MASTERARBEIT

Multivariate optimization of residential building envelopes and massing: A model proposal to balance solar building performance with design criteria

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Diplom-Ingenieurs
unter der Leitung

Georg Suter
Ao.Univ.Prof. Dipl.-Arch. Dr.phil.

E259

Institut für Architekturwissenschaften

eingereicht an der Technischen Universität Wien
Fakultät für Architektur und Raumplanung
von

Fabian Hübner
0507969

Wien, am 21.10.2017

Abstract

Während Algorithmen integraler Bestandteil in den verschiedensten Gebieten moderner Produktion geworden sind, bleibt der Arbeitsalltag von Architekten erstaunlich unberührt von dieser Entwicklung. Gleichzeitig können Umweltprobleme wie Erderwärmung, Luftverschmutzung oder eine zunehmende Ressourcenknappheit zu großen Teilen auf schlechte Planung und ineffizientes Verhalten von Gebäuden zurückgeführt werden. Um diese Probleme wirkungsvoll bekämpfen und strengere gesetzliche Auflagen in Zukunft erfüllen zu können müssen Gebäude effizienter werden, hierbei könnte algorithmische Leistungsoptimierung eine bedeutende Rolle spielen. Doch nachdem in der Architektur eine Vielzahl von Faktoren die Güte eines Gebäudes bestimmen, ist der Einsatz von Algorithmen in der Architektur ungleich schwieriger als beispielsweise im Ingenieurswesen. Einige dieser Faktoren, wie beispielsweise Ästhetik, sind zu abstrakt, um von einem Computer optimiert zu werden, was die Skepsis von Architekten gegenüber algorithmischer Optimierung erklären könnte. Um dieses Problem zu überwinden, muss der Versuch, algorithmische Optimierung im Architekturalltag zu etablieren, auf die Eigenheiten von Architektur eingehen. Aus diesem Grund wird in dieser Arbeit eine alternative Herangehensweise an die Thematik vorgeschlagen, die Unzulänglichkeiten von vielen herkömmlichen algorithmischen Optimierungsversuchen vermeidet. Im Rahmen der Arbeit wird zudem ein Tool vorgestellt, das Entwürfe entsprechend mehrerer solarer Kriterien optimieren kann, das aber nicht zwangsläufig nach der besten Gebäudeperformance sucht, sondern darauf ausgerichtet ist, das allumfassend beste Ergebnis im Einklang mit dem Architekten zu liefern, das alle Entwurfskriterien berücksichtigt. Die Anwendbarkeit des Tools im tatsächlichen Arbeitsalltag von Architekten steht dabei im Vordergrund. Um das Tool zu testen, wird es auf drei Fallbeispiele angewandt mit dem Ziel, ein Ergebnis zu erhalten, das die Qualität abstrakter, nicht optimierbarer Kriterien zumindest nicht mindert, die Effektivität des Gebäudes jedoch steigert.

Schlagwörter: Algorithmische Optimierung, Gebäudehülle von Wohngebäuden, solare Gebäudeperformance, Generativer Entwurf, Entwurfsalternativen

Abstract

While algorithms have become an essential tool for optimization in various fields of modern industries, the everyday architecture routine remains surprisingly untouched by that development. At the same time, environmental problems such as global warming, air pollution or an increasing scarcity of resources can be traced back to poor and inefficient building performance. In order to successfully fight those problems and to meet stricter law regularities in future, buildings have to become more efficient, algorithmic performance optimization could be a promising tool in that process. Yet, optimizing architecture is a challenging enterprise, as, in contrast to most cases in engineering, various aspects have to be considered for architectural design. Some of them, such as aesthetics for example, are too abstract for optimization by a machine, which may explain the scepticism of architects towards algorithmic design. In order to overcome this problem, an approach for successful implementation of algorithmic optimization in architecture has to respect the characteristics of architecture. Therefore, in this thesis a defect of many common optimization algorithms for architectural application is detected and a more suitable alternative is proposed. A tool will be presented, which is optimizing design according to multiple passive solar criteria, but which is not necessarily aiming at the most efficient design solution, but at an overarching best solution respecting all relevant design criteria. Stress is put to a high practicability of the tool for the usual working routine in architecture businesses. In order to test the tool, it will be applied on three different projects with the goal to come up with a result that is not touching the quality of abstract design criteria, but which is superior in its optimized performance.

