

KATJA KNECHT**Topic: Architecture****Authors:****Katja Knecht**

Bauhaus-University
Weimar, Chair Computer
Science in Architecture
Germany

<http://infar.architektur.uni-weimar.de>

Reinhard König

Bauhaus-University
Weimar, Chair Computer
Science in Architecture
Germany

References:

[1] Jon Louis Bentley, "K-d Trees for Semidynamic Point Sets", in Proceedings of the Sixth Annual Symposium on Computational Geometry, Berkley, California, United States, 1990

[2] Tomor Elezkuraj, "Evolutionäre Algorithmen zur Unterstützung des kreativen architektonischen Entwerfens", TU Wien, Vienna, Austria, 2004

[3] Kazuhiro Kado, "An Investigation of Genetic Algorithms for Facility Layout Problems", University of Edinburgh, Edinburgh, UK, 1995

Contact:

katja.knecht@uni-weimar.de

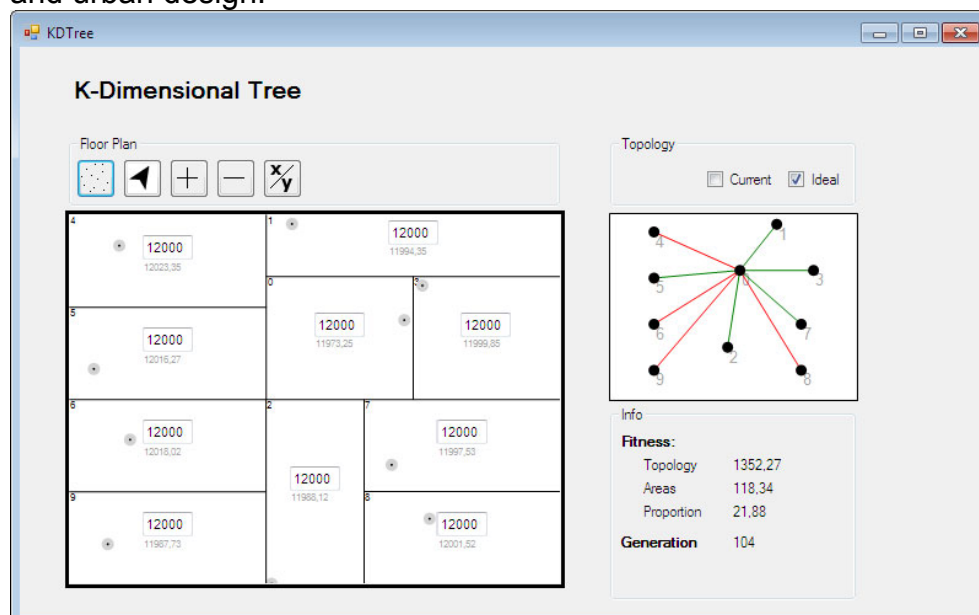
Paper : GENERATING FLOOR PLAN LAYOUTS WITH K-D TREES AND EVOLUTIONARY ALGORITHMS

Abstract:

K-dimensional trees, abbreviated as k-d trees in the following, are binary search and partitioning trees which represent a set of n points in a multi-dimensional space [1]. K-d tree data structures have primarily been used for nearest neighbor queries and several other query types for example in database applications. [1]

In the context of a research project at the Bauhaus-University Weimar concerned with the development of a creative evolutionary design method for layout problems in architecture and urban design, spatial partitioning with k-d trees has been applied as a partial solution to generate floor plan layouts. Unlike, for example, packing algorithms in [2] and slicing tree structures in [3] the employment of k-d tree algorithms in combination with evolutionary algorithms to generate floor plan layouts has not previously been examined in the scope presented here.

In the application developed in this project the k-d tree algorithm is initially used to subdivide a given rectangular area. The dividing lines thereby correspond to eventual spatial boundaries. By combining the k-d tree algorithm with genetic algorithms and evolutionary strategies, layouts can – in the current version - be optimized in three criteria dimensions (size, ratio and topology). Through user interaction the layouts can be dynamically adjusted and altered in real time. The result is a generative mechanism that provides an interesting and promising alternative to existing well-established algorithms for the creative and evolutionary solution of layout problems in architecture and urban design.



Application prototype

Keywords:

Generative layout design, architecture

Generating Floor Plan Layouts with K-d Trees and Evolutionary Algorithms

Dipl.-Ing. (FH) K. Knecht

Chair Computer Science in Architecture, Bauhaus-University Weimar, Germany.

<http://infar.architektur.uni-weimar.de>

e-mail: katja.knecht@uni-weimar.de

Interim Prof. Dr.-Ing. R. König

Chair Computer Science in Architecture, Bauhaus-University Weimar, Germany.

Abstract

K-dimensional trees, abbreviated as k-d trees in the following, are binary search and partitioning trees that represent a set of n points in a multi-dimensional space. K-d tree data structures have primarily been used for nearest neighbor queries and several other query types, for example in database applications.

In the context of a research project at the Bauhaus-University Weimar concerned with the development of a creative evolutionary design method for layout problems in architecture and urban design, spatial partitioning with k-d trees has been applied as a partial solution to generate floor plan layouts. Unlike, for example, packing algorithms and slicing tree structures, the employment of k-d tree algorithms in combination with evolutionary algorithms to generate floor plan layouts has not previously been examined in the scope presented here.

