This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

Recent Advances in Procedural Generation of Buildings: From Diversity to Integration

Damian Kutzias and Sebastian von Mammen

***Abstract*—We survey the state-of-the-art in procedural generation of buildings (PGB). We analysed the design choices, capabilities and limitations of 71 PGB approaches. We structured our analysis based on a taxonomy of procedural content generation (PCG) that we adapted to PGB by adding relevant details. Overall, we identified 13 taxonomical aspects which we organised in three groups: 1) *Generative Capabilities*, 2) *Accessibility Aspects* and 3) *Technological Aspects* of the implementation. It is shown that the generation of different parts of buildings is easiest to accomplish with different PGB techniques. While some approaches built on other ones or even integrate two or more of them to arrive at solutions with a greater effective generative scope, all of them still exhibit rather basic limitations. For instance, the choice of room geometries might be limited to rectangular rooms, they might require numerous manual creation steps, or they might yield potentially invalid output. In addition, only few approaches implement fitting interiors and exteriors which can be used as complete, traversable buildings in virtual environments. Hence, the combination and therefore integration of different approaches for different parts of buildings is a major concern of procedural building generators, rendering their compatibility a characteristic of central importance.**

***Index Terms*—Procedural Buildings, Procedural Content Generation, PCG, Procedural Generation of Buildings, PGB**

I. Introduction

PCG can be defined as the algorithmic creation of game content with limited or indirect user input [1]. It has the potential to speed up development, decrease costs and increase the scale of virtual worlds with a proper level of detail with given resource restrictions. It is applied in a wide range of application areas, e. g. for urban understanding [2] or cultural heritage reconstruction [3]. PCG is used for numerous diverse purposes in game development: The content can range from 3D objects to abstract puzzles [4] including levels, weapons, personalities, quests etc. [5]. It can even include PCG as a game artefact in itself, featuring in-game content creation [6] such as the generation of new levels during play. PCG in games has been successfully applied to numerous popular games such as in Deep Rock Galactic (2020) for cave-system generation [7] or in Valheim (2021) for biome-based maps [8]. It is likely to grow in relevance both for assisting in the design and development of games as well as a game element by itself, e. g. for adapting games to particular demographics [9]. There are several reasons for game developers to use procedural methods. Popular reasons have been, for instance, maintaining a low footprint on computational load by providing content on demand, saving time and effort as compared to manually creating content, and fostering the generation of novel content, also by overcoming our own imagination [10]. Another reason is the replay value of games, experiencing the same games in a different way each time [11].

Buildings are an important category of assets to populate virtual worlds. The application of PGB in practice seems to be rare when it comes to traversable buildings, but is applied more commonly for building exteriors such as in Star Citizen [12]. Another indicator for the rare occurrences of PGB in games is, that the success stories of ArcGIS CityEngine, a commercial tool supporting PGB, only contain one game use case in November 2022: a destroyed city in a student project [13]. Depending on the use case, such as a specific type of game, buildings may only be part of the virtual stage designed to provide an atmospheric environment that a player is immersed into. In this case, of course, their exterior architecture is all that matters. Alternatively, buildings might themselves define individual environments that can be explored by the player. Their great diversity in functionality and design in combination with the need for potentially large numbers of buildings, e. g. in the context of open-world games, emphasise the need and potential benefit of PGB.

This article provides the results of a structured analysis of PGB approaches for use in virtual, interactive 3D environments. It presents the current state-of-the-art and takes steps to comprehensively and comparatively understand the individual approaches, best practices, and open challenges. Based on the gathered data, we recognise numerous highly specialised solutions and identify the need for their integration as an important next step in PGB, as hinted at by the title of this article.

The remainder of this article is structured as follows. Section II details the method that was followed in conducting the survey and the analysis of the identified relevant works. Section III introduces a taxonomy for the different aspects of buildings as well as their procedural generation. The taxonomical aspects and groups of aspects are used in Section IV to present and classify the identified PGB approaches. In addition, the results are analysed and discussed to yield insights about the development and challenges of PGB. Finally, a conclusion is provided in Section V.

II. Methodology

In order to achieve a comprehensive understanding of PGB approaches, best practices and open challenges, the following steps were taken.

First, we had to identify the various aspects according to which PGB approaches should be classified. To the best

D. Kutzias and S. von Mammen are with the Julius-Maximilians-Universität Würzburg (JMU), Sanderring 2, 97070 Würzburg, Germany. E-mails: {damian.kutzias,sebastian.von.mammen}@uni-wuerzburg.de

Manuscript received DATE, YEAR; revised DATE, YEAR.

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

of our knowledge, there exists no specialised taxonomy for distinguishing between the design and capabilities of PGB systems so far. However, we decided to start this investigation based on a generic taxonomy of PCG by Togelius et al. [14]. We adapted its distinct classifications with regard to validation and optimisation of PCG approaches.

Next, we identified relevant works based on an extensive online search for scientific literature. We started with search terms derived from surveys we got and added possibly relevant additional terms to our list during the whole process. We dropped several other search terms when realising that the search has converged (i. e. the same publications were found repeatedly) or no relevant publications at all. We consider the search terms which lead to publications for our list core search terms of which we have the following: "Procedural Building Generation", "Procedural Architecture", "Computer Generated Architecture", "Semantic Procedural Buildings", Procedural Interior Placement", "Procedural Content Generation for Buildings", "Procedural Stairs, "Procedural Roofs", "Procedural Interior Design" and "Procedural Furnishings". Each core search term was used in each of the following search engines and publication repositories: SciTePress Digital Library [15], arXiv [16], MDPI [17], Dimensions [18], Google Scholar [19], IEEE Xplore [20], ScienceDirect [21], emerald insight [22], ACM Digital Library [23], and SpringerLink [24].

Next, we identified patterns and alignments in publications we found on PGB. Based on these, we were able to propose an extended taxonomy for PGB which considers 13 aspects in total. It is detailed in Section III. While working on the taxonomy, we realised that PGB has quickly evolved within the last several years, had received little attention before, and all state-of-the-art publications we found have been published since 2006. Even though we had not restricted our search in any way, the included publications range from 2006 to 2022 for our taxonomy-based analysis. A few important publications prior to 2006 deserve special credit as important baseline research which is not part of the state-of-the-art and thus is handled separately at the beginning of Section IV. Despite of fulfilling our criteria for PGB systems, we also excluded several identified works. The first reason for this was missing novelty for PGB generators, e. g. by solely reporting about the application of an existing approach such as the usage of the software ArcGIS CityEngine. The second reason are cases where some authors were involved in several works with largely overlapping contributions. We embraced opportunities to only refer to works that covered their contributions as a whole.

For the following step, 71 publications remained which we analysed in accordance with the extended PGB taxonomy. Hence, we identified the necessary taxonomical aspects of the respective works and evaluated them in comparison to all the other works included in this study. This process of regression required numerous iterations to have the domains of the taxonomic aspects converge: the aspects were already fixed, but the possible set of values was adapted during this process. After each change in the possible values of one taxonomical aspect, all previously analysed publications were analysed again for the changed aspects. Since some aspects did not allow for quantifiable comparisons, we directly quoted the respective details in the tabular overview we composed (Table I). In case we decided on quantifying one or the other aspect, even though the taxonomic alignment was not immediately obvious, we added according justifications in the collection of surveyed works.

Tutenel et al. state that complete (virtual) buildings are those "consisting of not only a facade, but also interiors, stairs, furniture, etc." [25]. Leblanc et al. speak of "coherent interior and exterior" and give examples by referring to staircases and furniture layouts. Yet, the authors do not explicitly state the requirements a complete building would have to fulfil [26]. Surveying PCG, Smelik et al. distinguish between the exterior and interior of buildings and consider floor plan generation and furniture layout solving belonging each to either category [27].

Having these rough schemata of virtual buildings in mind, we analysed the selected PGB literature. We identified six recurring building constituents, which we consider necessary to reach a "complete building". Based on their descriptions, which can be found in Subsection III-A, one can determine which ones are addressed by a given PGB approach, and which ones are not (represented by opaque and transparent pictograms in Table I and Figure 10, respectively).

A more detailed separation of the constituents is possible, e. g. by individually considering doors, windows etc. We decided against it, since these construction elements are typically considered in the respective contextual generative tasks such as the generation of floor plans or outer walls. Also, numerous optional elements of a building such as chimneys could be addressed, but these are rare occurrences within the PGB literature and we focus on those components necessary for the generation of "simple buildings", analysing the inclusion of the previously mentioned six building constituents.

The goal of this survey is to cover **complete**, **simple** and **common** buildings for the game-domain. A game-focussed, justified definition of these terms could be subject to a survey on its own and, to the best of our knowledge, is missing in public literature. We fixed our pragmatic understanding based on literature occurrences, especially in surveys from the game domain and PCG. In addition to our previously given explanations for simple and complete, we are targeting buildings commonly found in human settlements as "common buildings". As a result, complex dungeon and cave systems typically designed for riddles, traps, or battles which in addition are often generated with different requirements by different algorithmic approaches such as cellular automata, are not subject to our analysis.

PGB is tightly linked to the procedural generation of cities. An often-cited survey work in this area from 2006 highlights that built environments can be an important factor for determining form, function and style of buildings, especially considering the available lot spaces and geometric features shared across a neighbourhood [28]. Yet, since the consideration of the spatial and functional relationships between sets of buildings as well as architectural and cultural influences can be seen as parameters and restrictions for PGB given by a city generator, we limited this survey to building generation. We further do not consider aesthetics of any PGB approaches, including, for instance,

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

different styles or visual attractiveness. Whereas such aesthetics are an important aspect of many game assets, they can hardly be captured by a literature-based survey and would require different research approaches such as user polls.

III. Taxonomy

As mentioned above, Togelius et al. introduced a taxonomy for PCG in general [14]. They proposed the following aspects adapted for our taxonomy: *automatic generation* vs. *mixed authorship* and *constructive* vs. *generate-and-test*. *Automatic generation* for PCG means that generation is done automatically, whereas *mixed authorship* implies involvement of humans throughout the design process. *Constructive* PCG means that content is generated with the constraints of the overall artefact in mind, allowing to yield usable results in just one pass. On the opposite end, *generate-and-test* means that content is generated in a loop and evaluated until an artefact has been produced that satisfies all constraints at once. We used these two cited taxonomic aspects as our starting point. They describe how a concrete generation process is utilised to arrive at usable artefacts and, thus, are applicable to all sorts of generative approaches. At the same time, they are closely linked to the concrete use cases of PCG (see, for instance, the goals of PGB as outlined in Section I).

Despite being important taxonomic aspects, the other taxonomic aspects of the general PCG taxonomy were not further explicitly considered in this survey. While they apply to PGB just as much as for any other PCG area, they only play a secondary role when comparing approaches with respect to their generative outcomes. In that sense, our extended PGB taxonomy can be understood as an **extension with slight modifications, not a replacement** of the taxonomy of Togelius' et al. The following exclusions and corresponding arguments are not about the general relevance of the taxonomic aspects. They are about their relevance for this survey and the included comparison of PGB generators.

Whether the content is generated during playtime (online) or not is highly dependent on hardware and use cases and none of the analysed solutions includes online generation based on player behaviours (excludes *online versus offline* and *generic versus adaptive*). Whether a building is necessary or not for game progression is a design element and not directly depending on the generator (excludes *necessary versus optional*). Some publications did not address the presence or absence of stochastic elements without explaining enough details to reliably classify this aspect. In addition, this aspect can usually be switched with little effort: a stochastic generator can be made deterministic by fixing a seed for the underlying random number generator and the other way around inputs and aspects of a deterministic generator can be randomised (excludes *stochastic versus deterministic*). *Degree and dimensions of control* is partially dependent on the *stochastic versus deterministic* which is not available. Some information about it are included in our aspects *Means of Control* and *Interaction Required*, though.

To specifically provide a taxonomy for PGB, we aimed for more details and structure by groups of aspects, i. e. specialised characteristics and generative capabilities, related to buildings. Without finding prior works to support this goal, we chose the generative capabilities of the PGB approaches as the primary taxonomic group. Generative capabilities target the different parts of a building and are described in detail in Section III-A. Next, we focus on the approaches' accessibility and usability in Section III-B. Finally, we investigate taxonomic aspects which provide the basis for a technical comparison of the considered approaches (Section III-C).

Most of the aspects of the proposed PGB taxonomy are directly comparable, i. e. the domains are either binary or categorial. Yet, some also contain free text information to provide a better impression of the analysed approaches in the overview. Note that several of the aspects can also be thought of as a continuum. They were grouped into categories and binaries for measurability. The presented form was chosen to make the survey more informative, thus it might not be the final and polished form of a taxonomy. We consider it as an elaborated and good basis for such a taxonomy and discuss measures to make a better taxonomy from the presented aspects after their explanation in the end of this section.

