure. As an exploratory tool, it allows for the fast generation of different design options. Aggregation strategies can be further coupled with heuristics to guide the assembly. However, navigating often complex constraints or boundary conditions can be very challenging in the very large design space. There, top-down methods can offer an alternative, starting directly from geometric constraints, such as a building or site boundaries that get subdivided into smaller units. For this, different subdivision or packing strategies can be deployed. Third, referential methods are being investigated to make use of existing buildings and datasets. Geometric properties of existing or premade layouts can be fit or adapted to a new context. Fueled by recent advances in machine learning algorithms, spatial relationships have been captured as graphs or bitmap images and encoded into neural networks, enabling lookup and synthesis.

The further accessibility of machine learning algorithms and advancements of computational tools integrated into traditional geometric modeling environments used in architectural design could help bridge the interdisciplinary gap for architects to apply more domain specific knowledge. In our survey, we can show how floorplan layout automation is a dynamic field, both in terms of industry developing new tools, and business cases, as well as in academic research. We can see different disciplines engaging with the topic, ranging from architectural design research, civil engineering, building physics and technology, as well as computer graphics.

Showing the opportunities of hybrid approaches that go beyond purely spatial properties (e.g. proportions, areas or connectivity) to create believable floorplans, we see potential to further evaluate layouts based on environmental, and structural constraints that can serve the occupants. We propose the hybridization of the three methods, coupled with a new set of interdisciplinary metrics and performance indicators to guide future building layout automation. Working together in an iterative loop, the strengths of the different strategies can be applied at different points in the design process.

We show how automating building layouts can have a wide range of value propositions. Current use cases in the real estate industry can be expanded to create design tools that utilize automated floorplan layouts to give feedback about program, occupancy, or embodied carbon in the early stages of design. Using algorithmic and data driven solutions, they can optimize building layouts during the design process or explore creative solutions for new construction. Furthermore, they could lead to developing a better understanding of existing building stock or changes in building policy: empowering architects, urban designers, law makers and the public to make more informed decisions towards creating sustainable cities of the future.

7. Acknowledgment

This research was primarily sponsored by the Sustainable Design Lab and the Digital Structures group at MIT.

8. References

[1] M. Wörsdörfer, T. Gül, J. Dulac, T. Abergel, C. Delmastro, P. Janoska, K. Lane, A. Prag, Perspectives for the Clean Energy Transition – The Critical Role of Buildings, Paris, 2019. www.iea.org.

[2] M.C. Rehm, Complicit: The creation of and collaboration with intelligent machines, Archit. Des. 89 (2019) 94–101. https://doi.org/10.1002/ad.2417.

[3] M. Del Campo, S. Manninger, A. Carlson, Imaginary Plans the potential of 2D to 2D Style transfer in planning processes, Ubiquity Auton. - Pap. Proc. 39th Annu. Conf. Assoc. Comput. Aided Des. Archit. ACADIA 2019. (2019) 412–418. ISBN: 9780578591797.

[4] R.S. Liggett, Automated facilities layout: Past, present and future, Autom. Constr. 9 (2000) 197–215. https://doi.org/10.1016/S0926-5805(99)00005-9.

[5] V. Calixto, G. Celani, A literature review for space planning optimization using an evolutionary algorithm approach : 1992-2014, in: Siggradi, Blucher Design Proceedings, 2015. https://doi.org/10.5151/despro-sigradi2015-110166.

[6] K. Dutta, S. Sarthak, Architectural space planning using evolutionary computing approaches: a review, Artif. Intell. Rev. 36 (2011) 311. https://doi.org/10.1007/s10462-011-9217-y.

[7] J. Rhee, R. Krishnamurti, P. Veloso, J. Rhee, R. Krishnamurti, Multi-agent Space Planning A literature review (2008-2017), in: J.-H. Lee (Ed.), "Hello, Cult. 18th Int. Conf. CAAD Futur. 2019, Proc., Daejon, Republic of Korea, 2019. ISBN: 978-89-89453-05-5.