In the application developed in this project the k-d tree algorithm is initially used to subdivide a given rectangular area. The dividing lines thereby correspond to eventual spatial boundaries. By combining the k-d tree algorithm with genetic algorithms and evolutionary strategies, layouts can – in the current version - be optimized in three criteria dimensions (size, ratio and topology). Through user interaction the layouts can be dynamically adjusted and altered in real time. The result is a generative mechanism that provides an interesting and promising alternative to existing well-established algorithms for the creative and evolutionary solution of layout problems in architecture and urban design.

1. Introduction

K-dimensional trees (k-d trees) are binary data structures originating from the field of computational geometry which are employed in numerous search algorithms. Due to the efficiency and the speed with which they can be traversed k-d trees are employed in different areas such as data base applications and ray tracing methods. Their efficiency stems from the space partitioning algorithm with which they organize objects in k-dimensional space and code the data structure. In addition, the space

partitioning algorithm produces a geometrical, plot-like structure, whose creative potential has as yet not been investigated in detail.

Algorithms such as this and their generative and creative potential have, however, been studied for a while in the field of generative architecture and have, for example, been applied to generate floor plan layouts. The first attempts in computer-aided layout generation date back to the 1960s [1]. The design and development of layouts i.e. the arrangement of rooms, buildings and building complexes in architectural and urban contexts are a central task in architecture and urban planning. As this involves both functional as well as creative problem solving, generative processes are often combined with evolutionary methods. The work of Elezkurtaj [2] is an example that combines a packing algorithm with evolutionary strategies and genetic algorithms.

In the scope of a research project at the Bauhaus University Weimar on the development of a creative evolutionary design method for layout problems in architecture and urban planning (Kremlas) we developed a mechanism which, for the first time, employs space partitioning by k-d trees to generate floor plans in combination with generative algorithms and evolutionary strategies.

Chapter 3 of this paper gives an insight into the current development and results of our investigations in the Kremlas research project. Chapter 4 discusses the potential of the k-d tree algorithm as a creative and generative means whose possibilities for further development are described in chapter 5. To begin with chapter 2 provides a short introduction to k-dimensional trees along with a brief overview of the current state of research and related work in the field of generative architecture.

2. Background and related work

2.1 An introduction to k-d trees

K-dimensional trees, according to Bentley [3], are binary search trees that organize a set of n points in a multidimensional space [4] and thus represent data structures that can be used for space partitioning [5].

K-d tree data structures are commonly used for nearest neighbor queries [6] and other search algorithms, in particular for traditional database applications [4]. K-d tree data structures in connection with search algorithms are used for diverse applications (see for example [7, 8, 9]) due to their high traversal speed, accuracy and efficiency of use. This includes, for example, facial recognition in the interactive tracking of facial features in video frames [10], for which the speed and accuracy of the results is highly relevant. The use of k-d tree data structures in ray tracing techniques has also proved worthwhile (see, for example, [11] and [12]).

Previous applications of k-d trees have, therefore, focused on their capacity to organize and store spatial data as well as to search and traverse data structures quickly and efficiently.

Starting from a known, finite set of points, the k-dimensional space is divided by so called partition planes perpendicular to one of the coordinate axes. In two-dimensional space, subdivision corresponds to a line, in three- and multidimensional

spaces it corresponds to a hyper-plane that is aligned to one axis [5]. For this purpose, the median or average of the point coordinates based on the point set P is calculated in the split dimension. The result is the first node N with the calculated median or average as split value, which divides the point set P into two subsets P_1 and P_2 , the so-called sub-trees. All points whose coordinates are smaller than the split value in the respective split dimension, form the left branch and are part of sub-tree P_1 ; all points whose coordinates are larger form the right branch and are part of sub-tree P_2 . N is considered as the root node for the sub-nodes N_1 and N_2 of the sub-trees P_1 and P_2 ; N_1 and N_2 are child nodes of N . These subsets are then further subdivided following the same system described for the initial point set. The subdivision continues until further subdivision is no longer possible, or a certain threshold or a given depth has been reached.

A region is assigned to every node which in turn can contain child nodes and subspaces. Within the search tree, nodes are called internal nodes. They divide the space by a split plane in one of the k dimensions [4]. In addition, the internal nodes store information on the partition plane as well as references to the right and left child nodes. The endpoints of the tree are referred to as leaves, or as external nodes, or buckets [4].

The data structure thus performs three functions at once: it stores the data set, subdivides the space into hyper-rectangles and at the same time provides a directory for these hyper-rectangles [13].