A. Generative Capabilities

The terminology we propose for the six *Generative Capabilities* below emerged from the goal to avoid ambiguities. Some approaches require user interaction. Such solutions were not excluded and instead the user interaction requirement is one aspect of this taxonomy (cf. Subsection III-B). For the *Generative Capabilities* this has the following effect: there is a fluent transition between design tools and procedural generators. For this taxonomy and the corresponding analysis, it was handled in the following way: manual generation steps are counted towards *Generative Capabilities* of the analysed generators as long as they are integrated with PCG steps, i. e. are part of the same generation-process. We made this decision for two reasons: 1) Many state-of-the-art PGB generators we found during our survey include user interaction and 2) depending on the required details, it is likely that artists will play an important role in addition to PCG aligning generators with use cases and complementing them with their skills. Note that deterministic generation (from input data) is counted as PCG and full user control is permitted.

Floor Plans () — Floor plan generation encompasses the processes of generation, arrangement and connection of rooms. Examples are provided by [29], [30], [31].

We consider this aspect fulfilled, if a PGB approach is capable of generating the outline and relative placement of more than one room for a single building. Note that it is possible to have a generator capable of generating *Floor Plans* but not *Outer Walls*, if the *Outer Walls* are a required input for the generator.

Outer Walls () — Outer walls or facades, as in [32], [33], [34], connect the foundation of a building with its different levels. Based on our survey, the term "facade" is more frequently used than "outer walls". However, it also linked to the processes of generation, extraction and UV-mapping of textures for outer walls such as in [35], [36], [37]. From a geometric perspective, facades can be generated based on data specifying the enclosing planes. As a common case, 2D

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

areas depicting the outer walls in floor plans are extruded into the third dimension, therefore generation of the outline of a building is counted as the generation of *Outer Walls*.

Texturing — The procedural generation of textures is a research field on its own providing vital benefits for interactive entertainment and computer games [38]. Input data such as parameters, random values or existing images are processed and combined to generate new textures [39]. For example, recent works used semantic descriptions [40] or real-world examples with convolutional neural networks [41] for texture generation. Despite its general relevance, we have rarely found PGB approaches with other generative capabilities in combination with the generation of textures, besides the previously mentioned example on facades. Instead, existing textures are often applied during generation such as in [42], [43], [44].

A PGB approach is considered capable of handling textures, if it generates building textures or applies given textures to at least one of the five building constituents corresponding to the other five *Generative Capabilities*. Note that pictures with textures are not taken into account if textures are not mentioned in the text, since textures may have been applied by hand for presentation or have been already applied, e. g. for placed furniture.

Roofs — Roofs have been the sole generative targets of PGB approaches as in [45] or alongside outer wall generation as demonstrated in [46].

We consider PGB approaches capable of handling roofs which generate anything more elaborate than flat rooftops, i. e. created by means of planar triangulation.

Stairs — The positioning of stairs in a multilevel building is a challenging task. Accessible areas, floors and openings at different levels have to be carefully set and aligned, and the flight of stairs itself has to be generated accordingly [34].

A PGB approach is considered capable of handling stairs, if it can generate or select and place a flight of stairs to provide an effective functional geometry for transitioning between two or more floors of a building.

Furnishings — As stated by all three basic building definitions quoted above, furnishings are part of complete buildings (cf. Section II). Some PGB approaches make use of agents optimising the placement by moving around, e. g. in [47]. Others optimise global cost functions quantifying the quality of arrangements, as in [48].

We consider a PGB approach capable of handling furnishings, if it can place instances of at least one type of furniture.

B. Accessibility Aspects

Generative modelling is related to 3D modelling and programming [49]. Such software historically had issues with usability and productivity [50]. The acceptance of procedural modelling has been persistently hindered by the lack of intuitive controls and disparity of isolated techniques that generate only specialised features [27].

To this end, we introduce the group of *Accessibility Aspects* comprising the following three aspects of PGB approaches that impact dependencies, applicability, and thus, accessibility and usability.

Means of Control — PGB approaches are often difficult to control [51]. They often require the definition or adaptation of non-intuitive rules and input parameters. This may result in steep learning curves for mastering the necessary, underlying technology [27]. They often also rely on statistics or effects of structural emergence, both of which may result in effects that are difficult to anticipate. Many current procedural approaches utilise a special purpose language to encode the parameters and behaviours of the generative procedures [50]. In contrast, a Graphical User Interface (GUI) can facilitate the work by visualising the intricacies of a specific approach. For instance, it might highlight the configurable flow of data, provide instant feedback on possibly erroneous input data, or give real-time feedback on the effect of various introduced changes [52].

Considering these diverse challenges at the application level, we address the corresponding skill requirements of a PGB approach. To this end, we introduce the *Means of Control* which covers a range of five values:

- *GUI* means that a GUI is provided for applying the respective PGB approach, e. g. working with a multi-window editor as in [53].
- *Special Purpose Language* means that users have to learn and use a special purpose language, such as the selection-expression commands of the procedural modelling language SelEx as in [54].
- *Formal Configuration* means that users have to design formal rules, encoded, for instance, in logical expressions, or mathematical constraints as in [55].
- *Parametric Configuration* means that the core task of the user is to tweak parameters of the generator, e. g. setting parameters such as building width, length, shape etc. for model instantiation as in [34].
- *Data Provision* means that the users have to generate or collect data such as building images, existing floor plans or sensor-based point clouds. For instance, [56] derives grammatical rules from sensor data to generate floor plans.

Most approaches require several, maybe even all of these *Means of Control*. We made this choice based on the effects regarding skill requirements and level of control over the final outcome. For example, in [57], users define building profiles in an interactive editor, but can also set a few parameters such as a width for repetitive elements. We considered interactive visual modelling to be more decisive for users tasks than setting simple parameters. We know that this cannot be fully objective for general publications, but consider it helpful, especially for comparison.

Interaction Required — According to Smith [58], the interaction type is one of the mechanical aspects of PCG and can range from no human interaction (i. e. *None*), as seen, for instance in automatic furniture layout generation by means of a genetic algorithm [59], to human input to actively altering the generated content (i. e. *Direct Manipulation*), as in [53].

Although in her elaborations, Smith focussed on gameplay, she pointed out that the classification of interaction is equally applicable to a potentially independent design process. While we express the type of interaction and control in the *Means* *of Control* above, we now introduce an aspect that captures the need for interaction during the actual generative process. Configuration or programming tasks that take place before the generative process itself, as well as optional opportunities to interfere with contents or the process during generation are not taken into account. Considering any required interactions is all the more important due to the general need for automation as one of the main motivations for PCG—to overcome resource limitations, to create infinite worlds, or to increase replay value [60], [9], [61]. With this aspect we show whether interaction is required or not as binary information.

Based on External Data — By external data we refer to data that is provided to and processed by a PCG approach as, e. g., described in [27], [58]. Providing some given data may simplify the overall process in terms of amount, adaptation, and diversity of the generation efforts. Yet, the availability of the respective data sets, in sufficient type, number, and quality, needs to be warranted in the first place [55]. Some approaches might only require external data once, e. g. to train a Bayesian network one time and to use it repeatedly in subsequent generative steps [30]. Other approaches require new data for each execution, for instance for creating architectural geometry from images as in [62]. Although there may be edge cases, we do not consider expert knowledge of system developers or users as external data. These deliberations result in three possible values:

- *Execution* means that external data is required for each run and thus for every generated artefact or set of artefacts.
- *Initial* means that external data is initially required for configuration or learning.
- *None* means that no external data is required at all.

C. Technological Aspects

In general, it is difficult to choose the right technology due to the increasing number of alternatives and complexity [63]. When creating procedural generators, the central choice concerns the selection of data structures and technological approach, i. e. types of algorithms such as formal grammar processing or procedural growth. Depending on the underlying representation, certain modelling operations may be difficult or impossible to implement [50]. Addressing the crucial technological aspects of generators, this group of aspects contains information about the technical details of the evaluated PGB approaches. Our survey includes the following four aspects within this group:

Shape Description — According to Havemann, none of the existing shape description methods is entirely satisfactory. He also roughly divides the methods in two classes. The first class follows the “list of primitives” approach to describe three-dimensional objects and scenes. Compositions of elementary objects such as points, triangles, cubes, spheres and many others are used. The second class uses procedural shape representations including formal grammars, shape programming languages, physically based simulations and others [64, p. 1]. These classes can occur together: A formal grammar that places given primitives instead of generating everything from scratch combines elements of both classes. Considering these challenges in addition to the restricted scope of our analysis (focussing on PGB), our approach was to collect the methods used in the analysed publications and manually group them by similarity. This allows us to provide detailed information about the shape description methods used for the PGB without having to describe all possibilities in general.

With this technological aspect we provide categorial information about the geometric representation handled by the system with six possible values:

- *Points and Vectors* means that points and vectors are manipulated. For example, existing objects can be placed or polygons can be modified by procedural growth algorithms as used in [34].
- *Polygonal Surfaces* means that geometric shapes are processed as polygonal surfaces and altered by geometric operations such as splitting or Boolean intersections. This was done with formal grammars using convex polyhedra to generate buildings [65].
- *Parametric Representations* means that folded or encoded parametric content such as geometric functions is used for generation. For example, spline curve mesh generation can be used to construct complex roofs [66].
- *Grid-Cells / Voxels* means that occupation matrices are processed either in 2D or 3D such as using grid cells for room growth as in [31].
- *Predefined Building Blocks* means that prepared assets are arranged and properly combined. van Aanholt et al. arrange tiles (e. g. stairs and wall fragments) by a solver according to their properties [55].
- *Pixels / Images* means that images or pixels of images are processed for generation. For example, facade reconstruction can be done based on real world images [35].

Note again that several of these values occur in many of the approaches and we picked the one seeming to be the most prominent one regarding the representation of the final geometry. For example, in [66], spline curves for geometry and component instantiation (based on the spline curves) are used. For the classification, *Parametric Representations* was chosen to be more important than component instantiation for the procedural generation, since the latter is more restrictive and dependent on the first one. We know that this cannot be fully objective for general publications, but consider it helpful, especially for comparison.

Data Structures — The selection of data structures not only influences the computational efficiency, but also the ease of implementation by supporting or disallowing different basic operations. For example, tree structures grown through grammatical production can provide semantic, contextual information such as groupings and compositions, and support attribute management by inheritance. In contrast, octrees are typically used to capture and describe spatial relationships, allowing for geometrical selections and operations. One of these operations is efficient solving of the interference problem, i. e. the detection of different objects occupying the same section in space [67].

By means of this aspect, we provide free text information about the central data structures of PGB approaches addressed in the corresponding publication.

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

Core Technique — Different fundamental approaches of PCG determine many aspects of the generators' application as well as the result space. For procedural buildings, Schwarz and Müller have argued that grammar-based approaches are most common [46]. Grammars can easily express hierarchical dependencies which often occur in human-made architecture and foster repeated structures. In addition, grammars have several known challenges and limitations: Dependencies can be unclear and grammatical rules can be cumbersome to handle [68], (non-repeating) details can be difficult to implement [66] and different derivation branches are hard to coordinate [54]. Several other approaches also occur in PGB such as (re-)construction based on images or point clouds, interactive sketching or arrangement, procedural growth algorithms, agent-based placements and machine learning-based approaches. All these approaches have their own advantages and disadvantages, making the *Core Technique* a distinguishing technological aspect. While the type of application and also the user skill requirements are addressed by the accessibility aspect's *Means of Control* of our taxonomy, they are also affected by the *Core Technique*. As an example, a formal grammar requires the definition of production rules in an according syntax. If one tried to keep this chore from the user, the rules would have to be generated with different degrees of *Interactions Required*.

The *Core Technique* aspect captures categorial information about the core principle used in the PGB approach with four possible values:

- *Substitution* means that details of the buildings are defined by (hierarchical) substitutions. These are usually found in approaches based on L-systems (parallel rewriting systems motivated by cellular growth originating from 1968 [69]) or shape grammars (sequential rewriting systems using shapes in form of polygonal surfaces as geometric representations conceived by Stiny in 1971 [70]).
- *Agents* means that agents are used for generation. For example, agents can represent furniture optimising arrangements as in [47].
- *Rules* means that rules given by constraints, ontologies etc. are the core concept of generation such as rule-based layout solving [29].
- *Machine Learning* means that the utilisation of data by means of machine learning is the core concept of generation. For example, convolutional neural networks can be used for floor plan generation as done in [71].

Validation and Optimisation — This aspect provides categorial information about a PGB's method to find valid or high quality solutions with three possible values:

- *Constructive* means that the system directly generates valid or high quality solutions (with high probability) such as building blueprint generation in clear phases from start to end as in [72]
- *Automatically Evaluated* means that the system repeatedly generates solutions and assesses their fitness to find fitting ones. Yu et al. use a cost function to evaluate the fitness of samples from the search space for furniture arrangements [48].
- *Designer Evaluated* means that users interactively have to care about the quality of the results such as interactively drawing building profiles in an editor as in [57].

