Article

International Journal of
Architectural Computing
2018, Vol. 16(3) 212–226
© The Author(s) 2018
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/1478077118798395
journals.sagepub.com/home/jac
SAGE

Computational urban design prototyping: Interactive planning synthesis methods—a case study in Cape Town

Yufan Miao[1], Reinhard Koenig[2], Katja Knecht[1], Kateryna Konieva[1], Peter Buš[3] and Mei-Chih Chang[3]

Abstract
This article is motivated by the fact that in Cape Town, South Africa, approximately 7.5 million people live in informal settlements and focuses on potential upgrading strategies for such sites. To this end, we developed a computational method for rapid urban design prototyping. The corresponding planning tool generates urban layouts including street network, blocks, parcels and buildings based on an urban designer's specific requirements. It can be used to scale and replicate a developed urban planning concept to fit different sites. To facilitate the layout generation process computationally, we developed a new data structure to represent street networks, land parcellation, and the relationship between the two. We also introduced a nested parcellation strategy to reduce the number of irregular shapes generated due to algorithmic limitations. Network analysis methods are applied to control the distribution of buildings in the communities so that preferred neighborhood relationships can be considered in the design process. Finally, we demonstrate how to compare designs based on various urban analysis measures and discuss the limitations that arise when we apply our method in practice, especially when dealing with more complex urban design scenarios.

Keywords
Procedural modeling, spatial synthesis, generative design, urban planning

Introduction

Modern urban design processes are characterized by increasing complexities and dynamics. However, traditional urban design methods still rely heavily on static and sectoral approaches. At the same time, from modeling to manufacturing, computers are playing an increasingly important role in design processes. Many aspects of the design process previously carried out by hand are now automated. In the context of urban

[1]Future Cities Laboratory, Singapore
[2]Department of Energy, Austrian Institute of Technology, Vienna, Austria
[3]Departement Architektur, Eidgenössische Technische Hochschule Zürich, Zürich, Switzerland

Corresponding author:
Yufan Miao, Future Cities Laboratory, 1 Create Way, 138602 Singapore.
Email: miao@arch.ethz.ch

design, automation enables the fast generation and exploration of different design options using computational tools so that designers can acquire a comprehensive overview and understanding of different design choices. We call this method computational urban design prototyping (CUDP). It aims to support the designer's tasks through the automated generation and evaluation of multiple urban design solutions.

CUDP is rooted in the concept of virtual prototyping (VP), which has been widely studied and implemented in the field of engineering design.[1–3] Song et al.[4] describe VP as "the process of simulating the user, the product, and their combined (physical) interaction in software through the different stages of product design, and the quantitative performance analysis of the product," which stresses the importance of the interaction between human and product in the design process facilitated by VP.[5] CUDP adapts the concept of VP to the context of urban design, placing emphasis on (1) computationally supported interaction between urban designers and urban design layouts, and (2) the corresponding spatial analysis. Like VP, CUDP has the potential to reduce time, save costs, and improve quality.[6] More specifically, it reduces the time that urban designers spend exploring different variants of urban layouts; it saves manpower costs and improves the quality of the design solutions by making it possible to test and evaluate different variants rapidly. It also makes the design process explicit and traceable and designs scalable and reproducible. Collaboration and communication platforms can additionally help to connect, engage, and better facilitate the discussion of alternative designs and their respective trade-offs, and to involve different stakeholders.

The research presented here evolved within the context of a collaboration with the Urban-Think Tank (UTT) at ETH Zurich as part of the Empower Shack project.[7] The project is located in the city of Cape Town in South Africa, where approximately 7.5 million people live in informal settlements and about 2.5 million housing units are needed. The Empower Shack project focuses on developing upgrading strategies for these informal settlements. Our research aims to enhance the design process with the help of new digital planning tools and an evaluation framework that can be used to replicate developed urban planning concepts for different sites. The CUDP method we developed makes it possible to adapt an urban planning concept to meet different site-specific requirements or changing specifications as defined by urban designers and stakeholders. Consequently, the tool was designed to be design- and user-centered with two main characteristics: first, the parameters guiding the generative process were tailored to fit the requirements defined by the designers based on their understanding of the design task. The designers can create design variants by adjusting parameter values. Second, the application allows designers to interact with the generated designs not only parametrically via numeric parameter inputs but also geometrically by changing geometric constraints. User interaction makes it possible to improve generated designs that do not fully meet the requirements and customized parameters help urban designers to naturally and precisely specify their needs. The interaction capability was realized using the software platform Rhino3D together with its add-on Grasshopper, which is a visual programming environment for parametric design. The parameters for the CUDP tool were determined through discussions with the UTT urban design team and local stakeholders. The aims of this study are as follows:

- To develop a new data structure to represent spatial configurations (street networks, blocks, and parcels).
- To translate the requirements of urban planners and stakeholder to parameters for the CUDP.
- To generate spatial configurations that fulfill all restrictions, which are primarily:
    - Efficient use of the available space on the site by densely packing housing units;
    - Fair allocation of private and public spaces;
    - Preserve local communities by distributing households based on their neighborhood preferences.
- To generate spatial configurations that are comparable with manually created designs in terms of the level of detail, geometric precision, and spatial qualities.