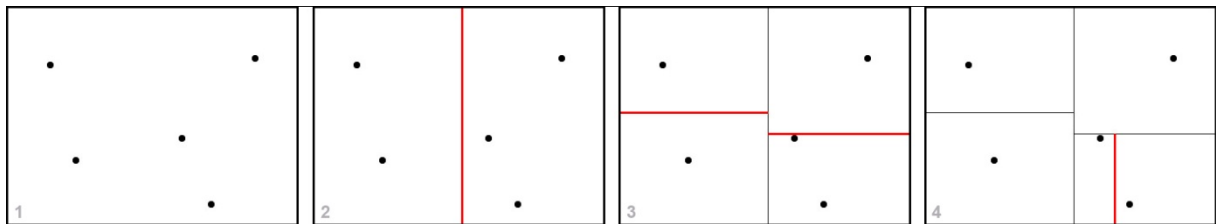


Fig. 1: Space partitioning with a k-d tree structure

The structure of the tree, and with it the geometric representation, depends on the splitting rule, i.e. how, in which dimension and in which sequence the partition planes are built [14]. One can distinguish two basic types, point-based and region-based k-d trees [5], which are based on different splitting rules.

How subdivisions are performed depends on the distribution of points in k -dimensional space for point-based k-d trees. An affiliated splitting rule is the standard splitting rule based on [15] where the split is performed in the dimension in which the points have the maximum spread [5]. The split value is in this case calculated from the median value of the points' coordinates. The sequence of split dimensions may also differ from the one used in the standard splitting rule. It can be set as a continuous sequence of splits along the x -, y - until k - axis [3, 16]. The cycle starts anew after the split in the k th dimension.

A region-based k-d tree can be built by applying the midpoint splitting rule [14]. The partition plane runs through the center of the region intersecting its longest side. The sliding midpoint rule presented in [14] is a further development that combines point- as well as region-based features. At first it also subdivides the region through its center but then moves the split plane to the nearest point.

As already stated, split values can also be determined by calculating the average instead of the median of the points' coordinates resulting in k-d trees with different structural compositions. If a k-d tree is built by median calculation, the partition planes run through the median points so that a point is assigned to each internal node. By contrast, all points in a tree built by average calculation lie in a leaf region as the partition planes run through new points whose coordinates are generated from the average value of the point coordinates.

2.2 An overview of automatic layout generation

The design and development of layouts is a central key task in architecture and urban planning and spans several levels of scale. The composition of plots, buildings or rooms in architectural and urban contexts respectively is supposed to be functional as well as creative. For several decades, architects and planners have increasingly made use of by software tools to assist in solving such layout problems. Research has been conducted not just in the development of drawing, modeling or representation tools but also in the development of generative systems to support the creative problem solving process.

The first approaches for computer-aided layout generation date back to the 1960s [1]. Numerous projects emerged in this field dealing with different algorithms, structures and models for generating urban and architectural layouts and assisting in the automation of the design process. Among the generative mechanisms developed and implemented are cellular automata which have been employed at both an urban and regional scale [17] and for architectural space [18], as well as Lindenmeyer systems [19], constraint- [20] and physically-based systems [21] and shape grammars, which, for example, have been used for the generative description of styles and to generate floor plans based on existing architectural archetypes [22].

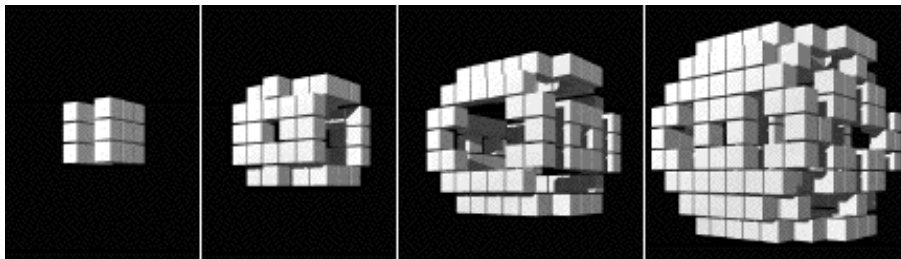


Fig. 2: Cellular Automata [18]

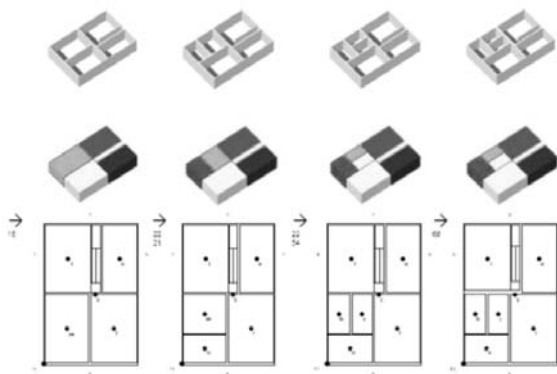


Fig. 3: Jose Duarte, Customizing Mass Housing [22]

With respect to k-d trees, a number of recent projects on Voronoi-Diagrams and slicing tree structures are of particular interest. Like k-d trees, Voronoi-Diagrams also partition space into subspaces, so called cells, dependent on the distribution of objects in space. The objects are referred to as cell centers. Each cell contains all the points that lie closer to its center than to any other [23]. The subdivisions, and with it the diagram, are formed by the borders of the cell, i.e. those points that have the same distance to several centers [16]. Foam-like structures and topologies emerge which are defined by the cell centers and their position, number and distribution. Voronoi-Diagrams have been employed to generate urban designs (see [24]) and architectural spaces (see for example [23, 25]).