This aspect is taken from the literature in which it was introduced as one of several taxonomical aspects with *Constructive* and *Generate-and-test* as values [10], [14]. We encountered several approaches which used PCG techniques for the assistance of users in an interactive process. Using these approaches, the systems do not care for the quality of the results alone. Instead, they provide capabilities and delegate responsibility to their users, resulting in a foundational difference in terms of guiding the generation efforts. Thus, we split *generate-and-test* in the two values *Automatically Evaluated* and *Designer Evaluated* to address this important difference.

D. Meta Aspects

In addition to the taxonomical aspects, we provide one meta aspect which we captured during our survey. We provide the *Code Accessibility* to ease getting in touch with generators and to estimate the necessary efforts for doing so.

Code Accessibility — Although software is an integral part of modern research, its publication, acknowledgement and citation are not common practice [73]. Regarding the importance of source code, it has been claimed that all scripts and parameters have to be published to ensure scientific reproducibility [74]. Even with published code of poor quality, other researchers are enabled to engage in corresponding research [75]. This is the only aspect which is independent of the operating principles of the PGB approach and thus not a taxonomical aspect for PGB generators. As such, it is not used to compare the solutions directly, but gives indications about the efforts scientists and practitioners have to invest when starting to work with the approaches. It also indicates how precise the core of the approaches can be (re-)created compared to the version of the corresponding publication.

This aspect provides categorial information with three possible values:

- *Not Accessible* means that no code is provided.
- *Partial* means that samples of (pseudo-)code are given as parts of the publication.
- *Open Source* means that major parts of the source code are made available on the Internet. Claims with broken or dead links provided in the literature were not considered as *Open Source*.

A few of the taxonomic aspects are not consistent in a way a robust taxonomy would require them, i. e. yielding the same results when different persons would use it for categorisation. We included these aspects in the presented form for making the survey more informative. For making the taxonomy more robust and independent of this survey, the following steps could be taken. First, *Data Structures*, being a free text aspect, could be formalised for clear categories or left out. Second, the *Shape Description* and *Means of Control* could be left out or made a multi select, providing separate binary information for each of the provided values.

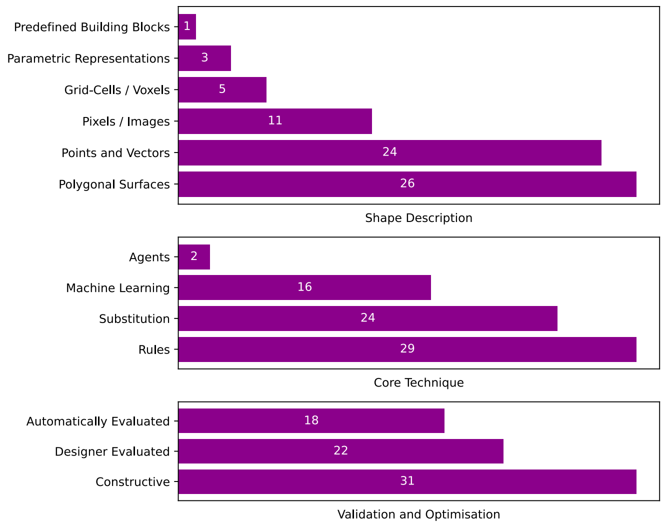

Fig. 1: Graphical overview of the categorial values of the *Technological Aspects*. The count of each value is given as a number in the corresponding bar.

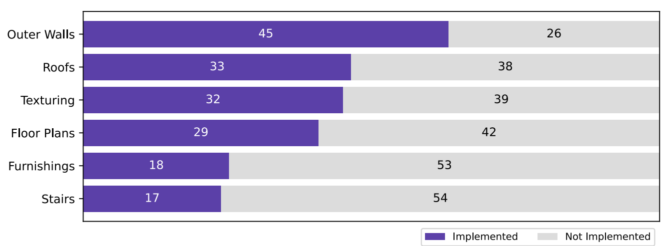

Fig. 2: Graphical overview of the *Generative Capabilities*: For each capability, the count of approaches implementing the corresponding building constituent is given. The count of each value is given as a number in the corresponding coloured bar and the count of approaches not being able to generate the corresponding building constituent is given as a number in the grey bar.

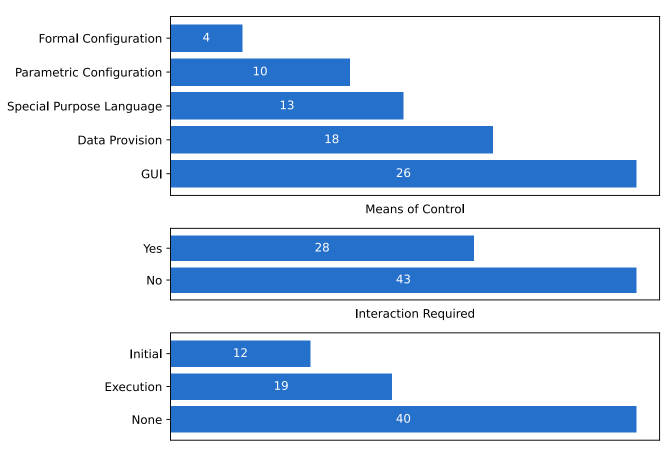

Fig. 3: Graphical overview of the categorial and binary values of the *Accessibility Aspects*. The count of each value is given as a number in the corresponding bar.

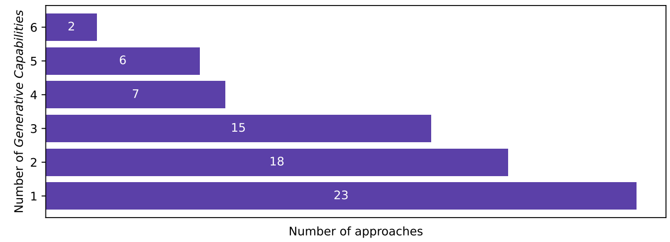

Fig. 4: Counts of approaches with certain numbers of *Generative Capabilities*. The counts are given as a number in the corresponding bar.

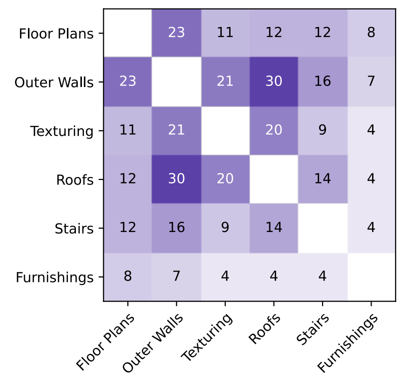

Fig. 5: Occurrences of *Generative Capabilities* in **absolute** pairwise combinations in matrix representation. Each cell shows the number of solutions implementing both, the *Generative Capability* listed in the corresponding row and column. The matrix is symmetric by definition and redundant symmetric values are given for reading usability.

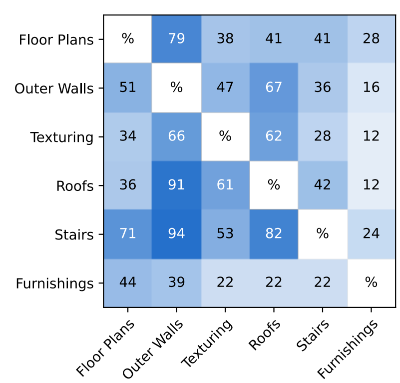

Fig. 6: Occurrences of *Generative Capabilities* in **relative** pairwise combinations in matrix representation. The cells of the matrix show the percentage of bilateral occurrences relative to the single occurrences of the *Generative Capabilities* of the corresponding rows. For example, 71 percent of all solutions implementing *Stairs* also implement *Floor Plans*. The percentages are rounded to integer values.

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

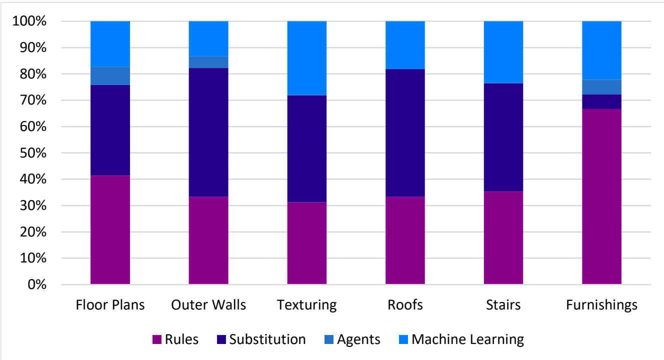

Fig. 7: The relative distribution of *Core Techniques* for the *Generative Capabilities*.

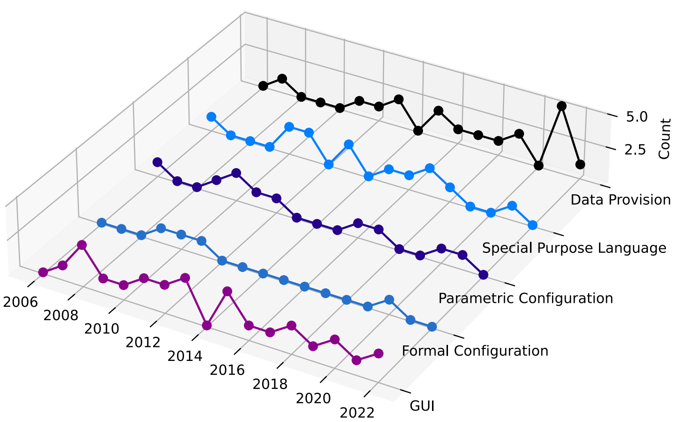

Fig. 8: Manifestations of the *Means of Control* with counts per year. Note that for the last year 2022, only publications before 1 June 2022 were considered.

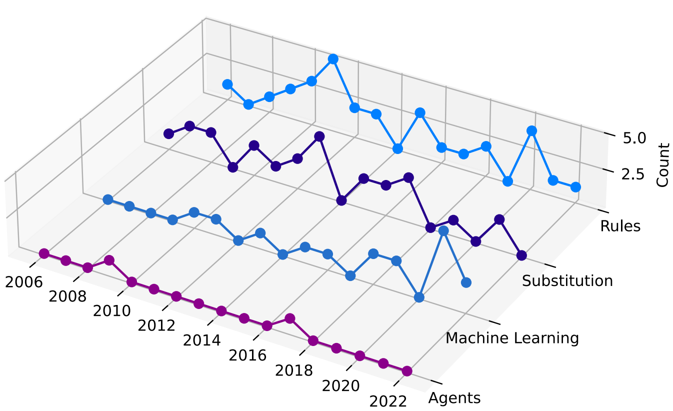

Fig. 9: Manifestations of the *Core Technique* with counts per year. Note that for the last year 2022, only publications before 1 June 2022 were considered.

IV. Procedural Generation of Buildings

In this section, the results of our analysis are presented. The underlying classification data for all graphics of this section are provided in Table I. Although our 71 samples are not sufficient to reveal reliable statistical insights, these overviews are considered helpful for an understanding of the domain in terms of relevant features, commonly used approaches and trends. We briefly discuss relevant publications prior to 2006 in the beginning of this section.

A cornerstone for PGB was published in 2003 [118]. The authors observed the suitability of L-systems for other procedural content such as plants and streets. Compared to these contents, they highlighted the different structure of buildings and proposed split grammars for procedural architecture. Split grammars are parametric shape grammars using rule type restrictions and controlled randomness. They are based on the work of Stiny et al., who introduced shape grammars a few decades earlier with a focus on visual arts and two-dimensional applications [70].

Later, grammar-based approaches were reported to be primarily used for PGB, hierarchically applying substitution rules [46], especially for building exteriors [26]. Our data confirms this claim with *Substitution* occurring 24 times (see Figure 1). 22 of these 24 occurrences of *Substitution* appear in combination with *Outer Walls* which is $22/45 \approx 49$ percent and 16 occur alongside of *Roofs* which is $16/33 \approx 48$ percent (cf. Figure 2 for the occurrences of the *Generative Capabilities*). In contrast, 10 occurrences appear alongside *Floor Plans* which is $10/29 \approx 34$ percent and only one with *Furnishings* ($1/18 \approx 6$ percent), which in turn are often handled by constraints and fitness functions (cf. Figure 7). In other words, different *Core Techniques* are usually used for the implementation of different *Generative Capabilities*. This indicates differences in the problem structure of the implementation of different *Generative Capabilities*.

Already more than a decade ago it was observed that most solutions for procedural buildings focus on the exteriors [72], [48], [119]. This observation was reasserted in recent years [103] and is again affirmed by our data. This can be seen in Figure 2, which shows the number of solutions implementing each *Generative Capability*.