Key words: algorithmic optimization, residential building envelope, solar building performance, generative design, design options

Table of Contents

1 Introduction

1.1 Optimization in architecture

It was in the age of Renaissance, when architecture started to focus on "effectiveness", when scientific approaches and mathematical calculations step by step replaced the method of *trial and error*, which had been state of the art for centuries before. Until then, the collapse of a building during its construction indicated, that its statics was insufficient and that the bearing structure had to be reinforced for the next attempt. Studying physical phenomena and thus being able to make proper predictions about the performance of a building not only improved the quality of architecture, but also distinguishes architecture from other art categories (Ji, 2012). Since then, optimization of structure, layout, materials, etc. has been an integral part of architecture with its climax in the modernists' denial of anything ornamental in design.

With the rise of the computer in the middle of the 20th century, the potential for optimization in all fields of engineering was pushed to a new level, as with the increasing performance of hardware more and more complex calculations could be solved. Today one of the omnipresent topics of our time is the gathering of so called *big data* and its optimization by algorithms (Mayer-Schönberger & Cukier, 2013). Despite some scepticism towards that development, we have to admit, that algorithms made many things in our everyday life to some extent simpler, cheaper or smarter.

However, the architect's working routine remains surprisingly untouched by algorithmic influence. In contrast to the related field of engineering, where algorithms have been helping to make work more efficient, architects still seem to refuse to do the same (see Figure 1).

Today, architecture or the building sector, respectively, is one of the main reasons, why a huge amount of solar energy remains unused and why we are facing problems like climate change or shortage of non-renewable resources (Kanters, 2011; Horvat, Dubois, Snow, & Wall, 2011). Especially through an on-going worldwide urbanisation and an increasing density in the cities, energy efficient planning has become more complex and difficult, but remains one of the most important tasks for urban planners and architects (Van, Miyamoto, & De Troyer, 2014). Algorithms have the potential to deal with and solve such complex problems. Architects shouldn't waste more time leaving this potential unused.

So finding an approach, how algorithmic optimization could be integrated into the architect's design process will be a goal of this thesis.