[8] T. Du, M. Turrin, S. Jansen, A. van den Dobbelsteen, J. Fang, Gaps and requirements for automatic generation of space layouts with optimised energy performance, Autom. Constr. 116 (2020) 103132. https://doi.org/10.1016/j.autcon.2020.103132.

[9] T. Du, S. Jansen, M. Turrin, A. van den Dobbelsteen, Effects of Architectural Space Layouts on Energy Performance: A Review, Sustainability. 12 (2020) 1829. https://doi.org/10.3390/su12051829.

[10] Y. Friedman, Flatwriter: choice by computer, Prog. Arch. 03 (1971) 89–101.

[11] T. Riley, S. Deyong, M. De Michelis, P. Apraxine, P. Antonelli, T. di Carlo, B. Cline, The Changing of the Avant-Garde, The Museum of Modern Art; D.A.P./Distributed Art Publishers, New York, 2002. ISBN: 0870700049.

[12] G. Stiny, Introduction to shape and shape grammars, Environ. Plan. B Plan. Des. 7 (1980) 343–351. https://doi.org/10.1068/b070343.

[13] G. Stiny, W.J. Mitchell, The Palladian grammar, Environ. Plan. B Plan. Des. 5 (1978) 5–18. https://doi.org/10.1068/b050005.

[14] H. Koning, J. Eizenberg, The Language of the Prairie: Frank Lloyd Wright's Prairie Houses, Environ. Plan. B Plan. Des. 8 (1981) 295–323. https://doi.org/10.1068/b080295.

[15] C. Alexander, A City is not a Tree, Parts 1 & 2, Archit. Forum. 122 (1965) 58–62, 58–61. ISBN: 978-0500275108.

[16] Esri, ArcGIS CityEngine, (2022). https://www.esri.com/en-us/arcgis/products/arcgis-

cityengine/overview (accessed May 15, 2022).

[17] M. Schwarz, P. Müller, Advanced procedural modeling of architecture, ACM Trans. Graph. 34 (2015) 1–12. https://doi.org/10.1145/2766956.

[18] A. Heuman, Innovator Spotlight: Part I - “Designing Design Tools,” (2020). https://www.youtube.com/watch?v=MicmKkkM_3o&t=1375s (accessed May 13, 2022).

[19] S. Das, C. Day, A. Hauck, J. Haymaker, D. Davis, Space plan generator, ACADIA 2016 Posthuman Front. Data, Des. Cogn. Mach. - Proc. 36th Annu. Conf. Assoc. Comput. Aided Des. Archit. (2016) 106–115. ISBN: 9780692770955.

[20] C. Derix, Mediating spatial phenomena through computational heuristics, Life Inf. Responsive Inf. Var. Archit. - Proc. 30th Annu. Conf. Assoc. Comput. Aided Des. Archit. ACADIA 2010. (2010) 61–66. ISBN: 9781450734714.

[21] E. Finucane, C. Derix, P. Coates, Evolving Urban Structures using Computational Optimisation, in: Proc. 9th Gener. Art Conf., 2006. http://www.generativeart.com/on/cic/papersGA2006/37.htm.

[22] L. Wilson, J. Danforth, C.C. Davila, D. Harvey, How to Generate a Thousand Master Plans: A Framework for Computational Urban Design, in: Proc. Symp. Simul. Archit. Urban Des., Society for Computer Simulation International, San Diego, CA, USA, 2019: pp. 113–120.

[23] B. Coorey, Archistar, (2020). https://archistar.ai/ (accessed May 13, 2022).

[24] C. Harness, R. Griege, Test Fit, (2019). https://testfit.io/ (accessed May 13, 2022).

[25] R. Gidei, M. Thorley, D. Flynn, Matterlab, (2019). https://www.matterlab.co/ (accessed May 13, 2022).

[26] H. Haukeland, C. Christensen, Spacemaker, (2016). https://www.spacemakerai.com/ (accessed May 13, 2022).

[27] KREO, Modular, (2020). https://www.kreo.net/ (accessed May 13, 2022).