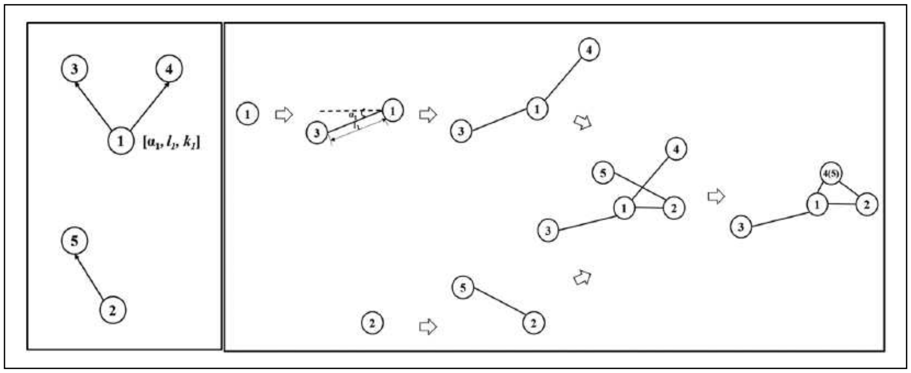


Figure 1. The instruction tree nodes contain the following three parameters per node: length *l*, angle α, and degree of connectivity *k*. On the left side, two instruction trees are presented with the arrows defining the relationship between nodes and the generative order (e.g. from node 1 to node 3 and node 4). On the right side, geometries are generated based on the parameters, with α defining the angle between itself and its child, *l* is the length of each street segment between nodes, and *k* is the maximum number of children. When the root nodes are combined together to form a graph, adjustments are made automatically. In this case, root nodes are node 1 and node 2, and node 4 and 5 are merged in the result because of crossing.

In the following description of the CUDP, we describe the methods and data structures for the generation of urban fabric and demonstrate its use through a case study. The article comprises three sections: "Methods," "Case study," and "Results and conclusion." In the "Methods" section, we present the data structure for the representation and the algorithm for the generation of spatial configurations consisting of street networks, blocks, parcels, and buildings. In the "Case study" section, we present the tool in use and the spatial configurations resulting from the requirements specified by UTT for the Empower Shack project. We show the analysis method employed to evaluate pedestrian accessibility on the site and present a way to preserve communities by allocating units based on neighborhood preferences using spatial clustering of households.

Methods

For generating urban layouts including streets, blocks, parcels, and buildings, it is necessary to establish a function that maps specified parameters to expected layouts. In the following, we present the abstract data structure we developed to realize such a mapping.

Representation of street networks as instruction trees

To represent a street network, we employ a so-called instruction tree,[8] which has been implemented and tested in the CPlan framework.[9] An instruction tree and its corresponding street network is shown in Figure 1. Each node in the instruction tree contains information on how to add a street segment. The following three parameters are stored with each node: the length of the connecting segment, the angle, and the degree of connectivity. Edges between the instruction tree nodes define the relationships between the nodes, and therefore to which parent node a segment is connected in the network.

Urban blocks are regarded as "rings" of enclosing street segments. Buildings are independent of the generated street network and blocks can be regarded as a separate layer, in which building volumes are projected onto the parcels. However, in the abstract representation of spatial configurations we encounter two additional problems: first, the instruction tree needs to be extended to include parcel structures; and second, the connection between the street network, the embedded blocks, and the parcels has to be defined.

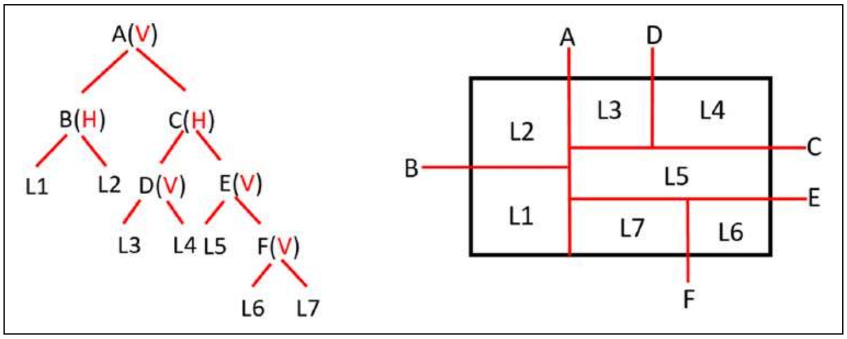


Figure 2. The slicing tree structure. Left: the geometric representation of the sliced parcels. Right: the tree representation of the sliced parcels where "V" indicates vertical slicing and "H" horizontal slicing. The indices of the parcels on the left correspond to the ones on the right.

Representation of parcels as slicing trees

To represent parcels, we employ a slicing tree data structure. This subdivision method has the advantage of being efficient in construction and for querying.[10] In computer science, such kinds of data structure are popularly used in data clustering techniques such as *k*-*d* trees.[11] The data structure and its mapping are shown in Figure 2. The geometric representation of the parcels resulting from the subdivision of a block can be seen on the left, and the abstract tree representation of the parcels and the instructions for slicing on the right. The slicing rules are encoded in the nodes of the slicing tree. The generative slicing process (depicted step by step in Figure 3) is recursive and the algorithm stops after reaching a certain threshold. As meaningful threshold parameter, we used the minimum area of a plot,[10] or the minimum width of the parcel on a street side. In the case study presented below, the stakeholders considered the width of the parcel to be more important.

To generate satisfying urban layouts, the data structure had to be adapted to urban design requirements. Normally, slicing is performed along the longest edges of the blocks or parcels to avoid the generation of very long and thin parcels. However, depending on the urban context, slicing may need to be performed only along certain street segments to ensure that the generated parcels face a street and buildings on the parcels can be accessed accordingly. For example, in Figure 2, parcel L5 is longer than the others, but it is not further subdivided because it has only one street-facing edge, which is not long enough to be further divided.

In reality, blocks are not always as regularly shaped as depicted in Figures 2 and 3. To deal with irregular shapes, Koenig and Knecht[10] frame the irregular block in a rectangular bounding box and then slice the rectangle, as shown on the left in Figure 4. The bounding box is defined by the minimum and maximum coordinate values of the polygon. This strategy ensures the slicing process is consistent. One disadvantage is that this can result in parcels with irregular shapes that are not ideal for urban design. To improve on this, we perform slicing on a minimum bounding box defined by the smallest width, as shown on the right in Figure 4. Moreover, to ensure the block is sliced perpendicular to the street segment, the shapes are rotated before being sliced (Figure 5). After being sliced, the shapes are rotated back to their original positions.