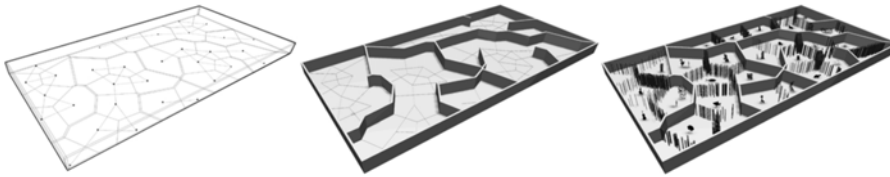


Fig. 4: Voronoi-Diagram [25]

Another method also based on space partitioning is the slicing tree structure. This was presented by Kado [26] to solve facility layout problems. Like k-d trees, they subdivide a region in a top-down-approach with split lines into sub-spaces which correspond to facilities. The split sequence is also represented as a tree structure; the partition planes, however, are not dependent on the distribution of a point set.

The most promising approach with regard to automatic layout generation offers a combination of generative structures and evolutionary algorithms. First attempts in the field of architecture can be found in Frazer [27] and Coates [28]. A design model for space layout planning with evolutionary methods has been developed and tested by Jo and Gero [29], for example. When using evolutionary methods one needs to differentiate between the optimization of existing floor plans and the generation of new floor plan layouts [19]. The second case is incomparably more complex as the creative design process incorporates the solution of operational as well as non-operational criteria. Criteria are operational when they describe a problem clearly and therefore incorporate the solution in their description. Non-operational criteria describe ambiguous problems for which there is no objectively correct solution.

A relatively advanced prototype which combines generative methods with evolutionary algorithms and also takes non-operational criteria into consideration can be found in the work of Elezkurtaj [2]. The system which he developed uses a packing algorithm which calculates and displays optimized layout proposals in real time. At the same time the user is given the opportunity to interact with the layout and customize it according to his own needs.

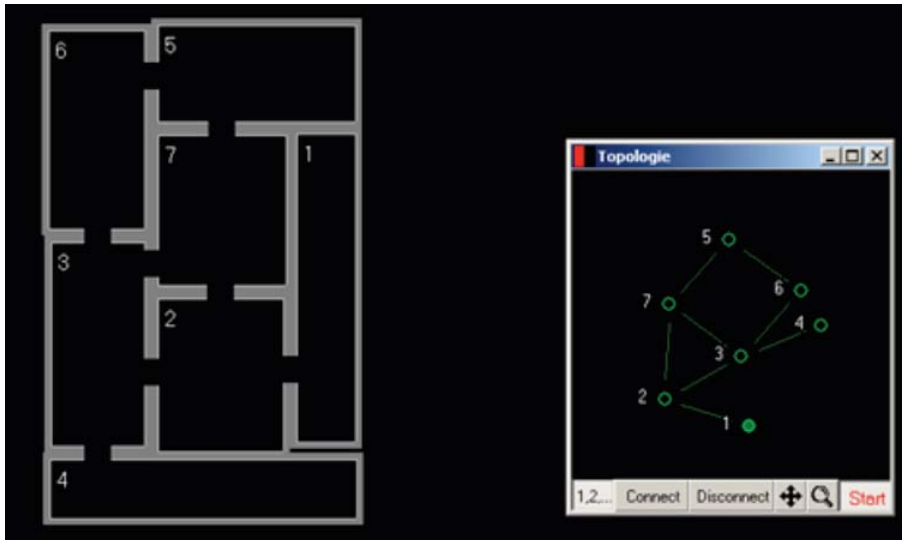


Fig. 5: Tomor Elezkurtaj, *Interactive Layout System* [2]

3. K-d Trees for the generation of floor plan layouts

3.1 Research background

In the scope of the Kremlas research project we look at layout problems on different levels of scale ranging from the urban neighborhood to the building structure and floor plan organization. Unlike other studies, which predominantly employ computer-based tools to optimize floor plans, our focus lies on the development of a method which includes the creative component and takes non-operational criteria into account (see [30]).

The design process is cyclical and consists of the consideration of a problem, the definition of criteria to solve the problem, the generation of solutions and their evaluation and optimization by adaptation of the criteria. It does not follow a linear sequence and is therefore characterized by the continuous adaptation of the solution space by the designer over the course of the working process. Consequently a design system has to be flexible and applicable for different design problems. Top-down and bottom-up strategies i.e. starting with both a minimal as well as a highly restricted description of the problem, need to be combined seamlessly with one another. In [30] we suggest the development of an adaptive design system for layout problems that allows the user to define, adapt and explore the design space flexibly.

Furthermore designing very often takes place on various scales at once. A system that solves layout problems should therefore be adaptively applicable on different levels of scale. Layout problems generally entail the sensible arrangement of different elements such as plots, buildings, functional areas, rooms, furniture et cetera. The definition and the requirements of the elements change on different scales and with them the demands they place on the system. In order to enable a seamless transition between scales during the design process with as few constraints as possible, we chose a quite universal definition of layout for the data structure of the system. Layout is defined as nested elements, i.e. the arrangement of elements in elements, in elements [30].

