SYMPOSIUM ON SIMULATION IN ARCHITECTURE + URBAN DESIGN
SimAUD 2020

Automated Parametric Building Volume Generation: a Case Study for Urban Blocks

Iuliia Osintseva[1], Reinhard Koenig[1,2], Andreas Berst[1], Martin Bielik[1], Sven Schneider[1]

[1]Bauhaus-Universität Weimar
Weimar, Germany
iuliia.osintseva@uni-weimar.de

[2]AIT Austrian Institute of Technology
Vienna, Austria
reinhard.koenig@ait.ac.at

ABSTRACT

The aim of this paper is to demonstrate a new developed parametric workflow for efficient semi-automatic generation of design options for multi-family residential buildings. The generation process is performed by modular components that generate design variants for building volumes automatically but enable the user to combine them in the preferred order, depending on the project goals. Such an approach enables the quick search in the vast solution space and, at the same time, allows the user to influence the generation process and guide it into the desired direction, thus allowing the designer to integrate his experiences and insights. The developed methodology is applied to a case study of a residential building design competition. In this case study, we aimed to 1) generate a multitude of diverse design options for the existing project and thus validate the developed method 2) generate options, that are visually similar to the manual designs submitted by architects and thus validate if the semi-automatic generation procedure is sufficient for including concept-related features and demands into the design process.

Author Keywords

Residential architecture; design automation; parametric design; computational design; parametric block typology.

1 INTRODUCTION

The output of the planning process – the building design – has a great and lasting impact on the social, economic, and ecological performance of the building. Therefore, finding a good solution at the early stage of planning, where most of the design decisions are taken, is crucial. In order to find the best possible solutions for the residential building planning at an early design stage, multiple design options must be created, analyzed, and compared with each other. It is not possible to directly find an individual optimal solution, because various design goals usually contradict each other, like the goals of achieving a maximum of density and maximum of daylight at the same time [12]. Therefore, a good solution will be represented by a trade-off between different design criteria.

Despite various existing approaches, utilizing generative design methods, the current design practice still mostly utilizes manual sketching as a method for the solution search process. As follows, only a relatively small set of design options is explored. Accordingly, good options may not even be regarded. Therefore, even if the final design satisfies the formal demands, there is no guarantee that a better trade-off was not overlooked during the solution search.

In this paper, we present a method for the efficient automatic generation of design options for multi-family residential buildings. The advantage of computational design methods is the enlarging of the design space. However, considering the number of theoretically possible design inputs, parameters, and criteria, the solution space can be infinite, and the computational expenses make it impossible to explore the complete solution space. Accordingly, the aim of this project was to develop a framework that would explore a large number of diverse options yet remain time-efficient. To achieve this, we based the parametric generation framework on the replication of the common existing residential typologies, thus restricting the infinite size of the design space to a smaller set of solutions that are well-established in the architectural practice. Such an approach allows avoiding the generation of non-practical or exotic solutions. The developed method is applied based on the widely used CAD-Software Rhino3d [18] and Grasshopper Plugin [17] in order to provide a seamless transition of the received outputs into the overall design process.

2 RELATED WORK

Although the usage of computational tools in design remains negligible [4], there already exists a multitude of generative tools applicable to different scales of architectural design, utilizing very different approaches on generation. The challenge of creating a highly complex architectural model from a simple set of inputs was addressed by Parish & Mueller [14]. The Decoding Spaces toolbox [1] contains a set of methods for both the generation and analysis of masterplans, starting with street network generation and followed by parceling and buildings placement. Rather than focusing on the search of a single optimized solution, the toolbox instead provides a highly adaptive generation workflow. Wilson et al. [21] developed a CUrBD method as an approach to performance-based master planning, outlining the most relevant trends for categorization of best-

performing design schemes as an instrument for negotiations between multiple stakeholders with contradicting interests. However, those approaches primarily focus on the creation of the masterplans of large areas. Such masterplans demand methods on a subdivision of the area into blocks, parceling, distribution of the density program, defining the uses combination and the allocation of the public spaces, etc. However, most of the masterplans contain numerous blocks that must be further elaborated. For each of the blocks, the solution space remains vast. As for the block scale, various tools and prototypes are available on the market. Many of those tools are online applications [20, 9] and let the user insert their goal criteria and then explore the set of automatically generated solutions. Another group of tools parametrizing the generation process, and automating manual steps is built upon existing CAD software [6]. Such tools allow for a quick adaptation of design to the new variables, but on the downside, explore only one design option at a time. The most recent approaches suggest an application of trained computation models for the solution search [19].

