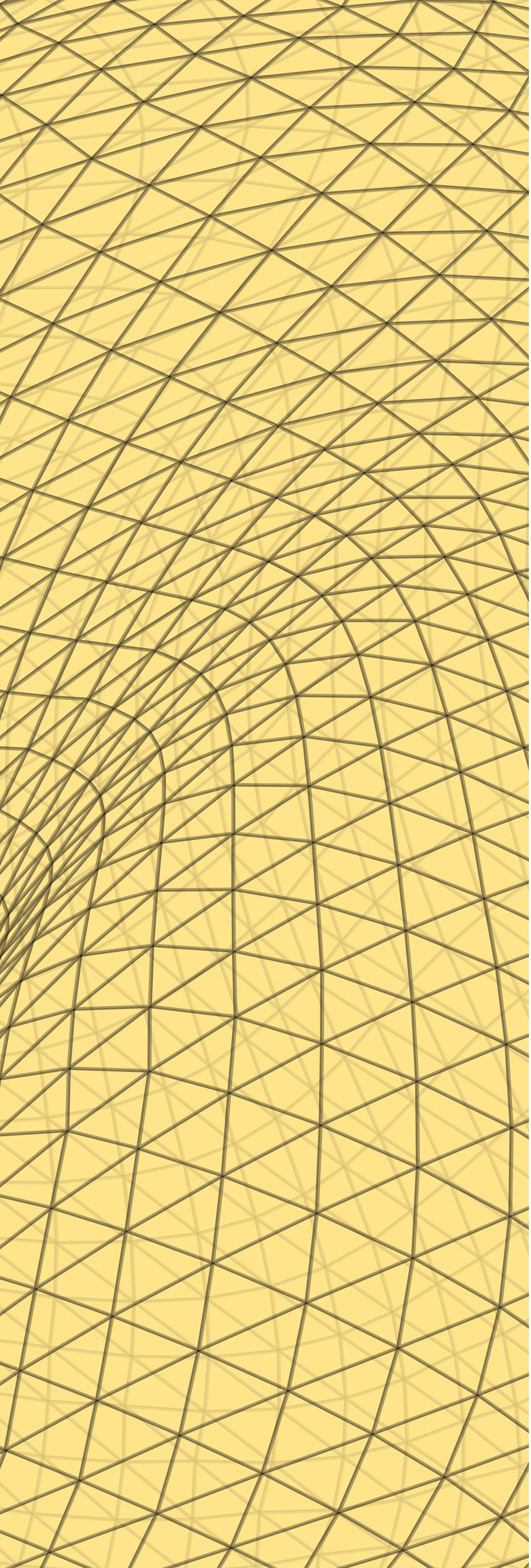
FLOOR PLAN GENERA-TION

Raban Ohlhoff - 000457528
Master in Architecture

Architecture and Design
Faculté d'Architecture La Cambre-Horta
Université libre de Bruxelles
Belgium - Brussels
May 17, 2022

Contents



Introduction

The increasing application of computer science in a wide range of industries has become a reality since the vulgarization of the technology and the facilitated access of a broad public. Applications are as diverse as the underlying mechanisms, ranging from simple code-based optimizations to simulations in all directions to fully digitized frameworks. That scientific fields such as natural, structural, economic, and engineering sciences benefit from increasing digitization is fairly natural and easily explained by the relevance of mathematical modeling, computational simulations, and the need for significant computational power, but this interaction is far less intuitive when considering the example of artistic or humanistic fields. This report examines the interaction between computer science and the field of architecture, which is situated between science and art. In particular, the topic of automatic floor plan generation is explored as a starting point for exploring applications such as simulation, analysis, data manipulation, and machine learning based on the generated data. Each of these topics has been extensively researched by academia in recent years and therefore provides extensive documentation of scientific reports. This explains why this work is partly based on third party work, but attempts to bring each topic together in a relevant way to create a workflow that can potentially provide new insights. Due to the significant amount of existing research and information, a comprehensive review and thorough reading of relevant research is essential. In addition, a significant emphasis is placed on the interplay between computer-based algorithms and the inherent creativity of the architectural profession, which is difficult to combine with digital tools. Thus, this is neither a purely scientific work nor an exclusively experimental investigation, but rather the structured documentation of a problematic investigation with sensitive consideration of interdisciplinary aspects.

Abstract

This project attempts to generate a synthetic dataset of architectural entities using parametric modeling to enable automation within a defined range of variability. To achieve this, several steps with different software libraries and algorithms are necessary, as the results of each step have to be analyzed and verified against different features. First, the focus is on the parametric generation of an apartment floor plan using Python, Blender's Sverchok, and various algorithms such as Voronoi diagrams, KD trees, and genetic algorithms. After generating a manifold spatial configuration, it can be analyzed from different geometric aspects using the Python library Topologic. It is important that these two steps are in constant exchange to combine the geometric data with the evaluation of the spatial analysis and to store the information in a file format suitable for the data. These formats can range from simple two-dimensional files to database formats or graphical data. With the help of topological analysis, apertures such as doors and windows can be integrated into the basic geometry in a variable pattern within the framework of defined spatial and architectural rules. The final step involves reconstructing the data in geometric form into a three-dimensional model that is finally enhanced, improved, stored and displayed in a common open source BIM format such as IFC using IFCOpenshell, BlenderBIM, Topologic and Opencascade.

In addition, a requirement is that the dataset can be easily supplemented with physical and environmental analyses, such as the use of light and radiation-based simulations with Radiance, Vi-Suite, Honeybee and OpenStudio, or a simulation of the energy behavior of the architectural object with Energy+, Vi-Suite-Energy and the core component of OpenStudio in order to add an evaluation layer. The evaluated geometric synthetic data can thus be used as training data for a machine learning model and should be able to counteract the traditional bias thanks to the synthetic data origin, in contrast to the common use of real life data by real estate companies or architecture firms.

Special attention is paid to the consistent use of python-based libraries to ensure the best possible compatibility of the individual software interactions. Furthermore, only open source projects are used for didactic, ideological and compatibility reasons. During the execution of the individual steps, possible problems, suggestions, solutions, proposals for improvement and last but not least ideas for further research of the topic are recorded, which are described in detail in the following report.

The apartment layouts generated show a high variance and allow a high degree of intervention and control in the generation process thanks to the parametric rule-based generation method. The advantages in the application in connection with machine learning are convincing, but will only be proven in the continuation of this work in a direct comparison with models trained on conventional data sets.

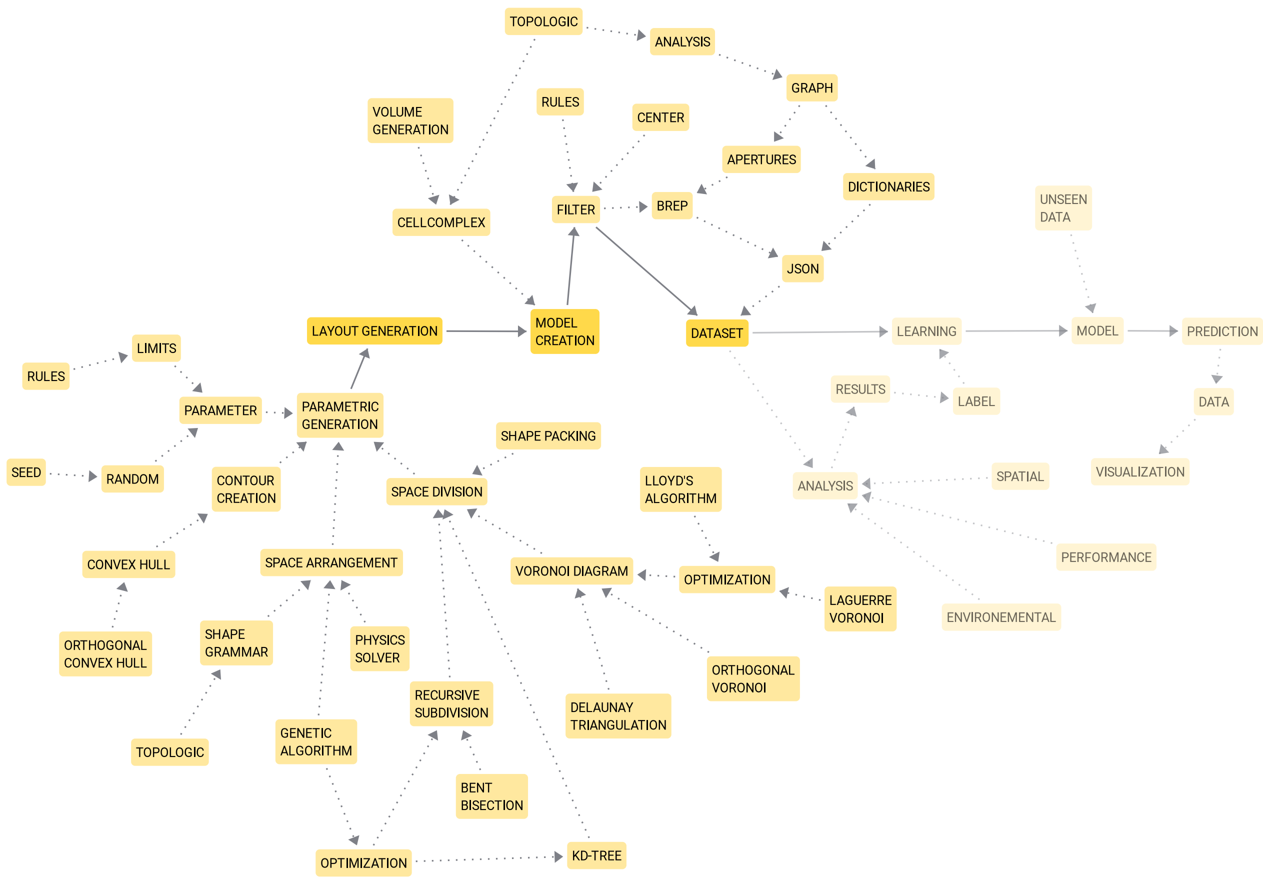


Figure 1: Process

Framework

This report is part of a long series of research on the interaction between machine learning and architectural design. The following project was developed within the framework of the Architecture and Design Studio at the Université libre de Bruxelles, which is characterized by an openness to new technologies associated with architecture. The focus of the studio is to raise interdisciplinary questions between technology, research and architecture that can be freely developed without having to remain in their limited domains.

The goal is to develop hypotheses about the future of architecture, taking into account all areas that may be of interest to the design process of the project. In every aspect of project development, it is important to consider the analysis of new scientific knowledge and explore the achievements of current science. However, it is also important to base experiments on inventions that have emerged throughout history.

In this studio, the primary goal is not to invent new objects, but rather to deepen the investigation of a particular topic using all existing publications and research to formulate a relevant research hypothesis. The goal is then to begin a process of experimentation accompanied by meticulous documentation. The goal of this process is to find answers to the questions raised in the preliminary phase. In this way, the experimentation cycle advances the research and possibly leads to a concrete solution, but in any case raises new questions that can subsequently be pursued.

The operation of the studio is closely linked to the FabLab of the Faculty of Architecture, which provides access to numerous tools for design and experimentation. This space will also serve as a library for the skills acquired by each individual and thus for the emergence of collective knowledge. As a result of the restructuring related to the Covid 19 pandemic, the workshop has become a paperless studio, which means that the visual representations and research objects are predominantly digital.

Hypothesis

The use of intelligent neural networks and machine learning trained models to optimize traditionally manual processes is now the norm. Machine learning is no longer limited to computer science, but extends to any field as long as a database to be analyzed is involved or can be created. It is not surprising, therefore, that learning models have also found their way into architectural optimization. However, the application of machine learning in architecture is far-reaching and can be useful in any project development process. For example, intelligent parameterization can help find the appropriate shape even before a concrete project is modeled; mechanical analysis of existing conditions and constraints can be helpful in determining approximate volumetrics. In addition, it is possible to use intelligent algorithms to generate the layout of the interior space, proposing several adapted plans that can lead to a qualitatively improved experience for the occupants. This optimization is not limited to the two-dimensional space and can therefore provide suggestions for optimal circulation or daylighting optimization throughout the building. In addition to the conceptual phase, it is also possible to optimize the BIM model through various machine learning algorithms. All these processes are no longer visions of the future, but have become the norm, albeit often automated and therefore not directly visible. This report will focus mainly on the application of geometric, pseudo-intelligent and evolutionary algorithms in the conceptual design phase to try to automate the generation of floor plans in order to obtain an optimized plan through subsequent steps.

The premise of this work is the assumption that there is a direct relationship between external conditions such as spatial connections, solar radiation, shading, humidity, wind flow, heat generation, soil conditions, air quality, pedestrian traffic or traffic load and the quality of housing perceived by the occupants.

The main hypothesis addressed in this report concerns whether and to what extent synthetic architectural datasets generated by different algorithms can simplify, accelerate, and/or optimize the architectural design process and to what extent training machine-learning models on synthetic datasets leads to diversity in the results. Is it possible to automatically generate spatially meaningful and architecturally sophisticated floorplan layouts? What are the advantages of synthetic datasets in architecture and what problems can be avoided? Furthermore, this thesis investigates to what extent it is possible to perform a complete workflow from generation to simulation, analysis, prediction and back to generation without having to resort to proprietary software, thus describing a step towards the democratization of architecture and its digital tools.

State of the Art

Since this project deals with diverse topics, the collection of previous works is accordingly divided into hierarchical subgroups. First, the role and emergence of computational design with parametric, generative and algorithmic design as subgroups should be considered. Its origins go back to 1960, when Sutherland's SKETCHPAD software was a major step towards the automation of architectural drawings and the digital parameterization of the relationships between individual geometric units. In the following years, several individual approaches to building information modeling inspired by important computational design conferences developed. With the establishment of various computer-aided design software, the way to visual programming interfaces was paved and with it the access to parametric and generative design tools for the masses. In this work, the geometric parametric tools apart from backend python coding will focus on the visual interface of the Sverchok addon for the three-dimensional processing software blender, which makes use of various python libraries. The first goal of this work, the automatic generation of synthetic floorplans, is as a topic widely researched and this with many very diverse tools, which can be roughly divided into simple algorithms, intelligent algorithms and machine learning enhanced algorithms. The group of simple algorithms is subdivided into geometric space partitioning by Voronoi diagrams and its derivatives like Delaunay triangulation, Lloyd's Algorithm, Orthogonal Voronoi Diagram and weighted Voronoi Diagrams. Also mentioned are aproximating schemes for subdivision such as Cutmull-Clark and interpolating ones such as Butterly subdivision Surfaces. Kdimensional trees, originally from the family of such algorithms, differ from the previous ones, despite the essential similarities, by a subdivision of input vertices as opposed to polygons, which thus allows a sensitive control over control points. The method of slicing tree structure forms the connection between point and area oriented subdivision. Another method of simple algorithms is the shape grammar methodology which allows rule based geometry generation and thus space for variation in defined limits. This topic has gone through many variations such as the CGA shape or parametric shape grammar and is recognized as a programming language in its own right through successful simulation of turing machines. Far enough away from these methods to define themselves as a group of their own are physics solver based methods like attraction force models or MagnetizingFPG algorithm to some degree. In this paper, however, we will mainly focus on the latest and more interesting topology based method developed by Wassim jabi. Topologic is not a layout generation method but rather an opencascade based geometry processor that processes non-manifold topologies and thus reduces the architectural spaces to simple cells and cell complexes. This method allows layout generation by combining the above mentioned methods and paves the way to a simple geometry to file workflow. Furthermore, topologic's boundary representation method facilitates environmental simulations for the subsequent plan evaluation stages. However, by far the most significant advantage of using topologic is the analytical approach to continuous space and relations awareness, which allows the arbitrary addition

of apertures such as doors and windows, as well as the graph generation of various topological parameters and last but not least an interface to common BIM file types such as IFC, BREP and JSON.
More important in the literature of floorplan generation than the mentioned approaches are intelligent methods like evolutionary algorithms since the 1990s. These are input populations that go through biology-inspired mechanisms such as reproduction, mutation, recombination, and selection through fitness evolutionary phases, resulting in an optimization of the fitness function. Evolutionary algorithms can be combined with simple geometric methods by means of goal definition and can also be used for optimization by means of evaluation analyses. Computational intelligence also includes machine learning and neural networks, which are by far the most widely used methods in the field of automated floorplan generation. However, the above methods differ drastically from artificial neural network applications in that the latter are based on the learning of features in defined training datasets. In the context of plan generation, these datasets are real plans designed by humans. Thus, the computer generated floor plans resemble the input drawings but leave room for differentiation which is limited by the learning process based on existing plans or evaluation based on suitably defined examples.

Process

This work is an experimentally exploratory approach, meaning the primary goal is not to achieve an optimized process, but rather to critically analyze the individual steps. Thus, by repeatedly questioning the method, insights can be gained that will be beneficial in the following phases. Moreover, an essential focus lies on answering the formulated questions presented in the hypothesis, which means that the individual steps should be reected on several layers in order to gain not only technical, but also moral, ethical, and social insights.
First, the topic of parametric automated plan generation must be addressed in depth. In today's world, artificial intelligence is used as a kind of selling point, a solution to all complex problems, which raises the question of whether this assessment is truthful, or is this term simply associated with idealized solutions? Once these questions are answered and a concrete concept has emerged from the abstract term, it becomes possible to think about connections between architecture and machine learning that simplify existing design processes, simulations or constructive procedures.
Just as important as a clear understanding of the subject matter an in-depth review of the available libraries and their functions, in order to create a customized network diagram covering the interactions. The open source community, through platforms such as Github, Gitlab, and OSArch-community, provides an appropriate and direct exchange with developers and interested individuals and a complete understanding of the functions and procedures, as well as, in most cases, detailed documentation. With the help of various forums and exchanges with developers, it was possible to gain a comprehensive understanding of the software in question in a relatively short period of time, thus advancing the main intentions.

Objectives

The experimental freedom explained above also leads to flexibility in terms of the defined aims. In general, there are objectives for each stage, but this does not mean that a step has failed if they turn out differently than expected or formulated in advance. Thus, the stage of parametric generation has as a desired result the generation of synthetically generated models that can be constrained by certain parameters. These can be the number of living rooms, the number of residents, the area, the volume or the shape of the floor plan. The end result of this phase should be a database of the different geometries that describes each situation as accurately as possible while still requiring a minimum amount of points. In the next stage, the simulations would generate new evaluation values that could be added to the geometric database. Further, the next stage is to train a model that describes the relationship between the individual values as accurately as possible by means of a graph. Finally, the last stage is to provide an accurate visualization of the predicted scenario.