[28] S.V. Patel, C. Weijenberg, Digital Blue Foam, (2022). https://www.digitalbluefoam.com/ (accessed May 13, 2022).

[29] Sidewalklabs, Delve, (2020). https://hello.delve.sidewalklabs.com/ (accessed May 13, 2022).

[30] J. Wallgren, Finch 3d, (2020). https://finch3d.com/ (accessed May 13, 2022).

[31] B. Wood, Seismic School App, (2019). https://seismic-school-app.io/ (accessed October 21, 2020).

[32] Mayor of London, Prism App, (2019). https://www.prism-app.io/ (accessed October 16, 2020).

[33] Autodesk, Project Refinery, (2020). https://www.autodesk.com/campaigns/refinery-beta (accessed October 21, 2020).

[34] M. Dennemark, M. Rudolph, B. Wannemacher, Form Follows You, (2020). https://formfollowsyou.com/typenhausplus/ (accessed May 13, 2022).

[35] Kohn Peterson Fox (KPF), Scout, (2020). https://scout.build/ (accessed October 30, 2020).

[36] Ramboll, SiteSolve, (2021). https://www.site-solve.co.uk/ (accessed May 16, 2022).

[37] A. Andrejevic, J. Spiegeler, B. Worms, omrt - ostate, (2021). https://www.omrt.tech/ (accessed May 13, 2022).

[38] B. Copley, D. Cairns, citybldr, (2021). https://www.citybldr.com/ (accessed May 16, 2022).

[39] A. Hengels, B. Ahmes, Parafin, (2021). http://parafin3d.com/ (accessed May 16, 2022).

[40] A. Pacheco, Gensler launches Blox, an algorithm-powered design visualization and computation tool, Archinect.Com. (2020). https://archinect.com/news/article/150202814/gensler-launches-blox-an-algorithm-powered-design-visualization-and-computation-tool (accessed May 13, 2022).

[41] J. van Lith, PlanFinder, (2022). www.planfinder.xyz (accessed May 16, 2022).

[42] R.G. Black, S. Duff, A model for teaching structures: Finite element analysis in architectural education, J. Archit. Educ. 48 (1994) 38–55. https://doi.org/10.1080/10464883.1994.10734621.

[43] T. Van Mele, L. Lachauer, M. Rippmann, P. Block, Geometry-based understanding of structures, J. Int. Assoc. Shell Spat. Struct. 53 (2012) 285–295. https://block.arch.ethz.ch/brg/files/2012-jiass-vanmele-lachauer-rippmann-block_1380094579.pdf.

[44] E. Ben-Joseph, H. Ishii, J. Underkoffler, B. Piper, L. Yeung, Urban simulation and the luminous planning table: Bridging the gap between the digital and the tangible, J. Plan. Educ. Res. 21 (2001) 196–203. https://doi.org/10.1177/0739456X0102100207.

[45] T. Dogan, E. Saratsis, C. Reinhart, The optimization potential of floor-plan typologies in early design energy modeling, in: 14th Int. Conf. IBPSA - Build. Simul. 2015, BS 2015, Conf. Proc., 2015: pp. 1853–1860. https://web.mit.edu/sustainabledesignlab/publications/BS2015_FloorPlanOptimisation.pdf.

[46] R.E. Weber, C. Mueller, C. Reinhart, Generative Structural Design for Embodied Carbon Estimation, in: Proc. IASS Annu. Symp. 2020/21 7th Int. Conf. Spat. Struct., 2021.

[47] N.C. Brown, V. Jusiega, C.T. Mueller, Implementing data-driven parametric building design with a flexible toolbox, Autom. Constr. (in press) (2020) 103252. https://doi.org/10.1016/j.autcon.2020.103252.

[48] O. Tessmann, A. Rossi, Geometry as Interface: Parametric and Combinatorial Topological Interlocking Assemblies, J. Appl. Mech. 86 (2019) 1–13. https://doi.org/10.1115/1.4044606.