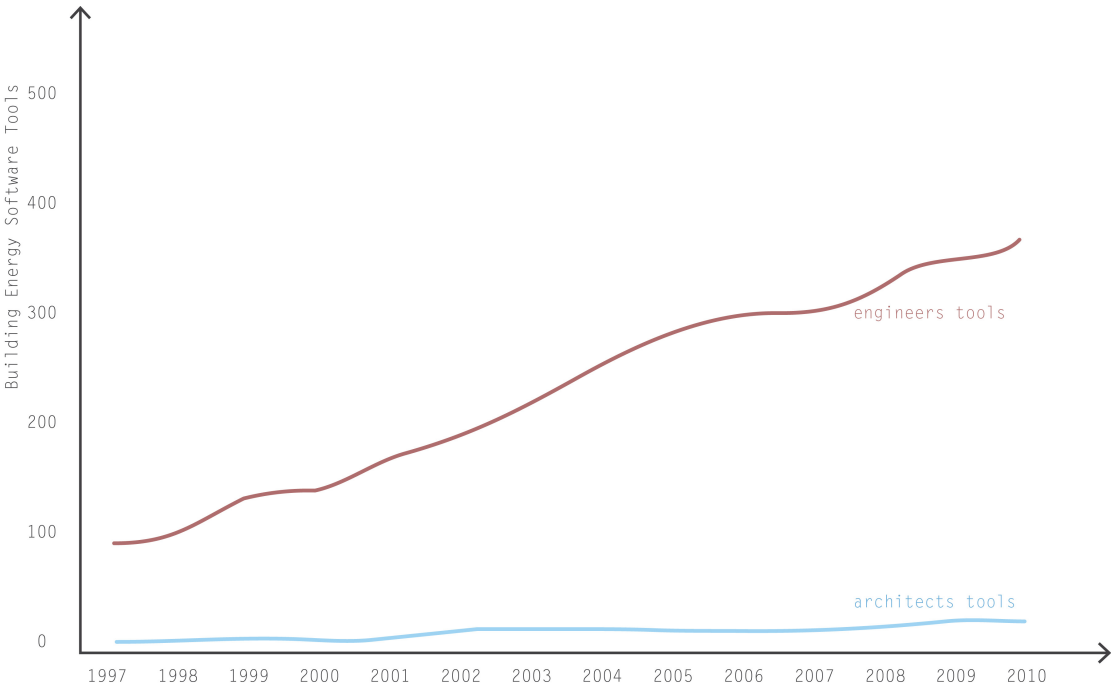


Figure 1: Building Energy software tools development (Horvat, Dubois, Snow, & Wall, 2011)

1.2 Problem statement

For the past decades, and especially in the past couple of years, the notion of architecture done by algorithms has become increasingly popular amongst scientists. From form finding to rapid prototyping, the whole design process, that's the belief, could be done by the computer and machines. From the architects point of view, the best-case scenario would include the architect as a general supervisor of the design process, the worst case scenario would make his job redundant. Most practising architects are watching this development with scepticism, but not only because they are afraid of losing their jobs, but also because there is a serious problem with algorithms in architecture, which shall be demonstrated now.

1.2.1 Design criteria in architecture

Architects' design approaches usually decompose a design problem into sub-problems or individual components, respectively, which are easier to solve separately. As soon as their solution is achieved, the components can be assembled to one design solution (Mahmoodi, 2001). Those components can be of two different kinds: tangible criteria, and subjective criteria (Elezkurtaj &

Franck, 2002). Tangible ones are objective, measurable quantities, which can be expressed in numbers and units.

Examples for tangible criteria are:

- Natural and artificial light
- Views
- Accessibility
- Distance matrix
- Ratio of circulation area/usable area
- Ratio of volume/usable area
- Dimensions of rooms
- Noise control
- Cost

Subjective, unmeasurable criteria can be almost any criteria that influence the design or inspire the designer, such as:

- Urban environment
- Sociological factors
- Cultural factors
- Political factors
- Historic factors
- Psychological factors
- Aesthetics
- Geometric composition
- Shape

Obviously, the differentiation, if criteria are tangible or subjective is not always clear. In the case of views e.g. one can just measure the volume of the view cone and analyse its quantity (tangible), but one could also focus on the aesthetic value of a view, its quality (subjective) (e.g. for hotels, rooms with sea views are more valuable than others, even if they have the same volume of the view cone)

The ability to express phenomena in measurable and quantified terms has the advantage that they can be put into numbers and units, they can be tabulated and analysed, a precondition for an algorithm to optimize them (Mayer-Schönberger & Cukier, 2013). Vice versa, a computer cannot (yet?) deal with abstract terminology like aesthetics, as there is no way to translate aesthetics into numbers. Louis Kahn once put it like this: “What is unmeasurable is the psychic spirit. The psyche is expressed by feeling, and also by thought, and I believe it will always remain unmeasurable” (Stöckli, 1992). This constitutes the

problem in the notion, that computer could substitute the designer, because “the design of buildings becomes an art precisely through dealing with the subjective needs and wants of the users and beholders. Our sensitivity to architectural qualities goes further than our ability to describe the needs and wants in explicit terms” (Elezkurtaj & Franck, 2002).

Subjective aspects distinguish architecture from engineering, rational aspects distinguish it from arts (Ji, 2012), that’s what makes architecture unique. But despite the understandable scepticism of architects towards algorithms, the latter can, if they are used as an attendant design tool properly, help the architect to bring the design to a maximum of performance (Glassman & Reinhart, 2013) without losing the important subjective design elements. But how to make sure that subjective design elements remain untouched while the algorithm is adjusting the design?

In order to assure this, algorithmic optimization has to be applied very carefully. Not every project may be suitable for being optimized, and when optimization is used, the timing of its application is crucial. So what is a proper way for using algorithmic optimization and when can it be used?