The common feature of most of the above-mentioned applications is their level of automation, which demands the user to insert important parameters and indicate key criteria, like the desired density or the apartment mix to be placed. Further work is taken over by the application, which enables a rapid overview concerning the possible building morphology for a site. In the current project we intended to set up a framework that remains quick while searching automatically for the good design alternatives and as well lets the user interact with the generation process. Such interaction should allow the designer to include his desired experience-based characteristics for a particular project into the generation process and therefore enable the consideration of non-quantifiable parameters, the lack of which is often being mentioned when criticizing computational design tools.

3 METHOD

The presented generation method is based on the replication of the common residential typologies. Most of the residential architecture examples can be categorized into a list of typologies [2, 3, 15]. In this research, we differentiate between three principal typologies (see Figure 1) and consider other residential building forms as combinations of these.

Figure 1. Principal typologies: block, slab and solitary.

Our framework allows us to replicate those three typologies in their basic form and to further perform variations over them to produce diverse yet buildable design options. In the following chapters, we will demonstrate the generation process on the example of an urban block typology.

3.1 Inputs

The inputs needed for the generation can be categorized into three groups: geometric, numeric, and regulations. By the geometric inputs, we mean the construction plot and, if available, the context information, such as streets, public spaces, and the neighboring buildings. Except for the construction plot outline (further referred to as plot) that must be provided by the user himself, the rest of the context data can be taken from publicly available sources, like the OpenStreetMap [13]. Context information is evidently important conceptually, as we evaluate the architectural designs, among other criteria, by their integration into the existing environment. Nevertheless, the context data is also used to derive important factors for the generation, such as the average height of the neighbor buildings. Context data is as well useful for the evaluation of the generation's outputs when it comes to the evaluation of the performance of our design for the surrounding buildings. For example, it is a common practice for the approval procedure of a residential project to provide daylight analysis. The goal of such an analysis is to prove that the daylight situation for the neighbors did not deteriorate significantly because of the new planning. At last, by regulations input, we mean the consideration of the spacing from the plot boundaries. In this case, spacing is defined as a distance from the plot outline till the building edge, that must remain empty [8]. By indicating the geographical location of the plot, we may derive the corresponding spacing indexes and consider them during the generation to ensure that the developed designs remain buildable according to the local building regulations.

3.2 Basic Block

First, the basic form of the chosen typology is applied to the construction plot. In the case of the block typology, the basic block is represented by the offsetting of the plot outline inwards of the plot by the chosen building depth. Depending on the selected building height, the needed spacing is calculated before placing the block. Once the basic block (or their range) is set, we can vary the form to explore the possible design space (see Figure 2).

3.3 Actions

Manipulations of the basic form (further: actions) are developed as modular grasshopper components. Each of the actions performs one type of manipulation over the current block form and can be applied after the basic block or after another action; multiple actions create a sequence. Each action has a set of unified inputs and outputs for the geometry representation as well as a section of user-inputs for manual control over the generation.