[49] Y. Zhang, C. Mueller, Shear wall layout optimization for conceptual design of tall buildings, Eng. Struct. 140 (2017) 225–240. https://doi.org/10.1016/j.engstruct.2017.02.059.

[50] L. Helme, C. Derix, Spatial configuration: Semi-automatic methods for layout generation in practice, J. Sp. Syntax. 5 (2014) 35–49. http://joss.bartlett.ucl.ac.uk/journal/index.php/joss/article/view/201/pdf.

[51] F. Amtsberg, Y. Huang, D.J.M. Marshall, K.M. Gata, C. Mueller, Structural upcycling : Matching digital and natural geometry, in: Adv. Archit. Geom., Paris, 2020. http://web.mit.edu/yijiangh/www/papers/AAG2020_Structural_Upcycling.pdf.

[52] O. Green, An Introspective Approach to Apartment Design, in: DC I/O 2020, Design Computation Ltd., 2020. https://doi.org/10.47330/DCIO.2020.THHT2676.

[53] L. Wilson, J. Danforth, D. Harvey, N. Licalzi, Quantifying the urban experience: establishing criteria

for performance based zoning, Simul. Ser. 50 (2018) 237–244. https://doi.org/10.22360/simaud.2018.simaud.031.

[54] Clarivate, Web of Science, (2022). https://clarivate.com/webofsciencegroup/solutions/web-of-science/ (accessed April 11, 2022).

[55] Google, Google Scholar, (2022). https://scholar.google.com (accessed May 15, 2022).

[56] Elsevier, Scopus, (2022). https://www.scopus.com/ (accessed May 15, 2022).

[57] B. Martens, Z. Turk, T. Cerovsek, CumInCAD, (2016). http://papers.cumincad.org/ (accessed May 15, 2022).

[58] G. Staib, A. Dörrhöfer, M. Rosenthal, Components and Systems, Birkhäuser, 2008. https://doi.org/10.11129/detail.9783034615662.

[59] B.M.A. Rosenman, J.S. Gero, Evolving Designs by Generating Useful Complex Gene Structures, in: P. Bentley (Ed.), Evol. Des. by Comput., Morgan Kaufmann, San Francisco, CA, USA, 1999: pp. 345–364. http://pdf.aminer.org/000/743/586/evolving_design_genes_in_space_layout_planning_problems.pdf.

[60] M. ROSENMAN, Evolutionary Case-Based Design, in: C. Fonlupt, J.-K. Hao, E. Lutton, M. Schoenauer, E. Ronald (Eds.), Artif. Evol., Springer Berlin Heidelberg, Berlin, Heidelberg, 2000: pp. 53–72. https://doi.org/10.1017/S0890060400141022.

[61] S.A. Arvin, D.H. House, Modeling architectural design objectives in physically based space planning, Autom. Constr. 11 (2002) 213–225. https://doi.org/10.1016/S0926-5805(00)00099-6.

[62] M. Inoue, H. Takagi, Layout algorithm for an EC-based room layout planning support system, SMCia/08 - Proc. 2008 IEEE Conf. Soft Comput. Ind. Appl. (2008) 165–170. https://doi.org/10.1109/SMCIA.2008.5045954.

[63] P. Merrell, E. Schkufza, V. Koltun, Computer-Generated Residential Building Layouts, ACM Trans. Graph. 29 (2010) 1–12. https://doi.org/10.1145/1882261.1866203.

[64] F. Bao, D.M. Yan, N.J. Mitra, P. Wonka, Generating and exploring good building layouts, ACM Trans. Graph. 32 (2013). https://doi.org/10.1145/2461912.2461977.

[65] H. Yi, Y.K. Yi, Performance based architectural design optimization: Automated 3D space layout using simulated annealing, 2014 ASHRAE/IBPSA-USA Build. Simul. Conf. (2014) 292–299. http://www.scopus.com/inward/record.url?scp=84938862924&partnerID=8YFLogxK.