Roadmap

- ☒ Information Gathering
  - ☒ Resources
    - ☒ Code Repositories
    - ☒ Academic Papers
    - ☒ Books / Reports / Articles
    - ☒ Student Works
  - ☒ Bibliography
  - ☒ Adequate Software
    - ☒ Code Documentation
    - ☒ Explanatory Resources
- ☒ Geometric Generation
  - ☒ Methods
    - ☒ Algorithm
      - ☒ Evolutionary Generation
      - ☒ KD Tree
      - ☒ Physics Simulation
      - ☒ Shape Grammar
      - ☒ Shape Packing
      - ☒ Convex Hull
      - ☒ Voronoi
      - ☒ Orthogonal Voronoi
      - ☒ Lloyd
      - ☒ Delaunay
      - ☒ Power Diagram
      - ☒ Polygon Division
      - ☒ Recursive Subdivision
      - ☒ Recursive Bisection
    - ☒ Machine Learning

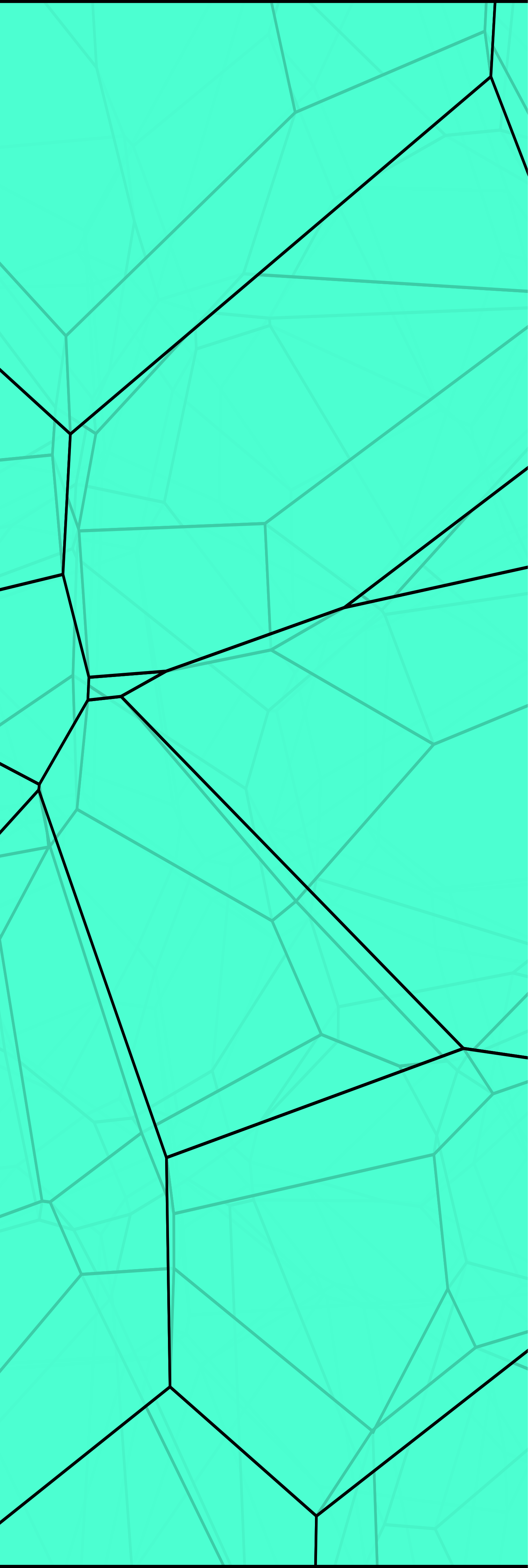
            - ☒ Neural Network
            - ☒ Classification
    - ☒ Parametric Generation
        - ☒ Sverchok
        - ☒ Python
    - ☐ Evaluation
        - ☐ Data Set Search
        - ☐ Comparison
        - ☐ Esthetic Verification
        - ☒ Functional Verification
        - ☐ Ideological Verification
            - ☐ A Pattern Language

- ☐ Simulations / Analysis
    - ☐ Environmental
        - ☐ Energy
            - ☐ Energy+
            - ☐ Openstudio-Energy
            - ☐ Ladybug
            - ☐ Vi-Energy
        - ☐ Lightning
            - ☐ Radiance
            - ☐ HoneyBee
            - ☐ Openstudio
            - ☐ Vi-Radiance
        - ☐ Air
        - ☐ Crowd
        - ☐ Urban
        - ☐ Life Cycle
    - ☐ Spatial
        - ☐ Topologic
    - ☐ Structural
    - ☐ Usage
    - ☐ Data Type Examination
    - ☐ Verification

- ☒ Data
    - ☒ Manipulation
    - ☒ Data / Database Types
    - ☒ Structure

- ☐ Training
    - ☐ Methods

- ☐ Visualization
    - ☐ Geometry Viewer



Layout Generation

The first stage of this work deals with the automated generation of different apartment layouts using various geometry processing software. An alternative to the generative approach is the acquisition of existing architectural datasets provided by real estate or research groups. Advantages of reality-based datasets are the certainty of architectural feasibility and construction of the individual plans, but disadvantages are a conservative approach in creative terms and a high risk of traditional bias due to the predominance of local and traditionally influenced architectural methods. Therefore, this experimental phase is primarily concerned with testing different floorplan generation methods and their derivatives as well as their possible combination. Simple algorithms like voronoi diagrams, intelligent methods like evolutionary algorithms and simulation constrained methods like physics solvers are explained and evaluated through different experiments. Criteria for the evaluation are simplicity in the construction process, computing power, time and memory, integration and interaction with used software, purity of the generated geometry, degree of readiness for the simulation stage, simplicity of the layout for data storage but above all spatial quality, creativity in form finding and feasibility. After experimentation, the most appropriate method or combination of algorithms for the following steps is selected and an appropriate set of different two dimensional layouts is generated. In order to increase the variation in the synthetic dataset, results from different methods will be mixed, but they must not deviate from the generally defined quality level. Furthermore, it is significant to understand the functionality of the algorithms sufficiently to allow adaptations and combinations with other functions and to achieve variation in the individual applications. In general, simplicity is preferred to complexity, creativity to ordinariness and variation to repetition.

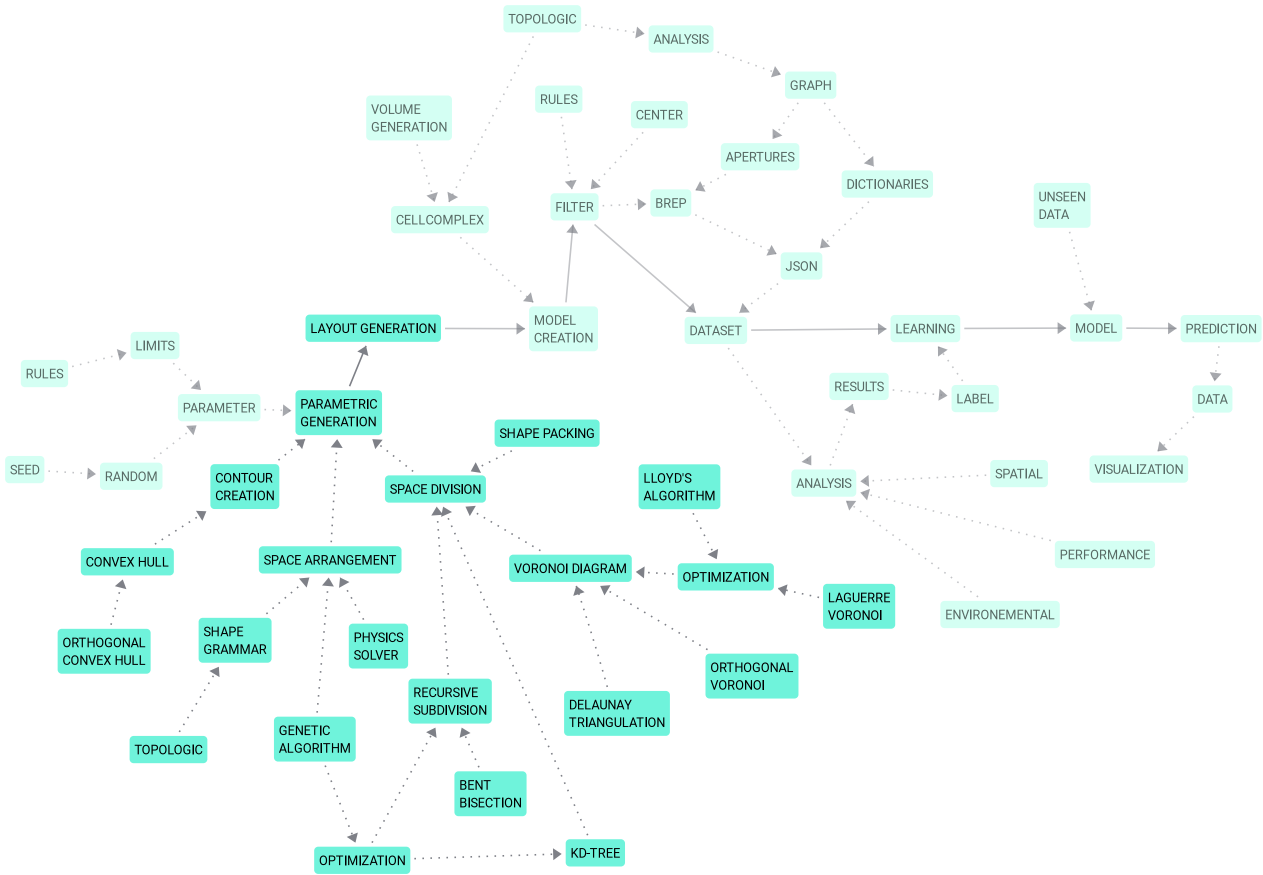
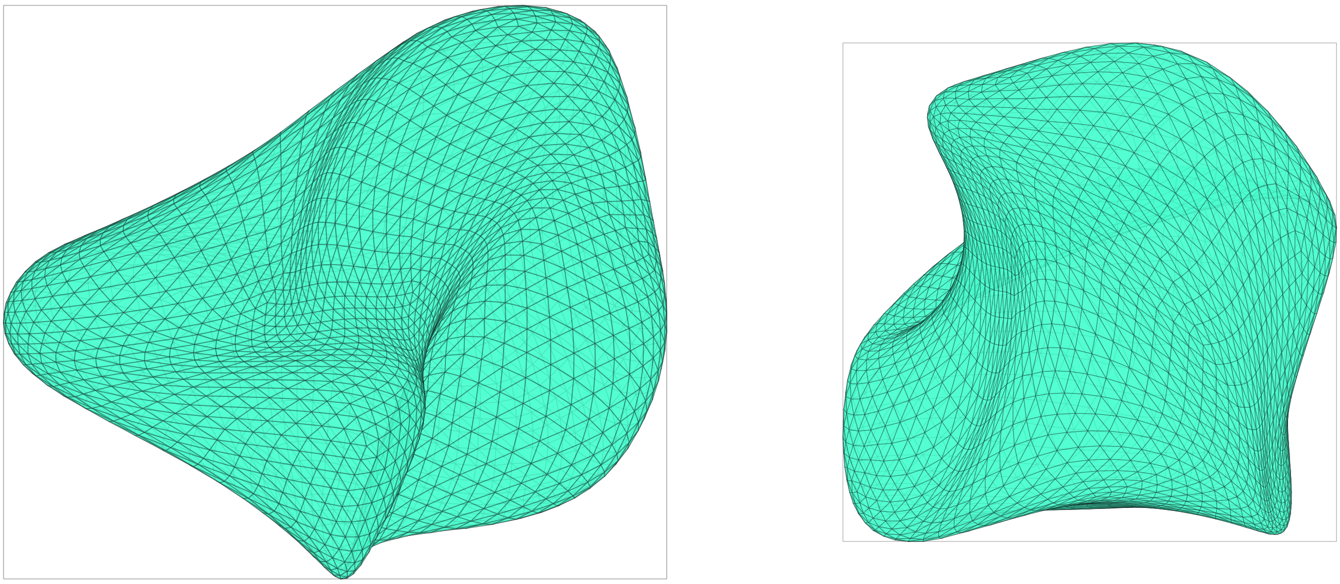


Figure 2: Layout Generation

Genetic Algorithm

The Genetic Algorithm developed by Holland and De Jong is an optimization model based on Darwin's natural selection-based theory of evolution in biology. Its operation is based on biological mechanisms such as crossover, recombination, mutation and selection by fitness evaluation acting on a population of defined size. Due to their mode of operation, genetic algorithms and their derivatives are particularly adaptable and offer a parallelization of the solution finding. However, for optimal application, the right parameters have to be set, such as crossover and mutation rate, population size, iterations and fitness boost. These parameters are problem-dependent and there is a risk of an undesired and non-optimized result if the starting conditions are set incorrectly. The construction of an evolutive algorithm begins with the determination of the variable function to be optimized, which consists of an unlimited number of components and has a tendency towards zero. After the framework conditions of the algorithm have been defined, the variables to be varied must also be determined, whereby it is important to ensure a proportional relationship between parameter variance and fitness evaluation function. In Figure 16, the surface area of the X- and Y-dimensional bounding box of an irregular three-dimensional body was defined as the fitness function. The population of the genetic algorithm acts on the three Eulerian rotation axes of the body and thus achieves a minimization of the Z-section of the irregular shape by iterative mutation and fitness evaluation.



(a) (b)

Figure 3: (a) Initial Rotation (b) Optimized Rotation

With increasing iterations and an increase of the population number, the optimum is progressively approached, but the question arises how such a method works theoretically and how it proves itself on more complex problems? In Python there are several established evolutionary algorithm libraries with different integration mechanisms ranging from native python implementation to Keras and Pytorch or even scikit-learn integration. Since this work is primarily focused on generation and manipulation of visual and three-dimensional entities, the python natieve Blender implementation of the sverchok Genes solver was chosen as the experimentation tool. This provides a seamless integration into a visual scripting environment and thanks to sverchok's Blender integration, Blender's features such as the variety of available export file formats can be easily used.
In order to test the floorplan-layout-generation capabilities of the evolution algorithm in Blender, different rectangles with defined X and Y sizes are first generated parametrically, each of these space representations being anchored by its origin location and rotation in two-dimensional space. The shape of each unit remains constant and represents the space sizes of the different apartment components. However, the lacolization and orientation of these modules remains free and is defined as a population parameter.

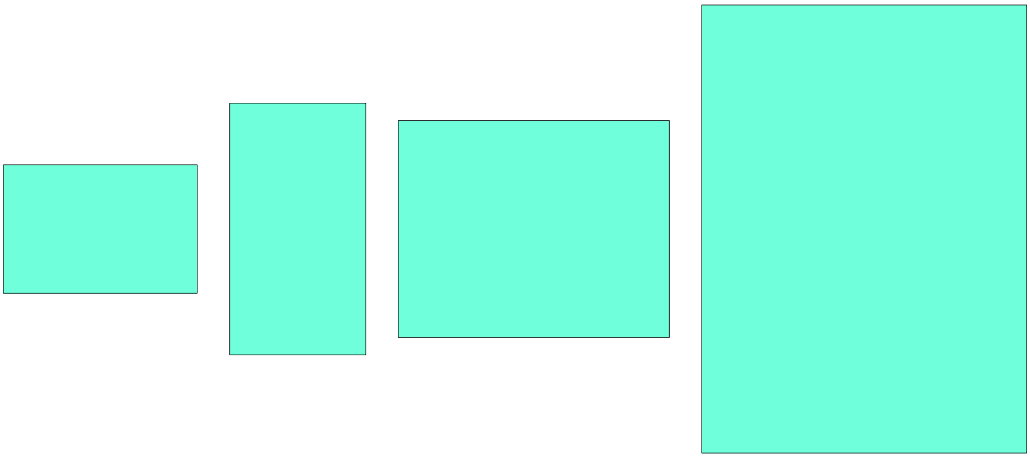
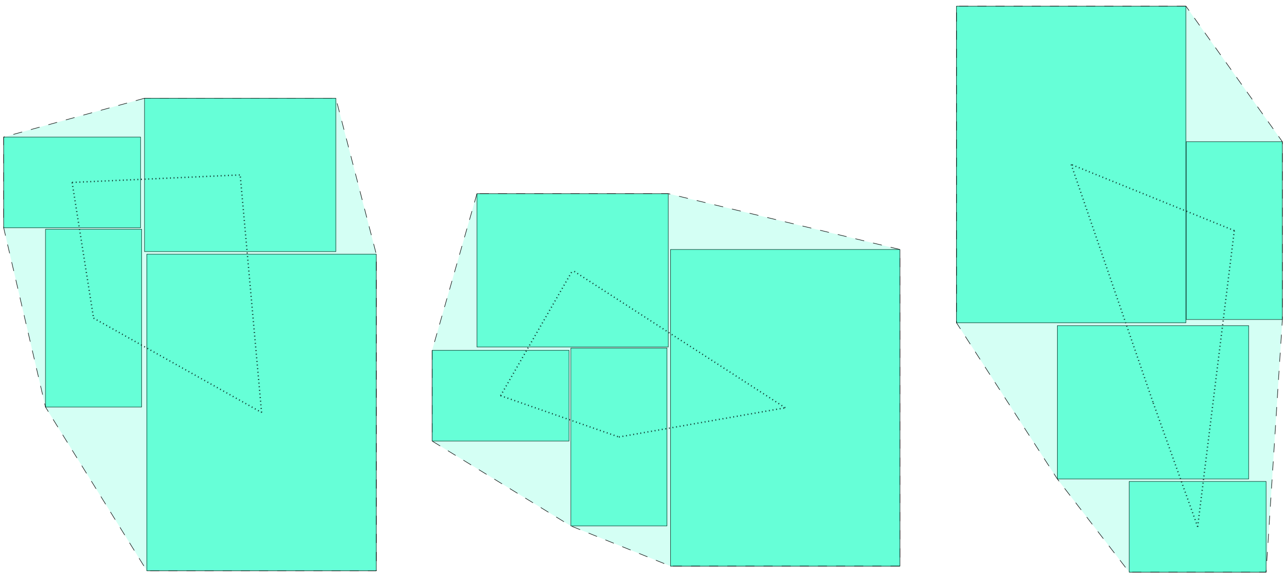


Figure 4: Inital Layout

The fitness function is composed of the total length of the path of the UV connection of the individual rectangle origins, added to the total area of the two-dimensional convex hull with respect to the Z-axis of the overall geometry. Thus, the smaller distance between the individual units is rewarded in parallel by the total area and the individual distance of the center points, while also avoiding the overlap of the individual rectangles. This is achieved by an irradiation function which evaluates the number of geometries formed by a constant check of the boolean intersection to see whether the total number increases. If this is the case, a defined irradiation variable is added to the fitness function. The population size is set to 500, the fitness boost to 5, the mutation rate to 0.3 with an iteration of 5.



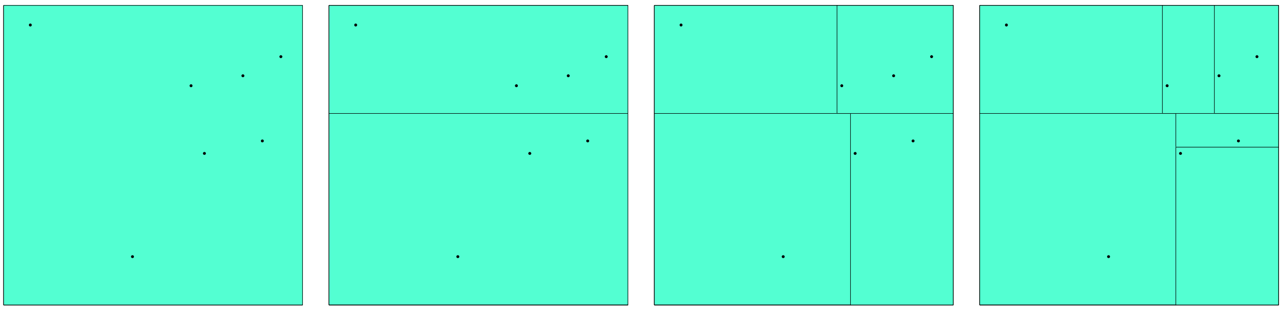
(a) (b) (c)

Figure 5: (a) (b) (c) Optimized Layouts

The different results generated parametrically by changing the random seed of the algorithm give random results with an amazing variance. Advantages of this method are the possibility of parameterization during the execution of the algorithm, a parallelization of the optimization, the possibility of defining different spatial relations by multipliers, the adaptability of the framework conditions and a variability of the obtained results. However, the main disadvantage is the constant distance between the spaces, which complicates the generation of boundary representation geometries and thus the complexity with respect to later environmental simulations or topological analyses. Furthermore, this increases the impurity of the geometry files to be stored. Possible solutions to this problem are a change of the framework and an extension of the irradiation function due to the overlap of the single units, which could integrate the size of the overlap surface. Furthermore, it would be possible to implement a second method that could lead to the cleanup of the overall geometry, but there is a risk of altering the basic geometry by causing alternating angles.

K-D Tree

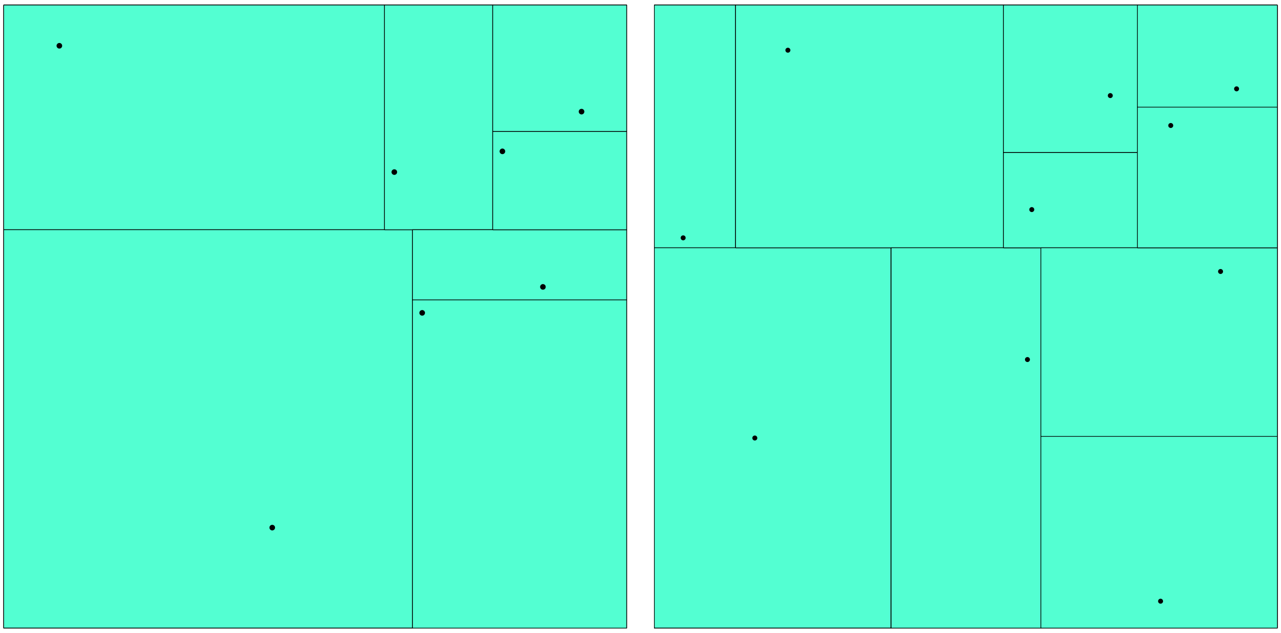
The k-dimensional tree data structure is a space partitioning primarily used in computer science for search algorithms and thus belongs to the family of binary space partitioning trees. It is the partitioning of pointclouds in a k-dimensional space. In this work I will limit myself to the two-dimensional space, since the floorplan layout can be described sufficiently in this dimension. In such a data structure, the dataset is divided into branches by nodes and branches, creating successive levels. Each of these branches leads to a leaf, which carries the coordinates of exactly one point.