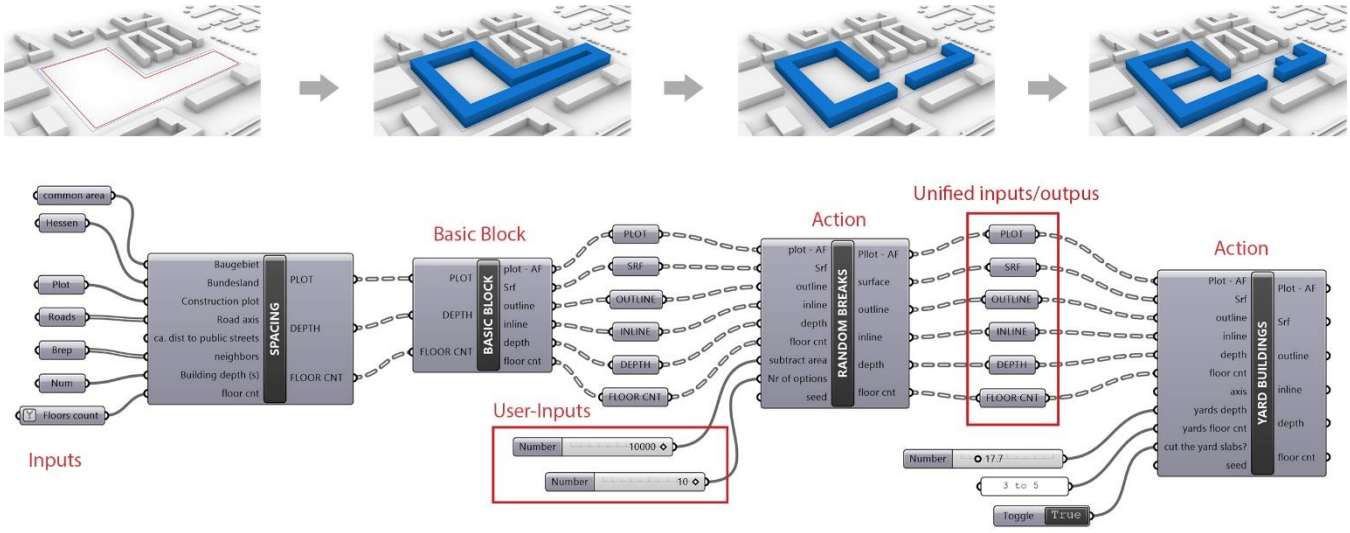


Figure 2. Simplified scheme of the developed parametric flow: Inputs→ Basic form of typology (here: block) →Actions

Because of the unified principal inputs and outputs (see Figure 2), the actions can be applied in any order. The unified inputs/outputs contain the following parameters: plot form after applying the spacing, surface (building footprint), contours of the surface for further facades generation, building's width, and the number of floors per building.

In fact, each action adds or subtracts some area from the current block variation. For example, the action "Towers" splits the block form into smaller fragments and extrudes one or several of them (see Figure 3). Each action runs automatically and is controlled by a set of user inputs (see Table 1). Most of the inputs are numerical and define the amount of area to be manipulated. For the "Towers" action, the user can select the range of areas for possible fragments placement, as well as the maximal number of the floors and the total amount of the area to add. Based on that, multiple options with randomly placed towers will be generated, whereas each of the options will add the desired amount of area to the development.

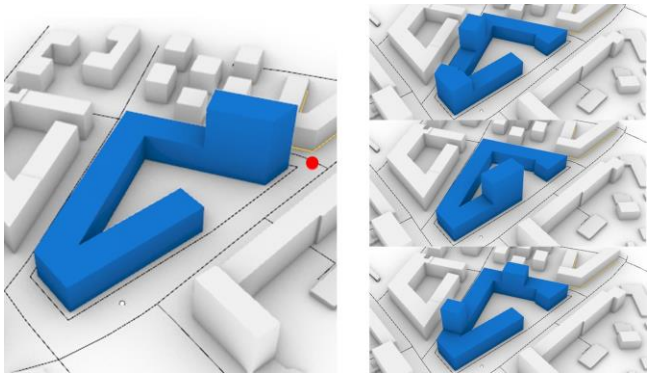


Figure 3. Comparison of the user-driven tower placement using the red marked allocation attractor (left) with the randomized allocation (right).

Each of the actions can perform automatically and intervene at multiple random locations of the block. However, the user can indicate the intervention placement with an allocation attractor, and the action will be applied at the defined spot. In the example of the "Towers", the action will not place them at different random positions around the block, but instead at the preferred location (see Figure 3).

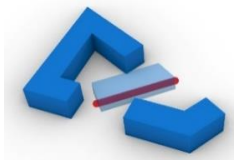
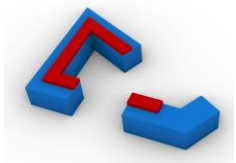
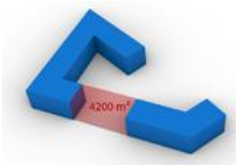
Type of the user inputs	Illustration
Attractor point(-s) for precise allocation of the needed intervention. Example: placement of "Setbacks".	
Attractor line(-s) for precise allocation of the needed intervention. Example: placement of the "Yard buildings".	
Numeric values to define geometry. Example: width of the "Rooftop terraces".	
Amount of area to manipulate (add or subtract).	

Table 1. Overview of the different possible types of user inputs.