Introduction of a revised slicing tree data structure to represent the relationship between urban layers

Given the hierarchical nature of the urban layers, the generation of parcels is highly dependent on the layout of the street network. It is important to ensure that the street networks are generated in a way that enables the generation of subsequent layers. To generate extended urban layouts, the new data structure has to be able to

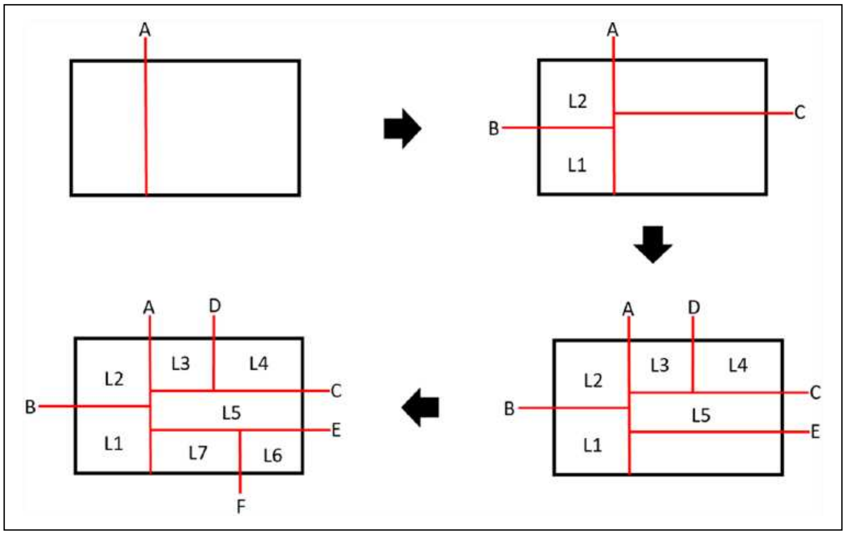


Figure 3. The slicing process works top-down from the root of the slicing tree to the leaves. The four steps correspond to the four levels of the tree.

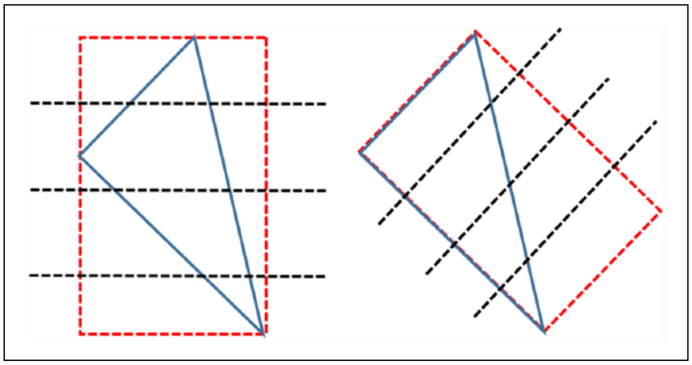


Figure 4. Slicing of a triangular shape (blue triangle) based on a bounding box (red rectangle). On the left, the slicing operation is performed on a bounding box defined by the minimum and maximum *x* and *y* coordinates of the polygon whereas on the right, slicing is performed on the minimum bounding box defined by the shape's smallest edge.

capture interconnections between different layers, for example, between street network, blocks, parcels, and buildings. To define the relationship between street network and parcels computationally, we propose merging the previously presented tree structures in the form of a revised slicing tree.

The revised slicing tree is illustrated in Figure 6 using two blocks as examples, in which we have applied different slicing rules. In the block on the left, slicing (A) starts from the long edge 1. After this, edge 2 on the left side is sliced at B and edge 4 on the right at C. The process continues recursively until the stopping criteria is reached. In general, this process is the same as that described above in Figure 2 except that in order to retain information on the interconnection between street segments and parcel structure, the revised slicing tree stores the slicing line in relationship to the edge it slices. At the same time, the slicing lines are ranked based on their levels in the tree structure and the rank of each slicing line in the tree is also stored as an

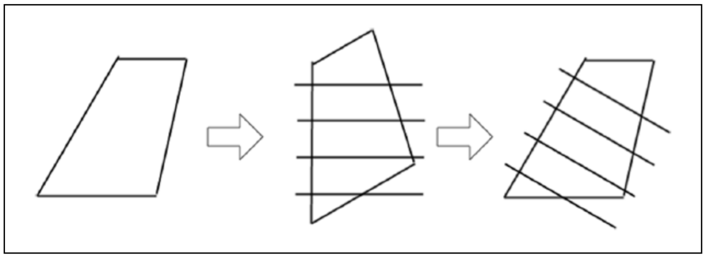


Figure 5. Slicing on a rotated shape to adapt the slicing process to the nature of the shape.

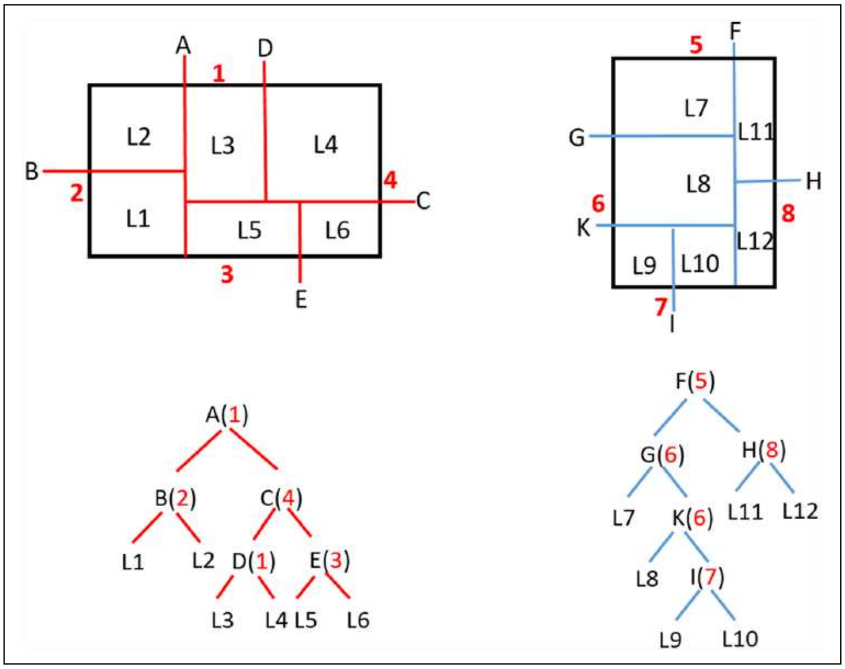


Figure 6. Revised slicing tree representation (bottom) for two parcels (top). The numbers indicate the indices of the street segments, the letters the indices of the slicing lines. L_1, L_2, …, L_n are leaves of the tree as well as the parcels.

additional attribute to the corresponding edge. The closer the levels of nodes are to the root, the higher their rank. Moreover, the higher the rank a node has, the earlier the slicing operation is performed. If a slicing line crosses more than one edge, the slicing line is assigned to one edge randomly. If a rank number is missing, then an edge with a lower rank is promoted to be sliced first.