Of the 18 approaches implementing *Furnishings*, seven accompany *Outer Walls* and four accompany *Roofs* (see Figure 5). In contrast, *Outer Walls* and *Roofs* appear together 30 times. We refer to this phenomenon as the “interior-exterior-gap”. This can be seen from a different perspective when investigating the relative pairwise occurrences of the *Generative Capabilities* shown in Figure 6. Some of the *Generative Capabilities* commonly occur together: *Roofs* are rarely implemented without the *Outer Walls* (91 percent) and *Stairs* are rarely implemented without the *Outer Walls* (94 percent) and *Roofs* (82 percent). The other way around, *Outer Walls* are rarely handled together with *Stairs* (36 percent) and *Furnishings* (16 percent). Generally, furnishings appear more rarely together with other *Generative Capabilities*. This can also be witnessed when investigating the more extensive PGB solutions: Only 2 of the 13 solutions with four or five *Generative Capabilities* implement *Furnishings*

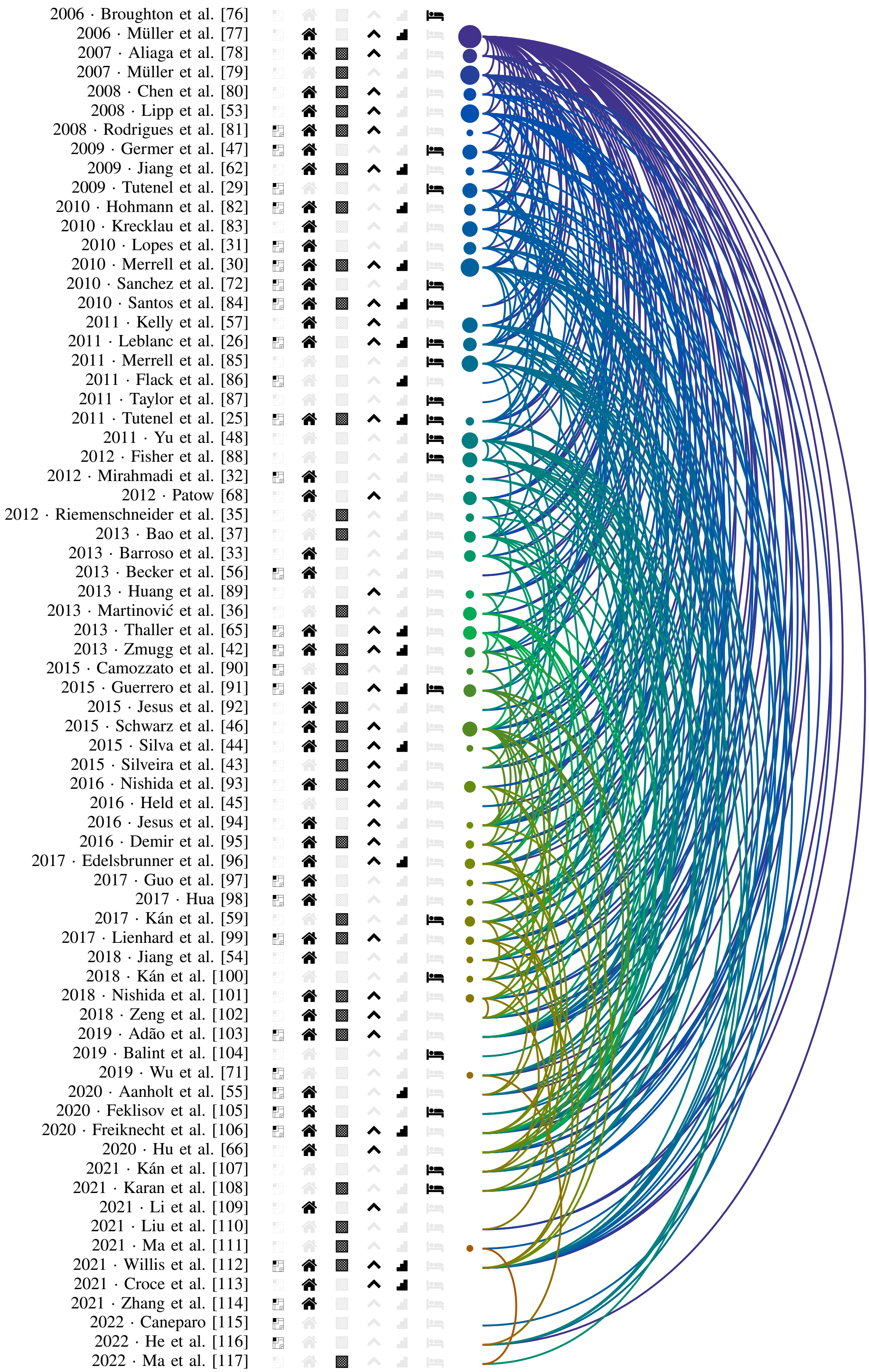

Fig. 10: Overview of the 71 analysed publications with their implemented *Generative Capabilities* and mutual citations. For each publication, the number of citations within the set of the 71 publications is visualised by the radius of a circle which is calculated logarithmically from the number of citations. Each publication has a unique colour for the circle and connections to all publications citing it.

(see Figure 10 and Table I). With *Furnishings* being the most detailed interior aspect of the six *Generative Capabilities*, this emphasizes the interior-exterior-gap. These observations also give a second indication for the differences in the problem structures of the different *Generative Capabilities*.

From a technological perspective (see Figure 1), the problem domain of procedural buildings has a strong rule-based character according to the values of the *Core Technique* aspect: *Rules* occur 29 times and the special case of applying replacement rules by means of *Substitution* occurs 24 times, totalling to 53 out of 71. This shows a distinguishing feature of buildings compared to other areas of PCG. A possible cause is that buildings are usually constructed by humans in a structured manner following rules and implications of the corresponding culture and environment such as climatic conditions. Such rules are often explicitly expressed by construction planners and architects, and they may be formalised for automatic rule checking of building designs [120]. This is also a possible cause for the rare occurrence of evolutionary algorithms and other approaches which behave well for complex challenges which are, on the contrary, typically hard to formalise. For the broader field of PCG, Togelius et al. conducted a survey and identified an over-representation of evolutionary approaches [10].

Machine Learning-based *Core Techniques* follow with 16 occurrences showing a clear upwards trend over time (see *Data provision* in Figure 8 and *Machine Learning* in Figure 9). This aligns with the advances of machine learning and artificial intelligence in science and practical applications especially over the past decade. This was also noted by Summerville et al. in their survey on PCG via machine learning published in 2018, but the authors also emphasized that much work has to be done, especially since most efforts so far concerned 2-D levels [121]. Machine learning still is an area of rapid evolution with many novel concepts currently being investigated and not all of them require (plenty of) data for application. An example for such a solution is the open-ended, novelty-guided evolution of Minecraft buildings, using autoencoders for 1D representation of 3D-structures [122]. This is based on DeLeNoX (Deep Learning Novelty Explorer), a system that autonomously creates artifacts in constrained spaces according to its own evolving interestingness criterion [123].

Regarding *Validation and Optimisation*, the values are more balanced: All three values occur rather often with a minimum of 18 for *Automatically Evaluated* and a maximum of 31 for *Constructive* methods, where the latter directly generates a (valid) result (see Figure 1). The *Designer Evaluated* value often accompanies the *GUI* from the *Means of Control* and likewise *Interaction Required* (see Table I). Both these accessibility values, *GUI* and *Yes* for the interaction requirement, are frequent values within the *Accessibility Aspects* (cf. Figure 3). Considering the values *Designer Evaluated* for the aspect *Validation and Optimisation* and *Yes* for the aspect *Interaction Required* it can be seen, that many current solutions do not implement or aim for fully automatic solutions (cf. Figures 1 and 3). However, full automation is not always a goal. Some even consider the worst case for the level of control over a procedural approach to be limitation to the definition of the model and its parameters [27]. Some of the approaches directly name artists as their target group and leverage procedural modelling as a means to augment the artists' skill sets. For example, a grammar-based approach can be combined with a visual workflow to enable artists to create buildings on a GUI level [53].

For practical application and research, code accessibility is important for the utilisation of the corresponding approaches. Although it can foster future research or application, many authors may be interested in maintaining their edge in research experiments by not sharing their implementation work, future commercial usage or simply do not want to prepare their code for public presentation. *Open Source* rarely occurs (3 out of 71), 29 publications do not provide any code and the majority shares parts of the code within the publications.

Besides the aforementioned machine learning trend, we could not identify other clear trends in the distributions of the occurrences over time, neither for the *Technological Aspects* nor for the *Accessibility Aspects*. In addition to the two printed trend diagrams in this article (cf. Figure 8 and Figure 9), all other binary and categorial aspects were investigated for trends without the identification of further peculiarities.

A. Challenges and Limitations

Challenges and limitations are important for choosing an approach for implementation and application. They can also be seen as open research questions, guiding the direction of future research. Many published approaches have limits in what they can produce or handle. For example, several approaches can only produce rectangular rooms, others are restricted to convex rooms [65], [124]. More complex geometry can be generated by operations such as Boolean subtraction which extends the capabilities of the approach. This often requires handling of special cases in the underlying data structures to maintain the required invariant properties and may come with numerical challenges during implementation [112].

Furthermore, different solutions are suitable for different challenges: Grammar-based solutions, as stated in the literature and confirmed by our analysis (cf. *Substitution* as the *Core Technique* in Figure 1) are a very common approach for the PGB. However, they usually are not used for the implementation of each and every *Generative Capability*. In particular, furniture placements are typically done without the usage of grammars and use different rule systems or some kind of self-organisation such as agents whereby each agent represents a piece of furniture, optimising local goals to find proper placements (see Table I). In contrast, self-organisation and evolution are rarely used for the geometric generation of architecture.

Shape-based grammars as the most popular concrete approach for implementation also come with challenges when generating architecture, most notably reaching a realistic level of architectural complexity. This is often discussed, also as an open challenge from publications providing grammar-solutions themselves [46], [98]. While performing well at producing repetitive structures and hierarchies, grammars struggle at the generation of complex details [82], [89]. The complexity of grammars can also affect the users—they often have to learn and apply scripting in a grammar language [37], [36]. One very important challenge for which no conclusive, scalable solution has been presented so far, is the selection problem: Hierarchical structures resulting from shape splitting operations are independent and unrelated, which renders it difficult to maintain relations between different sub-trees during generation or manual post-processing [44], [54]. Approaches that address this challenge often include complex tree operations on the substitution tree. This has led to a search for extensions or different solutions: Jiang et al. replace the traditional grammar approach with label-based selections. Therein, several hierarchies are maintained as views and a selection-based language allows for complex selection queries [54].

In general, we could not identify a single approach ready for generating complete buildings without further research and development efforts.

B. Generating Complete Buildings

Depending on the use case, different parts of buildings are of importance. A survey on PCG in 2017 lists *entire accessible buildings*, but states them to be only vaguely discussed [11]. The freedom associated with open world games is perceived more intensely when the players encounter fewer barriers that prevent them from exploring surroundings [125]. One of the most common types of barriers in urban settings are certainly buildings that the player cannot enter and that are only represented by facades [25], [106]. For this reason, traversable buildings with interiors shift into focus [34].

Bidarra et al. have compared manual modelling with PCG in terms of strength and weaknesses. Completeness was listed as a strength of manual modelling, but not of procedural modelling [126]. Our results indicate that this also holds for the area of PGB (cf. Figure 4). Numerous PGB approaches have one or the other *Generative Capability*. Similar approaches are typically pursued to implement similar *Generative Capabilities*. In particular, interior-related content generation is very different from generating exteriors such as facades [27]. We have also found indications for this based on the analysed publications, discussed in the context of the interior-exterior-gap, Section IV. Only a few approaches consider several or all aspects at the same time. Either such approaches are ensembles of different generators or the results are not convincing in terms of their restrictions or results. Two publications were found addressing all six building constituents [84], [25], however also exhibiting some limitations. In addition, several solutions exist for every building constituent as can be seen in Figure 2. Thus, it is theoretically possible to procedurally generate complete buildings using the current state-of-the-art. But it is not easy to do, since in order to generate complete buildings different approaches have to be used together—and they offer hardly any infrastructure to support such a multi-facetted engineering process.