(a) (b) (c) (d)

Figure 6: (a) 0 Iterations (b) 1 Iteration (c) 2 Iterations (d) 3 Iterations

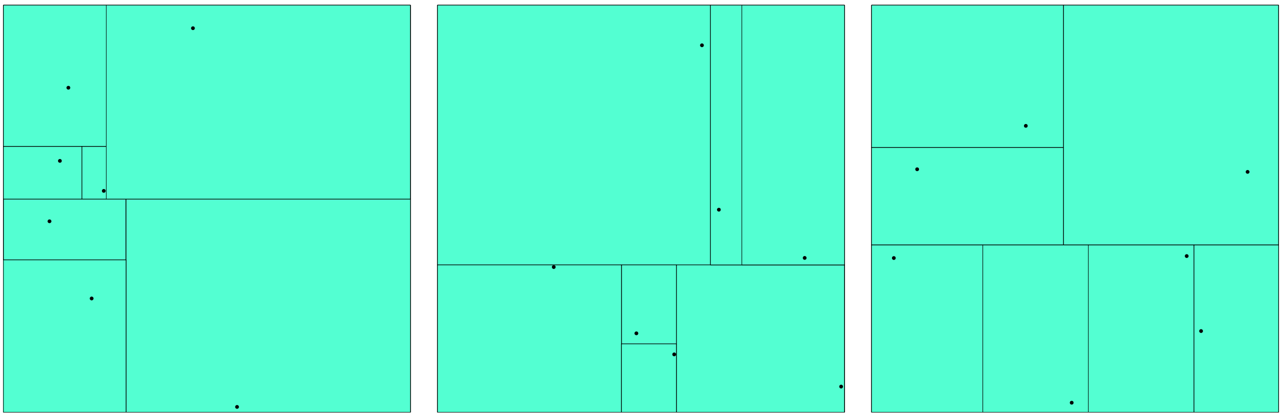
The functionality of such a data tree can vary, but the basic principles remain the same and can be implemented in a simple way in python without using external libraries, building the tree from the root upwards. First, the entire set of data points is considered and a sorting axis is determined based on the depth of the points. After these points have been sorted according to the axis, the node point is determined. This is located on the median of the point list and determines the coordinates of the subarea, which in the following step divides the point list into two child lists. The orientation axis of this intersection is also determined by the depth of the dataset to be split. If the data set to be divided consists of only one point, the leaf of the tree is reached and the iteration is terminated at this node. Thus, by repeating these steps, the complete tree is created using the logarithm.



(a) (b)

Figure 7: (a) 7 Room Layout (b) 10 Room Layout

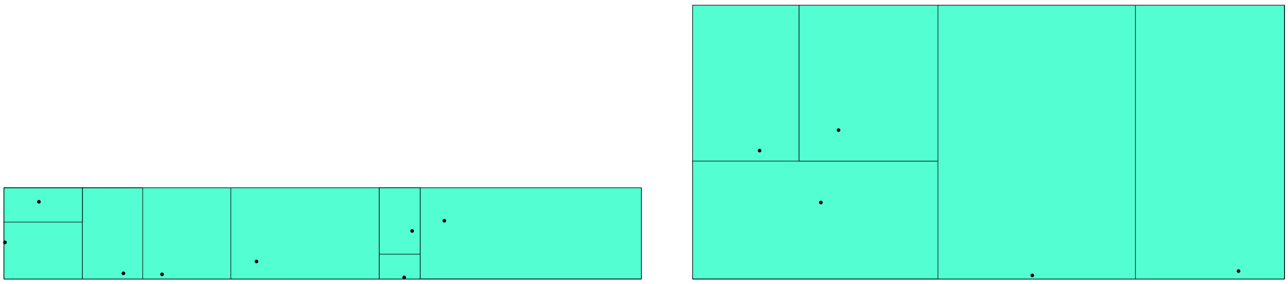
In Blender's Sverchok, creating a spatial K-d tree partition is relatively simple and requires only list manipulation, mathematical operators, slicing planes and loops. The points to be split and the corresponding planar area can be freely defined and are described in the examples shown by X and Y size defined rectangles and randomly selected points on this area. To avoid too small sheet spaces, a minimum distance between these points is defined. The number of these points is unlimited and can be defined accordingly, whereby their number determines the number of spaces of the floorplan.



(a) (b) (c)

Figure 8: (a) (b) (c) K-D Tree Layouts

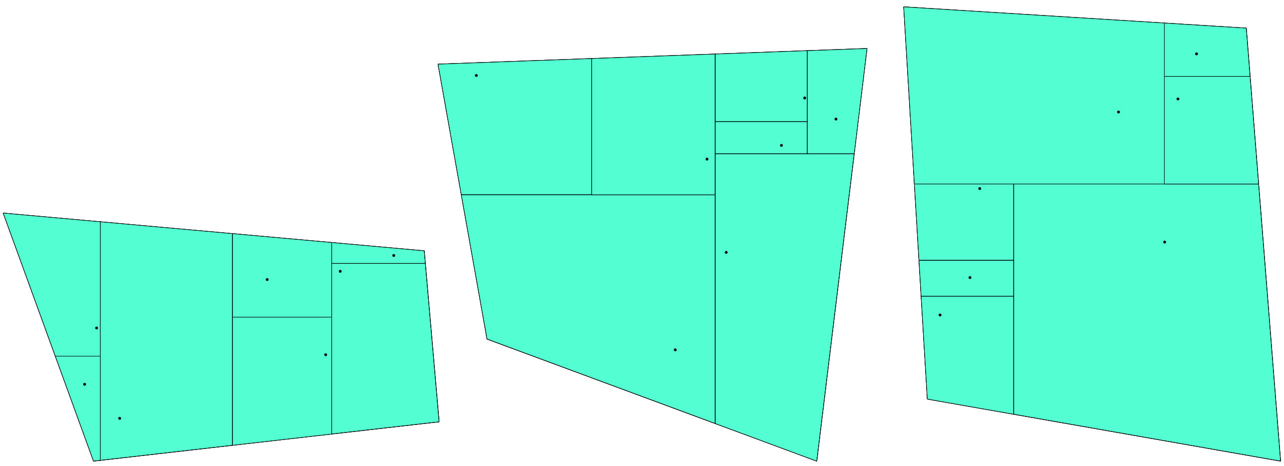
The main advantages of the described method are its speed and simplicity, the generation of natively connected units which simplify the export to boundary condition geometries, the possibility to parameterize the generation process, the determination of localizations of the individual spaces and the choice of the underlying volumetry of the contours of the plan.



(a) (b)

Figure 9: (a) (b) Rectangle Variations

The outlines of the individual shapes can assume any shape of the rectangles and are thus adaptable to the different frame conditions. Furthermore, it is possible without problems to determine more complex shapes as input geometries, from deformed rectangles to irregular polygons. However, this method also has its drawbacks, such as the difficulty of generating composite apartments in which individual units protrude beyond the floorplan contour as isolated shapes. Thus, it becomes complicated to generate shapes such as terraced garages or irregular facades.



(a) (b) (c)

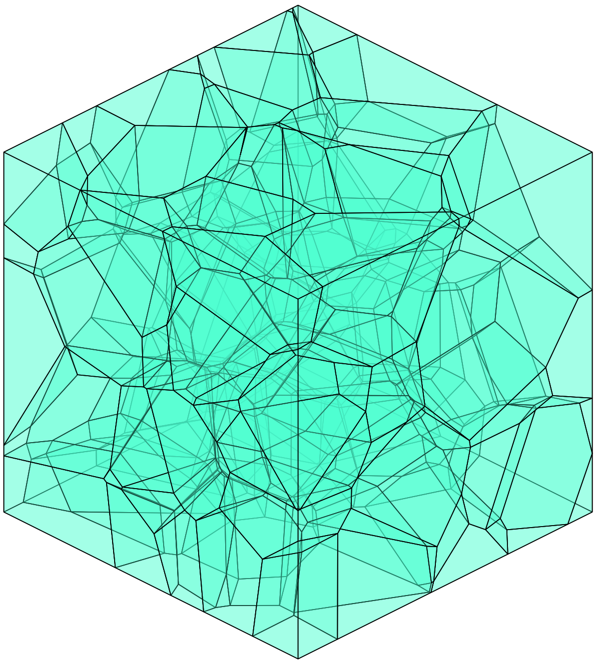
Figure 10: (a) (b) (c) Polygon Variations

KD-Tree and Genetic Algorithm

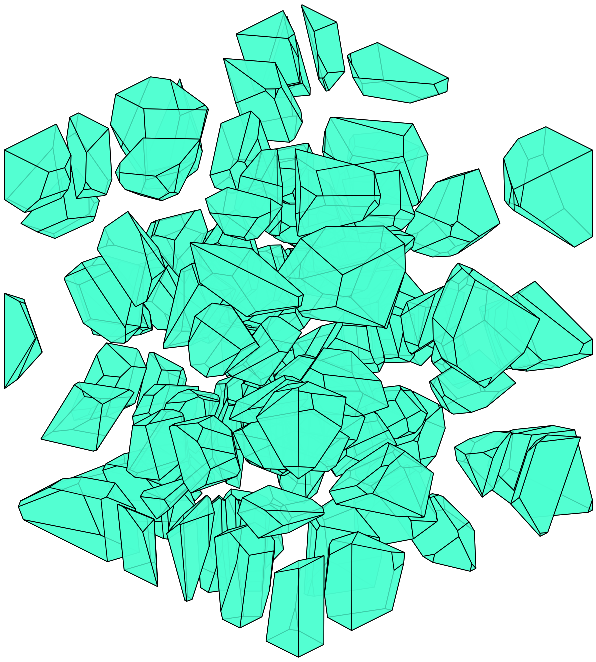
A possible combination between the k-dimensional space partitioning method and the evolutionary algorithm could be to optimize the variables of space contour, space number, but especially the point coordinates with a user defined fitness function according to the desired results. Such a combination would optimize the floorplan generation according to the objectives, but it would not solve the above mentioned problems.

Voronoi Diagram

The Voronoi diagram is the decomposition of a defined body into so-called Voronoi regions. Here, points in the nth dimension are defined as the centers of the subdivision and the spans of these spaces are all points that are closer to its center than to any other point. Thus it is possible to subdivide any surface or volume into regions and avoid overlap or spacing between the subdivisions.



(a)



(b)

Figure 11: (a) Spatial Voronoi (b) Separated Spatial Voronoi

For a successful Voronoi subdivision of a geometric body, only the initial geometry has to be defined, the dimension and the subdivision method have to be chosen, and last but not least, the points of the Voronoi centers have to be determined. If possible, these points should be located on the basic body to be subdivided according to the dimension, but they can also be projected onto it by a defined thickness in the second dimension. With the Populate mesh node, these points can be placed on the body using different distribution methods.

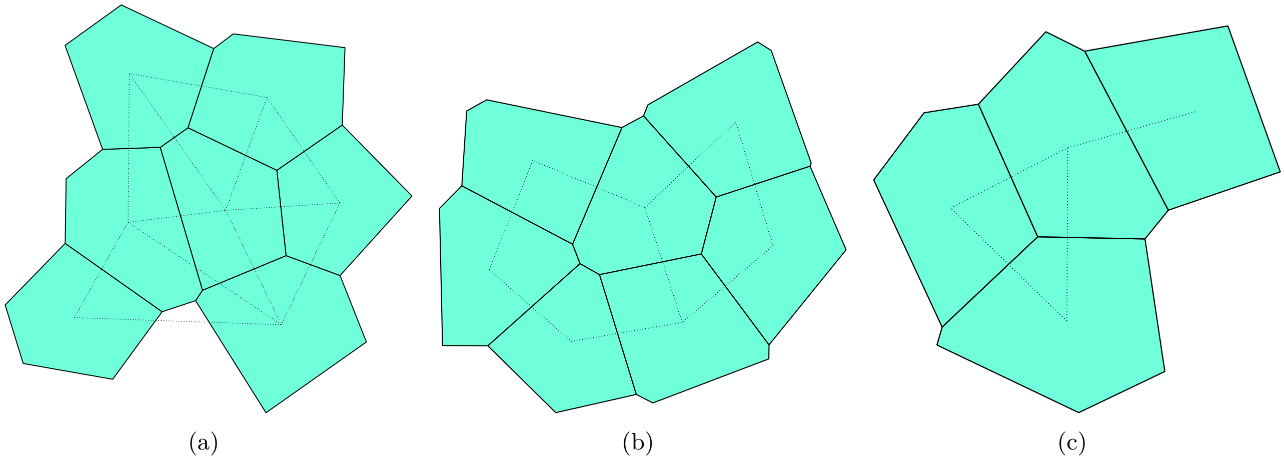


Figure 12: (a) (b) (c) 2D Voronoi Diagrams

The resulting room partitions can be varied with the advantage that the weight point of each room can be defined in advance. Basically, two different floorplan typologies can be generated. Those whose contour is predefined and therefore the process is directed from shape to interior partitioning and those whose geometric shape is determined by the interior partitioning and therefore the contour depends on the number of rooms and position of the seeds. The main difference is the organic irregular contour of the latter typology, which tends to vary but is difficult to reconcile with a predefined building framework.

Delaunay Triangulation

The seed point set that determines the Voronoi regions can also be represented in a mesh. The most common of these mesh representations is directly geometrically related to the Voronoi diagram and is called Delaunay triangulation. More precisely, the delaunay mesh describes the dual graph of the voronoi pattern. Every single triangle side intersects a Voronoi edge at a right angle and this exactly at its center.

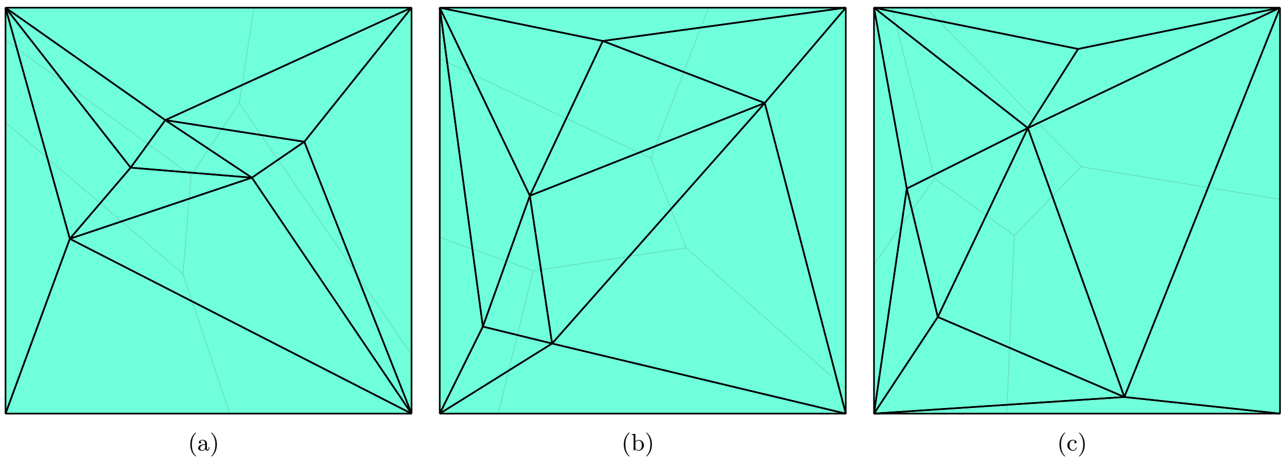
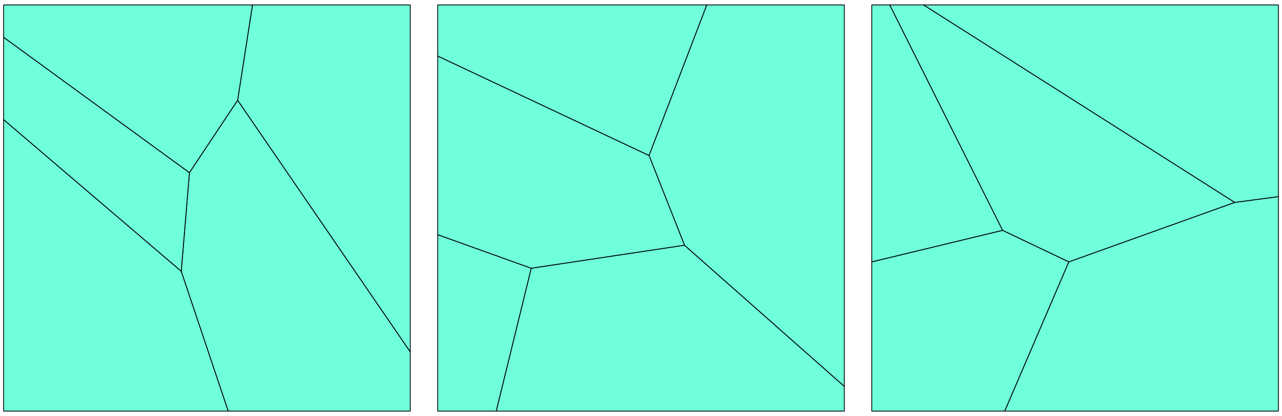


Figure 13: (a) (b) (c) Delaunay Triangulation Variants

Lloyd's Algorithm

One of the most striking features of the voronoi subdivision based on randomly chosen points is the irregular shape of the individual regions. These shapes are all convex by definition, but their side length and number can vary greatly, and so can the interior angle of each side. In traditional architecture, the norm of designing orthogonal spaces has been established for simplicity and interior design aspects, and thus architects are accustomed to designing rectangular floor plans. To make a Voronoi subdivision approximate regular shapes, the seed points can be iteratively shifted using the Lloyd algorithm, thus increasing the compacity of the regions. This method consists of three distinct steps: first, the voronoi pattern of points is calculated, in the following step the centroid of each of these voronoi regions is calculated, and in the last step the seed point is shifted to the calculated voronoi centroid. After a few repetitions, a fair division of the total area into Voronoi regions and thus more regular shapes of the regions are obtained.

Apart from Lloyd's method, maximization, average and minimization of the individual region areas or edge lengths can also be used for relaxation of the generated mesh.