Application of the data structure to enable evolutionary multi-objective optimization

The system presented in this article constitutes a so-called simple reflex agent,[12] which can only take actions based on specified rules and given parameters. Simple reflex agents are considered the most rudimentary form of artificial intelligence systems, as they do not possess a memory, are not aware of goals,

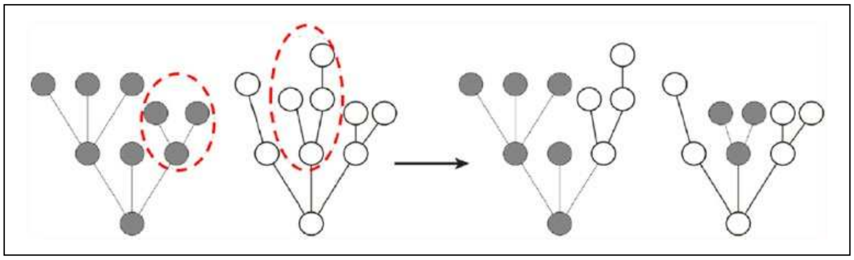


Figure 7. Illustration of the crossover operation as an important operation of an EMO. A branch of the black tree and a branch of the white tree are exchanged to form two new trees.

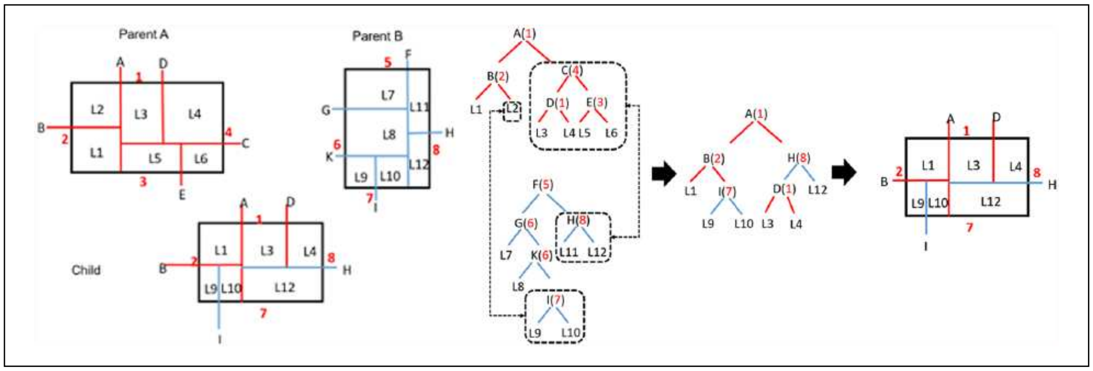


Figure 8. The crossover process using the revised slicing tree structure. The generated geometry of the block on the far right inherits parts of the spatial configurations of its parents.

and are not able to evaluate their actions and learn from them. To improve the design system and equip it with the above-mentioned capabilities in future, we propose employing evolutionary multi-objective optimization (EMO).[13] Although the actual implementation of the EMO is beyond the scope of this article and constitutes future work, we designed the presented data structure to be usable as a genotype representation for this EMO process.

Koenig et al.[8] have shown that EMO can be applied to the generation of street networks. Figure 7 illustrates a recombination process in EMO based on existing instruction trees for representing street networks.[8] However, as discussed in the previous section, a different data structure is required to be able to include more elements, namely, parcels and buildings, in the generation of urban layouts and to address the relationship of the elements in their genotype representation.

The crossover process for the revised slicing tree is illustrated in Figure 8. A feature of this kind of data structure is that it preserves the slicing sequence of blocks while enabling the crossover process of swapping branches of trees, which is exactly what is needed in the recombination process of EMO. As the generated parcels in Figure 8 show, the revised slicing tree helps the descendants preserve the spatial configurations of their parents. The revised slicing tree structure reduces the number of genotype representations of different urban layers into one. If we represent the slicing direction (horizontal or vertical) as (d_i), the position of the slicing line on the street segment as (a_i), and the rank of each slicing line as (r_i), then the parameters that determine the chromosomes i are $[\alpha_i, l_i, k_i, d_{i1}, p_{i1}, r_{i1}, \ldots, d_{in}, p_{in}, r_{in}]$, where n is the number of slicing lines on each street segment.

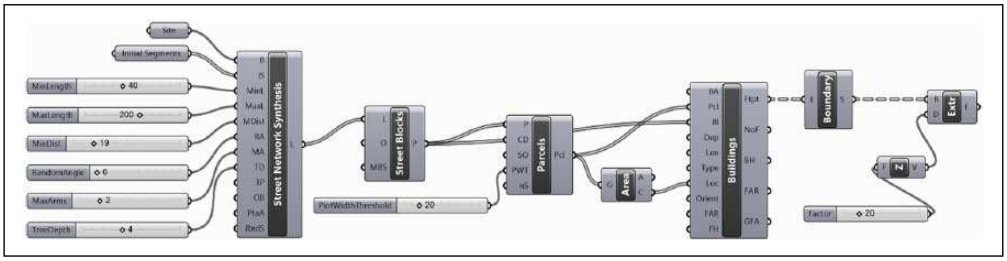


Figure 9. The Grasshopper components for generating street networks, street blocks, parcels, and buildings.