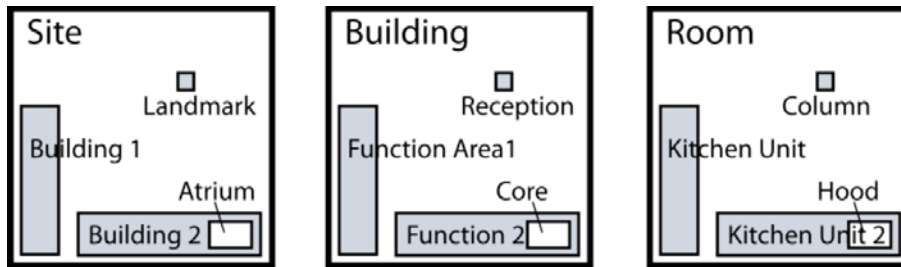


Fig. 6: The same layout at different scales with different interpretations of the elements [30]

This results in a hierarchical structure across many levels in which homogenous as well as heterogeneous elements can mix. In general it has to be possible to create relationships between elements of different hierarchical order as well as across different layout levels. This concerns, for example, the formation of required or desired neighbor relations between elements in both horizontal and vertical dependency. The resulting data structure is object oriented. A layout element is defined as an object with properties and constraints as well as defined relations to other objects.

As previously stated, the design process is cyclical which means that some problems are only identified during the process. Consequently, the design system used cannot be purely linear and solution-oriented but has to enable the designer to adjust and even redefine the problem during the design process. Consequently we follow an explorative approach based on a recursive, circular trial and error system [30].

The system employs generative and evaluative mechanisms based on evolutionary algorithms. An important role in evolutionary algorithms is played by the evaluative function which evaluates the generated solutions according to certain criteria and therefore determines the direction of the solution space. These criteria vary from designer to designer and according to design task which is why we propose employing criteria which are as neutral and multifunctional as possible so that they are valid for all scales and can be flexibly assigned to elements as well as controlled by parameters [30]. These include criteria such as topology, orientation, proportion and size. A multi-criteria fitness function facilitates flexible and problem-oriented weighting during the evaluation process. Criteria specific to a certain scale, such as density, building line, opacity and connectivity, can also be applied to evaluate the layout.

We have studied several generative strategies to build a generative mechanism that leads to results with different formal characteristics. The first strategy assesses and evaluates the quality of elements. In this case we applied a packing algorithm to generate solutions in which possible spatial configurations arise from the arrangement, relocation and proportional adjustment of the elements. The second approach focuses on the implementation of space partitioning rules. We employed a k-d tree algorithm as well as a slicing tree algorithm. Initially both strategies only support rectangular forms.

3.2 Generative mechanism

The idea of using a k-d tree algorithm to form a generative mechanism emerged from the examination of its geometric characteristics, the potential of its partitioning structure as well as its data structure. The data structure of the k-d trees fits the concept of the nested elements. Layout elements correspond to external nodes i.e. elements of a subordinate hierarchical level that can subsequently be assigned specific functions within the layout. Internal nodes represent elements of super ordinate hierarchical order that can bundle subordinate internal and external nodes into, for example, functional areas. Internal and external nodes represent central objects in the implementation which are defined by properties, such as data point and position as well as region. At the same time the k-d tree itself can be seen as an object that represents the tree structure as references between root and child nodes.

At first the investigation was explicitly limited to the application of k-d trees to the building scale, to generate floor plan layouts. The first step consisted of the realization of a simple k-d tree algorithm which subdivides a given area by mean value calculation. Thus each point of the point set represents a room. The form of representation described has the advantage that the size of the point set determines the number of rooms generated in the floor plan. The split lines correspond to potential space boundaries.

The generative mechanism builds up on that in a second step. The generation of solutions is a combination of an evolutionary strategy to optimize room sizes and genetic algorithms to optimize topology as well as room ratios. The following section describes the three generative mechanisms which are assigned three populations (one per criteria dimension).

The first mechanism searches for specific predefined room sizes. The problem of room size in relation to the k-d tree algorithm is determined by the sizes of the generated rooms through partitioning which depend on the position and distribution of the points in space. Evenly distributed points usually generate subspaces of similar size. In order to optimize the sizes the positions of the single points have to be manipulated in such a way that their distribution after partitioning creates the desired size. The use of an evolutionary strategy proved to be useful for optimizing which locations are mutated. The desired room size can be set by the user for every single room and interactively modified.

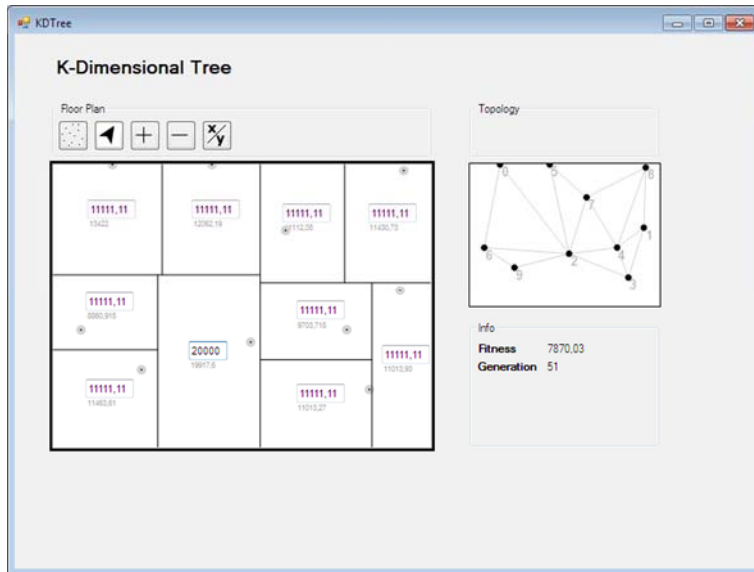


Fig. 7: Optimization of room sizes

The second criterion concerns neighbor relationships between points and the assigned spaces respectively. The problem of topology consists in the assignment of functions to spaces so that they have the desired topological relationships [31]. The optimization can be supported by a genetic algorithm in which the indices, coded as a genome, represent the spatial sequence. Mutational operations exchange the indices of individual rooms. The desired neighbor relationships between rooms are defined by the user and set in the topology.