[66] I. Chatzikonstantinou, A 3-Dimensional Architectural Layout Generation Procedure for Optimization Applications : DC-RVD, Proc. 2014 ECAADe Conf. 1 (2014) 287–296. http://papers.cumincad.org/cgi-bin/works/paper/ecaade2014_163.

[67] H. Hua, Irregular architectural layout synthesis with graphical inputs, Autom. Constr. 72 (2016) 388–396. https://doi.org/10.1016/j.autcon.2016.09.009.

[68] B. Dillenburger, Raumindex. Ein datenbasiertes Entwurfsinstrument, ETH Zurich, 2016. https://doi.org/https://doi.org/10.3929/ethz-b-000161426.

[69] Z. Guo, B. Li, Evolutionary approach for spatial architecture layout design enhanced by an agent-

based topology finding system, Front. Archit. Res. 6 (2017) 53–62. https://doi.org/10.1016/j.foar.2016.11.003.

[70] A. Bahrehmand, T. Batard, R. Marques, A. Evans, J. Blat, Optimizing layout using spatial quality metrics and user preferences, Graph. Models. 93 (2017) 25–38. https://doi.org/10.1016/j.gmod.2017.08.003.

[71] S. Bisht, Transforming an Adjacency Graph into Dimensioned Floorplan, 0 (2022) 1–18. https://doi.org/10.1111/cgf.14451.

[72] J. Sanchez, Architecture for the Commons: Participatory Systems in the Age of Platforms, Routledge, 2020. ISBN: 9781138362369.

[73] G. Retsin, Toward Discrete Architecture: Automation takes Command, in: Ubiquity Auton. - Pap. Proc. 39th Annu. Conf. Assoc. Comput. Aided Des. Archit. ACADIA 2019, 2019: pp. 532–541. ISBN: 9780578591797.

[74] P. Merrell, D. Manocha, Model synthesis: A general procedural modeling algorithm, IEEE Trans. Vis. Comput. Graph. 17 (2011) 715–728. https://doi.org/10.1109/TVCG.2010.112.

[75] A. Newgas, Tessera: A Practical System for Extended WaveFunctionCollapse, Association for Computing Machinery, 2021. https://doi.org/10.1145/3472538.3472605.

[76] M. Gumin, Wave Function Collapse, (2021). https://github.com/mxgmn/WaveFunctionCollapse (accessed May 15, 2022).

[77] J. Tóth, J. Pernecký, MONOCEROS, (2021). https://monoceros.sub.digital/ (accessed May 16, 2022).

[78] T. Hosmer, P. Tigas, D. Reeves, Z. He, Spatial assembly with self-play reinforcement learning, in: Proc. 40th Annu. Conf. Assoc. Comput. Aided Des. Archit. Distrib. Prox. ACADIA 2020, 2020: pp. 382–393. ISBN: 9780578952130.

[79] T. Schwinn, A. Menges, Fabrication Agency: Landesgartenschau Exhibition Hall, Archit. Des. 85 (2015) 92–99. https://doi.org/10.1002/ad.1960.

[80] D. DeFord, M. Duchin, J. Solomon, Recombination: A Family of Markov Chains for Redistricting, Harvard Data Sci. Rev. (2021). https://doi.org/10.1162/99608f92.eb30390f.

[81] N.A. Sherwani, Algorithms for VLSI Physical Design Automation, Springer US, Boston, MA, 1993. https://doi.org/10.1007/978-1-4757-2219-2.

[82] P. Pérez-Gosende, J. Mula, M. Díaz-Madroñero, Facility layout planning. An extended literature review, Int. J. Prod. Res. 59 (2021) 3777–3816. https://doi.org/10.1080/00207543.2021.1897176.

[83] A. Drira, H. Pierreval, S. Hajri-Gabouj, Facility layout problems: A survey, Annu. Rev. Control. 31 (2007) 255–267. https://doi.org/10.1016/j.arcontrol.2007.04.001.

[84] W. Wu, L. Fan, L. Liu, P. Wonka, MIQP-based Layout Design for Building Interiors, Comput. Graph. Forum. 37 (2018) 511–521. https://doi.org/10.1111/cgf.13380.