Implementation of Grasshopper components to facilitate parametric urban design

In order to demonstrate our new data structure in a case study, we used the software platform Rhino3D and Grasshopper. We use this software as the basis for our system because it is widely used in design practice and has a very flexible modular extension system. Using custom components, we can overcome the limitations of Grasshopper while taking advantage of it as a design environment. The custom components, we developed are provided as part of the DeCodingSpaces Toolbox (http://decodingspaces-toolbox.org/). In Figure 9, we illustrate the use of the new components for the process of generating an urban layout, starting with the generation of street networks (Street Network Synthesis component on the left in Figure 9). In the sequence of Grasshopper's data flow model from left to right, we then extract the blocks (Street Blocks component) from the street network and slice the blocks into parcels (Parcels component) before finally placing buildings in the parcels (Buildings component). This generic procedure is adapted for the case study of the Empower Shack project in the following section.

Case study

Our case study was conducted within the context of the Empower Shack project led by the Urban-Think Tank (UTT) group at ETH Zurich in 2016. The objective of the project is to densify and upgrade an area previously covered by informal single-story buildings. The aim of the collaboration was to provide an interactive planning tool which would make it possible to generate a layout of streets and parcels that makes efficient use of the available space on the site by densely packing housing units, but, at the same time, ensures their accessibility to facilities and fairly allocates private and public spaces. Furthermore, the aim was to support the preservation of local communities by assisting in the allocation of parcels to households based on their neighborhood preferences. The study area Enkanini in Cape Town City, South Africa, is shown in Figure 10. The aim of the collaboration was to provide an interactive planning tool for urban designers that makes it possible to rapidly prototype variations of street and parcel layouts based on given design requirements in a fast and interactive way.

Stakeholder design requirements

The requirements from UTT and the stakeholders can be summarized as follows: the tool should allow to make efficient use of the available space on the site by densely packing housing units, but, at the same time, ensuring their accessibility to facilities, as well as a fair allocation of private and public spaces. Furthermore, it should support the preservation of local communities by providing assistance in parcel allocation by observing neighborhood preferences. In addition, the tool should allow the manual adjustment of street layouts and the ensuing automatic regeneration and redistribution of parcels. To support discussion and

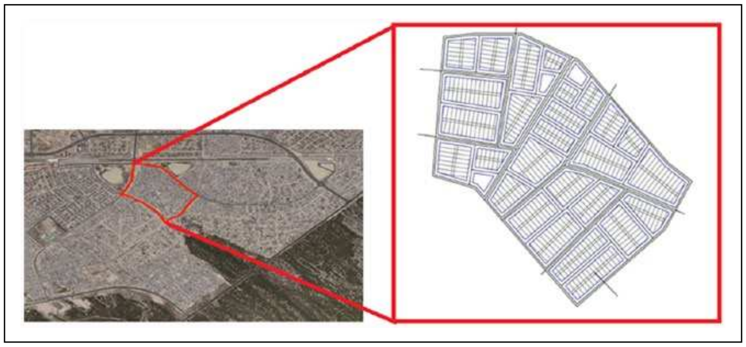


Figure 10. The study area selected from the Empower Shack project with a generated urban fabric of streets, blocks, and parcels.

negotiation among stakeholders, the Grasshopper interface should connect to a web interface for design exploration. To ensure a fair allocation of private space while making maximum use of land, the designers needed to be able to specify the exact dimensions of the parcels in terms of their width or depth and to fit as many regularly shaped parcels as possible in the study area.

Generation of street networks with regularly shaped parcels and buildings using nesting

Using the methods described below, we were able to address these requirements and generate basic urban layouts as depicted in Figure 11. For the generation of such layouts, the designer need only specify the border of the planning area, the initial street segments, and the dimensions of the individual parcel (width and depth). The generation of the street layout starts from initial street segments, which are used as root nodes for the instruction tree.

To address the requirements, we had to adapt the original method. The top-down, recursive way in which parcels were generated could not guarantee the generation of equal-sized blocks of a defined width and length required for the plots. Since the slicing algorithm terminates on reaching a certain threshold, parcels with a greater width could be generated rather than the desired size. To overcome this shortcoming, we nested block and parcel generation. In the resulting Grasshopper definition, we first partition the street blocks into smaller blocks of pre-defined width using the parcel component and only then do we divide them into parcels (Figure 12).

This nesting approach resulted in more homogeneous parcel sizes. In order to fully comply with the required precision as well as the requirement to place as many units on the site as possible, we further customized the definition. Blocks with a depth larger than 2.5 times of the desired parcel depth were offset to contain a courtyard. The resulting space around the block edges was subdivided into parcels of the desired widths. Buildings at the specified dimensions were placed within the generated parcel outlines using the building generation component (Figure 13).

Exploring design variants

Using the Speckle plug-in for Grasshopper,[14,15] generated urban layouts can be made accessible online to share and communicate the design as well as to allow stakeholders to explore design alternatives (Figure 14). In accordance with the initial specifications, the interface permits the dimensions of the parcels to be varied within a defined parameter range using sliders to adjust the parcels' width and

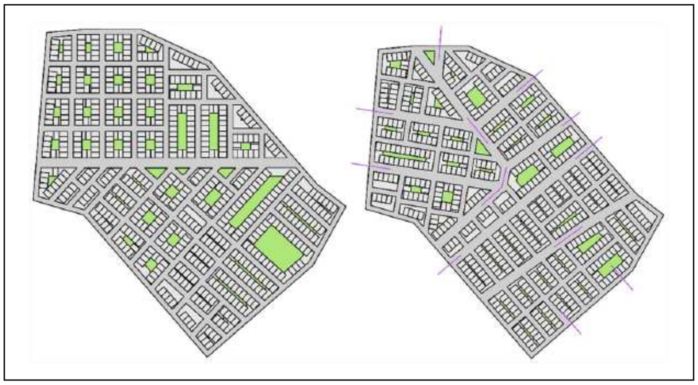


Figure 11. Example of two urban layouts for the sliced parcels, generated freely (left) and using initial pre-defined street segments (red lines) to guide street network generation (right).