The third mechanism is for finding sensible room proportions. The problem of proportion regards the ratio of the longest to the shortest side of a rectangular sub space and arises from the split dimension chosen. The solution to this problem is implemented using a genetic algorithm in which the split sequence (the sequence of splits in x and y direction) is coded as a genome and is recombined and mutated. Unlike conventional k-d trees the split dimensions in the nodes are not determined statically in advance or by the distribution of the points but vary dynamically. The ideal proportion of a room depends on its function in the floor plan and is set by the user. As long and narrow rooms are not practical for most uses square room ratios are deemed advantageous. Access spaces such as corridors and entrances represent an exception to this rule.

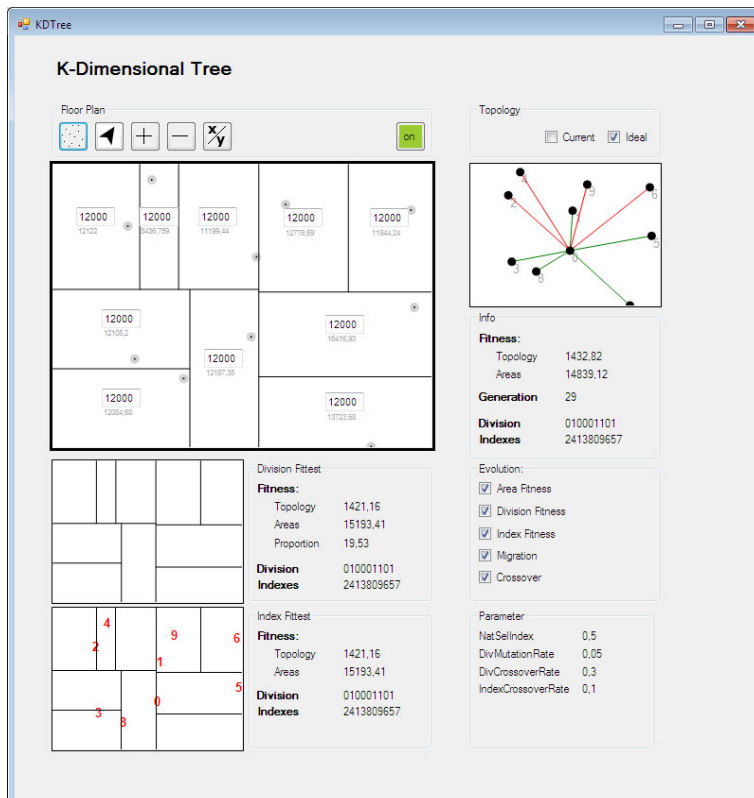


Fig. 8: Optimization according to three criteria

The three populations are optimized in a co-evolutionary fashion resulting in permanently changing and dynamic problem descriptions. Through migration they approach a collective Pareto front of optimal solutions over time. The employment of populations in evolutionary algorithms can be regarded as an analogy of the characteristics of the architectonic design process. Each population represents a design solution whereby several different solutions are developed in parallel, compared with one another and combined. This description also applies for the early phases of the design process when the aim is to initiate creative thought processes. The evolutionary generative methods assist the designer by facilitating a complex and computer-aided search for design solutions [32].

3.3 User interaction

The user should be able to search the solution space created by the generative mechanisms flexibly and to adjust it according to his needs. By means of direct user interaction non-operational criteria are incorporated in the solution-generating process, which was previously only based on the description and optimization of operational criteria through automatic floor plan generation. The system has to offer sensible functions and interaction possibilities to allow the user to influence the optimization process directly. The impact of such changes has to be reflected instantly, i.e. the system has to work in real-time [30].

User intervention takes place directly in the graphical representation of the solution by mouse click, mouse movement or keyboard input. As the generative mechanisms work on a k-d tree basis, the interaction possibilities encompass first and foremost functionality for point manipulation. This includes adding and deleting as well as moving points by which the user can influence the number of rooms as well as their

distribution and shape. The values for desired room sizes can be manipulated by manual input. In addition, the user can define neighbor relationships between room functions in the representation of the topological structure.

The actions of the users result in the immediate adaptation of the populations based on the changed properties of the objects. As the evolutionary process continues to run in the background, new solutions are almost simultaneously generated allowing the user to readjust immediately.

Based on the software prototype described it is possible to examine how the generated layouts react to user interaction. The adjustments usually take place gradually and are easily understood by the user even when, for example, the point distribution changes considerably through relocation so that a new split sequence and new topological relationships emerge.