1.2.2 The design process

Research shows that early design stages have the greatest potential for optimization, as the massing and urban settings are usually the first things that are designed and they are hard to change in later design stages. Nevertheless they have the largest impact on the building’s energy consumption and efficiency (Horvat & Wall, 2012). But as mentioned before, the computer is unable to respect all design criteria, that’s why it is crucial that all important design constraints are set by human designers before. Hence, for a holistic design, optimization cannot happen at the very first step. Expressed in design stages used in Figure 2, in the *Pre-Conceptual Stage*, the designer is gathering information about the site and setting some meta goals, that the design is supposed to fulfil. When proceeding to the *Conceptual Stage*, the designer will come up with some first ideas concerning the design. After having a clear vision about which path the design should follow and what are the crucial design elements that are characterizing that path, the designer will review their constraints and then has to decide, whether there are variables that could be optimized without decreasing the quality of his already set constraints. In many projects, there may be no variables that could be manipulated by the algorithm because everything is already defined by the specific circumstances of the project and the design idea, so the design process has to continue without algorithmic optimization in a conventional way. But in case optimization can be executed, the main optimization like massing and urban setting should be done in the conceptual stage, minor optimizations regarding façade optimization can stretch until the *Detailed Design Stage*, following a *from large scale to small*

scale logic. Though it is important to stress that the use of optimization tools in late design stages is less efficient (Yezioro, 2009).

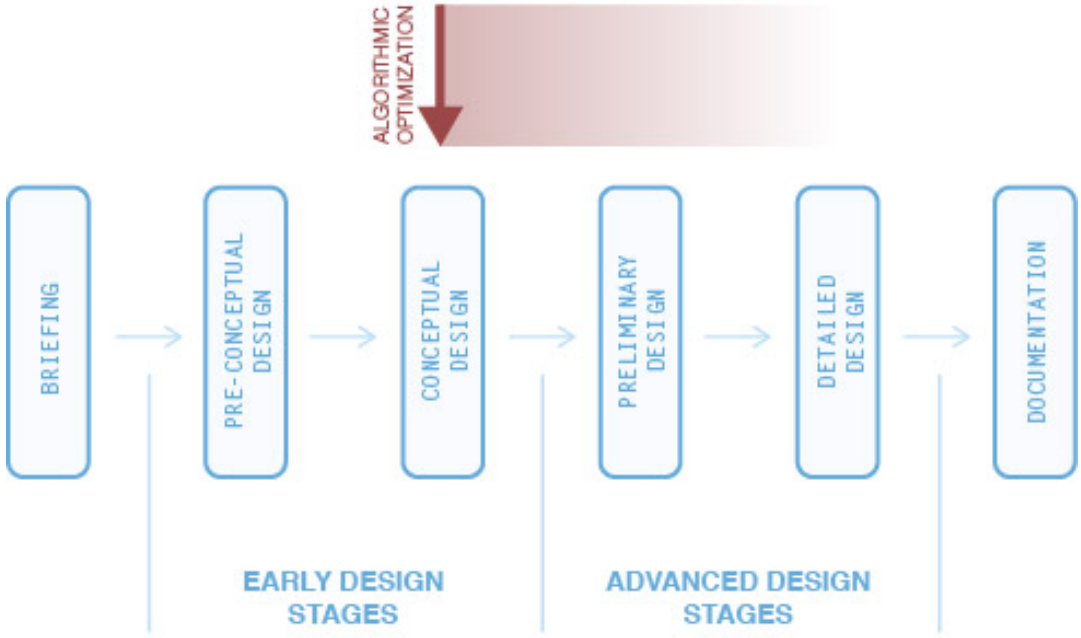


Figure 2: The position of algorithmic optimization in the design process

1.2.3 Reasonable optimization

To be more precise about what is meant by *reasonable optimization* and to show which kind of projects have a high potential for optimization, the process will be explained with the help of a residential project done by *ppag architects* in Vienna (see Figure 3).