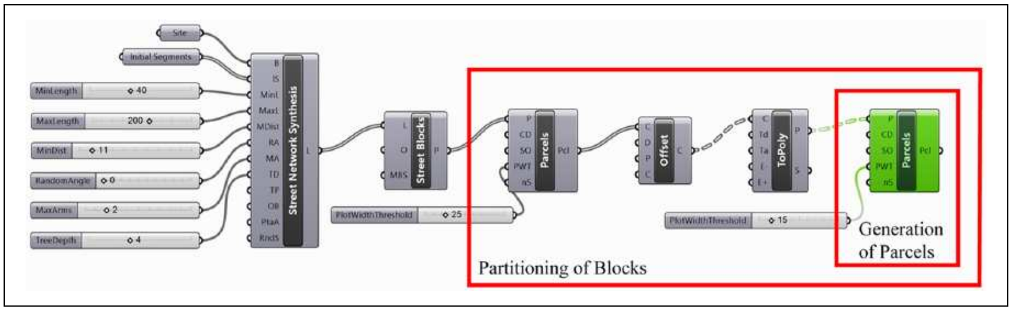


Figure 12. An illustration of the nested parcel generation method. Blocks are first partitioned into smaller ones (large red box) using the slicing tree and then partitioned into parcels (small red box).

depth. The impact of changes on the distribution of the parcels and the changes of the overall layout of the site can be evaluated in the comparative view of the online viewer. Furthermore, basic design performance indicators can be displayed, for example, the number of units placed on the site in a specific design variant.

Preference-based clustering

The generated urban layouts were enriched with a preference-based clustering algorithm (Figure 15). Since most of the citizens of the new planned area already live there in informal housing, social networks between the people exist. Based on people's preferences expressed at the beginning of the planning process, households were assigned to certain parcels, respectively, buildings by taking into account preferences for unit sizes and proximity to other households. The aim here is to ensure that former residents can retain their social ties and communities can persist in the newly planned neighborhood. To cluster people who want to live near to each other, we applied social network analysis based on cognitive distances using the NodeXL algorithm.[16] The difficulty here is how to transfer the preference-based household clusters obtained from the questionnaires to a spatial distribution of housing to form corresponding spatial communities. We needed to ensure that the buildings constituting a spatial cluster were close to each other. The measurement of

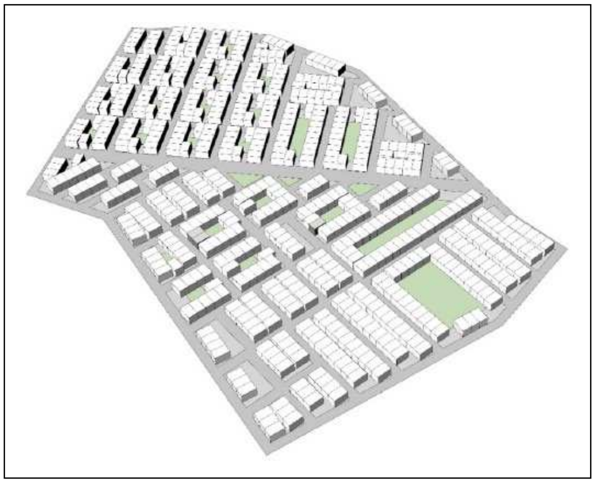
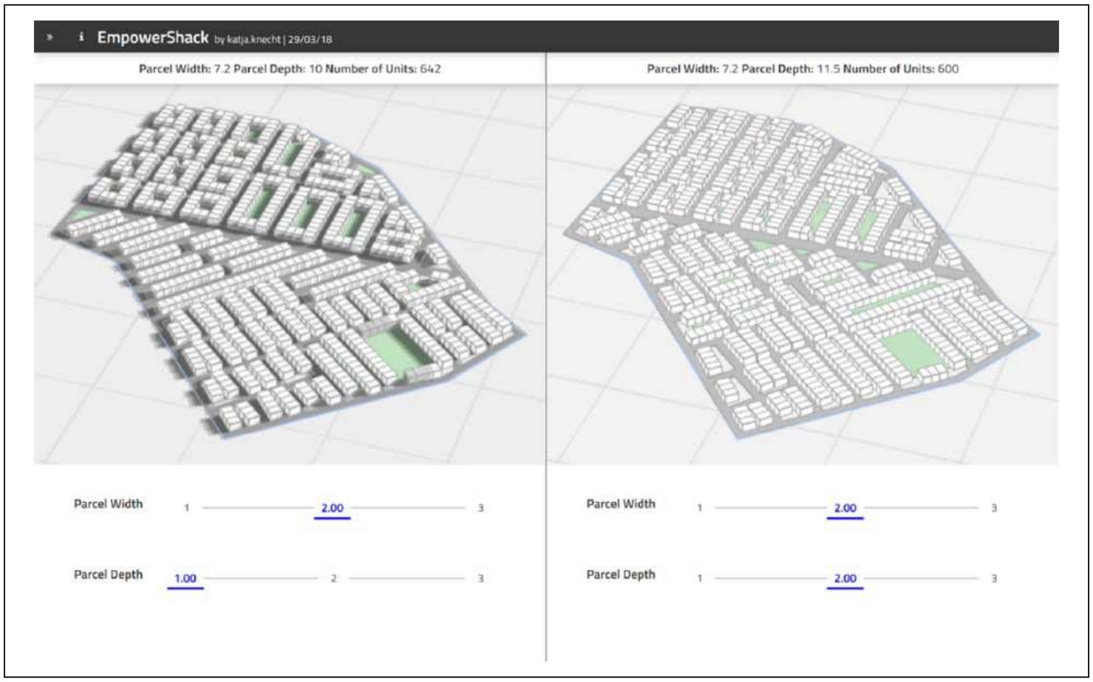


Figure 13. Placement of basic building masses on generated parcels.