(a) (b) (c)

Figure 14: (a) Voronoi I (b) Voronoi II (c) Voronoi III

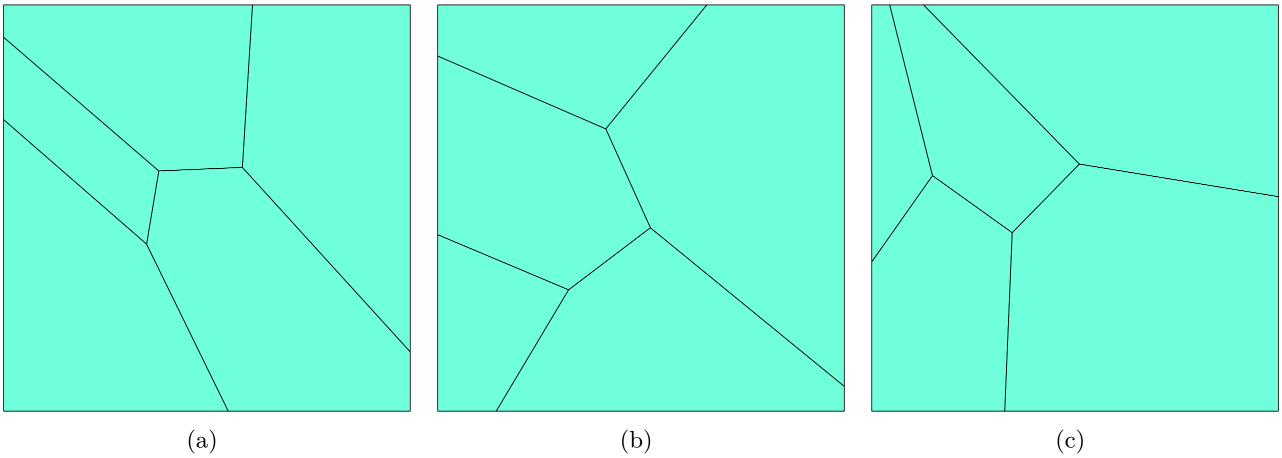


Figure 15: (a) Lloyd Variant I (b) Lloyd Variant II (c) Lloyd Variant III

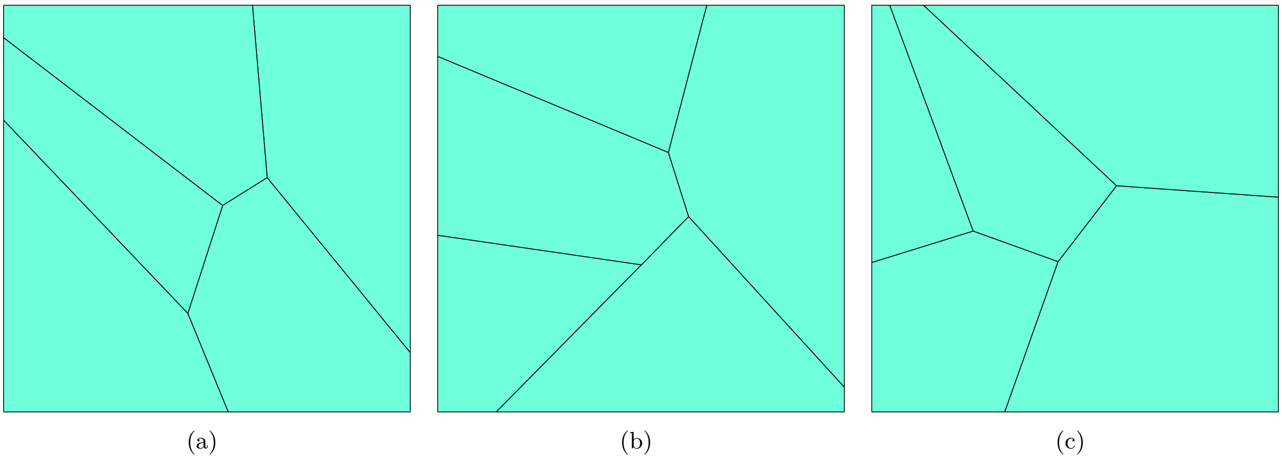
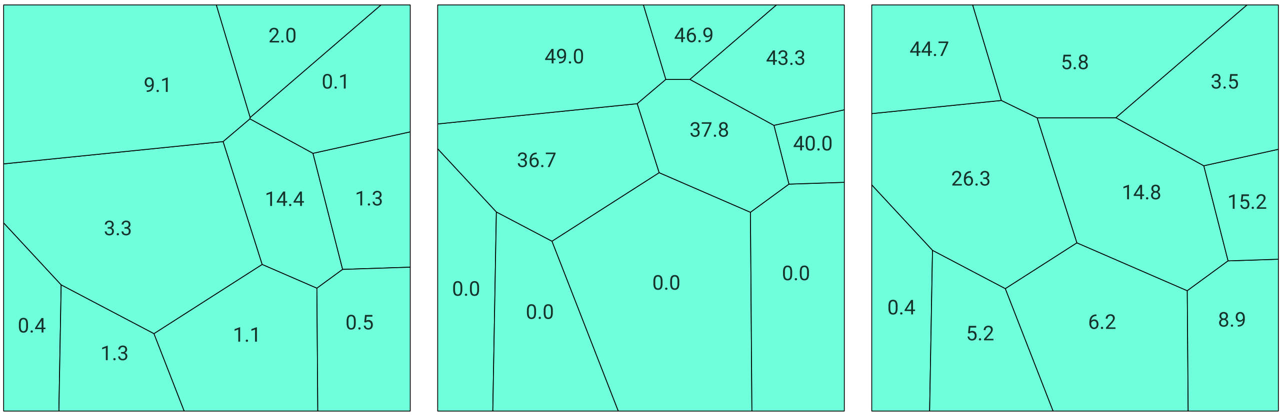


Figure 16: (a) Relaxed Diagram I (b) Relaxed Diagram II (c) Relaxed Diagram III

Laguerre-Voronoi

The most interesting variation of the voronoi daigram is called Power diagram or Laguerre-Voronoi diagram. In contrast to conventional voronoi's, the distance function is not described by half of the length but can be defined individually as a radius function of the respective regions. This allows a parametric control over the individual space sizes.



(a) (b) (c)

Figure 17: (a) (b) (c) Weighted Voronoi Diagrams

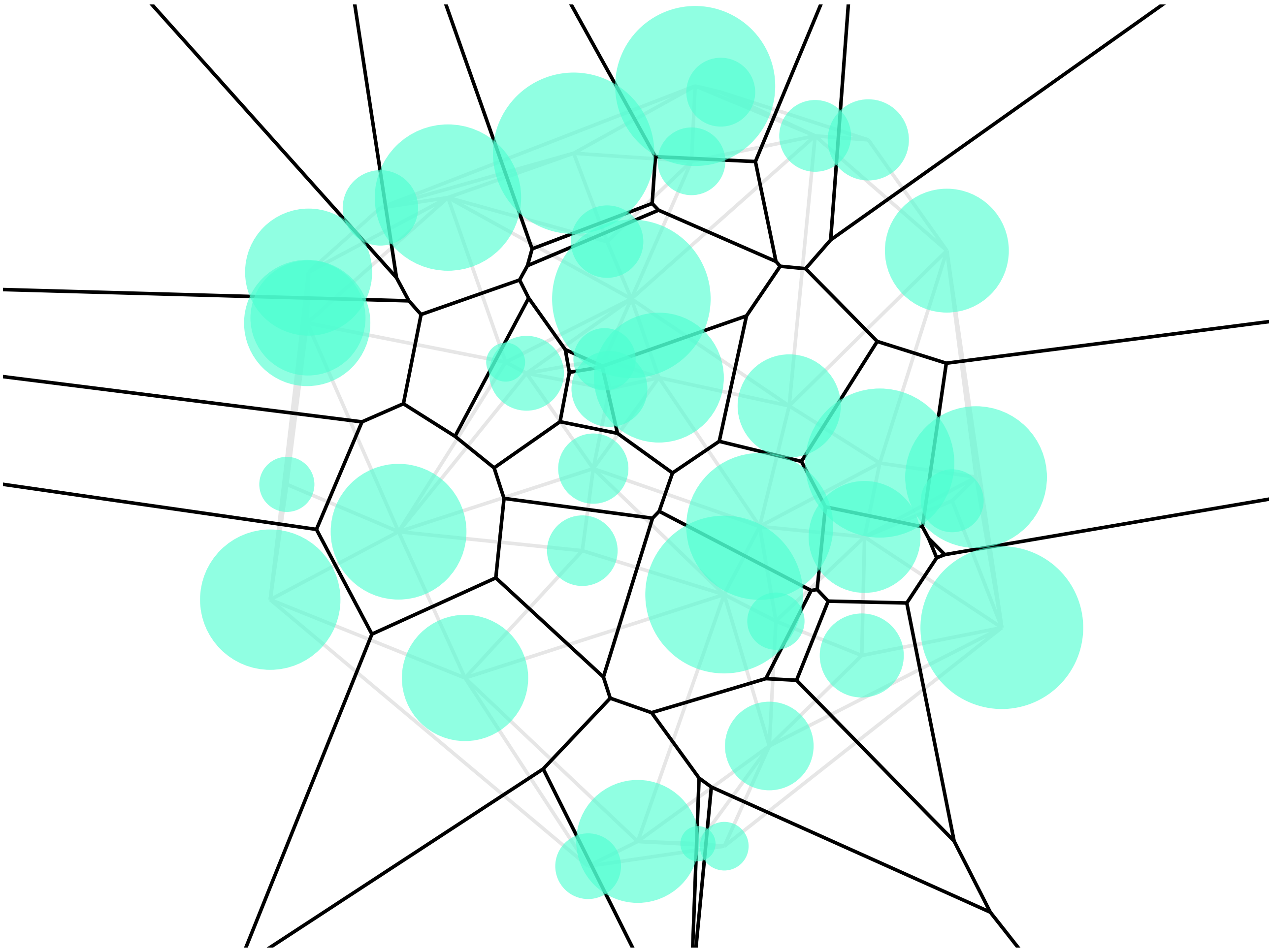


Figure 18: Power Diagram

To implement the radius distance function in sverchok blender the rule: every voronoi diagram in the nth dimension is the weighted diagram in the n-1th dimension was applied. In effect, this means that a fourth value w can be added to the x, y, z dimensions of the individual seeds of the two-dimensional diagram, which describes the weight and thus the weighted distance function.

Orthogonal Voronoi Diagram

Another less common variant is the rectangular voronoi which has the particularity that each voronoi region describes an orthogonal polygon and is therefore of special interest as an interface between traditional floorplan generation and irregular spatial planning.

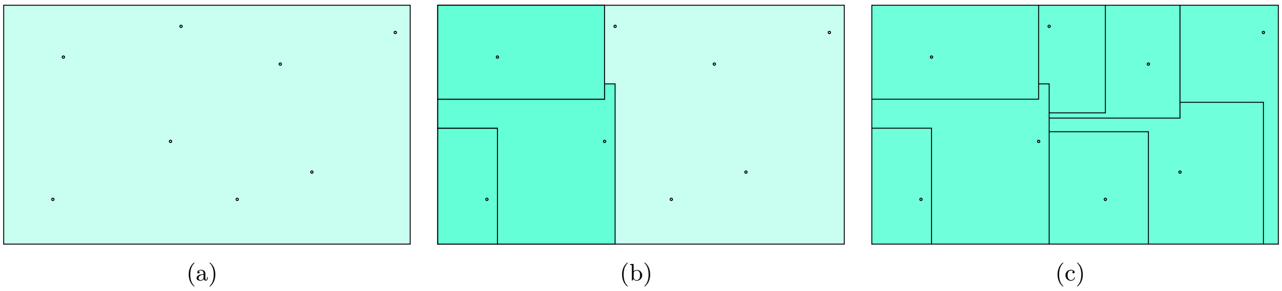


Figure 19: (a) Initial Seeds (b) 3 Iterations (c) Complete Orthogonal Voronoi Diagram

The generation of this diagram is based on a sweep algorithm with a predefined direction and skyline. First, the distance between the first two points in the dataset is considered, then a boundary is drawn orthogonal to the sweep line and at half the distance of the two points. The skyline is then moved to this limit and delimits the area orthogonal to the first point. These steps are repeated until each region is delineated.

Convex Hull

To determine the convex hull of a point cloud, different algorithms can be used and the final result is always the convex body containing all points. In the context of this work, this method is different from the previous ones, since it is not a method for dividing space, but rather one that defines the outline of the floorplan to be created.

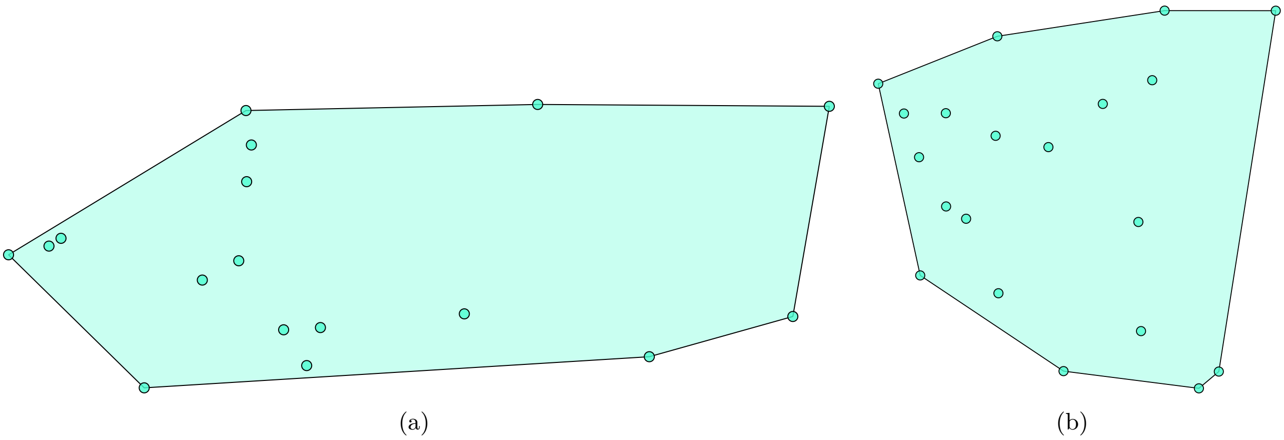


Figure 20: (a) (b) Convex Hull Variants

Orthogonal Convex Hull

Since the generated stress surface is an irregular convex polygon, this is difficult to use as a conventional floor plan because of its un-orthogonality. This can be remedied by the orthogonal convex hull, where the connected exterior points are connected by orthogonal lines.

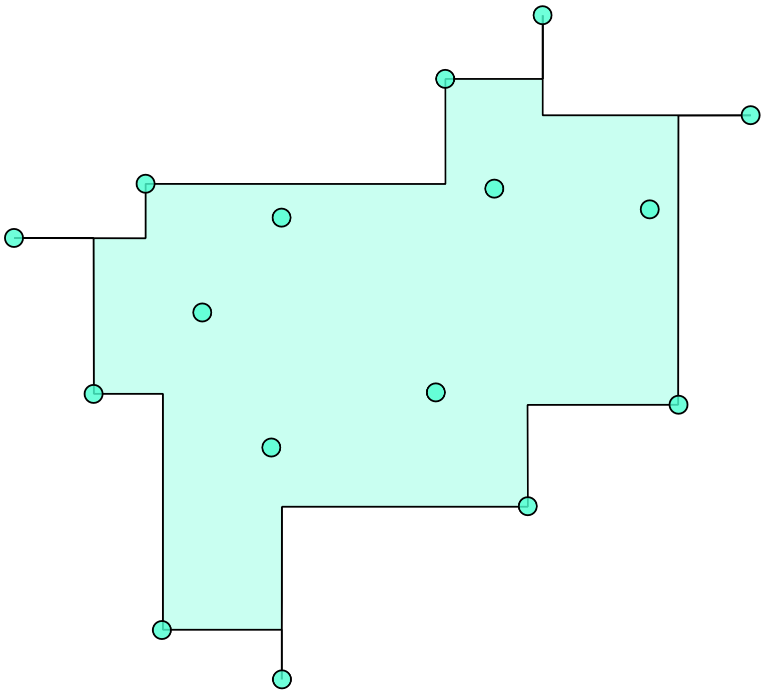
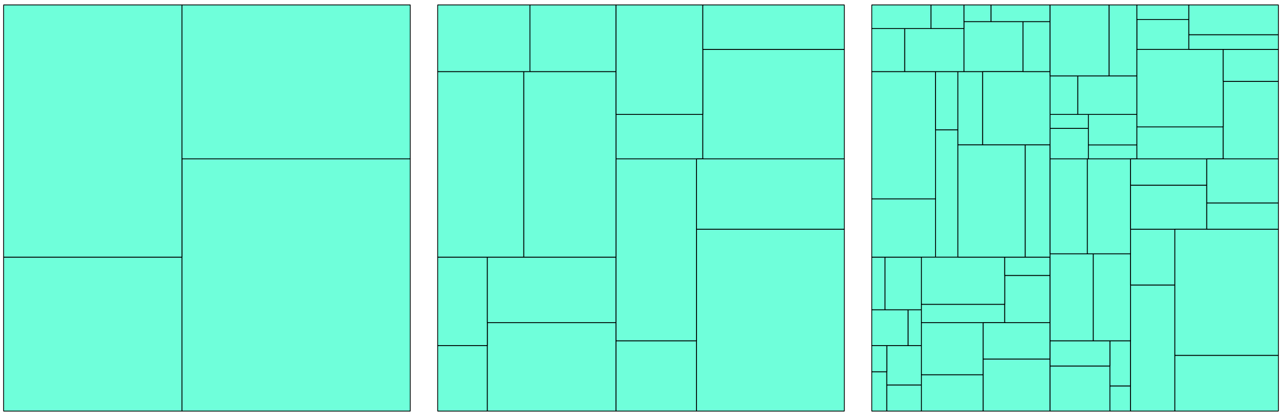


Figure 21: Orthogonal Convex Hull

The resulting outlines resemble those of a conventional floor plan and can be computed into spatial units by applying previously seen algorithms to the points located in the inner body. The disadvantage of this method of convex outline generation is, however, the separation caused by the convex orthogonality of some points which are bound to the generated plan only by a line.

Recursive Subdivision

This method is similar to the K-dimensional tree algorithm in that it is also an iterative subdivision of a total area into subunits. An important difference in the recursive subdivision is that there are no predefined points that define the space. Only a coordinate for the intersection axis point and its intersection axis is defined. The ratio of the intersections can be defined or randomly parameterized.



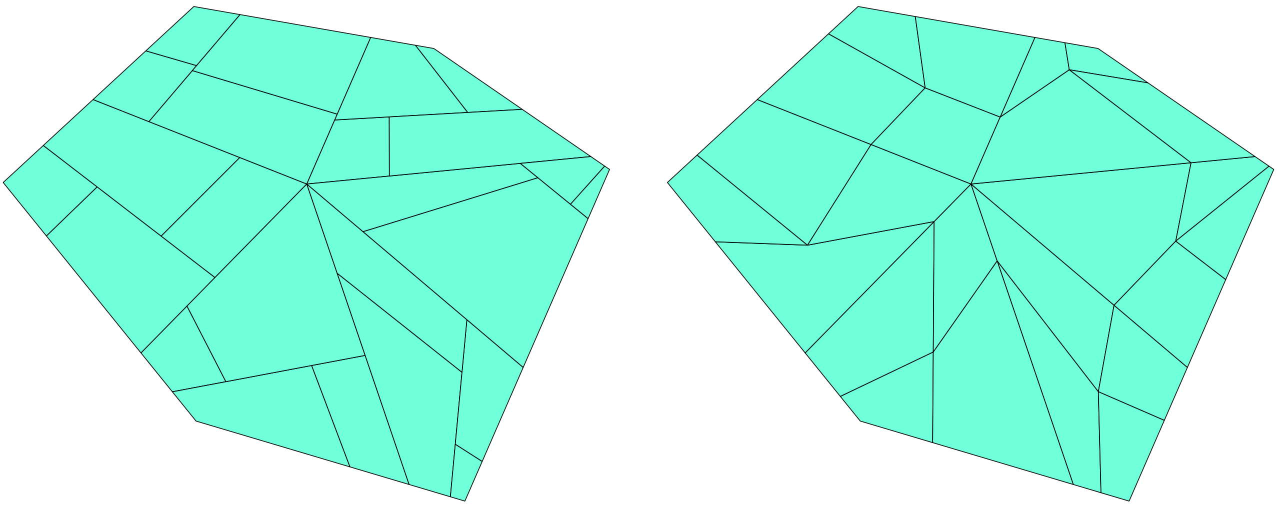
(a) (b) (c)

Figure 22: (a) 2 Iterations (b) 4 Iterations (c) 6 Iterations

A clear advantage of this algorithm is its simplicity, however, the basic bodies to be divided are limited to quads and the subdivided compartments are also quadrilaterals. Furthermore, in the normal iteration, each of the subdivided shapes is subdivided into the exact same number which provides regularity in the subdivision but at the same time makes variation difficult.

Bent Bisection

Since this method is based on the repetition of subdivision algorithms, the variation by combining different subdivision methods is virtually unlimited. Even slight modifications such as a randomly determined kink in the division plane can provide amazing variation. If one or more iterations are replaced by a division into quads by the centroid of the form, the regular character of this recursive subdivision can be varied again.