The architects' design consists of blocks which seem randomly spread across the site. There is no factor like solar orientation or the urban environment that makes the configuration of the blocks plausible. All of the surrounding buildings show that their orientation is deduced from the urban setting, the master plan of this area. That is obviously not the case for the *ppag* design. As there is no deduction from the environment of the site, there are infinitely more different solutions to place the blocks than the one suggested by *ppag*. The design in Figure 4 is not done by *ppag*, it is just another possible solution of the same design idea, but we could not say at the first look which one is better or worse, actually it even takes a second look to notice that they are different at all. We could now generate thousands more of those very similar layouts, but we are making the decision to choose the best one even harder. Still, all the different versions would fulfil the architects' design ideas of the previous design stage,

which were probably defining how many buildings they want to put on the site, their density, their massing, their maximum heights and depths and most characteristic, the scattered distribution on the site. As all of those criteria can remain untouched while generating design options, the quality and the architects' design ideas will not be diminished by an algorithm.

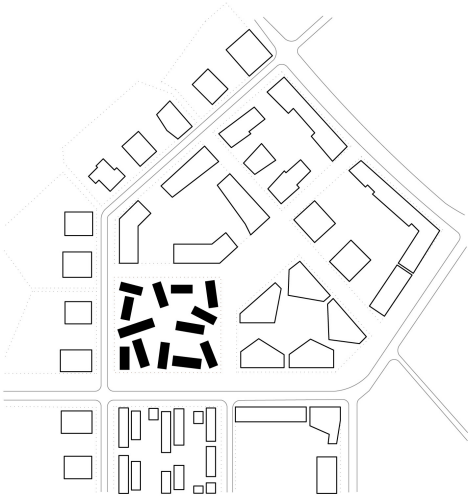


Figure 3: Site plan Slim City, Vienna (ppag architects, 2015)

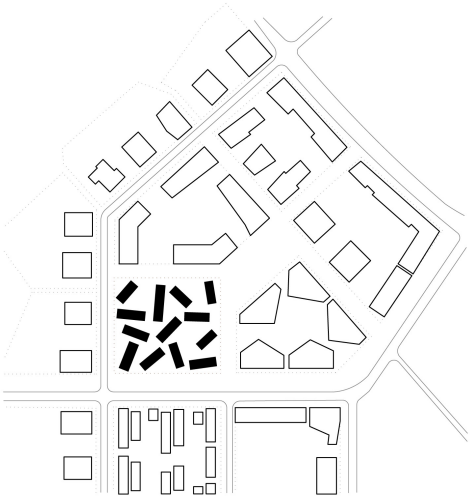


Figure 4: Site plan of design option

Moreover, by adding additional design goals like energy efficiency or in the *ppag* case optimized view sheds, the decision, which version is the best one, is drastically simplified, as we can choose the one providing the best views. But at the same time the complexity of the problem will increase enormously, as a change of a position of just one building will influence the view shed performance of the whole site. While the humans' capacity of dealing with such complex situations is strongly limited it appears to be a perfect occasion to use algorithm's qualities which are producing a high amount of data at a very high complexity. The algorithm can be used as a design tool of a new dimension, as long as it is assured that the designer never loses control over the design process (Chien & Flemming, 2002).

Put together, this thesis is founded on the problem that on the one hand, algorithmic optimization is very hard to apply reasonably on architecture, as various criteria (measurable and unmeasurable) define its quality, but on the other hand, by not using algorithmic optimization, a big potential for a more performance oriented and hence efficient way of building remains unused.

1.3 Research aim

The described example shows that algorithmic optimization, if applied reasonably, is not only able to increase the performance of design. It can actually add more freedom to the designer's creativity, as without optimizing their design, the architects would have problems justifying, why they chose the option they chose out of a spectrum of unlimitedly more equal solutions. In this example, the neighbouring projects can easily justify the orientations of their buildings by the underlying master plan or the compass, whereas none of those justifications work for the *ppag* design. In more general words, decisions taken by architects, which used to be dependent on external factors like urban settings e.g. can now be extended by factors provided by whatever the algorithm is optimizing for. Algorithms can increase creative freedom of designers.