EmpowerShack by katja.knecht | 29/03/18

Parcel Width: 7.2 Parcel Depth: 10 Number of Units: 642

Parcel Width: 1 — 2.00 — 3

Parcel Depth: 1.00 — 2 — 3

Parcel Width: 7.2 Parcel Depth: 11.5 Number of Units: 600

Parcel Width: 1 — 2.00 — 3

Parcel Depth: 1 — 2.00 — 3

Figure 14. Exploring design alternatives in the beta.speckle online viewer.

cognitive distance in urban space depends on the angular distance,[17,18] which represents a better empirical model of perceived distances than metric distance. The ranked-shortest angular paths for all buildings make it possible to map household clusters into building clusters (Figure 15). This method also allowed alternative building placements with different priorities of unit sizes as shown in Figure 16.

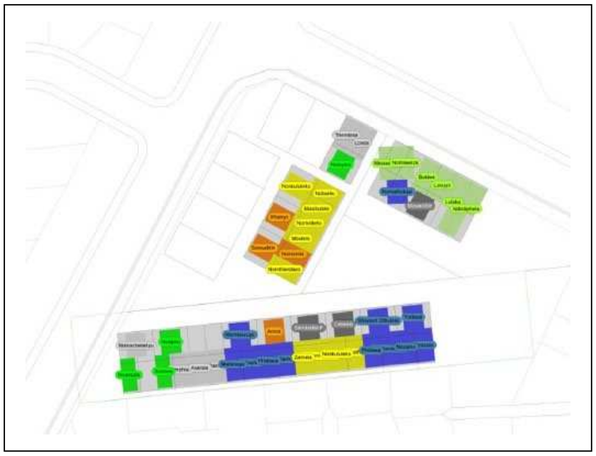


Figure 15. Building clusters reflecting the residents' neighborhood preference clustering. The color of the building shows the cluster group.

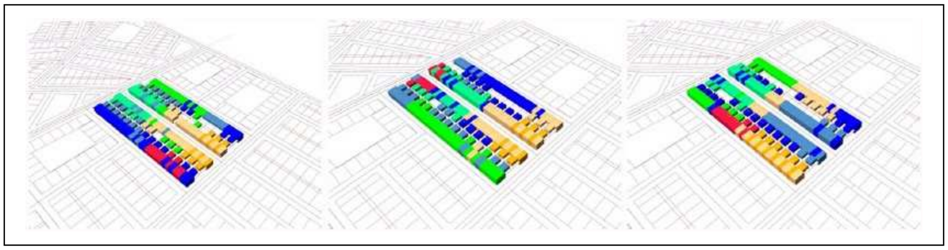


Figure 16. Alternative building clusters with different priorities of the residents for unit sizes. The color of the building indicates the cluster group.

Evaluation of pedestrian accessibility

The generated urban layouts can also be analyzed with respect to any characteristic affected mainly by its spatial configuration, of which pedestrian accessibility is one example. In contrast to a simplified buffer method (based on Euclidian distances), the distance along a street network is used as a basis for calculating the accessibility of each housing unit using a gravity-based method.[19] The value of the gravity index is proportional to the amount of accessible facilities and inversely proportional to the travel cost to reach them. In this case study, we assessed the pedestrian accessibility of educational facilities, with all of them weighted equally (i.e. attractiveness index of each facility equals 1), using metric distances to the destinations used as travel costs.

Using Open Street Map data, the configuration of the existing street network and facilities was extracted. Figure 17 shows an analysis of three generated design proposals. Moreover, quantitative indicators such as average walking distance to the facility and percentage of households with proper access to the facility were calculated.

Results and conclusion

With respect to the aims outlined in the introduction, the results presented in the case study show that we were able to translate the requirements of urban planners (UTT) into parameters for the generation of spatial

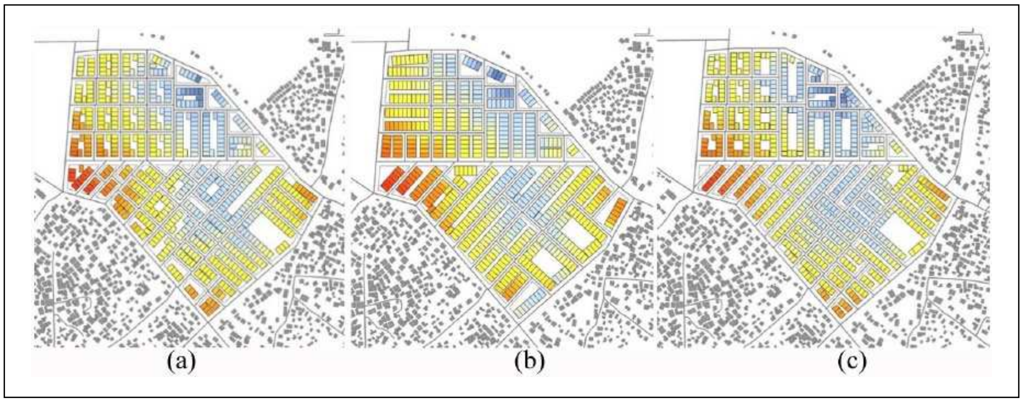


Figure 17. Street network analysis for three design proposals. The accessibility varies from cold color (dark blue) to warm color (dark red), with warmer colors representing higher accessibility of the households to the existing educational facilities. The average walking distance to the facilities for all three variations is between 812 and 818 m. Result (a) has the highest ratio (92%) for the households to have proper walkable access to the nearby facilities.

configurations and to represent these using our newly developed data structure. The results of the generative methods for the case study area made it possible to automatically generate urban layouts based on parameters such as width and length of the individual parcels, as specified by the designers. These layouts also satisfy the requirement to create densely packed housing units and to fairly allocate private and public space. Moreover, an urban designer can interactively revise the urban layout by moving street segments to achieve more satisfactory results.

Through our collaboration with UTT, we also discovered that our data representation for the parcels has some inherent limitations and did not fully address the needs of urban designers in practice. In the Empower Shack project, the planners defined basic row-house building types, which required exact plot widths, whereas the CUDP uses only threshold values for the minimum width resulting in approximate and varying plot widths. Although we were able to implement a remedy as an intermediate solution, this problem could not be solved completely.