At first glance, this renders the two publications covering all six *Generative Capabilities* all the more valuable. In 2010, Santos et al. proposed an ensemble approach based on external data and user interaction. Plans or photos are processed by automatic scale detection and are snapped to a spatial grid. The user can place staircases as floor transitions, draw walls with the cursor, define floor types and place assets like windows, doors and interior objects etc. [84]. In 2011, Tutenel et al. proposed a semantic approach to integrate different partial solutions by means of a semantic moderator. It requires the adaptation of partial solutions to work with the moderator concept. The semantic moderator is a central instance that coordinates the different approaches by means of a message bus: Communication, assessment and combination of parts are handled by the central semantic moderator [25]. Both solutions address the important question of how different approaches can work together—from a user experience and method integration perspective. Yet, they do not describe the interfaces between building constituents in a detailed and formalised way. This might explain why, so far, they have a comparably low influence on the other solutions we encountered: Santos et al. have not been cited and Tutenel et al. were cited twice within the set of analysed publications. The average citation of all analysed publications was 3.82 times, and the average citation of all analysed publications up to 2015 was 6.2 times (cf. Figure 10). These observations suggest that a focus on the integration of different approaches currently is the most promising option for the generation of complete buildings. Consequently, we infer that compatibility, i. e. the capability of a generator to integrate with other approaches, is one of the most important characteristics of PGB approaches to contribute to the generation of complete buildings.

Compatibility can be achieved in different ways. One possibility is the adaption of a semantic moderator as proposed in [25], utilising a central instance for integration of different approaches. Those approaches do not need to care (too much) about each other, but only implement interfaces for the moderator which then handles the coordination of the generation and composition.

Alternatively, interfaces between different *Generative Capabilities* can be optimised for compatibility by the availability of standardised or flexible information (e. g. parameters for directions, sizes etc. for staircases) or the absence of complex details in the interface areas, such as cornice at the interface between walls and roofs.

Another possibility is to generate building constituents in several steps starting with minimalistic generation, adding details in later steps considering the (intermediate) results of more than one generator to achieve integrated results.

Another option is the (compatibility for) deep integration between different approaches, e. g. by generating more than one building constituent together either with one algorithm or more than one optimised for their interworking. This approach most probably has the highest required effort, but also the potential for the best results since deep integration includes the possibility of complex details including two or more *Generative Capabilities*.

Each of these alternatives for increasing compatibility has different requirements for the corresponding interfaces. Having well-organised, standardised interfaces can help to keep the integration efforts manageable. It can affect the usage of elements such as data structures, defined formats, provided measures or a given hierarchy organising the assignment of details to building elements. Also, generic constraints could be supported by such a standardised interface allowing for an integrated control of the interplay of different generators as well as the users' expectations.

C. Design Decisions and Trade-offs

When selecting solution approaches or configuring given approaches, the capabilities and limitations of the resulting system are determined. This subsection discusses such design decisions and also trade-offs coming along with them.

There are only a few PGB approaches implementing several or even all *Generative Capabilities* and most analysed solutions have additional limitations such as restricted shapes. Depending on the use case, one or several approaches have to be selected to fulfil the given requirements and design goals. Their selection not only affects the results directly, but also indirectly by determining the approaches' usability, the necessary efforts for application, their compatibility among each other and the interplay with other virtual assets from the environment of the buildings. Our classifications (see Table I) can support these decisions by matching given requirements to individual approaches. To summarise, the decision process can be driven by PGB approach classifications with use case-specific pros and cons and subsequent optimisation of the faced trade-offs. In the following, we list design aspects we identified to be important and discuss their trade-offs and effects such as efforts and restrictions following certain decisions about selected technologies. Some of these directly come from the taxonomical aspects presented in Section III. Others are derived from the survey results.

Scope: The most obvious and nonetheless very important decision is about the scope of the solution, i. e. the decision about the *Generative Capabilities* and included details for generation. The number of different elements as well as the difference in the problem structures are likely to limit the solution quality or drastically increase the generators' complexity. Additionally, scope complexity can increase when going into detail. For example, some of the implementations for furniture arrangements have rules for specific types of furniture. Additional types of furniture would either require additional implementation or even exclude the specific solutions we analysed.

Compatibility: As stated in Subsection IV-B, compatibility may be one of the most important characteristics, at least when aiming for complete buildings. The strategy to ensure compatibility among complementary PGB approaches impacts the set of currently viable solutions to choose from. While some are easy to integrate, others come with intense challenges to do so. Those approaches with more flexible *Means of Control* and *Interaction Required*, are inherently easier to adapt. A rather challenging example nicely illustrating the issue would be a solver-based solution for outer walls and roofs. In addition to the given design constraints, it would have to provide and optimise additional interfaces to building interiors to be compatible to other solutions.

Interaction: Depending on the goal, PCG can be used to automatically create content or to augment the users' design capabilities. The involvement of users can allow for desirable, high quality solutions. But if the users provide inconsistent input or if the *Generative Capabilities* of the given approach simply do not meet their expectations, the process might fail as well. User involvement can thus be seen as a stream of introduced constraints increasing the computational challenge. Alternatively, considering an illustrative example, the users' introduction of assets or process configurations may also accelerate the convergence of the search process or even extend the search space.

Data Usage: The advances in data-based approaches such as the availability of sensor-systems and successes in artificial intelligence have also yielded promising approaches for PGB (see trend discussion in Section IV). Incorporating (real world) data can lead to high quality results and enable generators to take rules into account which humans have not even thought about, such as implicit aesthetics for furniture placements. On the other hand, and despite the increased availability of data in general, the availability of fitting data in adequate quantity and quality may still be a restriction and thus prevent such solutions from being used, especially if data is not only required for initial configuration or training, but also for each execution and thus for the generation of each artefact or set of artefacts. In addition, the fundamental challenge of uncertainty has to be considered for many data-based approaches. In particular, currently trending, induction-based machine learning models do not maintain ontological or logical knowledge. Instead, they just learn to reduce their error for approximation of likely results based on seen data.

V. Conclusion

We have presented an extensive survey on PGB. For categorisation and comparison, we started with the creation of a procedural building taxonomy which consists of 13 taxonomical aspects for procedural buildings in three groups: 1) *Generative Capabilities* (*Floor Plans*, *Outer Walls*, *Texturing*, *Roofs*, *Stairs*, and *Furnishings*), 2) *Accessibility Aspects* (*Means of Control*, *Interaction Required* and *Based on External Data*), and 3) *Technological Aspects* (*Shape Description*, *Core Data Structures*, *Core Technique*, and *Validation and Optimisation*). In addition, the *Code Accessibility* is given as an additional meta aspect. These aspects were inferred from relevant literature and, in turn, were used to classify 71 state-of-the-art approaches. We presented the survey's results taxonomically and in terms of analytical figures. The interplay of several aspects, the effects and reasons of their combined occurrences as well as trends and best practices were discussed.

An important goal is the PGB of complete buildings. We provided evidence that relying on current solutions, it is possible to achieve this goal. However, our survey also suggests that it is not easy to do so, requiring potentially great efforts for the conceptual and programmatic integration of complementary approaches. Due to the different nature of the various involved sub-problems, compatibility, i. e. the capability of an approach to integrate with other approaches, is an integral requirement and an important quality criterion. We emphasised this insight in our discussion on design decisions and trade-offs when selecting PGB for concrete applications.

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

TABLE I: The investigated solutions for procedural buildings and their taxonomy-based classifications.

2006 · Müller et al. [77] · Procedural Modeling of Buildings

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Grammar Hierarchy, Octree
Core Technique	Substitution
Validation	Constructive

2006 · Broughton et al. [76] · Introducing Clutter Into Virtual Environments

Code Accessibility	Partial
Means of Control	Parameter Configuration
Interaction Req.	✗
External Data	None
Shape Description	Coordinates
Data Structures	Attributed Clutter Regions
Core Technique	Rules
Validation	Automatically Evaluated

2007 · Aliaga et al. [78] · Style Grammars for Interactive Visualization of Architecture

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	Initial
Shape Description	Shapes
Data Structures	Hierarchical scene graph
Core Technique	Substitution
Validation	Designer Evaluated

2007 · Müller et al. [79] · Image-based Procedural Modeling of Facades

Code Accessibility	Partial
Means of Control	Data Provision
Interaction Req.	✗
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Shape tree, Symmetry-based pixel lists
Core Technique	Substitution
Validation	Contructive

2008 · Rodrigues et al. [81] · Incorporating Legal Rules on Procedural House Generation

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✗
External Data	None
Shape Description	Grid-Cells / Voxels
Data Structures	Tree of Rooms
Core Technique	Substitution
Validation	Automatically Evaluated

2008 · Lipp et al. [53] · Interactive visual editing of grammars for procedural architecture

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Shapes
Data Structures	Grammar Hierarchy
Core Technique	Substitution
Validation	Designer Evaluated

2008 · Chen et al. [80] · Sketching Reality: Realistic Interpretation of Architectural Designs

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Shapes
Data Structures	Graph
Core Technique	Rules
Validation	Designer Evaluated

2009 · Tutenel et al. [29] · Rule-based layout solving and its application to procedural interior [...]

Code Accessibility	Partial
Means of Control	Mathematics
Interaction Req.	✗
External Data	None
Shape Description	Coordinates
Data Structures	Placement areas
Core Technique	Rules
Validation	Automatically Evaluated

2009 · Jiang et al. [62] · Symmetric Architecture Modeling with a Single Image

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	Execution
Shape Description	Shapes
Data Structures	3D point set
Core Technique	Rules
Validation	Designer Evaluated

2009 · Germer et al. [47] · Procedural Arrangement of Furniture for Real-Time Walkthroughs

Code Accessibility	Partial
Means of Control	Parameter Configuration
Interaction Req.	✓
External Data	None
Shape Description	Coordinates
Data Structures	Parent Links
Core Technique	Agents
Validation	Automatically Evaluated

2010 · Santos et al. [84] · On the Expeditious Modelling of Buildings

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	Execution
Shape Description	Pixels / Images
Data Structures	-
Core Technique	Rules
Validation	Designer Evaluated

2010 · Sanchez et al. [72] · Procedural generation of building blueprints for real-time applications

Code Accessibility	Partial
Means of Control	Parameter Configuration
Interaction Req.	✗
External Data	None
Shape Description	Coordinates
Data Structures	-
Core Technique	Rules
Validation	Constructive

2010 · Merrell et al. [30] · Computer-Generated Residential Building Layouts

Code Accessibility	Not Accessible
Means of Control	Parameter Configuration
Interaction Req.	✗
External Data	Initial
Shape Description	Coordinates
Data Structures	Bayesian network
Core Technique	Machine Learning
Validation	Automatically Evaluated

2010 · Lopes et al. [31] · A Contrained Growth Method for Procedural Floor Plan Generation

Code Accessibility	Partial
Means of Control	Mathematics
Interaction Req.	✗
External Data	None
Shape Description	Grid-Cells / Voxels
Data Structures	2D-Matrices
Core Technique	Rules
Validation	Constructive

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

2010 · Krecklau et al. [83] · Generalized Use of Non-Terminal Symbols for Procedural Modeling

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Shape Trees
Core Technique	Substitution
Validation	Constructive

2010 · Hohmann et al. [82] · A GML shape grammar for semantically enriched 3D building models

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Stack, Evaluation Tree
Core Technique	Substitution
Validation	Constructive

2011 · Yu et al. [48] · Make it Home: Automatic Optimization of Furniture Arrangement

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✗
External Data	Initial
Shape Description	Coordinates
Data Structures	-
Core Technique	Rules
Validation	Automatically Evaluated

2011 · Tutenel et al. [25] · Generating Consistent Buildings: A Semantic Approach for Integrating [...]

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✓
External Data	None
Shape Description	-
Data Structures	Semantic Model
Core Technique	Rules
Validation	Designer Evaluated

2011 · Taylor et al. [87] · Randomness + Structure = Clutter: A Procedural Object Placement [...]

Code Accessibility	Not Accessible
Means of Control	Mathematics
Interaction Req.	✓
External Data	None
Shape Description	Grid-Cells / Voxels
Data Structures	Hierarchical coloured petri net
Core Technique	Rules
Validation	Automatically Evaluated

2011 · Flack et al. [86] · Evolution of Architectural Floor Plans

Code Accessibility	Partial
Means of Control	Parametric Configuration
Interaction Req.	✓
External Data	None
Shape Description	Grid-Cells / Voxels
Data Structures	Grid chromosomes
Core Technique	Machine Learning
Validation	Automatically Evaluated

2011 · Merrell et al. [85] · Interactive Furniture Layout Using Interior Design Guidelines

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Coordinates
Data Structures	Layout tuple
Core Technique	Rules
Validation	Designer Evaluated

2011 · Leblanc et al. [26] · Component-Based Modeling of Complete Buildings

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Components trees
Core Technique	Substitution
Validation	Constructive

2011 · Kelly et al. [57] · Interactive architectural modeling with procedural extrusions

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Coordinates
Data Structures	Collection of polyline segments
Core Technique	Rules
Validation	Designer Evaluated

2012 · Riemenschneider et al. [35] · Irregular lattices for complex shape grammar facade parsing

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✗
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Parse tree
Core Technique	Substitution
Validation	Automatically Evaluated

2012 · Patow [68] · User-friendly graph editing for procedural modeling of buildings

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Shapes
Data Structures	Directed Acyclic Graph
Core Technique	Substitution
Validation	Designer Evaluated

2012 · Mirahmadi et al. [32] · A Novel Algorithm for Real-time Procedural Generation of Building [...]