On top of that, in their approach published 2002, Elezkurtaj & Franck stressed the interesting notion that algorithms could actually increase not only the performance of a building, but also support the creative process of the designer by coming up with unexpected results.

> *Being able to surprise means to be creative. Architecture is an art in that it offers the opportunity of satisfying more needs and wants than the users and beholders were conscious of having.*
> (Elezkurtaj & Franck, 2002)

The aim of the thesis is therefore, to balance different point of views on architectural optimization. On the one hand the engineer's view, which is only looking for the most efficient solution and on the other hand the artist's view, which is mostly focusing on aesthetics. Balanced in the right way, algorithmic optimization is able to achieve more than just a compromise of the two. It has the potential to combine the strengths of both to an overarching, superior design.

Put together, generic design and optimization can bring four big improvements to the design process:

1. They can create unexpected and surprising results
2. They can increase the building quality
3. They can help designers justify new ways of designing
4. They can save resources!

The goal is to propose a way, how algorithmic optimization could be successfully implemented in the architect's working routine. Therefore, a tool should be established that could be used in early design stages in order to optimize residential massing and the building envelope. The tool is supposed to deliver design options to the initial design idea, which do not touch the quality of

abstract design criteria, but which are superior in their optimized performance. As in early design stages many design decisions have not been made, yet, the goal is to find a way how to overcome this lack of information and to indicate the specific potential of each design option, nonetheless. Moreover, the tool should be able to cover different levels of detail, so after optimizing the massing, the best massing result could be further optimized regarding its façade, for instance.

Although the inclusion of more functions and a broader scope of applications would be desirable for an encompassing tool, the framework of this thesis requires a restriction to the before motioned goals, for the following reasons:

Maximizing energy efficiency of buildings is one of the most researched topics in recent history of architecture. There are numerous studies and approaches how energy efficiency could be measured. But still, calculating the actual overall energy efficiency is a highly complex and complicated topic, even for already built houses standardized calculation methods sometimes come up with very different efficiency results (Weeber, Sahner, & Bosch-Lewandowski, 2007). If we also take changes in climate conditions or changing environments into account (Glassman & Reinhart, 2013), probably there is no such thing as an absolute number for energy efficiency, especially when we talk about simulation of unbuilt houses.

However, for a very accurate simulation result many different factors need to be considered, such as the environment, solar radiation, wind flow, materials, window openings, floor plan layout, user behaviour and many more. Respecting all of these factors in this thesis is impossible for two reasons:

1. Most of the mentioned factors are highly complex and in strong dependence on each other. Including all of them would explode the framework of this thesis
2. As the thesis focuses on an implementation of optimization tools in early design stages, factors like materials or window openings may not be defined yet

Hence, this thesis aims to focus on solar factors mainly, such as *radiation, light incidence* and shading, as they are of fundamental importance in residential projects. In order to test multi-criteria optimization, *views* are also included. The result of optimization will not be sufficient for a very accurate prediction about energy efficiency, but it will show a rough direction, which constellation has the best potential to be energy efficient. The focus of the tool is put to early design stages, which include the definition of massing and in an increased level of detail optimization of rough façade design, such as jutties, recesses or window openings. Further optimization steps in an advanced design stage with an even more increased level of detail, better defined design components and hence, a more precise prediction about the actual performance of the design will not be dealt with in this thesis, although for a holistic design optimization process those advanced steps would be the consequential continuation of the proposed tool.

Moreover, this thesis is limited on residential buildings in high-density areas, as this building typology is largely depended on access to sunlight and views. Thus the potential of optimizing residential buildings according to solar factors is high and because of increasing heating and cooling expenses in the residential sector quite relevant. For other building typologies such as commercial, cultural and industrial buildings, the significance of solar factors is less and depending on the specific project very diverse. In that case, other aspects can be more relevant.

In low-density areas, optimization might be obsolete, as the design is mostly independent from the neighbourhood and guidelines for energy efficient building can be applied easily. In contrast, algorithmic optimization in high-density areas is especially useful as the building task is very complex because of various factors influencing each other. For that reason, this thesis focusses on high density only.