[85] F. Marson, S.R. Musse, Automatic Real-Time Generation of Floor Plans Based on Squarified Treemaps Algorithm, Int. J. Comput. Games Technol. 2010 (2010) 624817. https://doi.org/10.1155/2010/624817.

[86] D. Nagy, L. Villaggi, D. Benjamin, Generative urban design: Integrating financial and energy goals for automated neighborhood layout, Simul. Ser. 50 (2018) 190–197. https://doi.org/10.22360/simaud.2018.simaud.025.

[87] R.S. Liggett, W.J. Mitchell, Optimal space planning in practice, Comput. Des. 13 (1981) 277–288. https://doi.org/10.1016/0010-4485(81)90317-1.

[88] J.H. Jo, J.S. Gero, Space layout planning using an evolutionary approach, Artif. Intell. Eng. 12 (1998) 149–162. https://doi.org/10.1016/S0954-1810(97)00037-X.

[89] J.S. Gero, V.A. Kazakov, Evolving design genes in space layout planning problems, Artif. Intell. Eng. 12 (1998) 163–176. https://doi.org/10.1016/S0954-1810(97)00022-8.

[90] R. Jagielski, J.S. Gero, A Genetic Programming Approach to the Space Layout Planning Problem, CAAD Futur. 1997. (1997) 875–884. https://doi.org/10.1007/978-94-011-5576-2_67.

[91] P.J. Bentley, The revolution of evolution in design: From coffee tables to hospitals, Proc. Recent Adv. Soft Comput. (1999). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.909.

[92] B. Medjdoub, B. Yannou, Separating topology and geometry in space planning, CAD Comput. Aided Des. 32 (2000) 39–61. https://doi.org/10.1016/S0010-4485(99)00084-6.

[93] J.J. Michalek, R. Choudhary, P.Y. Papalambros, Architectural layout design optimization, Eng. Optim. 34 (2002) 461–484. https://doi.org/10.1080/03052150214016.

[94] R. Baušys, I. Pankrašovaité, Optimization of architectural layout by the improved genetic algorithm, J. Civ. Eng. Manag. 11 (2005) 13–21. https://doi.org/10.3846/13923730.2005.9636328.

[95] H. Homayouni, A Genetic Algorithm Approach to Space Layout Planning Optimization, (2007) 137. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.114.7729.

[96] K. Terzidis, AutoPLAN, Z Jt. Study J. (2007) 84–87. http://ftp.formz.com/jointstudy/JS2008/11AutoPlan.pdf.

[97] A. Doulgerakis, Genetic Programming + Unfolding Embryology in Automated Layout Planning, University College London, 2007. http://eprints.ucl.ac.uk/4981/.

[98] A. Banerjee, J.C. Quiroz, S.J. Louis, A model of creative design using collaborative interactive genetic algorithms, Des. Comput. Cogn. '08 - Proc. 3rd Int. Conf. Des. Comput. Cogn. (2008) 397–416. https://doi.org/10.1007/978-1-4020-8728-8_21.

[99] M.K. Thakur, M. Kumari, M. Das, Architectural layout planning using Genetic Algorithms, Proc. - 2010 3rd IEEE Int. Conf. Comput. Sci. Inf. Technol. ICCSIT 2010. 4 (2010) 5–11. https://doi.org/10.1109/ICCSIT.2010.5565165.

[100] K. Knecht, R. Koenig, Evolutionäre Generierung von Grundriss-Layouts mithilfe von Unterteilungsalgorithmen, Arbeitspapiere Inform. Der Archit. 10 (2011). https://e-pub.uni-weimar.de/opus4/files/1653/InfAR_10_Layout_Unterteilung_pdfa.pdf.

[101] R. Koenig, S. Schneider, Hierarchical structuring of layout problems in an interactive evolutionary layout system, AIEDAM Artif. Intell. Eng. Des. Anal. Manuf. 26 (2012) 129–142. https://doi.org/10.1017/S0890060412000030.