Code Accessibility	Partial
Means of Control	Parameter Configuration
Interaction Req.	✗
External Data	Initial
Shape Description	Coordinates
Data Structures	Connectivity graph
Core Technique	Rules
Validation	Constructive

2012 · Fisher et al. [88] · Example-based Synthesis of 3D Object Arrangements

Code Accessibility	Open Source
Means of Control	GUI
Interaction Req.	✓
External Data	Initial
Shape Description	Coordinates
Data Structures	Static support hierarchy
Core Technique	Rules
Validation	Designer Evaluated

2013 · Barroso et al. [33] · Visual copy & paste for procedurally modeled buildings by ruleset rewriting

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	Initial
Shape Description	Shapes
Data Structures	Directed Acyclic Graph
Core Technique	Substitution
Validation	Designer Evaluated

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

2013 · Zmugg et al. [42] · Procedural architecture using deformation-aware split grammars

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	-
Core Technique	Substitution
Validation	Constructive

2013 · Thaller et al. [65] · Shape grammars on convex polyhedra

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Space Partitioning Trees
Core Technique	Substitution
Validation	Constructive

2013 · Martinović et al. [36] · Bayesian Grammar Learning for Inverse Procedural Modeling

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✗
External Data	Initial
Shape Description	Pixels / Images
Data Structures	(Grammar) Parse Tree
Core Technique	Machine Learning
Validation	Automatically Evaluated

2013 · Huang et al. [89] · Ting tools: interactive and procedural modeling of Chinese ting

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Parametric Representations
Data Structures	Skeletal ting frame
Core Technique	Rules
Validation	Designer Evaluated

2013 · Becker et al. [56] · Combined Grammar for the Modeling of Building Interiors

Code Accessibility	Partial
Means of Control	Data Provision
Interaction Req.	✗
External Data	Execution
Shape Description	Coordinates
Data Structures	Grammar Hierarchy
Core Technique	Substitution
Validation	Constructive

2013 · Bao et al. [37] · Procedural facade variations from a single layout

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Tree-based hierarchy of regions
Core Technique	Rules
Validation	Designer Evaluated

2015 · Silveira et al. [43] · Real-time Procedural Generation of Personalized Facade and Interior [...]

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✗
External Data	Execution
Shape Description	Coordinates
Data Structures	-
Core Technique	Rules
Validation	Constructive

2015 · Silva et al. [44] · Procedural Content Graphs for Urban Modeling

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Procedural content graphs
Core Technique	Rules
Validation	Constructive

2015 · Schwarz et al. [46] · Advanced Procedural Modeling of Architecture

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Shape Trees
Core Technique	Substitution
Validation	Constructive

2015 · Guerrero et al. [91] · Learning shape placements by example

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	Initial
Shape Description	Coordinates
Data Structures	2D subspaces, Operations tree
Core Technique	Machine Learning
Validation	Designer Evaluated

2015 · Camozzato et al. [90] · Procedural floor plan generation from building sketches

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✗
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Grid
Core Technique	Rules
Validation	Automatically Evaluated

2015 · Jesus et al. [92] · Towards interactive procedural modelling of buildings

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Shapes
Data Structures	Procedural Tree, Instantiation Tree
Core Technique	Substitution
Validation	Designer Evaluated

2016 · Nishida et al. [93] · Interactive Sketching of Urban Procedural Models

Code Accessibility	Partial
Means of Control	Data Provision
Interaction Req.	✓
External Data	Initial
Shape Description	Shapes
Data Structures	Grammar Tree, Grammar Snippets
Core Technique	Machine Learning
Validation	Constructive

2016 · Jesus et al. [94] · Layered shape grammars for procedural modelling of buildings

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Clipping Tree
Core Technique	Substitution
Validation	Constructive

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

2016 · Held et al. [45] · Straight Skeletons with Additive and Multiplicative Weights and Their [...]

Code Accessibility	Not Accessible
Means of Control	Parameter Configuration
Interaction Req.	✗
External Data	None
Shape Description	Coordinates
Data Structures	-
Core Technique	Rules
Validation	Constructive

2016 · Demir et al. [95] · Proceduralization for Editing 3D Architectural Models

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✗
External Data	Initial
Shape Description	Shapes
Data Structures	Split tree
Core Technique	Substitution
Validation	Constructive

2017 · Kán et al. [59] · Automated Interior Design Using a Genetic Algorithm

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✗
External Data	None
Shape Description	Coordinates
Data Structures	Furniture Hierarchy
Core Technique	Rules
Validation	Automatically Evaluated

2017 · Edelsbrunner et al. [96] · Procedural Modeling of Architecture with Round Geometry

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Split Tree
Core Technique	Substitution
Validation	Constructive

2017 · Lienhard et al. [99] · Design Transformations for Rule-based Procedural Modeling

Code Accessibility	Partial
Means of Control	Parameter Configuration
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Shape Tree
Core Technique	Substitution
Validation	Constructive

2017 · Hua [98] · A Bi-Directional Procedural Model for Architectural Design

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✓
External Data	None
Shape Description	Shapes
Data Structures	Spatial hierarchical tree
Core Technique	Substitution
Validation	Designer Evaluated

2017 · Guo et al. [97] · Evolutionary approach for spatial architecture layout design enhanced by [...]

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✗
External Data	None
Shape Description	Grid-Cells / Voxels
Data Structures	3D-Grid
Core Technique	Agents
Validation	Automatically Evaluated

2018 · Zeng et al. [102] · Neural Procedural Reconstruction for Residential Buildings

Code Accessibility	Open Source
Means of Control	Data Provision
Interaction Req.	✗
External Data	Execution
Shape Description	Shapes
Data Structures	Neural Networks
Core Technique	Machine Learning
Validation	Constructive

2018 · Nishida et al. [101] · Procedural Modeling of a Building from a Single Image

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	Execution
Shape Description	Shapes
Data Structures	Neural Networks
Core Technique	Machine Learning
Validation	Constructive

2018 · Kán et al. [100] · Automatic Furniture Arrangement Using Greedy Cost Minimization

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✗
External Data	None
Shape Description	Coordinates
Data Structures	Parent-child relationships
Core Technique	Rules
Validation	Automatically Evaluated

2018 · Jiang et al. [54] · Selection Expressions for Procedural Modeling

Code Accessibility	Partial
Means of Control	Special Purpose Language
Interaction Req.	✗
External Data	None
Shape Description	Shapes
Data Structures	Shape Hierarchies
Core Technique	Rules
Validation	Constructive

2019 · Wu et al. [71] · Data-driven Interior Plan Generation for Residential Buildings

Code Accessibility	Partial
Means of Control	Data Provision
Interaction Req.	✗
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Neural Networks
Core Technique	Machine Learning
Validation	Automatically Evaluated

2019 · Balint et al. [104] · A Generalized Semantic Representation for Procedural Generation of Rooms

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✗
External Data	Initial
Shape Description	Coordinates
Data Structures	Graph, Content Chunks
Core Technique	Machine Learning
Validation	Constructive

2019 · Adão et al. [103] · Procedural Modeling of Buildings Composed of Arbitrarily-Shaped [...]

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Coordinates
Data Structures	Treemaps
Core Technique	Substitution
Validation	Automatically Evaluated

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

2020 · Feklisov et al. [105] · Procedural interior generation for artificial intelligence training and [...]

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Coordinates
Data Structures	Room Clusters
Core Technique	Rules
Validation	Designer Evaluated

2020 · Hu et al. [66] · Extended interactive and procedural modeling method for ancient chinese [...]

Code Accessibility	Partial
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Parametric Representations
Data Structures	Hierarchical Tree
Core Technique	Rules
Validation	Designer Evaluated

2020 · Freiknecht et al. [106] · Procedural Generation of Multistory Buildings With Interior

Code Accessibility	Partial
Means of Control	Parameter Configuration
Interaction Req.	✕
External Data	None
Shape Description	Coordinates
Data Structures	Polygons
Core Technique	Rules
Validation	Constructive

2020 · Aanholt et al. [55] · Declarative procedural generation of architecture with semantic [...]

Code Accessibility	Not Accessible
Means of Control	Mathematics
Interaction Req.	✕
External Data	None
Shape Description	Predefined Building Blocks
Data Structures	-
Core Technique	Rules
Validation	Automatically Evaluated

2021 · Kán et al. [107] · Automatic Interior Design in Augmented Reality Based on Hierarchical [...]

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	Execution
Shape Description	Coordinates
Data Structures	Tree of procedural rules
Core Technique	Machine Learning
Validation	Designer Evaluated

2021 · Karan et al. [108] · A Markov Decision Process Workflow for Automating Interior Design

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✓
External Data	Execution
Shape Description	Coordinates
Data Structures	Point Cloud, Markov Chain
Core Technique	Machine Learning
Validation	Designer Evaluated

2021 · Li et al. [109] · Relation-Constrained 3D Reconstruction of Buildings in Metropolitan Areas [...]

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✕
External Data	Execution
Shape Description	Shapes
Data Structures	Tree
Core Technique	Substitution
Validation	Automatically Evaluated

2021 · Liu et al. [110] · Translational Symmetry-Aware Facade Parsing for 3D Building Reconstruction

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✕
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Neural Net
Core Technique	Machine Learning
Validation	Constructive

2021 · Ma et al. [111] · Pyramid ALKNet for Semantic Parsing of Building Facade Image

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✕
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Neural Net
Core Technique	Machine Learning
Validation	Constructive

2021 · Willis et al. [112] · Volumetric Procedural Models for Shape Representation

Code Accessibility	Open Source
Means of Control	Special Purpose Language
Interaction Req.	✕
External Data	None
Shape Description	Shapes
Data Structures	Semantic Hierarchy of 3D Shape Elements
Core Technique	Substitution
Validation	Constructive

2021 · Croce et al. [113] · From the Semantic Point Cloud to Heritage [...]

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	✓
External Data	Execution
Shape Description	Points and Vectors
Data Structures	Point Clouds
Core Technique	Machine Learning
Validation	Designer Evaluated

2021 · Zhang et al. [114] · Generative design and performance optimization of residential buildings [...]

Code Accessibility	Not Accessible
Means of Control	Parametric Configuration
Interaction Req.	✕
External Data	Initial
Shape Description	Parametric Representations
Data Structures	Layered Design Schemes, Graphs
Core Technique	Rules
Validation	Constructive

2022 · Caneparo [115] · Semantic knowledge in generation of 3D layouts for decision-making

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✓
External Data	None
Shape Description	Points and Vectors
Data Structures	Constraint Hypergraph
Core Technique	Rules
Validation	Designer Evaluated

2022 · He et al. [116] · iPLAN: Interactive and Procedural Layout Planning

Code Accessibility	Not Accessible
Means of Control	GUI
Interaction Req.	✕
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Neural Networks
Core Technique	Machine Learning
Validation	Contructive

2022 · Ma et al. [117] · Progressive Feature Learning for Facade Parsing With Occlusions

Code Accessibility	Not Accessible
Means of Control	Data Provision
Interaction Req.	×
External Data	Execution
Shape Description	Pixels / Images
Data Structures	Neural Networks
Core Technique	Machine Learning
Validation	Contructive

REFERENCES

[1] J. Togelius, E. Kastbjerg, D. Schedl, and G. N. Yannakakis, "What is Procedural Content Generation? Mario on the borderline," in *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games*. New York, NY, USA: Association for Computing Machinery, 2011, pp. 1–6.

[2] J.-S. Kim, H. Kavak, and A. Crooks, "Procedural City Generation Beyond Game Development," in *SIGSPATIAL Special*. ACM, 2018, pp. 34–41.

[3] K. Dylla, B. Frischer, P. Müller, A. Ulmer, and S. Haegler, "Rome Reborn 2.0: A Case Study of Virtual City Reconstruction Using Procedural Modeling Techniques," *Computer Applications and Quantitative Methods in Archaeology*, vol. 37, pp. 62–66, 2010.

[4] M. Hendrikx, S. Meijer, J. van der Velden, and A. Iosup, "Procedural Content Generation for Games: A Survey," in *ACM Transactions on Multimedia Computing, Communications, and Applications*. New York, NY, USA: ACM, 2013, vol. 9, pp. 1–22.

[5] D. Gravina, A. Khalifa, A. Liapis, J. Togelius, and G. N. Yannakakis, "Procedural Content Generation through Quality Diversity," *Conference on Games*, pp. 1–8, 2019.

[6] R. Craveirinha, N. Barreto, and L. Roque, "Towards a Taxonomy for the Clarification of PCG Actors' Roles," in *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, A. Cox, Z. O. Toups, R. L. Mandryk, and P. Cairns, Eds. New York, NY, USA: ACM, 2016, pp. 244–253.