(a) (b)

Figure 23: (a) Quad And Recursive Subdivision (b) Recursive Quad Division

Shape Grammar

By means of established shape rules and a shape engine which acts as a selector and interpreter, different shapes can be generated on the basis of the formulated geometric rules such as orientation, boolean, multiplication and deplacement. Together with a start rule, an indefinite number of transformation rules and a termination rule, several shapes respecting different rules can be generated by serial or parallel iteration. In this work the parametric shape grammers are of interest, because here the rules are based on variables and thus a repeated execution of the generation with randomly generated, but certain limits located variables by variation of the random seed unpredictable shapes arise, which however always respect the implied geometric conditions. Furthermore, it is possible to view the process after the execution of each stage and thus have a finer choice of the output geometry. The floor plans generated in this way have the advantage that the spaces touch each other exactly on one or more lines, thus enabling further processing in boundary representation.

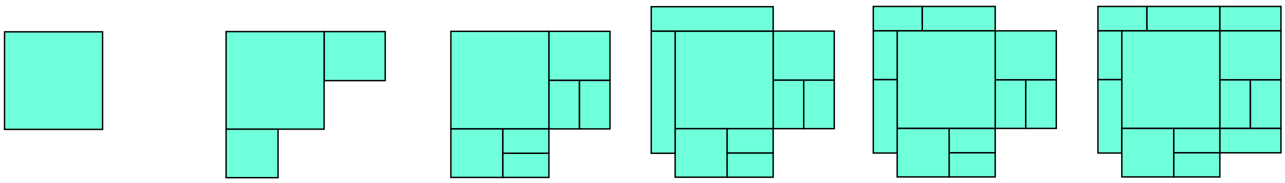
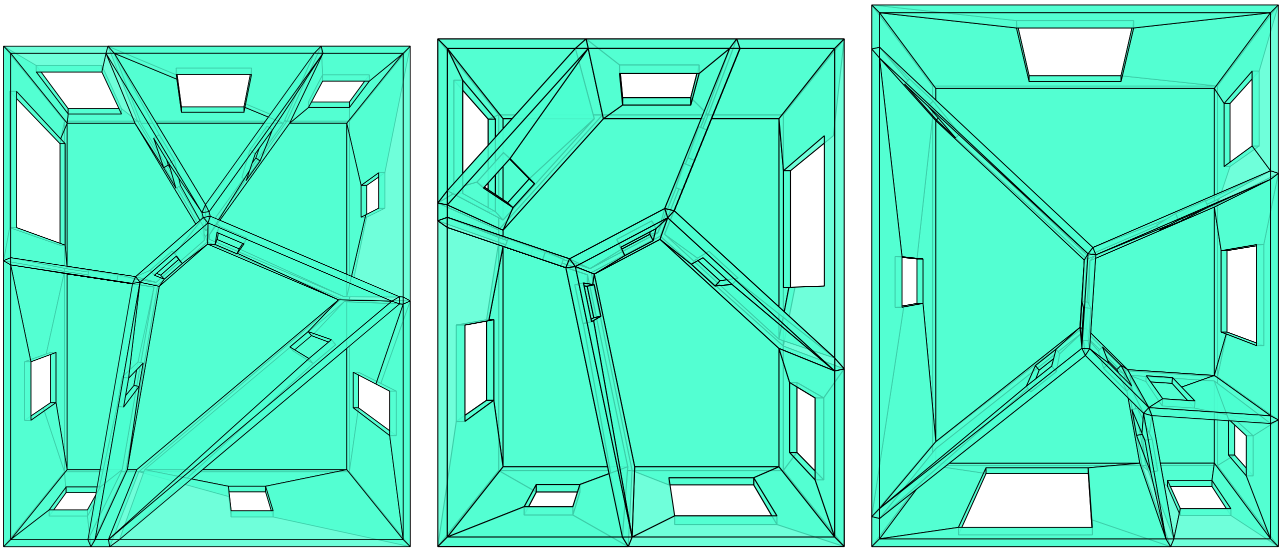


Figure 24: Shape Grammar Generation

Topologic

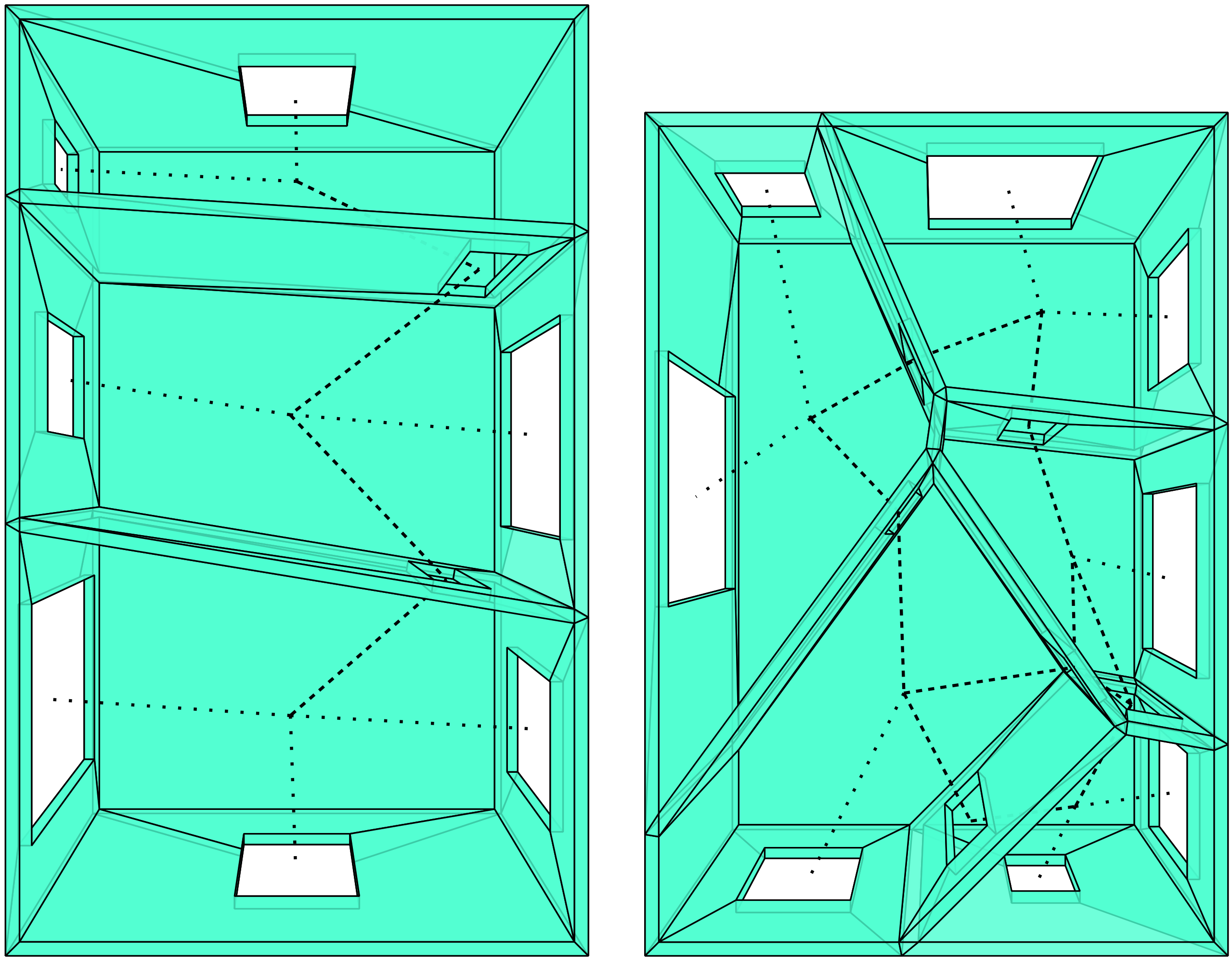
Topologic python library is not primarily a shape grammar engine, but by processing the topological relationships in the geometry, it is possible to establish shape rules and thus enhance the generated partitions. The functionality of topologicPy is based on non manifold topologies similar to boundary representations and opencascade shapes. The architectural geometry is extremely simplified and consists only of single layer surfaces. Thanks to the hierarchical structure of the library, it can easily switch from larger units like CellComplex, Cell and Cluster to smaller units like faces, edges and vertex by querying the entire topology. Furthermore, a major advantage of using topologic is a seamless interface between geometry processing and environmental simulation such as energy performance or light simulations thanks to Openstudio, honeybee and ladybug bindings. However, in terms of floorplan generation, the topological analysis capabilities are of primary importance.



(a) (b) (c)

Figure 25: (a) (b) (c) Aperture Enhanced Models

If the two-dimensional floorplan layouts generated by the aforementioned methods are converted into topologic geometries as a surface list, a proper data exchange between blender's sverchok geometry and the geometry processed by topologic can be ensured. Thus, named surfaces are first extruded along the z-axis and desired space height and registered as cells in a cellcomplex. At this point it is essential that the space contours touch each other on at least one side but without overlapping. After the cellcomplex generation, which is analogous to the apartment generation, the interior and exterior walls of the whole complex can be separated by parameterizable queries and retrieved with their respectieve geometry. In the following step it is possible to retrieve a point on each wall surface and, in combination with the geometry normal, to read its corresponding matrix, which in turn allows to generate parametric window and door surfaces on the walls. Thanks to topologics aperture integration, these openings can then be stored as surface apertures for each individual wall and integrated into the cell complex.



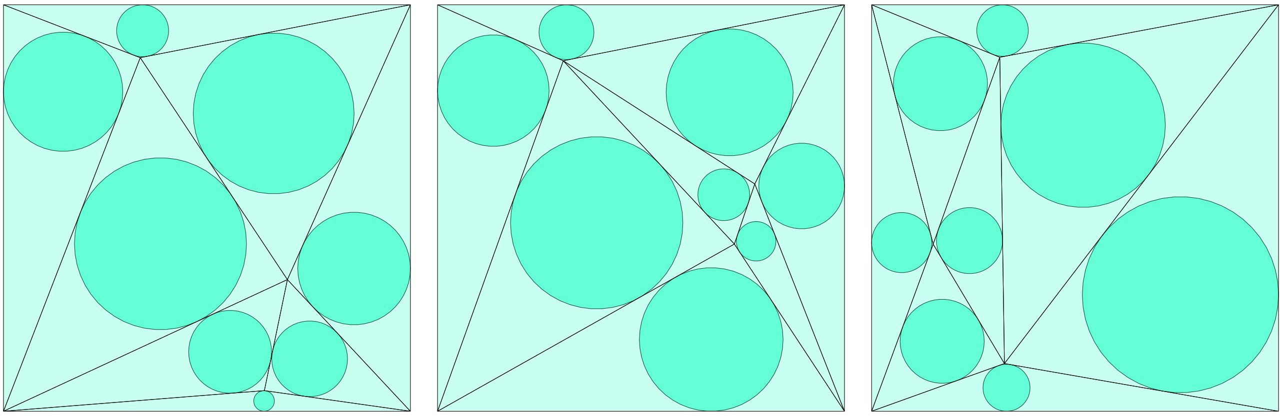
(a) (b)

Figure 26: (a) (b) Topological Graph To Exterior / Internal Apertures

Through these sequential processes, plausible three-dimensional apartment models can be generated with integrated circulation and window openings. Now that the apertures have been integrated into the cell complex, any relational connection of the spaces and apertures can be queried and used for analysis purposes using the topologics graph function.

Shape Packing

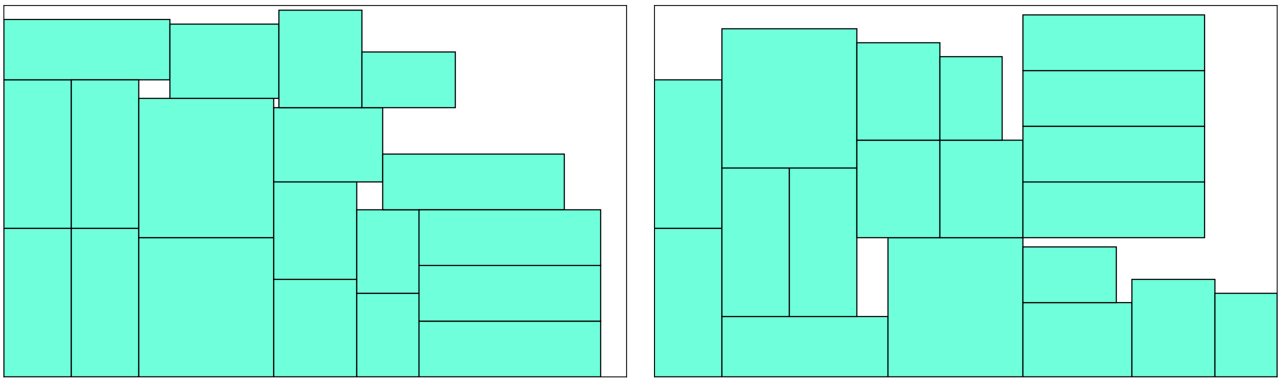
The family of shape packing algorithms deals with the topic of inscribing defined shapes with fixed dimensions into an equally fixed container. These methods are relevant in connection with the problem of automated floorplan generation, since almost all parameters can be defined in advance. These parameters include the number of rooms and their exact shape, but also the exact dimensions of the plan outline.



(a) (b) (c)

Figure 27: (a) (b) (c) Delaunay Triangulation With Inscribed Circles

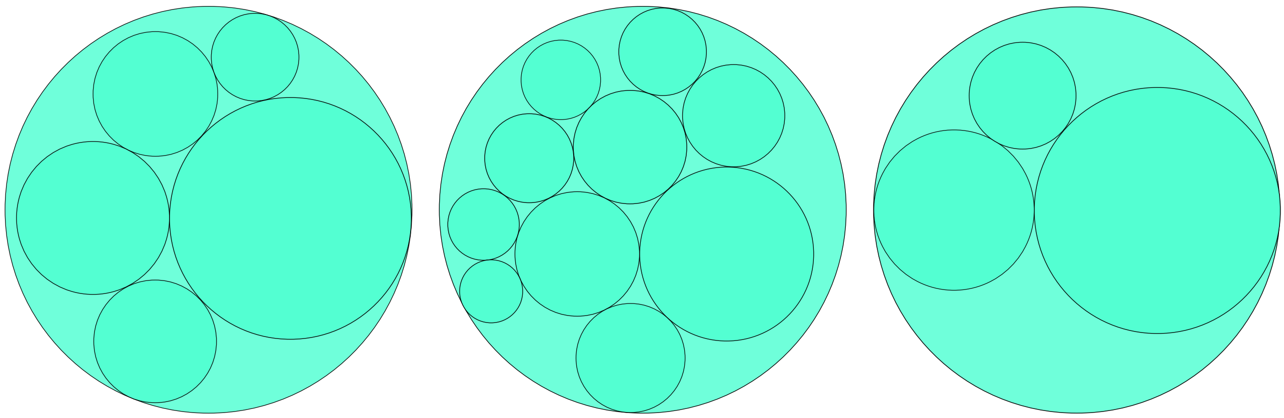
Three families of shape packing problems are distinguished, those where the container is unlimited and those where the container is limited in three or two dimensions. In this work, the methods of shape packing in two dimensions with limited container size are of particular importance. In a two-dimensional body triangulated by point projection delaunay, the largest possible circles can be inscribed in the respective triangles, thus creating a space division defined by the radius.



(a) (b)

Figure 28: (a) (b) Bin Packing Problem

The bin-packing problem describes the problem of the most space-efficient packing of an exact number of well-defined rectangles in a certain number of containers. Furthermore, the exact position of the best possible packing of a set of circles with defined radii can be calculated by a circles in circle packing algorithm.

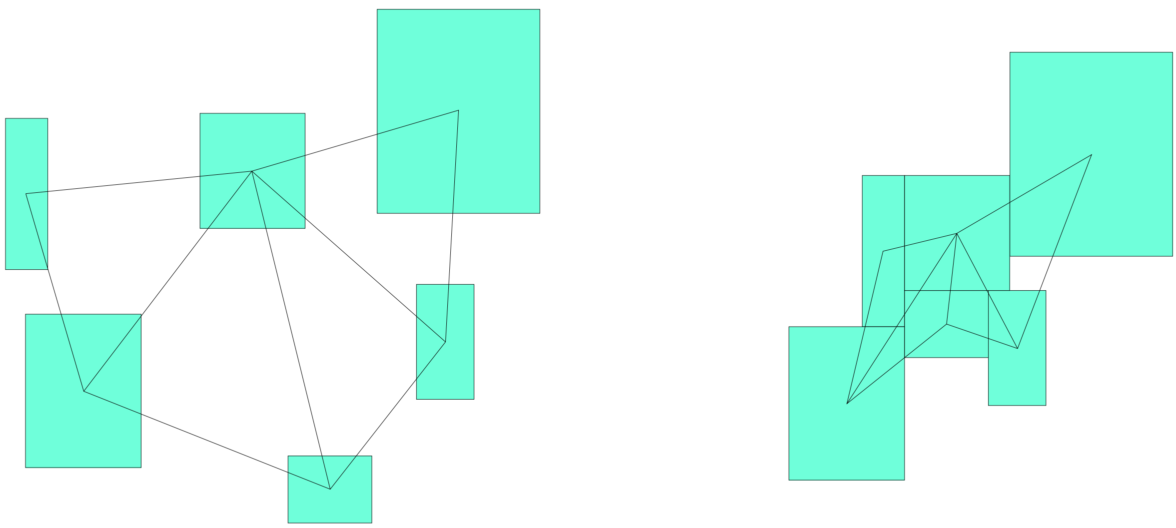


(a) (b) (c)

Figure 29: (a) (b) (c) Circles In Circle Algorithm

Physics Solver

Based on the genetic algorithm variant of floorplan generation, physic engines can be used to create different plan layouts. In this work, Bullet physics was used thanks to its blender integration and ease of interaction. Required for the successful execution of this generation method are only the pre-defined single room sizes as two dimensional geometries in euclidean space. In the next step, the relation network which is simulated by the attraction network has to be defined. This mesh should connect the centroids of the space compartments with each other to avoid undesired results.



(a) (b)

Figure 30: (a) Initial Position (b) Final Position

Thanks to the physic solver, the units can be defined as solid bodies with active collision, which prevents interpenetration. This is counteracted by the spring attractions forces that act on the net lines between the body centers of gravity to different degrees. During the execution of the simulation, the number of solver steps acts as a time variable and thus the animation can be played. The final results generated in this way can vary greatly in their layout

depending on the start position seed and the random noise seed which is added to the attraction forces. By varying the spring forces, the topological relationship between the spaces can be regulated and thus allows a variety in the generation.
Advantages of this method are the conclusive results in the traditional architectural sense and the possibility to define in advance the individual space dimensions, orientations and their topological relationships. However, similar to the results of the evolutionary algorithm, the main disadvantages are an impure end geometry with minimal distances between the single units and a time and memory consumption in the simulation process.

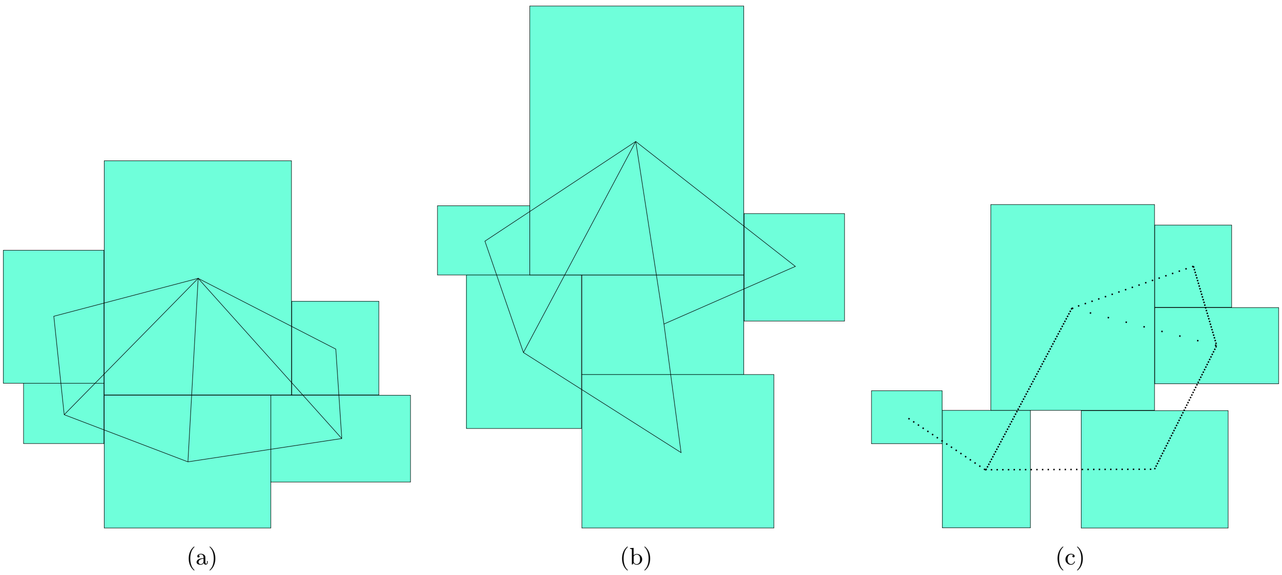


Figure 31: (a) (b) Physics Simulation Result Variants (c) Simulation With Different Attraction Forces

Data Driven Approaches

In the field of computer science, machine learning processes for architecture generation have been strongly established for several years. The different approaches vary strongly and therefore also their output and their application. However, there is an analogy in all data driven approaches which makes an application to layout generation inappropriate: The learning algorithms of amazing accuracy are always based on a real world dataset and thus show a constant variance bias which is caused by the functionality of such approaches. Either the model to be trained is trained directly on the basis of selected floor plans that are considered to be suitable, or a generator uses random noise to design objects that increasingly develop into apartment floor plans. However, also here the elementary part of the mechanism is formed by the discriminator which communicates to the random noise generation as feedback how far the generated object resembles a conventional plan.