The preference-based clustering methods of the CUDP also allowed the urban designers to better consider residents' needs for unit plot sizes and to preserve local communities by maintaining preferred neighborhood relationships in housing clusters on site. The clustering method was verified with 90%–94% success to meet the citizens' neighborhood preferences. In addition, the ability to visualize the generated results and make them accessible online facilitated early-stage decision-making discussions among urban designers, local community representatives, governmental bodies, and citizens themselves during the community workshops on site.

In our case study, the urban planners from UTT benefited from the interactive CUDP: it saved them time by obviating the need for manual design work in fitting parcels to the site and in turn improved planning efficiency. Our urban planning colleagues were satisfied with the design quality of the generated spatial configurations and found them essential for such complex urban redevelopment processes as that of the Empower Shack project. Despite the aforementioned drawbacks with respect to the precision of the generated parcel sizes, the results were regarded as being comparable with manually created designs in terms of the level of detail, geometric precision, and the spatial qualities of the case study area. Although the CUDP components we introduce in this article significantly simplify the generation of basic urban layouts from scratch, it is not yet universally applicable to all possible urban planning projects. Further customization is needed to address specific design requirements (e.g. pre-defined plot sizes) as well as the design constraints (e.g. public spaces distribution) of a particular design brief in order to ensure the generation of useful layouts that can serve as a basis for discussion and further development.

In future work, we plan to use the data structure we developed as chromosomes for EMO algorithms and to demonstrate the potential of optimization processes for urban design. For this, we also need further evaluation strategies that can be used as objective functions. It will be a challenge to explore applications of EMO in urban planning and the design of larger urban areas in close cooperation with designers. Another future perspective is that the computer may learn from the actions of urban designers as they revise generated urban layouts in order to improve the next generation of synthesized urban layouts. In addition, we see huge potential in using machine-learning methods to integrate not directly measurable quantitative criteria in the optimization process to increase the acceptance of greater automatization in urban design processes.

Acknowledgements

Additional accompanying material including videos and Grasshopper scripts are provided on this website: http://decodingspaces-toolbox.org/computational-urban-design-prototyping/

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The project was partially funded by SwissRe and partially conducted at the Future Cities Laboratory at the Singapore-ETH Center, which was established collaboratively between ETH Zurich and Singapore's National Research Foundation (FI 370074016) under its Campus for Research Excellence and Technological Enterprise program.

References

1. Reinhardt SK, Hill MD, Larus JR, et al. *The Wisconsin Wind Tunnel: virtual prototyping of parallel computers*. New York: ACM, 1993.
2. Malik SM, Lin J and Goldenberg AA. Virtual prototyping for conceptual design of a tracked mobile robot. In: *Proceedings of the Canadian conference on electrical and computer engineering*, Ottawa, ON, Canada, 7–10 May 2006, pp. 2349–2352. New York: IEEE.
3. Bringmann O, Ecker W, Gerstlauer A, et al. The next generation of virtual prototyping: ultra-fast yet accurate simulation of HW/SW systems. In: *Proceedings of the design, automation & test in Europe conference & exhibition*, Grenoble, 9–13 March 2015, pp. 1698–1707. New York: IEEE.
4. Song P, Krovi V, Kumar V, et al. Design and virtual prototyping of human-worn manipulation devices. In: *Proceedings of the design engineering technical conference*, ftp://swanson.seas.upenn.edu/pub/kumar/papers/1999/DETC99_CIE_9029.pdf (accessed 12 July 2018).
5. Wang GG. Definition and review of virtual prototyping. *J Comput Inf Sci Eng* 2002; 2: 232–236.
6. Rix J, Haas S, Teixeira J. Virtual prototyping: Virtual environments and the product design process. London, UK: Chapman & Hall, 1994.
7. Urban-Think Tank. Urban-Think Tank, http://u-tt.com/ (2016).
8. Koenig R, Treyer L, and Schmitt G. Graphical smalltalk with my optimization system for urban planning tasks. In: Stouffs R, Sariyildiz S (eds) *Computation & Performance: Proceedings of the 31st eCAADe Conference.* Delft, Netherlands: Delft University of Technology, 2013.
9. Koenig R. CPlan: An open source library for computational analysis and synthesis. In: Martens B, Wurzer G, Grasl T, et al. (eds) *Real Time Extending the Reach of Computation: Proceedings of the 33rd eCAADe Conference*. Vienna, Austria: Vienna University of Technology, 2015.

10. Koenig R and Knecht K. Comparing two evolutionary algorithm based methods for layout generation: dense packing versus subdivision. *AI EDAM* 2014; 28: 285–299.
11. Moore AW. *An intoductory tutorial on kd-trees*. Cambridge: Citeseer, 1991.
12. Russell SJ and Norvig P. *Artificial intelligence: a modern approach*. Malaysia: Pearson Education Limited, 2016.
13. Koenig R. Urban design synthesis for building layouts based on evolutionary many-criteria optimization. *Int J Archit Comput* 2015; 13: 257–269.
14. Stefanescu D. Future.Speckle: display and explore parametric models in your browser, https://github.com/didimitrie/future.speckle (2016, accessed 29 March 2018).
15. Stefanescu D. Speckle: open digital infrastructure for designing, making and operating the built environment, https://speckle.works/ (2018, accessed 29 March 2018).
16. Hansen D, Shneiderman B, Smith MA. Analyzing social media networks with NodeXL: Insights from a connected world. Burlington, USA: Morgan Kaufmann, 2010.
17. Turner A. Angular analysis. In: Peponis J, Wineman JD, Bafna S (eds) *Proceedings of the 3rd international symposium on space syntax.* Georgia, Atlanta, USA: Georgia Institute of Technology, 2001, pp. 30–31.
18. Turner A. From axial to road-centre lines: a new representation for space syntax and a new model of route choice for transport network analysis. *Environ Plann B* 2007; 34: 539–555.
19. Sevtsuk A. Location and agglomeration: the distribution of retail and food businesses in dense urban environments. *J Plan Educ Res* 2014; 34: 374–393.