4. Evaluation and outlook

During the realization of the generative mechanism we were able to demonstrate that k-d tree algorithms, in combination with evolutionary strategies and genetic algorithms can generate interesting and useful solutions for floor plan layouts due to their splitting rule and geometrical structure. The evolutionary algorithms applied have a positive impact on the geometrical properties of the k-d trees in return. They can help to control ratios, sizes and topologies of the regions and therefore influence the character of the overall structure. The layout solutions generated also appear more flexible and diverse as they are no longer subject to strict splitting rules but the split dimensions are dynamically adjusted instead.

A next step entails integrating the k-d tree data structure into the overall system which, due to its hierarchical structure, should benefit similarly. Connected to this is the examination of the application of the partitioning algorithm across scales as well as the interconnection of layout elements across levels and scales. An aspect that will need special consideration is the extent to which k-d trees support the arrangement of elements with fixed properties, for example staircases and other access and infrastructure units that often connect multiple levels. The relocation of a point in a k-d structure causes the dynamic adjustment of the size and ratio of hierarchically adjacent elements, which is inherently problematic for items with fixed properties.

The point-based approach of the k-d trees also complicates the design of the user interaction. The primary means of interaction by manipulating points is not very intuitive. When combining the generative mechanisms of the k-d tree with the overall system, it will be essential to devise a consistent user interface that facilitates the seamless transition between different levels of scale as well as incorporating other generative mechanisms which may employ area- or line-based interaction approaches.

A first prototypical application allows the movement of lines and regions. The difficulty resides, however, in calculating the location and distribution of the point set based on the location of split lines and regions, which is in effect an inversion of the current

algorithm. Herein too also lies the key to resolving the handover of areas of fixed position, size and ratio across levels and layers as mentioned above.

As part of the future development of the proposed system we also plan to extend the multi-criteria optimization model. Further investigations will examine extending the model through topological criteria such as direction as well as the integration of classical tools for analysis such as the angle of the sun and visibility.

Furthermore the generative mechanisms presently access only a limited part of the potential solution space as the examination was originally restricted to orthogonal partitions and rectangular spatial structures. Future developments could potentially also examine the generation of complex geometries, extending the present orthogonal pattern of splitting to non-orthogonal planes.

5. Conclusion

In the investigations presented in this paper we examined the geometrical qualities and creative possibilities of space partitioning with k-d tree algorithms to solve layout problems in architecture and urban design. Initially, we looked at the application areas of generative mechanisms on the scale of a building with a view to generating floor plan layouts. In combination with genetic algorithms and evolutionary strategies we developed methods for optimizing floor plans according to multiple criteria. The user can actively intervene in the generative method while searching for a solution.

The k-d tree algorithm supports the general data model of the design system as well as appropriate geometric layout solutions based on space partitioning due to its data structure. The point-based characteristic of the algorithm is advantageous for the generation of solutions as well as their evolution. However, there is still need for development with regard to the design of seamless user interaction and the seamless transition between hierarchical and layout levels.

To conclude we have demonstrated that space partitioning with k-d trees in combination with evolutionary algorithms presents an interesting and promising alternative to already well-known strategies for the creative algorithmic solution of layout problems in architecture and urban planning. Future investigations into the aspects referred to in chapter 4 will be of interest in helping to explore the possibilities that k-d data structures provide for this area of research.

6. Acknowledgements

The project is funded by the German Research Foundation (DO 551/19-1).

7. References

1. Frazer, J., *Reptiles*. Architectural Design, 1974(April): p. 231-239.

2. Elezkurtaj, T. and G. Franck, *Algorithmic Support of Creative Architectural Design*. Umbau, 2002. 19: p. 129-37.
3. Bentley, J.L., *Multidimensional Binary Search Trees Used for Associative Searching*, in *Communications of the ACM*. 1975, ACM: New York, NY, USA. p. 509 - 517.
4. Bentley, J.L., *K-d Trees for Semidynamic Point Sets*, in *Sixth Annual Symposium on Computational Geometry*. 1990, ACM: Berkley, California, United States. p. 187 - 197.
5. Goodrich, M.T. and R. Tamassia, *Algorithm Design: Foundations, Analysis and Internet Examples*. 2002, New York, United States: John Wiley & Sons
6. Moore, A.W. (1991) *An introductory tutorial on kd-trees*. Extract from Andrew Moore's PhD Thesis: Efficient Memory-based Learning for Robot Control Technical Report No. 209.
7. Carela-Español, V., et al., *K-Dimensional Trees for Continuous Traffic Classification*, in *Traffic Monitoring and Analysis*, F. Ricciato, M. Mellia, and E. Biersack, Editors. 2010, Springer Berlin / Heidelberg. p. 141-154.
8. Deng, K. and A.W. Moore, *Multiresolution Instance-Based Learning*, in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*. 1995, Morgan Kaufmann Publishers Inc.: Montreal, Quebec, Canada. p. 1233 - 1239.
9. Moore, A.W., *Very Fast EM-based Mixture Model Clustering using Multiresolution kd-trees*, in *Advances in Neural Information Processing Systems 11*, M.S. Kearns, S.A. Solla, and D.A. Cohn, Editors. 1999, MIT Press: Cambridge. p. 543 - 549.
10. Buchanan, A. and A. Fitzgibbon, *Interactive Feature Tracking using K-D Trees and Dynamic Programming*, in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2006, IEEE Computer Society. p. 626 - 633.
11. Fussell, D.S. and K.R. Subramanian, *Fast Ray Tracing Using K-d Trees*, in *Technical Report: CS-TR-88-07* 1988: Austin, TX, USA.
12. Wald, I. and V. Havran, *On building fast kd-Trees for Ray Tracing, and on doing that in $O(N \log N)$* , in *Proceedings of the 2006 IEEE Symposium on Interactive Ray Tracing*. 2006: Salt Lake City, UT, United States. p. 61 - 69.
13. Bentley, J.L. and J.H. Friedman, *Data Structures for Range Searching*, in *ACM Computing Surveys (CSUR)*. 1979, ACM: New York, NY, USA. p. 397 - 409.
14. Maneewongvatana, S. and D.M. Mount, *It's okay to be skinny if your friends are fat*, in *Center for Geometric Computing 4th Annual Workshop on Computational Geometry*. 1999: John Hopkins University
15. Friedman, J.H., J.L. Bentley, and R.A. Finkel, *An Algorithm for Finding Best Matches in Logarithmic Expected Time*, in *ACM Transactions on Mathematical Software (TOMS)*. 1977, ACM. p. 209 - 226.