Method of Choice

In order to generate a comprehensive data set, as many different floorplan layout design methods as possible are integrated, as well as their different variants. At the moment only the KD-Tree and Voronoi and its derivatives provide satisfying results. Possibly, a genetic algorithm could increase the variance of these methods by optimizing individual selection characteristics. Furthermore,

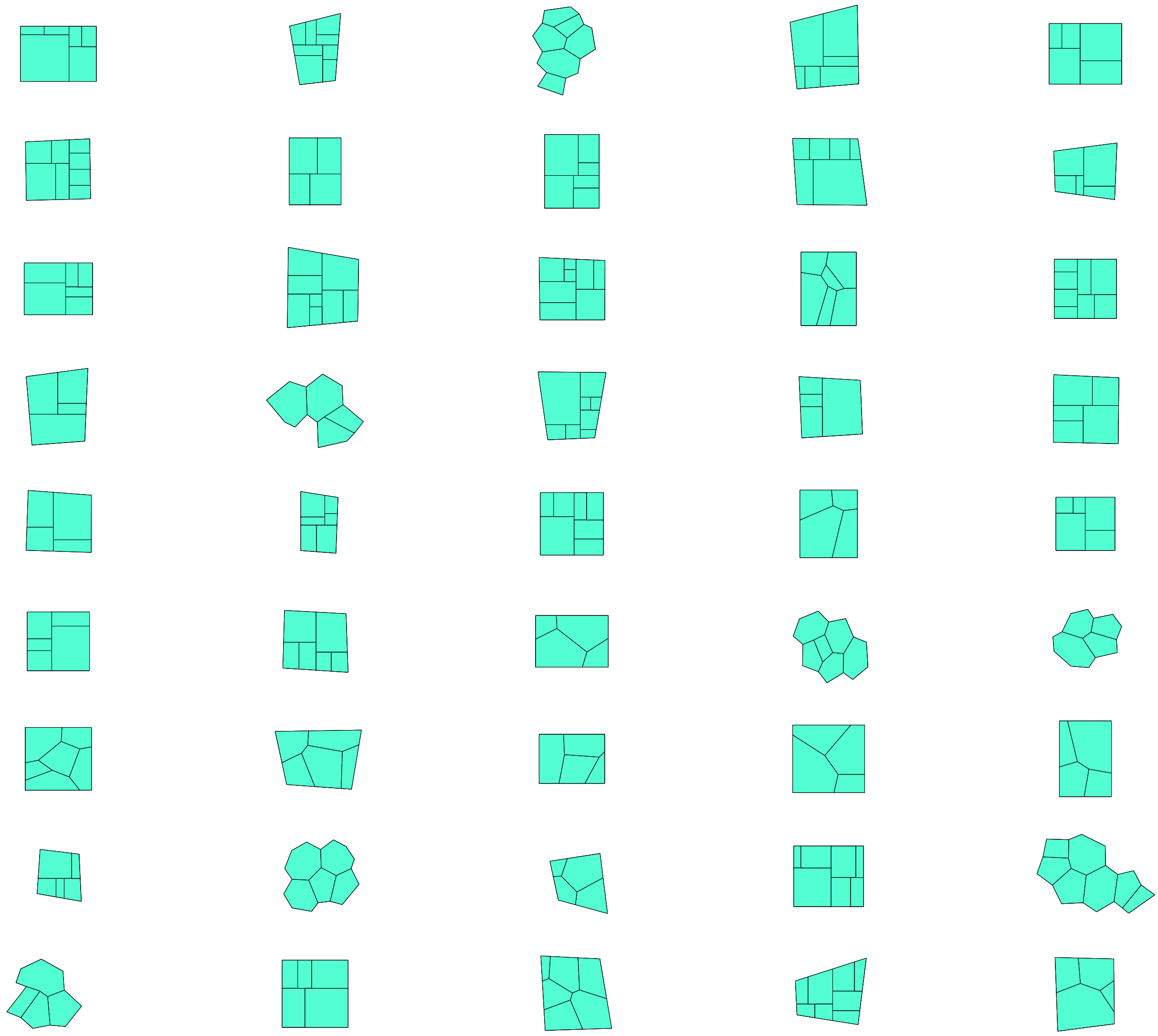
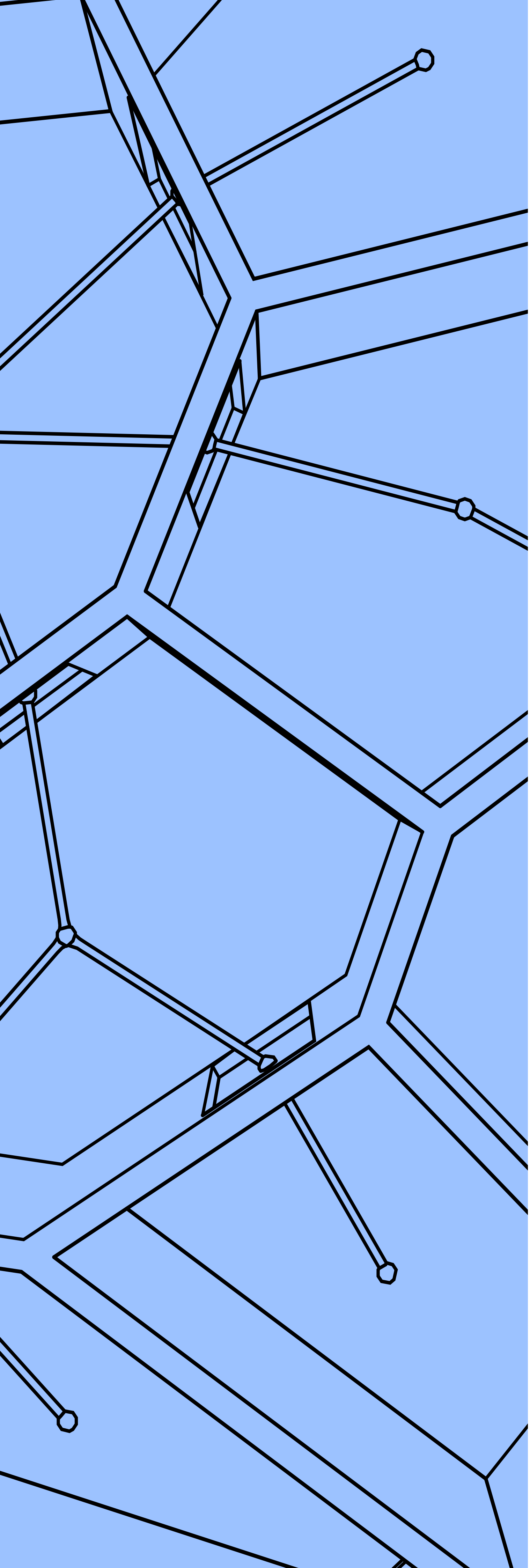


Figure 32: Results

an integration of the recursive subdivision method would be conceivable, but its results are similar to those of the KD-Tree, but are surpassed by the latter because of the regulability through the seed points. The remaining methods described have significant drawbacks that prevent their integration into the dataset generation process, such as the need for another gap removal algorithm between the individual areas. This is theoretically possible, but is not necessary at the moment, as it turns out that the dataset is sufficiently versatile due to a well-considered parameterization of the variables of the KD-Tree and Voronoi method. As explained in the next section, the generated geometric data are filtered after their generation. In order to avoid that a large part is sorted out, the parameterization must avoid the filter characteristics and, if necessary, cause a seed change by a feedback.



Data

The first stage of this work deals with the automated generation of different apartment layouts using different geometry processing software. An alternative to the generation approach is the acquisition of existing architectural datasets provided by real estate and research groups. Advantages of reality-based datasets are the certainty of architectural feasibility and construction of the individual plans, but disadvantages are a conservative approach in creative terms and a high risk of traditional bias due to the predominance of local and traditionally influenced architectural methods. Therefore, this experimental phase is primarily concerned with testing different floorplan generation methods and their derivatives as well as their possible combination. Simple algorithms like voronoi diagrams, intelligent methods like evolutionary algorithms and data driven approaches are explained and evaluated by different experiments. Criteria for the evaluation are simplicity in the construction process, computing power, time and memory, integration and interaction with used software, purity of the generated geometry, degree of readiness for the simulation stage, simplicity of the layout for data storage but above all spatial quality, creativity in form finding and feasibility. After experimentation, the most appropriate method or combination of algorithms for the following steps is selected and an appropriate set of different layouts is generated. In order to increase the variation in the synthetic dataset, results from different methods can be mixed, but they must not deviate from the generally defined quality level. Furthermore, it is a requirement to understand the functionality of the algorithms sufficiently to allow slight adaptations and combinations with other functions and to achieve variation in the individual applications. In general, simplicity is preferred to complexity, creativity to ordinariness and variation to repetition.

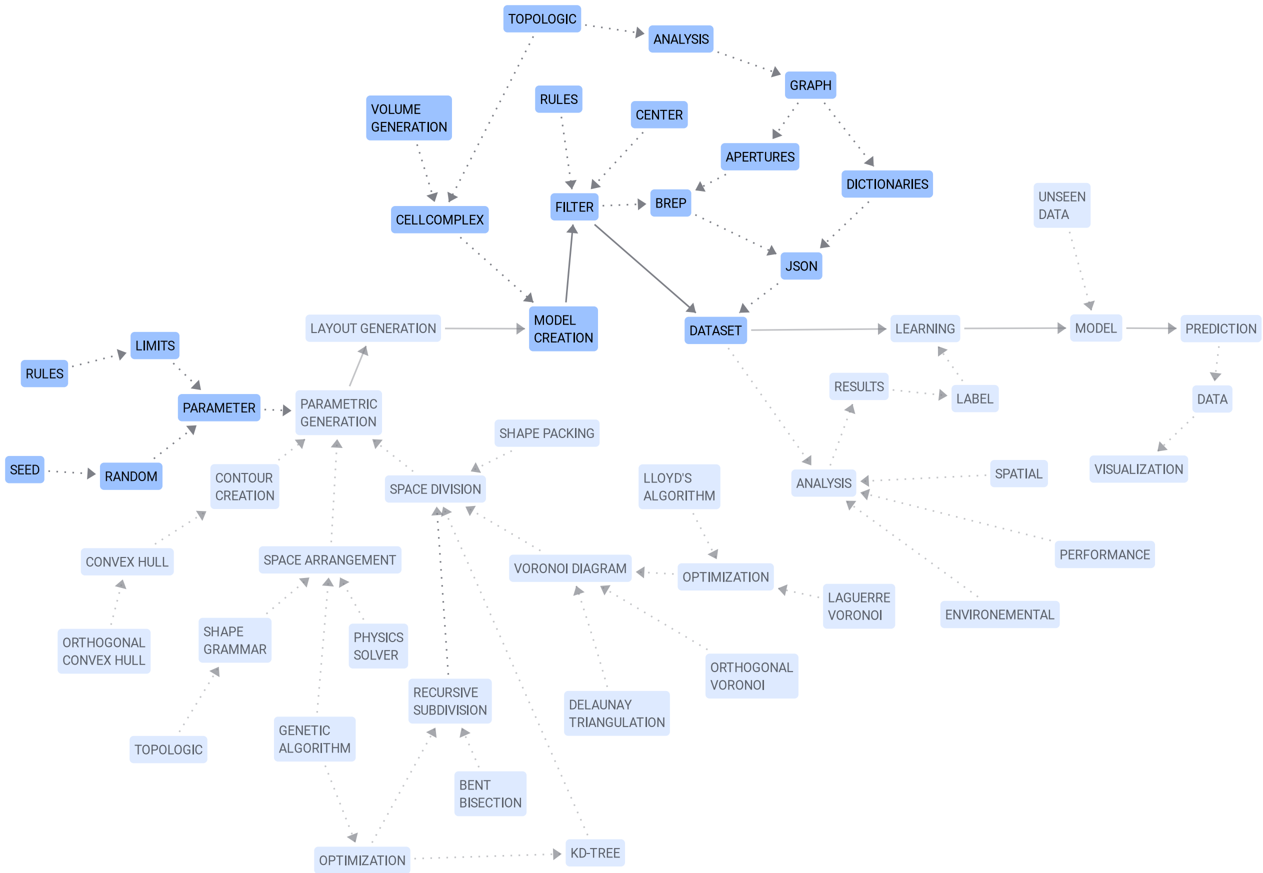


Figure 33: Data Augmentation

Volume Creation

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Boundary Representation

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

```
CASCADE Topology V1, (c) Matra-Datavision
Locations 0
Curve2ds 4
1 0.40997122167981925 0.88040078724103443 -1 0
1 0.40997122167981925 -0.88040078724103399 0 1
1 -0.40997122167981881 0.88040078724103443 0 -1
1 -0.40997122167981881 -0.88040078724103399 1 0
Curves 4
1 4.7081688703713462 1.0714996933938057
    1.659971221679819 1.1102230246251565e-16 0 -1
1 4.7081688703713462 2.8323012678758741
    1.659971221679819 0 -1 0
1 4.7081688703713462 1.0714996933938057
    0.84002877832018097 0 1 0
1 4.7081688703713462 2.8323012678758741
    0.84002877832018097 -1.1102230246251565e-16 0 1
Polygon3D 0
PolygonOnTriangulations 0
Surfaces 1
1 4.7081688703713462 1.9519004806348401
    1.2499999999999998 1 0 0 -0 0 1 0 -1 0
Triangulations 0

TShapes 10
Ve
1e-07
4.70816887037135 1.07149969339381 1.65997122167982
0 0

0101101
*
Ve
1e-07
4.70816887037135 1.07149969339381 0.840028778320181
0 0

0101101
*
```

```
Ed
1e-07 1 1 0
1  1 0 0 0.819942443359638
2  1 1 0 0 0.819942443359638
0

0101000
+10 0 -9 0 *
Ve
1e-07
4.70816887037135 2.83230126787587 1.65997122167982
0 0

0101101
*
Ed
1e-07 1 1 0
1  2 0 0 1.76080157448207
2  2 1 0 0 1.76080157448207
0

0101000
+7 0 -10 0 *
Ve
1e-07
4.70816887037135 2.83230126787587 0.840028778320181
0 0

0101101
*
Ed
1e-07 1 1 0
1  3 0 0 1.76080157448207
2  3 1 0 0 1.76080157448207
0101000
+9 0 -5 0 *
Ed
1e-07 1 1 0
1  4 0 0 0.819942443359638
2  4 1 0 0 0.819942443359638
0

0101000
+5 0 -7 0 *
Wi

0101100
+8 0 +6 0 +4 0 +3 0 *
Fa
0  1e-07 1 0

0101000
+2 0 *

+1 0
```

Listing 1: BREP Example

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

JSON

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Dictionaries

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Filtering

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Topologic

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Apertures

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Structure

```
[
```

```
    {
        "brep": "topology_00001",
        "cellApertures": [],
        "cellDictionaries": [
            {
                "dictionary": {
                    "area": 33.44,
                    "id": 0,
                    "type": "room"
                },
                "selector": [
                    1.491212715639101,
                    0.18688301956760373,
                    1.25
                ]
            },
            {
                "dictionary": {
                    "area": 14.2,
                    "id": 1,
                    "type": "room"
                },
                "selector": [
                    2.5635190468646725,
                    -3.0991602704456733,
                    1.25
                ]
            },
            {
                "dictionary": {
                    "area": 34.79,
                    "id": 2,
                    "type": "room"
                },
                "selector": [
                    -2.4800350291326527,
                    1.085227288601124,
                    1.2500000000000002
                ]
            }
        ],
        "dictionary": {
            "id": 1,
            "rooms": 3,
            "surface": 82.43,
            "type": "flat"
        },
        "edgeApertures": [],
        "edgeDictionaries": [],
        "faceApertures": [
            {
                "brep": "topology_00002",
                "dictionary": {
                    "id": 0,
                    "type": "door"
                }
            },
            {
                "brep": "topology_00003",
                "dictionary": {
                    "id": 1,
                    "type": "door"
                }
            },
            {
                "brep": "topology_00004",
```

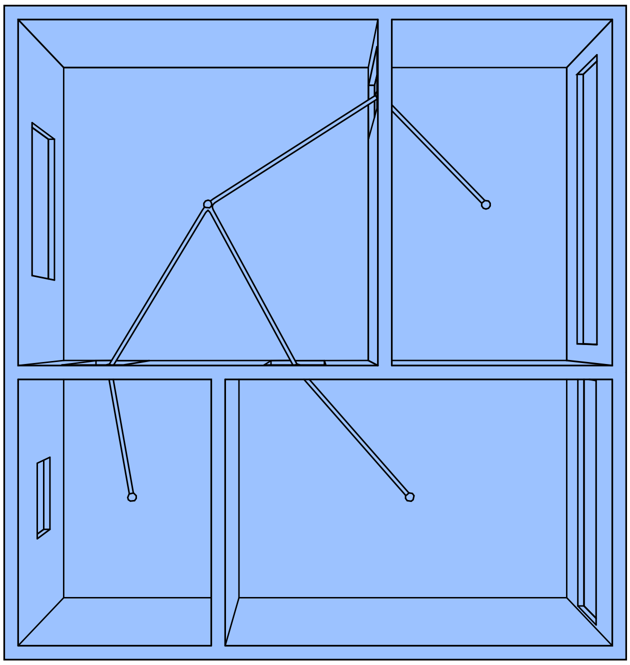
```
                "dictionary": {
                    "id": 0,
                    "type": "window"
                }
            },
            {
                "brep": "topology_00005",
                "dictionary": {
                    "id": 1,
                    "type": "window"
                }
            },
            {
                "brep": "topology_00006",
                "dictionary": {
                    "id": 2,
                    "type": "window"
                }
            }
        ],
        "faceDictionaries": [],
        "vertexApertures": [],
        "vertexDictionaries": []
    }
]
```

Listing 2: Topologic JSON Example

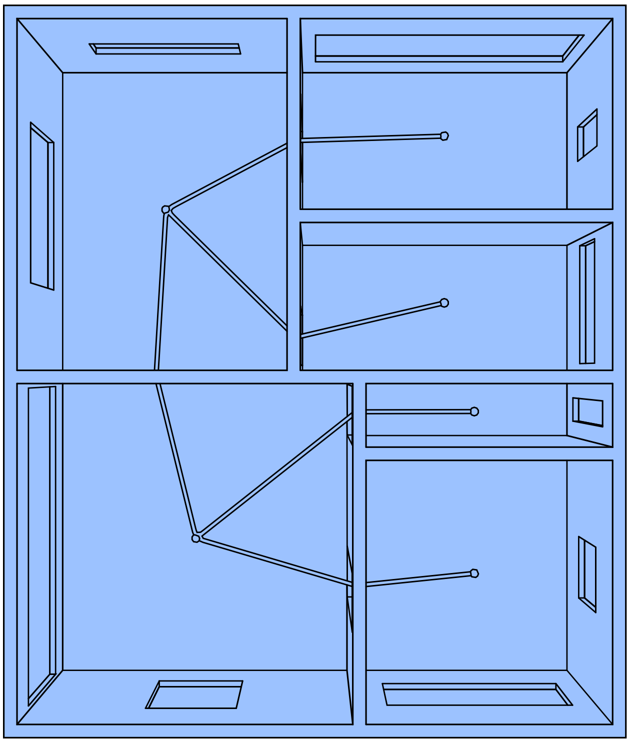
Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Results