[7] M. M. Pedersen, "Procedural Level Generation," 2017. [Online]. Available: https://www.ghostship.dk/blog/procedural-level-generation

[8] O. Toms, "Valheim map: biomes, placing map markers, how to ping the map," 2021. [Online]. Available: https://www.rockpapershotgun.com/valheim-map-biomes-placing-map-markers-how-to-ping-the-map

[9] N. Shaker, G. Yannakakis, and J. Togelius, "Towards Automatic Personalized Content Generation for Platform Games," *Artificial Intelligence and Interactive Digital Entertainment*, pp. 63–68, 2010.

[10] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne, "Search-Based Procedural Content Generation: A Taxonomy and Survey," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, pp. 172–186, 2011.

[11] J. Freiknecht and W. Effelsberg, "A Survey on the Procedural Generation of Virtual Worlds," *Multimodal Technologies and Interaction*, vol. 1, no. 4, 2017.

[12] "STAR CITIZEN || PROCEDURAL CITIES ARE HERE!" 2017. [Online]. Available: https://robertsspaceindustries.com/community/citizen-spotlight/9585-STAR-CITIZEN-PROCEDURAL-CITIES-ARE-HERE-MIC-DROP

[13] Esri, "ArcGIS CityEngine - Success Stories," 2022. [Online]. Available: https://www.esri.com/en-us/arcgis/products/arcgis-cityengine/success-stories

[14] J. Togelius, N. Shaker, and M. J. Nelson, "Chapter 1: Introduction," in *Procedural Content Generation in Games*, N. Shaker, J. Togelius, and M. J. Nelson, Eds. Springer, 2016, pp. 1–15.

[15] Science and Technology Publications, "SciTePress Digital Library." [Online]. Available: https://www.scitepress.org/HomePage.aspx

[16] arXiv Leadership Team at Cornell Tech, "arXiv." [Online]. Available: https://arxiv.org

[17] MDPI AG, "Multidisciplinary Digital Publishing Institute." [Online]. Available: https://www.mdpi.com

[18] Digital Science & Research Solutions, Inc., "Dimensions." [Online]. Available: https://app.dimensions.ai/discover/publication

[19] Google, "Google Scholar." [Online]. Available: https://scholar.google.com

[20] Institute of Electrical and Electronics Engineers, "IEEE Xplore." [Online]. Available: https://ieeexplore.ieee.org/Xplore/home.jsp

[21] Elsevier B.V., "ScienceDirect." [Online]. Available: https://www.sciencedirect.com

[22] Emerald Publishing Limited, "emerald insight." [Online]. Available: https://www.emerald.com/insight/

[23] Association for Computing Machinery, "ACM Digital Library." [Online]. Available: https://dl.acm.org

[24] Springer Nature, "SpringerLink." [Online]. Available: https://link.springer.com

[25] T. Tutenel, R. M. Smelik, R. Lopes, K. J. de Kraker, and R. Bidarra, "Generating Consistent Buildings: A Semantic Approach for Integrating Procedural Techniques," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, pp. 274–288, 2011.

[26] L. Leblanc, J. Houle, and P. Poulin, "Component-Based Modeling of Complete Buildings," *Graphics Interface*, pp. 87–94, 2011.

[27] R. M. Smelik, T. Tutenel, R. Bidarra, and B. Benes, "A Survey on Procedural Modelling for Virtual Worlds," *Computer Graphics Forum*, vol. 33, no. 6, pp. 31–50, 2014.

[28] G. Kelly and H. McCabe, "A Survey of Procedural Techniques for City Generation," *The ITB Journal*, vol. 7, no. 2, pp. 87–130, 2006.

[29] T. Tutenel, R. Bidarra, R. M. Smelik, and K. J. de Kraker, "Rule-based layout solving and its application to procedural interior generation," 2009.

[30] P. Merrell, E. Schkufza, and V. Koltun, "Computer-Generated Residential Building Layouts," in *ACM Transactions on Graphics*, G. Drettakis, Ed. New York, New York, USA: Association for Computing Machinery, 2010, pp. 1–12.

[31] R. Lopes, T. Tutenel, R. M. Smelik, K. J. de Kraker, and R. Bidarra, "A Contrained Growth Method for Procedural Floor Plan Generation," in *Proceedings of the 11th International Conference on Intelligent Games and Simulation*, 2010.

[32] M. Mirahmadi and A. Shami, "A Novel Algorithm for Real-time Procedural Generation of Building Floor Plans," 2012. [Online]. Available: http://arxiv.org/pdf/1211.5842v1

[33] S. Barroso, G. Besuievsky, and G. Patow, "Visual copy & paste for procedurally modeled buildings by ruleset rewriting," *Computers & Graphics*, vol. 37, no. 4, pp. 238–246, 2013.

[34] J. Freiknecht and W. Effelsberg, "Procedural Generation of Multistory Buildings With Interior," in *Transactions on Games*, vol. 12. IEEE, 2020, pp. 323–336.

[35] H. Riemenschneider, U. Krispel, W. Thaller, M. Donoser, S. Havemann, D. Fellner, and H. Bischof, "Irregular lattices for complex shape grammar facade parsing," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1640–1647.

[36] A. Martinović and L. van Gool, "Bayesian Grammar Learning for Inverse Procedural Modeling," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 201–208.

[37] F. Bao, M. Schwarz, and P. Wonka, "Procedural facade variations from a single layout," *ACM Transactions on Graphics*, vol. 32, no. 1, pp. 1–13, 2013.

[38] D. S. Ebert, F. K. Musgrave, D. Peachey, K. Perlin, and S. Worley, *Texturing and Modeling: A Procedural Approach: Third Edition*. San Francisco: Morgan Kaufmann Publishers, 2003.

[39] J. Dong, J. Liu, K. Yao, M. Chantler, L. Qi, H. Yu, and M. Jian, "Survey of Procedural Methods for Two-Dimensional Texture Generation," *Sensors (Basel, Switzerland)*, vol. 20, no. 4, 2020.

[40] J. Dong, L. Wang, J. Liu, and X. Sun, "A Procedural Texture Generation Framework Based on Semantic Descriptions," 2017. [Online]. Available: http://arxiv.org/pdf/1704.04141v1

[41] Y. Hu, J. Dorsey, and H. Rushmeier, "A Novel Framework For Inverse Procedural Texture Modeling," *ACM Transactions on Graphics*, vol. 38, no. 6, pp. 1–14, 2019.

[42] R. Zmugg, W. Thaller, U. Krispel, J. Edelsbrunner, S. Havemann, and D. W. Fellner, "Procedural architecture using deformation-aware split grammars," in *The Visual Computer*, vol. 30. Springer, 2013, pp. 1009–1019.

[43] I. Silveira, D. Camozzato, F. Marson, L. Dihl, and S. R. Musse, "Real-Time Procedural Generation of Personalized Facade and Interior Appearances Based on Semantics," in *2015 14th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. IEEE, 2015, pp. 89–98.

[44] P. B. Silva, E. Eisemann, R. Bidarra, and A. Coelho, "Procedural Content Graphs for Urban Modeling," *International Journal of Computer Games Technology*, pp. 1–15, 2015.

[45] M. Held and P. Palfrader, "Straight Skeletons with Additive and Multiplicative Weights and Their Application to the Algorithmic Generation of Roofs and Terrains," 2016. [Online]. Available: http://arxiv.org/pdf/1604.03362v1

This article has been accepted for publication in IEEE Transactions on Games. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/TG.2023.3262507

[46] M. Schwarz and P. Müller, "Advanced Procedural Modeling of Architecture," in *Transactions on Graphics*, vol. 34. ACM, 2015, pp. 1–12.
[47] T. Germer and M. Schwarz, "Procedural Arrangement of Furniture for Real-Time Walkthroughs," *Computer Graphics Forum*, 2009.
[48] L.-F. Yu, S.-K. Yeung, C.-K. Tang, D. Terzopoulos, T. F. Chan, and S. J. Osher, "Make it Home: Automatic Optimization of Furniture Arrangement," in *ACM SIGGRAPH 2011 papers on - SIGGRAPH '11*, H. Hoppe, Ed. New York, New York, USA: ACM Press, 2011, pp. 1–12.
[49] T. Ullrich and U. Krispel, "Compilation of Procedural Models," in *Proceedings of the 13th International Symposium on 3D Web Technology*. Los Angeles, California: Association for Computing Machinery, 2008, pp. 75–81.
[50] U. Krispel, C. Schinko, and T. Ullrich, "A Survey of Algorithmic Shapes," *Remote Sensing*, vol. 7, no. 10, pp. 12 763–12 792, 2015.
[51] J. O. Talton, Y. Lou, S. Lesser, J. Duke, R. Měch, and V. Koltun, "Metropolis Procedural Modeling," in *ACM Transactions on Graphics*, vol. 30. ACM, 2011, pp. 1–14.
[52] C. Depcik and D. N. Assanis, "Graphical user interfaces in an engineering educational environment," *Computer Applications in Engineering Education*, vol. 13, no. 1, pp. 48–59, 2005.
[53] M. Lipp, P. Wonka, and M. Wimmer, "Interactive visual editing of grammars for procedural architecture," *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 1–10, 2008.
[54] H. Jiang, D.-M. Yan, X. Zhang, and P. Wonka, "Selection Expressions for Procedural Modeling," *IEEE transactions on visualization and computer graphics*, vol. 26, no. 4, pp. 1775–1788, 2018.
[55] L. van Aanholt and R. Bidarra, "Declarative procedural generation of architecture with semantic architectural profiles," in *Conference on Games*. Osaka, Japan: IEEE, 2020, pp. 351–358.
[56] S. Becker, M. Peter, D. Fritsch, D. Philipp, P. Baier, and C. Dibak, "Combined Grammar for the Modeling of Building Interiors," *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 2, no. 4, pp. 1–6, 2013.
[57] T. Kelly and P. Wonka, "Interactive architectural modeling with procedural extrusions," *ACM Transactions on Graphics*, vol. 30, no. 2, pp. 1–15, 2011.
[58] G. Smith, "Understanding Procedural Content Generation: A Design-Centric Analysis of the Role of PCG in Games," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, M. Jones, P. Palanque, A. Schmidt, and T. Grossman, Eds. New York, NY, USA: ACM, 2014, pp. 917–926.
[59] P. Kán and H. Kaufmann, "Automated interior design using a genetic algorithm," in *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, M. Fjeld, M. Fratarcangeli, D. Sjölie, O. Staadt, and J. Unger, Eds. New York, NY, USA: ACM, 2017, pp. 1–10.
[60] E. J. Hastings, R. K. Guha, and K. O. Stanley, "Demonstrating Automatic Content Generation in the Galactic Arms Race Video Game," *Artificial Intelligence and Interactive Digital Entertainment*, vol. 5, no. 1, pp. 189–190, 2009.
[61] G. Smith, "The Future of Procedural Content Generation in Games," *Experimental Artificial Intelligence in Games*, pp. 53–57, 2014.
[62] N. Jiang, P. Tan, and L.-F. Cheong, "Symmetric Architecture Modeling with a Single Image," *ACM Transactions on Graphics*, vol. 28, no. 5, pp. 1–8, 2009.
[63] M. Torkkeli and M. Tuominen, "The contribution of technology selection to core competencies," *International Journal of Production Economics*, vol. 77, no. 3, pp. 271–284, 2002.
[64] S. Havemann, "Generative Mesh Modeling," 2005.
[65] W. Thaller, U. Krispel, R. Zmugg, S. Havemann, and D. W. Fellner, "Shape grammars on convex polyhedra," *Computers & Graphics*, vol. 37, no. 6, pp. 707–717, 2013.
[66] Z. Hu and X. Qin, "Extended interactive and procedural modeling method for ancient chinese architecture," in *Multimedia Tools and Applications*. Springer, 2021, pp. 5773–5807.
[67] D. J. R. Meagher, "Octree Encoding: A New Technique For The Representation, Manipulation and Display of Arbitrary 3-D Objects by Computer," New York, 1980.
[68] G. Patow, "User-friendly graph editing for procedural modeling of buildings," *IEEE computer graphics and applications*, vol. 32, no. 2, pp. 66–75, 2012.
[69] A. Lindenmayer, "Mathematical Models for Cellular Interactions in Development II. Simple and Branching Filaments with Two-sided Inputs," *Journal of Theoretical Biology*, vol. 18, no. 3, pp. 300–315, 1968.
[70] G. Stiny and J. Gips, "Shape Grammars and the Generative Specification of Painting and Sculpture," *IFIP Congress*, vol. 2, pp. 1460–1465, 1972.
[71] W. Wu, X.-M. Fu, R. Tang, Y. Wang, Y.-H. Qi, and L. Liu, "Data-driven interior plan generation for residential buildings," *ACM Transactions on Graphics*, vol. 38, no. 6, pp. 1–12, 2019.
[72] D. Sanchez, J. M. Solá-Sloan, and E. Lozano-Inca, "Procedural generation of building blueprints for real-time applications," *Proceedings of the 2010 Spring Simulation Multiconference*, pp. 1–4, 2010.
[73] D. S. Katz, K. E. Niemeyer, and A. M. Smith, "Publish Your Software: Introducing the Journal of Open Source Software (JOSS)," *Computing in Science & Engineering*, vol. 20, no. 3, pp. 84–88, 2018.
[74] G. K. Sandve, A. Nekrutenko, J. Taylor, and E. Hovig, "Ten simple rules for reproducible computational research," *PLoS computational biology*, vol. 9, no. 10, 2013.
[75] N. Barnes, "Publish your computer code: it is good enough," *Nature*, vol. 467, no. 753, 2010.
[76] R. Broughton and T. Howard, "Introducing Clutter Into Virtual Environments," 2006.
[77] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. van Gool, "Procedural Modeling of Buildings," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 614–623, 2006.
[78] D. G. Aliaga, P. A. Rosen, and D. R. Bekins, "Style Grammars for Interactive Visualization of Architecture," *IEEE transactions on visualization and computer graphics*, vol. 13, no. 4, pp. 786–797, 2007.
[79] P. Müller, G. Zeng, P. Wonka, and L. van Gool, "Image-based Procedural Modeling of Facades," *ACM Transactions on Graphics*, vol. 26, no. 3, pp. 85.1–85.9, 2007.
[80] X. Chen, S. B. Kang, Y.-Q. Xu, J. Dorsey, and H.-Y. Shum, "Sketching Reality: Realistic Interpretation of Architectural Designs," *ACM Transactions on Graphics*, vol. 27, no. 2, pp. 1–15, 2008.
[81] N. Rodrigues, M. Dionísio, L. G. Magalhães, J. P. Moura, and A. Chalmers, "Incorporating Legal Rules on Procedural House Generation," *Proceedings of the 24th Spring Conference on Computer Graphics*, pp. 59–66, 2008.
[82] B. Hohmann, S. Havemann, U. Krispel, and D. Fellner, "A GML shape grammar for semantically enriched 3D building models," *Computers & Graphics*, vol. 34, no. 4, pp. 322–334, 2010.
[83] L. Krecklau, D. Pavic, and L. Kobbelt, "Generalized Use of Non-Terminal Symbols for Procedural Modeling," *Computer Graphics Forum (CGF)*, vol. 29, no. 2, pp. 1–12, 2010.
[84] D. Santos, M. Dionísio, N. Rodrigues, and A. Pereira, "On the Expeditious Modelling of Buildings," in *2010 Second International Conference on Games and Virtual Worlds for Serious Applications*. IEEE, 2010, pp. 60–67.
[85] P. Merrell, E. Schkufza, Z. Li, M. Agrawala, and V. Koltun, "Interactive Furniture Layout Using Interior Design Guidelines," *ACM SIGGRAPH*, pp. 1–10, 2011.
[86] R. W. J. Flack and B. J. Ross, "Evolution of Architectural Floor Plans," *European Conference on the Applications of Evolutionary Computation*, pp. 313–322, 2011.
[87] J. Taylor and I. Parberry, "Randomness + Structure = Clutter: A Procedural Object Placement Generator," in *International Conference on Entertainment Computing*. Vancouver, BC, Canada: Springer, 2011, pp. 424–427.
[88] M. Fisher, D. Ritchie, M. Savva, T. Funkhouser, and P. Hanrahan, "Example-based Synthesis of 3D Object Arrangements," *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 1–11, 2012.
[89] C.-Y. Huang and W.-K. Tai, "Ting tools: interactive and procedural modeling of Chinese ting," in *The Visual Computer*, vol. 29. Springer, 2013, pp. 1303–1318.
[90] D. Camozzato, L. Dihl, I. Silveira, F. Marson, and S. R. Musse, "Procedural floor plan generation from building sketches," *The Visual Computer*, vol. 31, no. 6-8, pp. 753–763, 2015.
[91] P. Guerrero, S. Jeschke, M. Wimmer, and P. Wonka, "Learning shape placements by example," *ACM Transactions on Graphics*, vol. 34, no. 4, pp. 1–13, 2015.
[92] D. Jesus, A. Coelho, and A. A. Sousa, "Towards interactive procedural modelling of buildings," in *Proceedings of the 31st Spring Conference on Computer Graphics*, J. Jorge, L. P. Santos, and R. Ďurikovič, Eds. New York, NY, USA: ACM, 2015, pp. 109–112.
[93] G. Nishida, I. Garcia-Dorado, D. G. Aliaga, B. Benes, and A. Bousseau, "Interactive Sketching of Urban Procedural Models," in *Transactions on Graphics*, vol. 35. ACM, 2016, pp. 1–11.
[94] D. Jesus, A. Coelho, and A. A. Sousa, "Layered shape grammars for procedural modelling of buildings," *The Visual Computer*, vol. 32, no. 6-8, pp. 933–943, 2016.