Depending on the range of the inputs, each action can perform multiple variations over the same initial block input. Therefore, each next action added to the generation sequence multiplies the number of the output designs. Currently, the framework contains eight actions for the block typology (see Figure 4): “Open Block”(1) deletes one or multiple edges of the block; “Setbacks Edges”(2) and “Setbacks Corners”(3) place setbacks along the block perimeter in order to create plazas or to break the monotony of the long street-aligned facades; “Reduce Height”(4) subdivides the input form into fragments and reduces the density by sinking the height of one or multiple fragments; “Yard buildings”(5) places additional buildings in the inner yard space of the block; “Towers”(6) creates both spacing and solar envelopes [10], using the Ladybug plugin [11] for the given site and places the towers within the previously constructed envelopes; “Rooftop terraces”(7) creates the terraces on top of the block form; “Random Breaks”(8) creates passageways around the block perimeter. The actions listed above are performed automatically; however, it is on the user to choose the sequence of actions as well as to define their inputs. It is as well possible to use multiple sequences of actions in parallel (see Chapter 4, scenario 2). Depending on the selected actions sequences, the design variants can differ greatly. Therefore, the outcomes are more diverse than usual for parametric systems, which typically apply the same process for the changing inputs.

4 DEMONSTRATION

To validate the proposed framework, we demonstrate the possible application on two different scenarios for the same test case, represented by the competition “MK6 Theresienhöhe, Munich. [16]

In the first scenario, we tried to achieve optically similar designs like those created manually by the architects and compare the outputs with the winning competition entries by the density index (FAR). Such an approach allows us to test the methods of the user’s interaction with the generation process by trying to follow the formal criteria during the creation of the variants. With this, we can test if such a framework could be useful in the architectural workflow as a sketching tool for the early design phase.

One of the challenges that designers face is the great amount of changing goals and demands in the course of one project. Sufficient interaction methods would allow the implementation of the changes in the course of further project development. In addition, the various analysis, connected to the generation, directly report on the consequences of the morphology interventions, thus creating transparency in the decision making.

The first proposal is the closed block with the small breaks along one façade and a higher fragment at the southern corner. In order to achieve similar outputs, the following sequence was used: Basic Block → Random breaks→ Towers; resulting set: 200 options. Towers; resulting set: 200 options (see Figure 5).