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

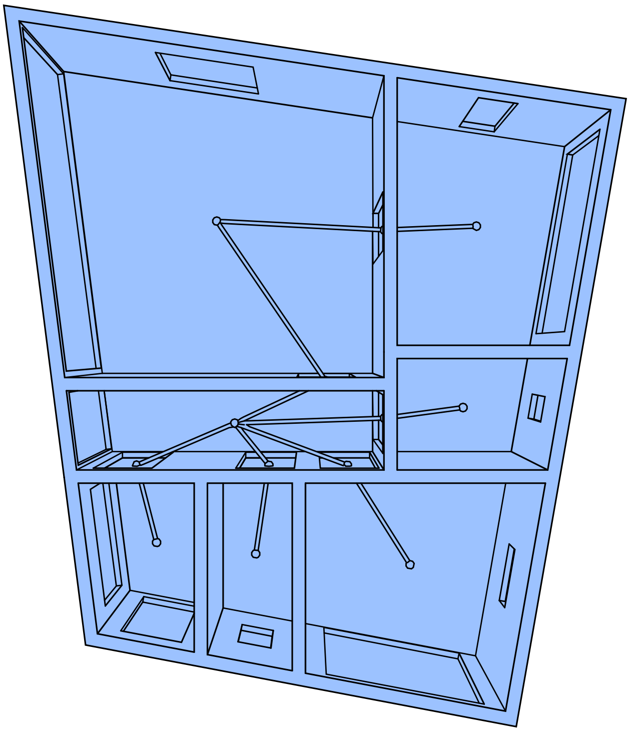


(a)

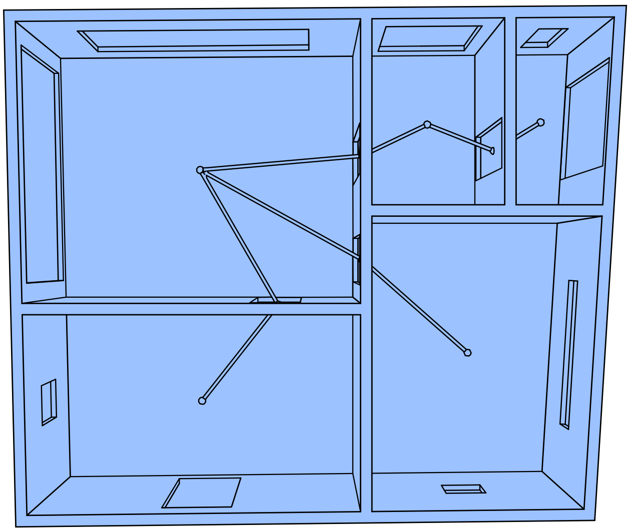


(b)

Figure 34: (a) (b) KD-Tree Variants

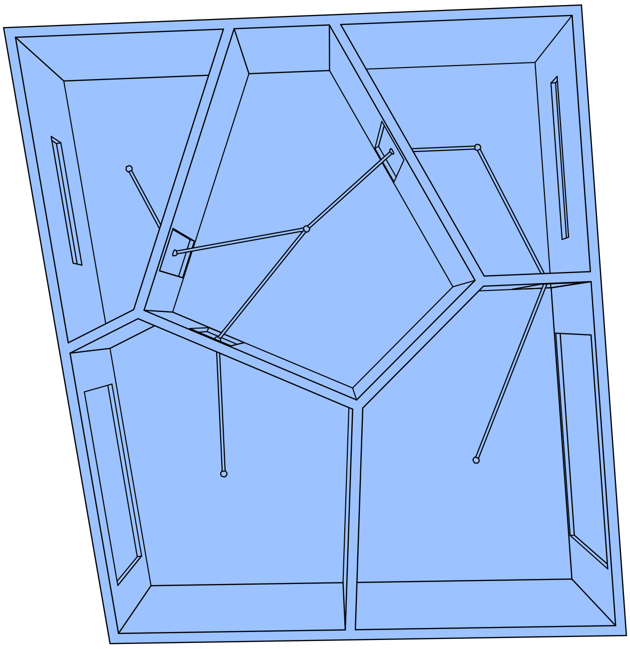


(a)

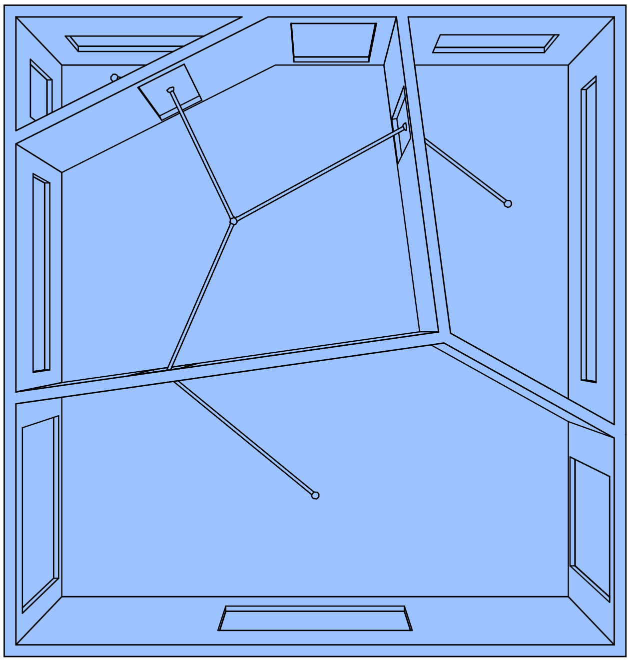


(b)

Figure 35: (a) (b) Deformed KD-Tree Variants

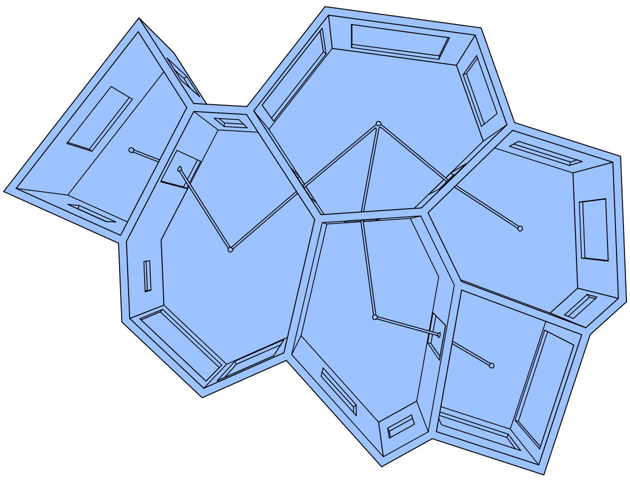


(a)

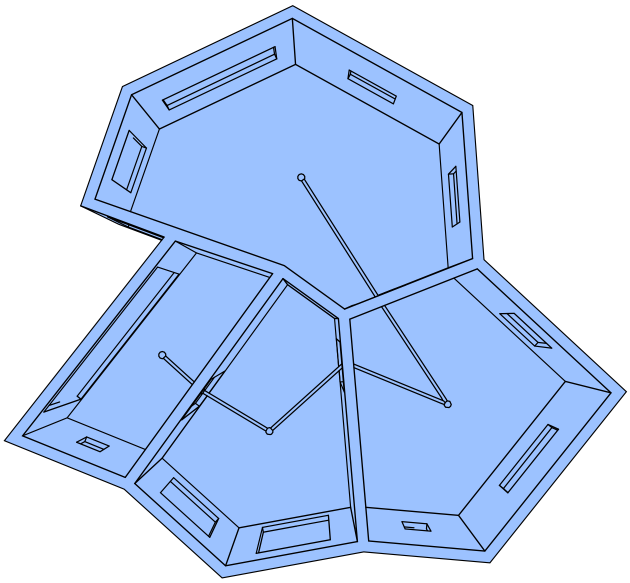


(b)

Figure 36: (a) Rectangular Voronoi Variant (b) Deformed Voronoi Variant

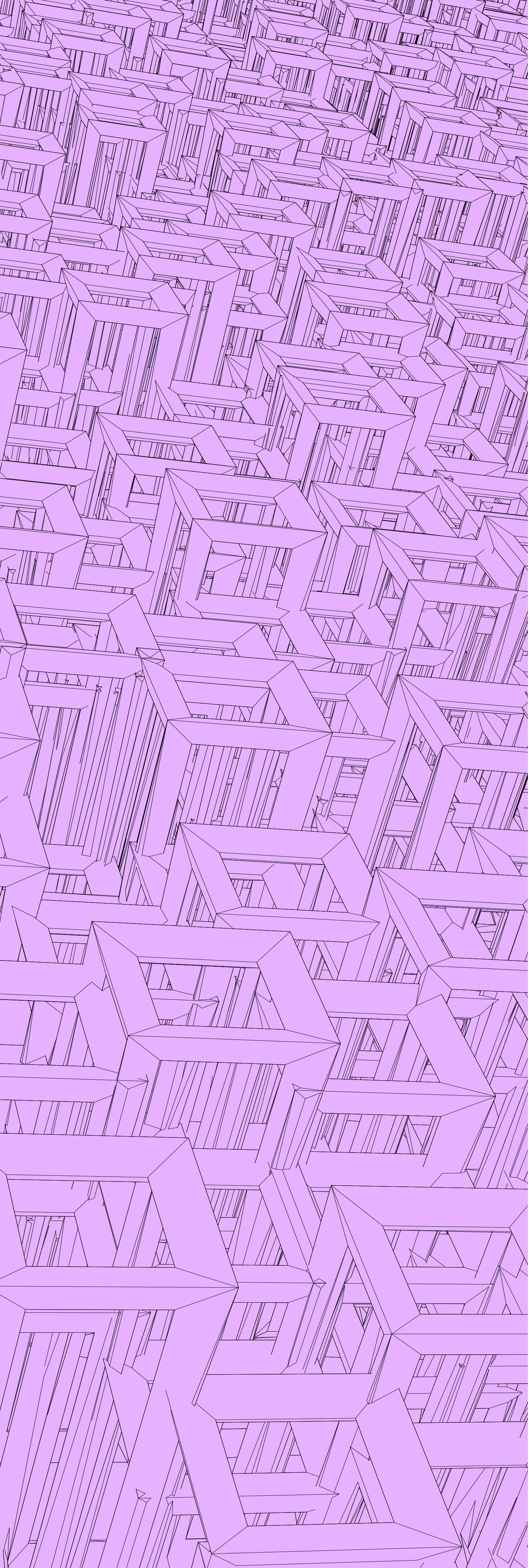


(a)



(b)

Figure 37: (a) (b) Voronoi Variants



Analysis

Due to time constraints, the data evaluation stage will not be an integrated part of this work, but it will be the main focus of the following work. Furthermore, in all preceding steps the possibility of data set evaluation by means of simulations and analyses was included as a relevant method of evaluation. Thus, the stored geometries consist of non-manifold geometry boundary representations which are excellent for envirenemental and topological analysis and furthermore, care was taken to keep the data size of the individual architectural objects and their geometries as small as possible. By storing typologies, individual identification numbers, geometric characteristics and geometric coordinates in the geometry JSON file as key value dictionnaries, such properties can be queried in the analysis stage in an effective way. Possible analysis approaches include energetic analysis using EnergyPlus, light analysis using Radiance, fluid dynamics using Openfoam, finite element analysis using Elmer, acoustics using I-Simpa, crowd simulation using vadere. Furthermore, topological analyses can be performed with Topologic and eventually additional simulations like firedynamics with FDS. A special focus will be put on graph-based analysis of geometric units, as this has proven to be a promising topological analysis option and allows several different graph machine learning methods as well as data storage in multidimensional graph datasets. Each of the different approaches also requires knowledge from related fields such as physics and chemistry, which must be acquired for successful evaluation and execution of the simulation.

After the completion of the simulation and analysis stage, the individual data with their evaluation results should be combined in an optimal way and thus complement the data set with labels. This requires an understanding of data set manipulation as well as a general understanding of data structure.

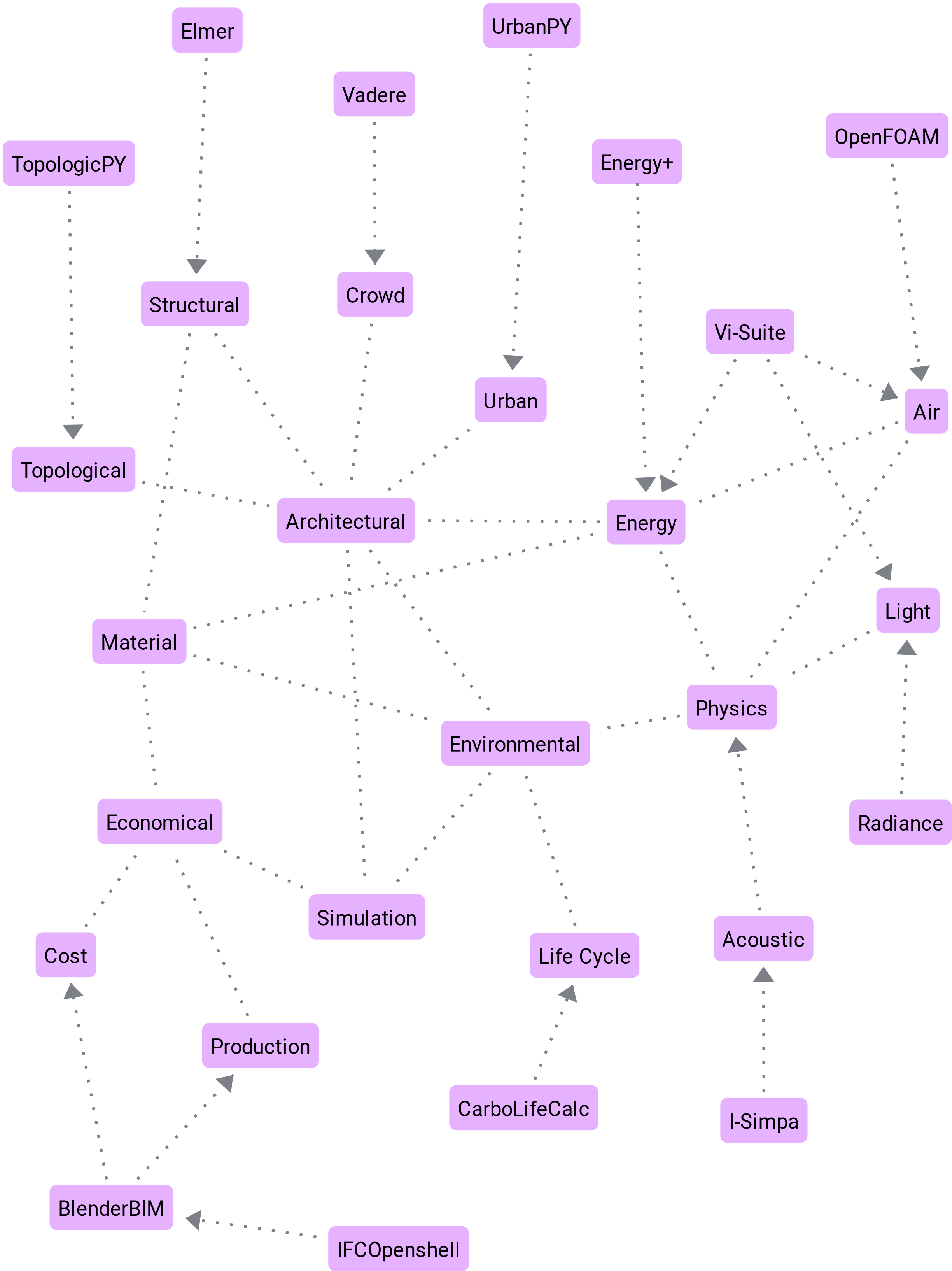
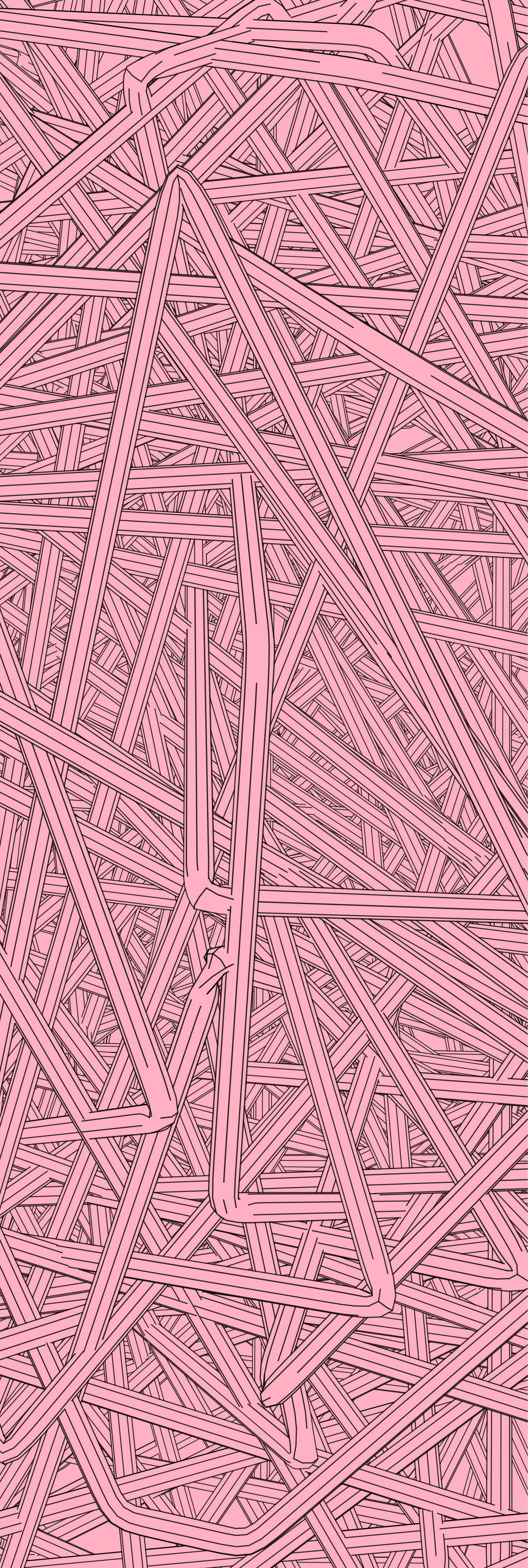
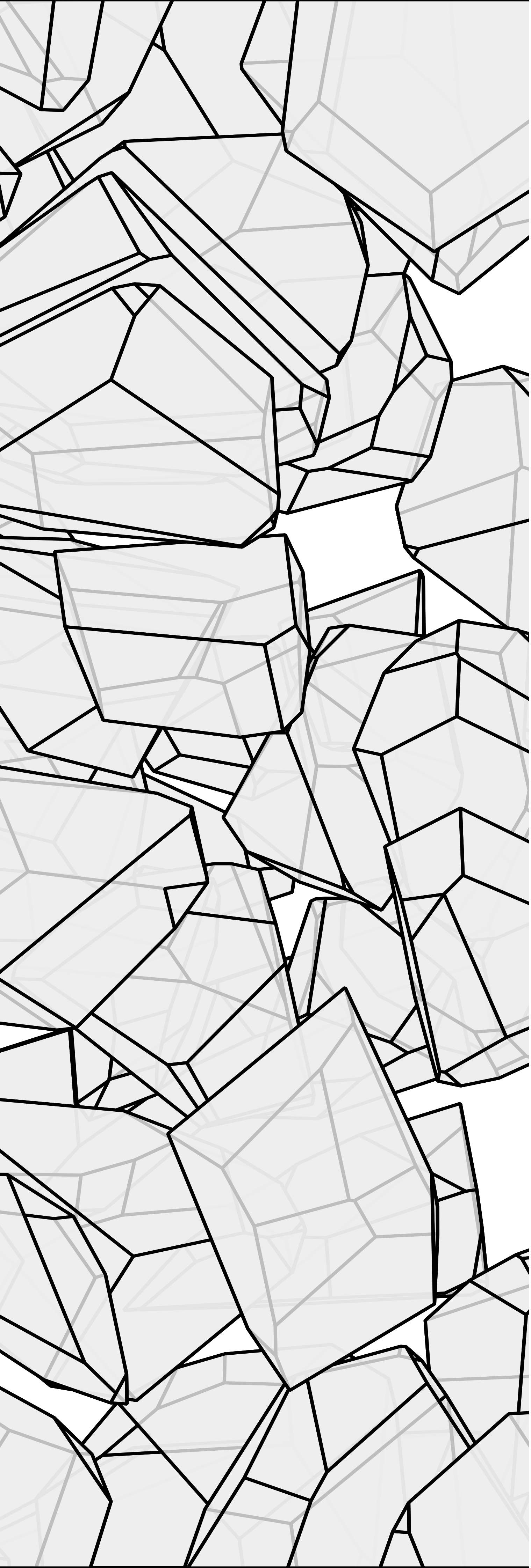


Figure 38: Data Augmentation



Learning

After the data set has been completed with the corresponding evaluation labels, it will be possible to analyze the relationship between the basic data and the achieved score using machine learning. Like the analysis step, the model training will not be part of this work but will be explored in detail as a separate topic in the following work. Nevertheless, the generation of the synthetic data set and its geometric data manipulation are the fundamental building blocks for a successful application of machine learning processes.
The exact methodology of the training stage cannot be defined at this stage, as the evaluation has not yet started and the training method and different ML approaches are highly dependent on the nature of the datasets. Basically, it is planned to try a variety of different methods and approaches with different data sets and to compare their performance, accuracy and results as well as potential applications.
This stage requires an extensive study of various computer science fundamentals to evaluate which method can lead to relevant results and in subsequent steps how the selected methods can be applied in a practical case. For this purpose a familiarization with file types like: CSV, GraphDB, Neo4j, SQL, JSON and DGCNN, data manipulation and visualization tools like Pandas, Numpy, matplotlib, pickle and mathutils, machine learnin libraries like Tensorflow, pytorch and scikit-learn. An essential part of this stage will be the exchange with professionals from the computer science field and feedback in various communities, since a fundamental basic understanding must be learned to effectively and properly arrive at the desired end result, an optimization in the architectural design process through instant feedback on the qualität of different characteristics of the object to be designed.