[102] H. Liu, Y.L. Yang, S. Alhalawani, N.J. Mitra, Constraint-aware interior layout exploration for pre-cast concrete-based buildings, Vis. Comput. 29 (2013) 663–673. https://doi.org/10.1007/s00371-013-0825-1.

[103] R. Koenig, K. Knecht, Comparing two evolutionary algorithm based methods for layout generation: Dense packing versus subdivision, Artif. Intell. Eng. Des. Anal. Manuf. AIEDAM. 28 (2014) 285–299. https://doi.org/10.1017/S0890060414000237.

[104] E. Rodrigues, A. Rodrigues, Á. Gomes, Automated approach for design generation and thermal assessment of alternative floor plans, Energy Build. 81 (2014) 170–181. https://doi.org/10.1016/j.enbuild.2014.06.016.

[105] C.H. Peng, Y.L. Yang, P. Wonka, Computing layouts with deformable templates, ACM Trans. Graph. 33 (2014). https://doi.org/10.1145/2601097.2601164.

[106] T. Feng, L.-F. Yu, S. Yeung, K. Yin, K. Zhou, Crowd-driven mid-scale layout design, ACM Trans. Graph. 35 (2016) 1–14. https://doi.org/10.1145/2897824.2925894.

[107] I.G. Dino, An evolutionary approach for 3D architectural space layout design exploration, Autom. Constr. 69 (2016) 131–150. https://doi.org/10.1016/j.autcon.2016.05.020.

[108] L. Villaggi, D. Zhao, D. Benjamin, Beyond Heuristics: Novel Design Space Model for Generative Space Planning in Architecture, in: ACADIA, Discip. + Disrupt., 2017: pp. 436–445. http://papers.cumincad.org/cgi-bin/works/paper/acadia17_436.

[109] N. Saha, J. Haymaker, D. Shelden, Space Allocation Techniques (SAT), in: ACADIA 2020 - Distrib. Prox., 2020: pp. 248–257. http://papers.cumincad.org/cgi-bin/works/paper/acadia20_248.

[110] G. Berseth, B. Haworth, M. Usman, D. Schaumann, M. Khayatkhoei, M. Kapadia, P. Faloutsos, Interactive Architectural Design with Diverse Solution Exploration, IEEE Trans. Vis. Comput. Graph. 27 (2021) 111–124. https://doi.org/10.1109/TVCG.2019.2938961.

[111] E. Neufert, Bauentwurfslehre, Bauwelt-Verlag, 1936. ISBN: 978-3-658-34236-4.

[112] T. Jocher, S. Loch, Raumplilot Grundlagen, kraemerverlag, 2012. ISBN: 978-3-7828-1551-2.

[113] O. Heckmann, F. Schneider, E. Zapel, Floor Plan Manual Housing, Fifth, rev, Birkhäuser, 2018. https://doi.org/10.1515/9783035611496.

[114] B. Bielefeld, Spaces in Architecture - Areas, Distances, Dimensions, Birkhäuser, Basel, 2019. https://doi.org/10.1515/9783035619706.

[115] Zonda, House Plans, (2021). https://www.houseplans.com/ (accessed May 15, 2022).

[116] N. Peters, Enabling Alternative Architectures - Collaborative Frameworks for Participatory Design, Harvard, 2018. https://www.nathanpeters.us/enabling-alternative-architectures.

[117] W. Wu, X.-M. Fu, R. Tang, Y. Wang, Y.-H. Qi, L. Liu, Data-driven interior plan generation for residential buildings, ACM Trans. Graph. 38 (2019) 1–12. https://doi.org/10.1145/3355089.3356556.

[118] S. Chaillou, AI Architecture: Towards a New Approach, Harvard, 2019. https://doi.org/10.9783/9781949057027-006.

[119] J. Landes, H. Dissen, H. Fure, S. Chaillou, Architecture as a Graph - A Computational Approach,

(2020). https://www.academia.edu/42059688/Architecture_as_a_Graph_A_Computational_Approach (accessed May 15, 2022).