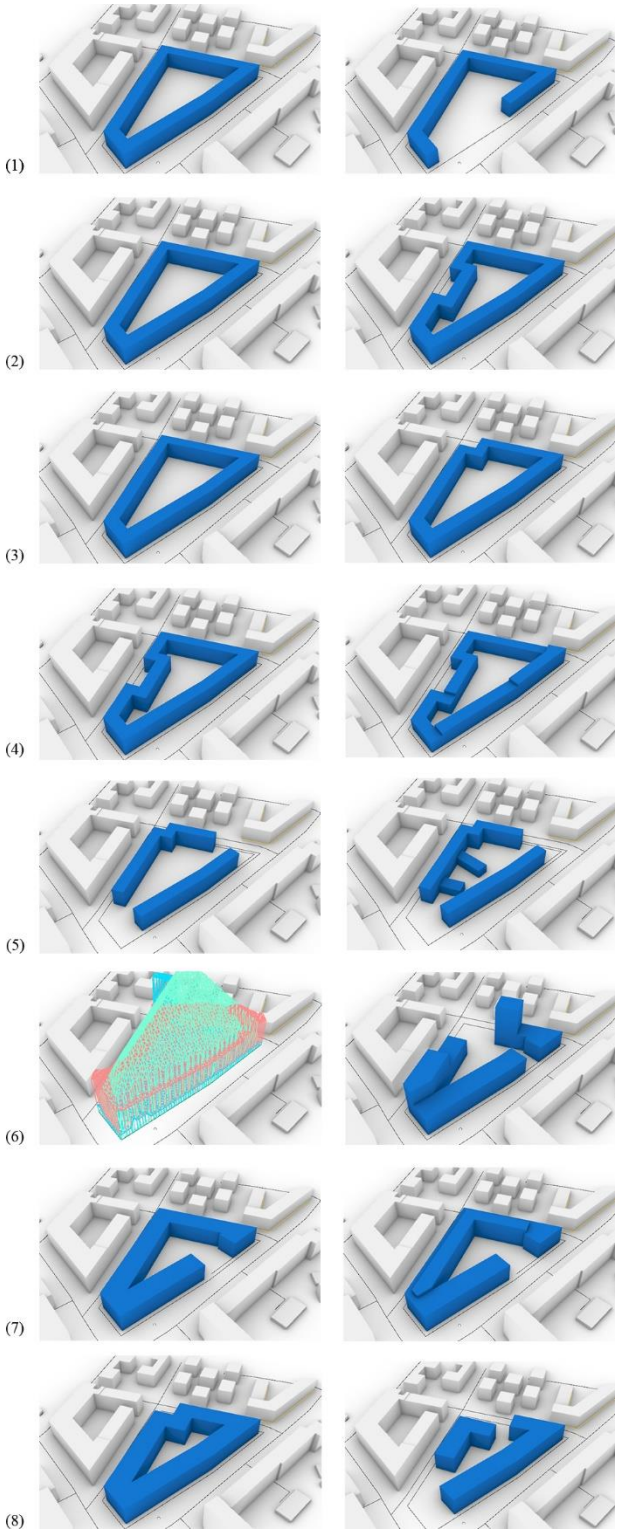


Figure 4. Actions modules, left column: geometry before action, right column: geometry after action. (1) Open block; (2) Setbacks Edges; (3) Setbacks Corners; (4) Reduce Height; (5) Yard buildings; (6) Towers; (7) Rooftop terraces; (8) Random Breaks. The examples of actions application shown above are not restricted by the input form and can be applied to any shape.

Figure 5. Proposal 1, competition entry [16] and three of the generated alternatives.

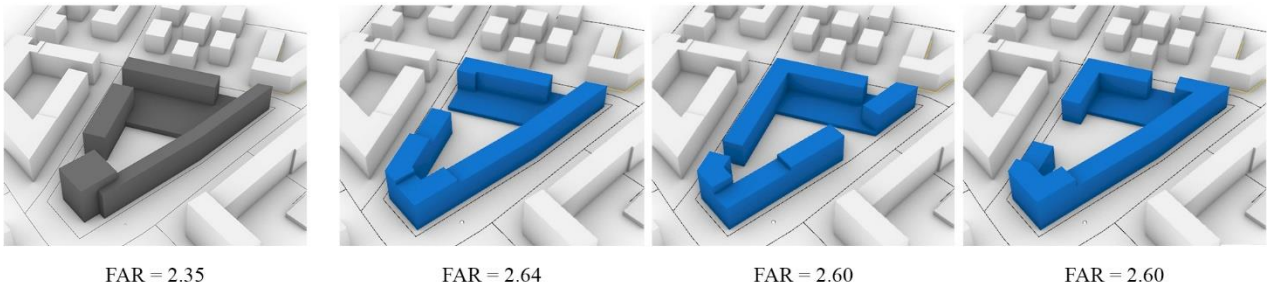


Figure 6. Proposal 2, competition entry [16] and three of the generated alternatives.

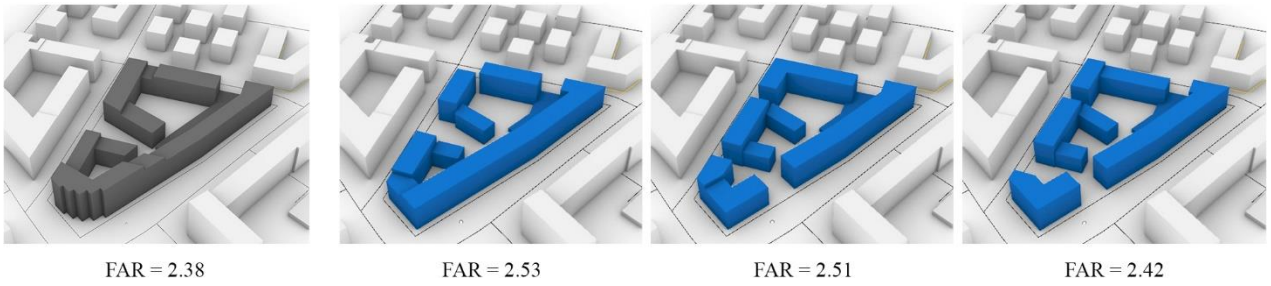


Figure 7. Proposal 3, competition entry [16] and three of the generated alternatives.

Here, the block outline, as well as height distribution was easy to achieve. However, the allocation of the passageways near each other in a sequence did not work (see Figure 5).

The second proposal is the block with multiple broader passageways. The applied sequence is: Basic Block→ Random breaks (delete larger area) → Towers; resulting outputs set: 200 options. In this iteration, it is noticeable that the depth manipulation as an action type is missing (see Figure 6).

The third proposal adds yard buildings to the perimetral block and cuts the southern corner. The sequence of actions: Basic Block → Random Breaks → Yard buildings; resulting outputs set: 200 options. Here as well the precise allocation of the passageways is missing (see Figure 7).