16. De Berg, M., et al., *Computational Geometry: Algorithms and Applications*. 1997, Berlin; Heidelberg; New York; London; Barcelona; Budapest; Hong Kong; Milan; Paris; Santa Clara; Singapore; Tokyo: Springer.
17. Koenig, R. and C. Bauriedel, *Computer-generated City Structures*, in *Generative Art Conference*. 2004: Milan.
18. Coates, P., et al., *The use of Cellular Automata to explore bottom up architectonic rules*, in *Eurographics UK Chapter 14th Annual Conference*. 1996, Eurographics Association UK: London.
19. Coates, P., T. Broughton, and H. Jackson, *Exploring three-dimensional design worlds using Lindenmeyer systems and Genetic Programming*, in *Evolutionary Design Using Computers*, P. Bentley, Editor. 1999, Morgan Kaufmann Publishers Inc.: San Francisco, California. p. 323 - 341.
20. Li, S.-P., J.H. Frazer, and M.-X. Tang. *A Constraint-Based Generative System for Floor Layouts*. in *Fifth Conference on Computer Aided Architectural Design Research in Asia (CAADRIA)*. 2000. Singapore
21. Arvin, S.A. and D.H. House, *Modeling architectural design objectives in physically based space planning*. *Automation in Construction*, 2002. 11: p. 213–225.
22. Duarte, J.P., *Customizing mass housing: a discursive grammar for Siza's Malagueira houses*, in *Faculty of Architecture*. 2000, Massachusetts Institute of Technology.
23. Coates, P., et al., *Generating architectural spatial configurations. Two approaches using Voronoi tessellations and particle systems*, in *Proceedings of the VIII Generative Art International Conference (GA2005)*. 2005: Milan, Italy.
24. Koltsova, A., et al., *A Case Study of Script-Based Techniques in Urban Planning*, in *Design Computing and Cognition DCC'10*, J.S. Gero, Editor. 2010, Springer: Stuttgart. p. 671 - 690.
25. Harding, J. and C. Derix, *Associative Spatial Networks in Architectural Design: Artificial Cognition of Space using Neural Networks with Spectral Graph Theory*, in *Design Computing and Cognition DCC'10*, J.S. Gero, Editor. 2010, Springer: Stuttgart. p. 305 - 323.
26. Kado, K., *An Investigation of Genetic Algorithms for Facility Layout Problems*. 1995, University of Edinburgh: Edinburgh.
27. Frazer, J., *An Evolutionary Architecture*. 1995, London: Architectural Association Publications.
28. Coates, P. and L. Hazarika. *The use of genetic programming for applications in the field of spatial composition*. in *Proceedings of the 2nd Generative Art Conference (GA1999)*. 1999. Milan: Generative Design Lab Milan Polytechnic University, Italy.
29. Jo, J.H. and J.S. Gero, *Space layout planning using an evolutionary approach*. *Artificial Intelligence in Engineering*, 1998. 12: p. 149-162.

30. Schneider, S., J.-R. Fischer, and R. König, *Rethinking Automated Layout Design: Developing a Creative Evolutionary Design Method for Layout Problems in Architecture and Urban Design*, in *Design Computing and Cognition DCC'10*, J.S. Gero, Editor. 2010, Springer Verlag: Stuttgart, Germany. p. 367 - 386.
31. Elezkurtaj, T., *Evolutionäre Algorithmen zur Unterstützung des kreativen architektonischen Entwerfens*, in *Institut für Architekturwissenschaften*. 2004, Technische Universität Wien: Vienna, Austria.
32. Von Buelow, P., *Using Evolutionary Algorithms to Aid Designers of Architectural Structures*, in *Creative Evolutionary Systems*, P.J. Bentley and D.W. Corne, Editors. 2001, Morgan Kaufmann Publishers: San Francisco, CA, USA. p. 315 - 336.