Conclusion

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Discussion

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Further Readings

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Future Works

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Used Software

Geometry

Blender (Version 3.2) [Computer Software]. (1994). Retrieved from https://www.blender.org

SciPy (Version 1.8.0) [Computer Software]. (2001). Retrieved from https://scipy.org/

Sverchok (Version 1.1.0) [Computer Software]. (2012). Retrieved from https://github.com/nortikin/sverchok

Shapely (Version 1.8.0) [Computer Software]. (2007). Retrieved from https://github.com/shapely/shapely

FreeCAD (Version 0.19.4) [Computer Software]. (2002). Retrieved from https://www.freecadweb.org

IfcOpenShell (Version 220330) [Computer Software]. (2011). Retrieved from https://blenderbim.org

SolveSpace (Version 1.0.4) [Computer Software]. (2019). Retrieved from https://solvespace.com/index.pl

Simulation

Topologic (Version 0.6.0) [Computer Software]. (2014). Retrieved from https://topologic.app

Radiance (Version 5.3) [Computer Software]. (1985). Retrieved from https://www.radiance-online.org

EnergyPlus (Version 22.1.0) [Computer Software]. (1996). Retrieved from https://energyplus.net

OpenFOAM (Version 9) [Computer Software]. (2004). Retrieved from https://openfoam.org

OpenStudio (Version 3.3.0) [Computer Software]. (2008). Retrieved from https://openstudio.net

Bullet (Version 3.21) [Computer Software]. (2012). Retrieved from https://pybullet.org

Vi-Suite (Version 0.7) [Computer Software]. (2013). Retrieved from https://github.com/rgsouthall/vi-suite07

Tools

Numpy (Version 1.22.3) [Computer Software]. (1995). Retrieved from https://numpy.org/

Pandas (Version 1.4.2) [Computer Software]. (2008). Retrieved from https://pandas.pydata.org/

LaTeX (Version 2022.02.24) [Computer Software]. (1984). Retrieved from https://www.latex-project.org

Inkscape (Version 1.1.2) [Computer Software]. (2003). Retrieved from https://inkscape.org

PyTorch (Version 1.11.0) [Computer Software]. (2016). Retrieved from https://pytorch.org

Scikit-Learn (Version 1.0.2) [Computer Software]. (2007). Retrieved from https://scikit-learn.org

Matplotlib (Version 3.5.1) [Computer Software]. (2003). Retrieved from https://matplotlib.org

Bibliography

[1] Asli Agirbas. Building energy performance of complex forms-test simulation of minimal surface-based form optimization. 2020.

[2] Rita Aguiar, Carmo Cardoso, et al. Algorithmic design and analysis fusing disciplines. 2017.

[3] Feyza Nur Aksin and Semra Arslan Selçuk. Use of simulation techniques and optimization tools for daylight, energy and thermal performance-the case of office module (s) in different climates. 2021.

[4] Firas Al-Douri. The employment of digital simulation in the planning departments in us cities-how does it affect design and decision-making processes? 2018.

[5] Amer Al-Jokhadar and Wassim Jabi. Humanising the computational design process: Integrating parametric models with qualitative dimensions. 2016.

[6] Pantea Alambeigi, Canhui Chen, Jane Burry, and Eva Cheng. Shape the design with sound performance prediction: a case study for exploring the impact of early sound performance prediction on architectural design. 2017.

[7] Christopher Alexander. *A pattern language: towns, buildings, construction*. Oxford university press, 1977.

[8] Abdulrahman Alymani, Wassim Jabi, and Padraig Corcoran. Machine learning methods for clustering architectural precedents-classifying the relationship between building and ground. 2020.

[9] Jayedi Aman, Nusrat Tabassum, James Hopfenblatt, Jong Bum Kim, and MD Haque. Optimizing container housing units for informal settlements-a parametric simulation & visualization workflow for architectural resilience. 2021.

[10] Dulce Andino and Sheng-Fen Chien. Embedding garifuna shape grammars in a parametric design software. *Blucher Design Proceedings*, 1(7):202–206, 2013.

[11] Mesrop Andriasyan, Alessandra Zanelli, Gevorg Yeghikyan, Rob Asher, and Hank Haeusler. Algorithmic planning and assessment of emergency settlements and refugee camps. 2020.

[12] V Anuradha and Vennila Thirumavalavn Minal Sabnis. Voronoi diagram voro [schemata]: Application of interactive weighted voronoi diagrams as an alternate master-planning framework for business parks. 2008.

[13] Logan Armstrong, Guy Gardner, and Christina James. Evolutionary solar architecture-generative solar design through soft forms and rigid logics. 2016.

[14] Hardik Arora, Jessica Bielski, Viktor Eisenstadt, Christoph Langenhan, Christoph Ziegler, Klaus-Dieter Althoff, and Andreas Dengel. Consistency checker-an automatic constraint-based evaluator for housing spatial configurations. 2021.

[15] Imdat As and Prithwish Basu. *The Routledge Companion to Artificial Intelligence in Architecture*. Routledge, Taylor & Francis Group, 2021.

[16] Imdat As, Siddharth Pal, and Prithwish Basu. Artificial intelligence in architecture: Generating conceptual design via deep learning. *International Journal of Architectural Computing*, 16(4):306–327, 2018.

[17] Fan Bao, Dong-Ming Yan, Niloy J Mitra, and Peter Wonka. Generating and exploring good building layouts. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013.

[18] Catarina Belém, Luís Santos, and António Leitão. On the impact of machine learning: Architecture without architects? 2019.

[19] Martin Bielik, Sven Schneider, Florian Geddert, and Dirk Donath. Addis building configurator: Computational design tool for efficient planning of mass housing in addis ababa. 2013.

[20] Jessica Bielski, Christoph Langenhan, Babara Weyand, Markus Neuber, Viktor Eisenstadt, and Klaus-Dieter Althoff. Topological queries and analysis of school buildings based on building information modeling (bim) using parametric design tools and visual programming to develop new building typologies. 2020.

[21] Christopher Boon, Corey Griffin, Nicholas Papaefthimious, Jonah Ross, and Kip Storey. Optimizing spatial adjacencies using evolutionary parametric tools: using grasshopper and galapagos to analyze, visualize, and improve complex architectural programming. *Perkins+ Will Research Journal*, 7(2):25–37, 2015.

[22] Nathan C Brown and Caitlin T Mueller. Design variable analysis and generation for performance-based parametric modeling in architecture. *International Journal of Architectural Computing*, 17(1):36–52, 2019.

[23] Inês Caetano, Luís Santos, and António Leitão. Computational design in architecture: Defining parametric, generative, and algorithmic design. *Frontiers of Architectural Research*, 9(2):287–300, 2020.

[24] Luísa G Caldas and Luís Santos. Generation of energy-efficient patio houses with gene_arch: Combining an evolutionary generative design system with a shape grammar. 2012.

[25] Victor Calixto and Gabriela Celani. A literature review for space planning optimization using an evolutionary algorithm approach: 1992-2014. 2015.

[26] D Campo, S Manninger, M Sanche, et al. The church of ai: An examination of architecture in a posthuman design ecology [c]. In *Proceedings of the 24th International Conference on Computer-Aided Architectural Design Research in Asia: Intelligent and Informed, CAADRIA 2019*, pages 767–772. CAADRIA Hong Kong, China, 2019.

[27] Matias Del Campo, Sandra Manninger, Leete Jane Wang, and Marianne Sanche. Sensibilities of artificial intelligence. In *Design Modelling Symposium Berlin*, pages 529–538. Springer, 2019.

[28] Giuseppe Canestrino. On the influence of evolutionary algorithm (ea) optimization in architectural design: A reflection through an architectural envelope's shadowing system design. 2021.

[29] Giuseppe Canestrino, Greco Laura, Francesco Spada, and Roberta Lucente. Generating architectural plan with evolutionary multiobjective optimization algorithms: a benchmark case with an existent construction system. 2020.

[30] Silvio Carta. Self-organizing floor plans. *Harvard Data Science Review HDSR*, 2021.

[31] Stanislas Chaillou. Ai architecture towards a new approach (2019). *URL https://www. academia. edu/39599650/AI_Architecture_Towards_a_New_Approach*.

[32] H Chalabee. Performance-based architectural design: optimisation of building opening generation using generative algorithms. *Master Sustain Archit Stud Univ Sheff Sch Archit*, 2013.

[33] Ioannis Chatzikonstantinou. A 3-dimensional architectural layout generation procedure for optimization applications: Dc-rvd. 2014.

[34] Aikaterini Chatzivasileiadi, Anas M Hosney Lila, Simon Lannon, and Wassim Jabi. The effect of reducing geometry complexity on energy simulation results. 2018.

[35] Paul Coates, Christian Derix, Ing Stefan Paul Krakhofer, and AbdulMajid Karanouh. Generating architectural spatial configurations. two approaches using voronoi tessellations and particle systems. 2005.

[36] Jan Cudzik and Kacper Radziszewski. Artificial intelligence aided architectural design. 2018.

[37] Subhajit Das, Colin Day, John Hauck, John Haymaker, and Diana Davis. Space plan generator: Rapid generationn & evaluation of floor plan design options to inform decision making. 2016.

[38] Matias del Campo. Architecture, language and ai-language, attentional generative adversarial networks (attngan) and architecture design. 2021.

[39] Matias del Campo, Alexandra Carlson, and Sandra Manninger. How machines learn to plan. 2020.

[40] Gözdenur Demir. Analysis of space layout using attraction force model and quadratic assignment problem. Master's thesis, Middle East Technical University, 2014.

[41] Theodoros Dounas, Wassim Jabi, and Davide Lombardi. Topology generated non-fungible tokens: blockchain as infrastructure for a circular economy in architectural design. 2021.

[42] Gavrilov Egor, Schneider Sven, Dennemark Martin, and Koenig Reinhard. Computer-aided approach to public buildings floor plan generation. magnetizing floor plan generator. *Procedia Manufacturing*, 44:132–139, 2020.

[43] Viktor Eisenstadt, Klaus-Dieter Althoff, and Christoph Langenhan. Student graduation projects in the context of framework for ai-based support of early conceptual phases in architecture. In *LWDA*, pages 174–179, 2020.

[44] Viktor Eisenstadt, Hardik Arora, Christoph Ziegler, Jessica Bielski, Christoph Langenhan, Klaus-Dieter Althoff, and Andreas Dengel. Comparative evaluation of tensor-based data representations for deep learning methods in architecture. 2021.

[45] Viktor Eisenstadt, Hardik Arora, Christoph Ziegler, Jessica Bielski, Christoph Langenhan, Klaus-Dieter Althoff, and Andreas Dengel. Exploring optimal ways to represent topological and spatial features of building designs in deep learning methods and applications for architecture. 2021.

[46] Viktor Eisenstadt, Christoph Langenhan, and Klaus-Dieter Althoff. Generation of floor plan variations with convolutional neural networks and case-based reasoning–an approach for unsupervised adaptation of room configurations within a framework for support of early conceptual design. In *eCAADe SIGraDi Conference, Porto*, 2019.

[47] Viktor Eisenstadt, Christoph Langenhan, Klaus-Dieter Althoff, and Andreas Dengel. Improved and visually enhanced case-based retrieval of room configurations for assistance in architectural design education. In *International Conference on Case-Based Reasoning*, pages 213–228. Springer, 2020.

[48] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

[49] E Friedrich, S Hanna, and C Derix. Emergent form from structural optimisation of the voronoi polyhedra structure. Generative Art International Conference, 2007.

[50] Ahmed Fawzy Gad. Pygad: An intuitive genetic algorithm python library, 2021.

[51] Katarzyna Grzesiak-Kopeć, Barbara Strug, and Grażyna Ślusarczyk. Evolutionary methods in house floor plan design. *Applied Sciences*, 11(17):8229, 2021.

[52] Tim Head, Manoj Kumar, Holger Nahrstaedt, Gilles Louppe, and Iaroslav Shcherbatyi. scikit-optimize/scikit-optimize, October 2021.

[53] Ruizhen Hu, Zeyu Huang, Yuhan Tang, Oliver Van Kaick, Hao Zhang, and Hui Huang. Graph2plan: Learning floorplan generation from layout graphs. *ACM Transactions on Graphics (TOG)*, 39(4):118–1, 2020.

[54] Mohamed Naeim A Ibrahim. Computing architectural layout. 2011.

[55] Tim Ireland. Emergent space diagrams: The application of swarm intelligence to the problem of automatic plan generation. 2009.

[56] Wassim Jabi. The potential of non-manifold topology in the early design stages. 2015.

[57] Wassim Jabi, Robert Aish, Simon Lannon, Aikaterini Chatzivasileiadi, and Nicholas Mario Wardhana. Topologic-a toolkit for spatial and topological modelling. 2018.

[58] Wassim Jabi, Aikaterini Chatzivasileiadi, Nicholas Mario Wardhana, Simon Lannon, and Robert Aish. The synergy of non-manifold topology and reinforcement learning for fire egress. 2019.

[59] Wassim Jabi, Barbara Grochal, and Adam Richardson. The potential of evolutionary methods in architectural design. 2013.

[60] Sam Conrad Joyce and Ibrahim Nazim. Limits to applied ml in planning and architecture-understanding and defining extents and capabilities. 2021.

[61] Ahti Kalervo, Juha Ylioinas, Markus Häikiö, Antti Karhu, and Juho Kannala. Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis. In *Scandinavian Conference on Image Analysis*, pages 28–40. Springer, 2019.

[62] Nariddh Khean, Lucas Kim, Jorge Martinez, Ben Doherty, Alessandra Fabbri, Nicole Gardner, and M Hank Haeusler. The introspection of deep neural networks-towards illuminating the black box-training architects machine learning via grasshopper definitions. 2018.

[63] Katja Knecht and Reinhard König. Generating floor plan layouts with kd trees and evolutionary algorithms. In *Generative Art Conf*, pages 238–253, 2010.

[64] Davide Lombardi, Theodoros Dounas, Lok Hang Cheung, and Wassim Jabi. Blockchain grammars for validating the design process. 2020.

[65] Yueheng Lu, Runjia Tian, Ao Li, Xiaoshi Wang, and Garcia del Castillo Lopez Jose Luis. Cubigraph5k-organizational graph generation for structured architectural floor plan dataset. 2021.

[66] Pascal Müller, Peter Wonka, Simon Haegler, Andreas Ulmer, and Luc Van Gool. Procedural modeling of buildings. In *ACM SIGGRAPH 2006 Papers*, pages 614–623. 2006.

[67] Danil Nagy, Damon Lau, John Locke, Jim Stoddart, Lorenzo Villaggi, Ray Wang, Dale Zhao, and David Benjamin. Project discover: An application of generative design for architectural space planning. In *Proceedings of the Symposium on Simulation for Architecture and Urban Design*, pages 1–8, 2017.

[68] Nelson Nauata, Kai-Hung Chang, Chin-Yi Cheng, Greg Mori, and Yasutaka Furukawa. House-gan: Relational generative adversarial networks for graph-constrained house layout generation. In *European Conference on Computer Vision*, pages 162–177. Springer, 2020.

[69] Nelson Nauata, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka Furukawa. House-gan++: Generative adversarial layout refinement networks. *arXiv preprint arXiv:2103.02574*, 2021.

[70] David Newton. Deep generative learning for the generation and analysis of architectural plans with small datasets. 2019.

[71] Maciej Nisztuk and Paweł Myszkowski. Tool for evolutionary aided architectural design. hybrid evolutionary algorithm applied to multi-objective automated floor plan generation. 2019.

[72] Maciej Nisztuk and Paweł B Myszkowski. Hybrid evolutionary algorithm applied to automated floor plan generation. *International Journal of Architectural Computing*, 17(3):260–283, 2019.

[73] Neri Oxman. Material-based design computation: Tiling behavior. 2009.

[74] Yuzhe Pan, Jin Qian, and Yingdong Hu. A preliminary study on the formation of the general layouts on the northern neighborhood community based on gaugan diversity output generator. In *The International Conference on Computational Design and Robotic Fabrication*, pages 179–188. Springer, 2020.

[75] A Papapavlou. *Structural Evolution: A genetic algorithm method to generate structurally optimal Delaunay triangulated space frames for dynamic loads.* PhD thesis, UCL (University College London), 2008.

[76] Greig Paterson, Sung Min Hong, Dejan Mumovic, and Judit Kimpian. Real-time environmental feedback at the early design stages. 2013.

[77] Bruno Postle. On pattern languages, design patterns and evolution.

[78] Mohammad Rahmani Asl, Subhajit Das, Barry Tsai, Ian Molloy, and Anthony Hauck. Energy model machine (emm)-instant building energy prediction using machine learning. 2017.

[79] Wiesław Rokicki and Ewelina Gawell. Voronoi diagrams–architectural and structural rod structure research model optimization. *MAZOWSZE Studia Regionalne*, (19):155–164, 2016.

[80] Richard Schaffranek. Parallel planning: An experimental study in spectral graph matching. In *Proceedings of the 10th International Space Syntax Symposium*, 2015.

[81] Adam Sebestyen and Jakub Tyc. Machine learning methods in energy simulations for architects and designers-the implementation of supervised machine learning in the context of the computational design process. 2020.

[82] Krishnendra Shekhawat, Nitant Upasani, Sumit Bisht, and Rahil Jain. Gplan: Computer-generated dimensioned floorplans for given adjacencies. *arXiv preprint arXiv:2008.01803*, 2020.

[83] Manav Mahan Singh, Patricia Schneider-Marin, Hannes Harter, Lang Werner, and Philipp Geyer. Applying deep learning and databases for energyefficient architectural design. In *Proceedings of the 38th International Online Conference on Education and Research in Computer Aided Architectural Design in Europe*, pages 79–87. eCAADe (Education and Research in Computer Aided Architectural Design in Europe), 2020.

[84] Kihoon Son and Kyung Hoon Hyun. A framework for multivariate data based floor plan retrieval and generation. 2021.

[85] Sherif Tarabishy, Stamatios Psarras, Marcin Kosicki, and Martha Tsigkari. Deep learning surrogate models for spatial and visual connectivity. *International Journal of Architectural Computing*, 18(1):53–66, 2020.

[86] Mahdi Valitabar, Mohammadjavad Mahdavinejad, and Peiman Pilechiha Henry Skates. Data-driven design of adaptive façades: View, glare, daylighting and energy efficiency. 2021.

[87] Yan-Chao Wang, Feng Lin, and Hock-Soon Seah. Orthogonal voronoi diagram and treemap. *arXiv preprint arXiv:1904.02348*, 2019.

[88] Nicholas Mario Wardhana, Wassim Jabi, Aikaterini Chatzivasileiadi, and Nikoleta Petrova. A spatial reasoning framework based on non-manifold topology. 2019.

[89] Shermeen Yousif and Daniel Bolojan. Deep-performance-incorporating deep learning for automating building performance simulation in generative systems. 2021.

[90] Xin Zhao, Yunsong Han, and Linhai Shen. Multi-objective optimisation of a free-form building shape to improve the solar energy utilisation potential using artificial neural networks. 2021.

[91] Hao Zheng, Keyao An, Jingxuan Wei, and Yue Ren. Apartment floor plans generation via generative adversarial networks. 2020.