In the second scenario, the goal was to evaluate if the method is efficient in terms of finding multiple diverse solutions in a short time. For this purpose, we used the following sequences of actions (see Figure 8). With the described semi-automatized generation approach, we achieved 10.000 designs in ca. 4 hours. (see Figure 9).

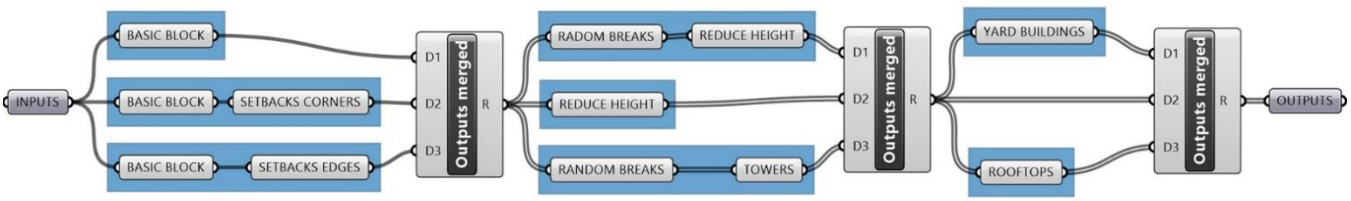


Figure 8. The sequences of actions applied for the second scenario of the test case.

As the goal of this scenario was to generate options as diverse as possible, we adjusted the inputs for each of the actions to ensure that we allow the required diversity. However, it is also possible to use readily combined sequences, where only the context inputs must be adjusted, and thus receive the outputs fully automatized in ca. 15 minutes. As follows, the interactive semi-automated process, where the designer controls the generation process, allows the application of one's experience and conceptual insights for the particular project case. The semi-automated process has the potential to reduce the number of generated designs to a user-friendly subset that does not need to be reduced further. However, by using the fully automated approach, a large number of design variants can be utilized by other participants of the planning process in order to apply additional evaluation criteria that were not included yet. When searching for a good solution among thousands of options, it is helpful to use exploration tools like the Design Space Exploration Framework [7] in order to find solutions with a set of defined properties. However, the goal of the second scenario test was to check if the diversity of the produced options is high enough. There are methods for grouping of similar objects based on pixel representation [7]. However, those methods only consider the two-dimensional space and ignore the third dimension - the height, although the height can have a great impact on the optical building perception. As follows, because of the missing method of distinguishing the visually diverse outputs, we manually selected three subsets of the generation outcomes. To highlight the diversity of the outputs with similar performance, we first filtered three different density groups from the generation population: for the first group the density of the competition entries was chosen (FAR = 2.22),

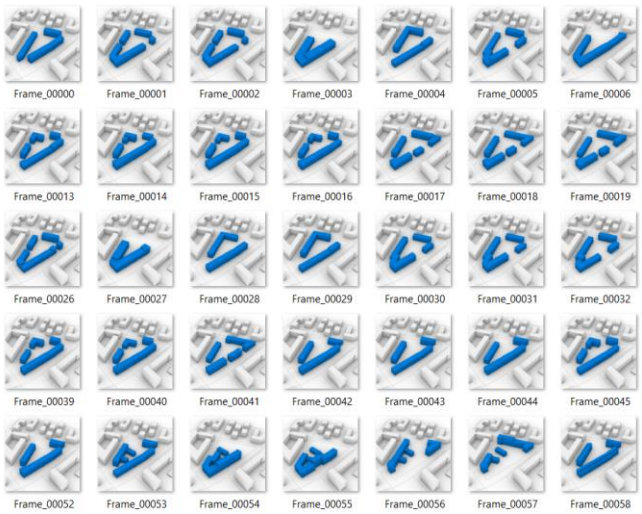


Figure 9. A subset of the 10.000 generations for the scenario 2.

another two groups were with lower density (FAR = 1.70) and high-density (FAR = 3.0). Each filtered subset has a density variation in the boundaries of +-0.05 from the goal value. From each subset (400 options for FAR 2.2, 1200 options for FAR 3.0 and 120 options for FAR 1.7) 5 optically diverse options were selected for demonstration (see Figure 10). The optical distinction between variants, all of which meet the formal criteria of the FAR index, proves that the broad exploration of the possible solutions at the early design stage is essential.

5 DISCUSSION AND OUTLOOK

In this paper, we presented a methodology for the efficient semi-automatic generation of multi-family-residential building designs. The principal goals for the developed methods were a quick generation of design variants (efficiency), exploration of the design space, and extended means of interaction for user influence on the generation process.