[95] I. Demir, D. G. Aliaga, and B. Benes, "Proceduralization for Editing 3D Architectural Models," in *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016, pp. 194–202.

[96] J. Edelsbrunner, S. Havemann, A. Sourin, and D. W. Fellner, "Procedural Modeling of Architecture with Round Geometry," *Computers & Graphics*, vol. 64, pp. 14–25, 2017.

[97] Z. Guo and B. Li, "Evolutionary approach for spatial architecture layout design enhanced by an agent-based topology finding system," *Frontiers of Architectural Research*, vol. 6, no. 1, pp. 53–62, 2017.

[98] H. Hua, "A Bi-Directional Procedural Model for Architectural Design," in *Computer Graphics Forum*, vol. 36. European Association for Computer Graphics, 2017, pp. 219–231.

[99] S. Lienhard, C. Lau, P. Müller, P. Wonka, and M. Pauly, "Design Transformations for Rule-based Procedural Modeling," *Computer Graphics Forum*, vol. 36, no. 2, pp. 39–48, 2017.

[100] P. Kán and H. Kaufmann, "Automatic Furniture Arrangement Using Greedy Cost Minimization," *IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 491–498, 2018.

[101] G. Nishida, A. Bousseau, and D. G. Aliaga, "Procedural Modeling of a Building from a Single Image," *Computer Graphics Forum*, vol. 37, no. 2, pp. 415–429, 2018.

[102] H. Zeng, J. Wu, and Y. Furukawa, "Neural Procedural Reconstruction for Residential Buildings," *European Conference on Computer Vision*, pp. 759–775, 2018.

[103] T. Adão, L. Pádua, P. Marques, J. J. Sousa, E. Peres, and L. Magalhães, "Procedural Modeling of Buildings Composed of Arbitrarily-Shaped Floor-Plans: Background, Progress, Contributions and Challenges of a Methodology Oriented to Cultural Heritage," *Computers*, vol. 8, no. 2, 2019.

[104] J. T. Balint and R. Bidarra, "A generalized semantic representation for procedural generation of rooms," in *Proceedings of the 14th International Conference on the Foundations of Digital Games*, S. Deterding, F. Khosmood, J. Pirker, and T. Apperley, Eds. New York, NY, USA: ACM, 2019, pp. 1–8.

[105] E. D. Feklisov, M. V. Zingerenko, V. A. Frolov, and M. A. Trofimov, "Procedural interior generation for artificial intelligence training and computer graphics," *Computing for Physics and Technology*, 2020.

[106] J. Freiknecht, "Procedural Content Generation for Games," Mannheim, 2020.

[107] P. Kán, A. Kurtic, M. Radwan, and J. M. L. Rodríguez, "Automatic Interior Design in Augmented Reality Based on Hierarchical Tree of Procedural Rules," *Electronics*, vol. 10, no. 3, 2021.

[108] E. Karan, S. Asgari, and A. Rashidi, "A Markov Decision Process Workflow for Automating Interior Design," *KSCE Journal of Civil Engineering*, vol. 25, no. 9, pp. 3199–3212, 2021.

[109] Y. Li and B. Wu, "Relation-Constrained 3D Reconstruction of Buildings in Metropolitan Areas from Photogrammetric Point Clouds," *Remote Sensing*, vol. 13, no. 1, 2021.

[110] H. Liu, Li Wentong, and J. Zhu, "Translational Symmetry-Aware Facade Parsing for 3D Building Reconstruction," 2021. [Online]. Available: http://arxiv.org/pdf/2106.00912v1

[111] W. Ma, W. Ma, S. Xu, and H. Zha, "Pyramid ALKNet for Semantic Parsing of Building Facade Image," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 6, pp. 1009–1013, 2021.

[112] A. R. Willis, P. Ganesh, K. Volle, J. Zhang, and K. Brink, "Volumetric procedural models for shape representation," *Graphics and Visual Computing*, vol. 4, 2021.

[113] V. Croce, G. Caroti, L. de Luca, K. Jacquot, A. Piemonte, and P. Véron, "From the Semantic Point Cloud to Heritage-Building Information Modeling: A Semiautomatic Approach Exploiting Machine Learning," *Remote Sensing*, vol. 13, no. 3, p. 461, 2021.

[114] J. Zhang, N. Liu, and S. Wang, "Generative design and performance optimization of residential buildings based on parametric algorithm," *Energy and Buildings*, vol. 244, 2021.

[115] L. Caneparo, "Semantic knowledge in generation of 3D layouts for decision-making," *Automation in Construction*, vol. 134, p. 104012, 2022.

[116] F. He, Y. Huang, and H. Wang, "iPLAN: Interactive and Procedural Layout Planning," 2022.

[117] W. Ma, S. Xu, W. Ma, X. Zhang, and H. Zha, "Progressive Feature Learning for Facade Parsing With Occlusions," *IEEE Transactions on Image Processing*, vol. 31, pp. 2081–2093, 2022.

[118] P. Wonka, M. Wimmer, F. Sillion, and W. Ribarsky, "Instant Architecture," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 669–677, 2003.

[119] T. Adão, L. Magalhães, A. Coelho, A. A. Sousa, N. Rodrigues, R. Rodrigues, F. Pereira, J. Moura, and L. P. Reis, "ERAS – An Ontology-Based Tool for the Expeditious Reconstruction of Virtual Cultural Heritage Sites," *20º Encontro Português de Computação Gráfica*, pp. 89–95, 2012.

[120] C. Eastman, J.-m. Lee, Y.-s. Jeong, and J.-k. Lee, "Automatic rule-based checking of building designs," *Automation in Construction*, vol. 18, no. 8, pp. 1011–1033, 2009.

[121] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmgard, A. K. Hoover, A. Isaksen, A. Nealen, and J. Togelius, "Procedural Content Generation via Machine Learning (PCGML)," *IEEE Transactions on Games*, vol. 10, no. 3, pp. 257–270, 2018.

[122] M. Barthet, A. Liapis, and G. N. Yannakakis, "Open-Ended Evolution for Minecraft Building Generation," *IEEE Transactions on Games*, pp. 1–10, 2022.

[123] A. Liapis, H. P. Martínez, J. Togelius, and G. N. Yannakakis, "Transforming Exploratory Creativity with DeLeNoX," *Proc. of the Intl. Conf. on Computational Creativity*, pp. 56–63, 2013.

[124] T. Adão, L. Magalhães, E. Peres, and F. Pereira, "Procedural Generation of Traversable Buildings Outlined by Arbitrary Convex Shapes," *Procedia Technology*, vol. 16, pp. 310–321, 2014.

[125] K. M. Maj, "On the pseudo-open world and ludotopian dissonance: A curious case of Cyberpunk 2077," *Journal of Gaming & Virtual Worlds*, vol. 14, no. 1, pp. 51–65, 2022.

[126] R. Bidarra, K. J. de Kraker, R. M. Smelik, and T. Tutenel, "Integrating semantics and procedural generation: key enabling factors for declarative modeling of virtual worlds," *Proceedings of the FOCUS K3D Conference on Semantic 3D Media and Content*, 2010.

Damian Kutzias holds a Master of Science from the Paderborn University. He currently is a doctoral student at the Julius-Maximilians University in Würzburg and research associate at the Fraunhofer Institute for Industrial Engineering IAO in Stuttgart. His main research interest in the area of games is the procedural generation of virtual worlds in 3D. One important aspect and the current focus of his work is the automatic procedural generation of complete buildings, i. e. buildings including not only the outer geometry, but also coherent floor plans and furniture arrangements. Besides that, he works in the areas of data science process models and time series analysis.

Sebastian von Mammen is an Associate Professor at the Julius-Maximilians University in Würzburg, Germany. After career steps at the universities of Erlangen, Calgary (PhD) and Augsburg (habilitation), he became Head of the Games Engineering Group in 2017. His research interests comprise Real-Time Interactive Systems, Interactive Simulations, Artificial Life, and Artificial Intelligence.