[120] R. Hu, Z. Huang, Y. Tang, O. van Kaick, H. Zhang, H. Huang, Graph2Plan: Learning Floorplan Generation from Layout Graphs, ACM Trans. Graph. 39 (2020) 1–14. https://doi.org/10.1145/3386569.3392391.

[121] Lifull, Home Dataset, (2015). https://www.nii.ac.jp/dsc/idr/lifull (accessed November 21, 2020).

[122] N. Nauata, K.-H. Chang, C.-Y. Cheng, G. Mori, Y. Furukawa, House-GAN: Relational Generative Adversarial Networks for Graph-constrained House Layout Generation, (2020). https://doi.org/10.48550/arXiv.2003.06988.

[123] K.-H. Chang, C.-Y. Cheng, J. Luo, S. Murata, M. Nourbakhsh, Y. Tsuji, Building-GAN: Graph-Conditioned Architectural Volumetric Design Generation, (2021). http://arxiv.org/abs/2104.13316.

[124] N. Nauata, S. Hosseini, K.-H. Chang, H. Chu, C.-Y. Cheng, Y. Furukawa, House-GAN++: Generative Adversarial Layout Refinement Networks, (2021). http://arxiv.org/abs/2103.02574.

[125] H. Wood, Essential House Plan Collection: 1500 Best Selling Home Plans, Home Planners, 2007. ISBN: 1931131708.

[126] C. Liu, A.G. Schwing, K. Kundu, R. Urtasun, S. Fidler, Rent3D: Floor-plan priors for monocular layout estimation, Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 07-12-June (2015) 3413–3421. https://doi.org/10.1109/CVPR.2015.7298963.

[127] A. Kalervo, J. Ylioinas, M. Häikiö, A. Karhu, J. Kannala, CubiCasa5K: A Dataset and an Improved Multi-Task Model for Floorplan Image Analysis, Lect. Notes Comput. Sci. 11482 LNCS (2019) 28–40. https://doi.org/10.1007/978-3-030-20205-7_3.

[128] D.B. Crawley, C.O. Pedersen, L.K. Lawrie, F.C. Winkelmann, Energy plus: Energy simulation program, ASHRAE J. 42 (2000) 49–56. https://gundog.lbl.gov/dirpubs/46002.pdf.

[129] R. Ward, G. and Shakespeare, Rendering with Radiance: The Art and Science of Lighting Visualization, 1998. ISBN: 0-9745381-0-8.

[130] E. Rodrigues, M.S. Fernandes, Á. Gomes, A.R. Gaspar, J.J. Costa, Performance-based design of multi-story buildings for a sustainable urban environment: A case study, Renew. Sustain. Energy Rev. 113 (2019) 109243. https://doi.org/10.1016/j.rser.2019.109243.

[131] V. Azizi, M. Usman, H. Zhou, P. Faloutsos, M. Kapadia, Graph-based generative representation learning of semantically and behaviorally augmented floorplans, Vis. Comput. (2021). https://doi.org/10.1007/s00371-021-02155-w.

[132] D. Schaumann, S. Moon, M. Usman, R. Goldstein, S. Breslav, A. Khan, P. Faloutsos, M. Kapadia, JOIN: an integrated platform for joint simulation of occupant-building interactions, Archit. Sci. Rev. 63 (2020) 339–350. https://doi.org/10.1080/00038628.2019.1662767.

[133] D. Nagy, L. Villaggi, J. Stoddart, D. Benjamin, The Buzz Metric: A Graph-based Method for Quantifying Productive Congestion in Generative Space Planning for Architecture, Technol. + Des. 1 (2017) 186–195. https://doi.org/10.1080/24751448.2017.1354617.

[134] E. Rodrigues, A.R. Amaral, A.R. Gaspar, Á. Gomes, An approach to urban quarter design using building generative design and thermal performance optimization, Energy Procedia. 78 (2015) 2899–2904. https://doi.org/10.1016/j.egypro.2015.11.662.