We believe that the developed parametric workflow for an automatic generation of design options allows for a more efficient exploration of the design space by avoiding manual form modeling and by a direct connection of the design outputs to the broad scope of spatial analysis and simulations thus showing the performances of the solutions immediately. According to the results of the test case application, we consider the developed framework to deliver a sufficient number of diverse designs (see Figure 10). We as well proved the possibility to consider the formal design aspects during the generation by achieving the outputs that visually resemble the manually designed competition entries.

However, the presented method has several limitations. Because of the nature of the parametric design methods, the design variants are predefined by the type and value-range of the used parameters. There are no surprising solutions as it would be possible by using optimization-based systems with a more advanced data-representation of designs [5]. As well, because of the initial narrowing of the design space to a limited set of possible solutions, better performing variants might be excluded in advance.

To increase the possible design space, we need to introduce the two remaining typologies (slab and solitary) and enable their combinations with the block. We also plan to further advance the options for the user interaction. In order to assure possible integration into an architectural workflow, we as well need to include the next levels of detail, such as the subdivision of the volumes into apartments and the further floorplans elaboration. Those levels of detail are as well necessary in order to conduct precise analysis and simulations for further qualitative design performance evaluation.

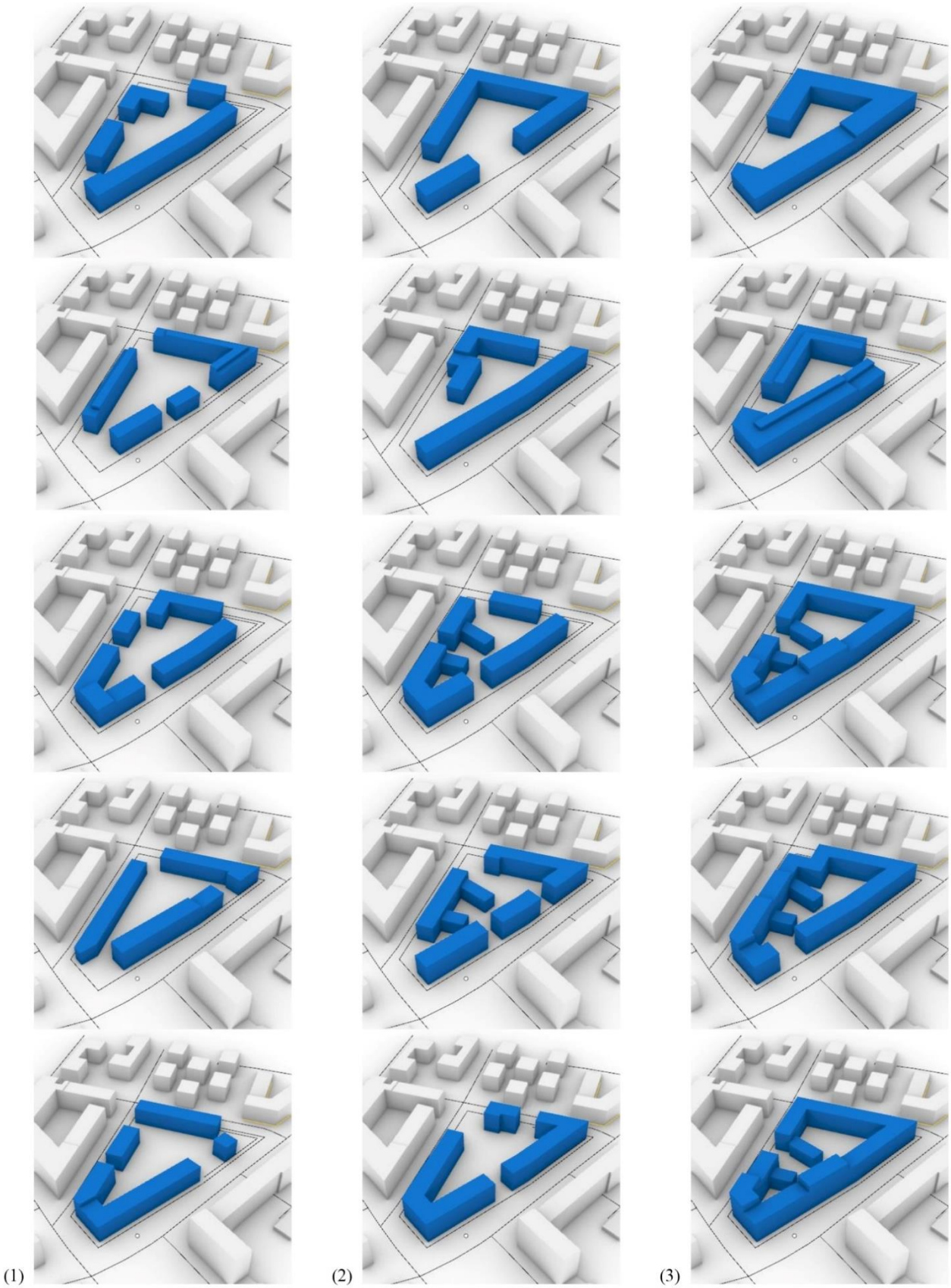


Figure 10. Examples of the designs for 3 sub-sets of the generation outputs: (1) FAR = 1.7 +- 0.5; (2) FAR = 2.2 +-0.5; (3) FAR = 3.0 +-0.05.

ACKNOWLEDGEMENTS

We would like to thank Egor Gavrilov for his contribution during the methodology development. We gratefully acknowledge the grant from research program “Zukunft Bau” of the German Federal Ministry of the Interior, Building, and Community together with DIPLAN Gesellschaft für Digitales Bauen und Planen GmbH and Decoding Spaces GbR.

REFERENCES

1. Abdumawla, A., Bielik, M., Buš, P., Mei-Chih, C., Dennemark, M., Fuchkina, E., Miao, Y., Knecht, K., König, R., and Schneider, S. Decoding Spaces, 2019.
2. a+t research group.Why density? Debunking the myth of the cubic watermelon. a+t architecture publishers, 2015.
3. Buerkin, T., Peterek, M., Petereck, M., Stadtbausteine. Birkhäuser, 2016.
4. Calskan, O. Parametric design in urbanism: A critical reflection. *Planning Practice & Research 32*, 4(2917), 17-443.
5. Dillenburger, B., M. Braach, L. Hovestadt, Building design as an individual compromise between qualities and costs, *CAAD Futures (2009)*, 458 - 471.
6. Finch3d [Webste/Tool]. Retrieved from https://finch3d.com/video-blog/ on 24.09.2019.
7. Fuchkina, E., Schneider, S., Bertel, S., Osintseva, I. (2018, September). Design Space Exploration Framework - A modular approach to flexibly explore large sets of design variants of parametric models within a single environment. *In Proceedings of the 36th eCAADe Conference* - Volume 2 (pp. 367-376).
8. German Building Regulations, retrieved from http://www.bauordnungen.de/html/deutschland.html on 28.09.2019.
9. Kreo [Webste/Tool]. Retrived from https://www.kreo.net/ on 24.09.2019.
10. Knowles, R.L., Berry, R. D., Department of Energy & Solar Energy Research Institute, Solar envelope concepts: moderate density building applications: final report. Solar Energy Information Data Bank, Golden, Colo, 1980.
11. Moudsari, R. S., Pak, M., Adrian Smith + Gordon Gill Architecture. Ladybug: a parametric environmental plugin for grasshopper to help designers create an environmentally-conscious design. *Proceedings of International Building Performance Simulation Assosiation, Chamberý, France* (2013), 3128-3135.
12. Nagy, D., Villaggi, L., and Baenjamin, D. Generative urban design: Integrating financial and energy goals for automated neighborhood layout. *SimAUD 2018* (pp. 247-254).
13. Openstreetmap [Website]: https://www.openstreetmap.de/, retrieved on 24.09.2019.
14. Parish, Y. I. H., Mueller, P., Procedural modeling of cities, *In Proceedings of ACM SIGGRAPH 2001*, ACM Press, E. Fiume, Ed., (2001), 301-308.
15. Reicher, C. Städtebauliches Entwerfen. Springer, Vieweg (2017), 54 - 81.
16. Results of the residential competition in Munich: https://www.competitionline.com/de/ergebnisse/241367 retrieved on 24.09.2019
17. Robert MCNeel & Associates. Grasshopper, 2014.
18. Robert McNeel & Associates. Rhinoceros 6.0, 2019.
19. Spacemaker [Webste/Tool]. Retrieved from https://spacemaker.ai/ on 24.09.2019.
20. Testfit [Tool]. Retreived from https://blog.testfit.io on 24.09.2019.
21. Wilson, L., Danforth, J., Davila, C. C., and Harvey, D. How to Generate a Thousand Master Plans: A Framework for Computational Urban Design, *SimAUD 2019* (pp. 113-119).