1.4 Methodology

After defining basics and terminology according to topic related scientific work, the thesis will come up with a suggested approach, how to put the findings of the research part into practice. The outcome will be a tool called *Building Performance Evaluation Tool* (BPET), which will be applied on three case studies in order to test its practicability and to figure out its potentials and problems. That optimization approach will be mainly executed in the *Rhinoceros* software by *McNeel* using the *Grasshopper* plugin. The following four steps will represent the framework of that approach:

1. Creation of a fully parametric model that incorporates the designers' ideas
2. Applying appropriate analysis tools to the model (such as solar radiation, light incidence, shaded areas or view sheds)
3. Exploring and classifying the design space with a high number of design options and its corresponding evaluations
4. Visualisation of the results and choosing the most appropriate ones

2 Literature review

2.1 Requirements of design optimization frameworks

Optimization in a scientific sense can be described as the “technique for finding a maximum or minimum value of a function of several variables subject to a set of constraints” (dictionary.com, 2017). The research on and the development of optimization processes reaches back to the *Age of Enlightenment*, when Isaac Newton came up with some first calculations on optimization problems. Though, the study on optimization is initially a mathematical discipline much older than computer science (Cassel, 2013), in this thesis the term *optimization* will mostly refer to computer-aided optimization. The following paragraph will provide a quick overview about requirements for optimization.

1. Computer

   For computer-aided optimization a computer is needed. That may seem tautological and redundant, but as optimization in architecture is a highly complex business, the performance of the computer as well as the software used can be a critical factor, especially for its application in architectural practice. The speed of an optimization process may determine its success or failure (Elezkurtaj & Franck, 2002). Moreover, in order to use a computer, the computer needs to be able to “understand” principles of architectural design. That requires a translation of the way the human mind grasps and designs architecture into a language the computer can deal with.

2. Quantifiable Object

   As mentioned in chapter 1.2.1 there are factors that are suitable for optimization and factors that are not. For example, it is impossible to optimize the aesthetic value of an art piece, as aesthetics depend on subjective taste. In order to optimize architecture, we need to provide objects that can be parameterized.

3. Parametric Design

   Parametric design can be defined as “coded relationships between objects” (Aish & Woodbury, 2005). For coding, software is needed that is able to translate the code (text based or node based) into architectural representations. Relations between objects can be expressed in variables and constraints, where variables are relations expressed in numbers that can be manipulated whereas constraints are fixed relations that cannot

be changed and that are limiting the possible output (Salim & Burry, 2010). There are two cases that have to be distinguished.

4. Singlevariate vs. Multivariate optimization

   If there is only one variable in the design optimization process it can be called singlevariate. In that case of simple computation, reaching the max or min of output, respectively, is the optimization goal that can be clearly defined. For example, if we want to have a high amount of daylight in our design, the maximum window size will provide maximum daylight.

   If there are more than one variable, the computation gets more complex and there may be no clear defined goal any more, as there are "multiple forms of optimality" (Zeleny, 1998). In the same example, if we still want to have a high amount of daylight, but at the same time we may want to keep the energy demand for cooling as low as possible in summer, which requires small window openings, we will have two conflicting variables. The more we maximize the one, the less the value for the other will get. In that case there is no best solution, only a so-called *Pareto front* can be found, a situation, in which no improvement for one criterion can be found without diminishing another criterion's quality (Ciftcioglu & Bittermann, 2008). In this thesis such multivariate optimization will be applied, as almost all architectural problems are multivariate. Therefore, a successful tool for optimization has to find a way to deal with that.

5. Algorithms

   When variables and constraints have been defined, the computer can start the optimization process accordingly. Changing the variables and calculating the corresponding output over and over again, a huge amount of design options can be created. That procedure is what is called an algorithmic calculation. An algorithm can be generally defined as

   > *a finite procedure, written in a fixed symbolic vocabulary, governed by precise instructions, moving in discrete steps, 1,2,3,..., whose execution requires no insight, cleverness, intuition, intelligence or perspicuity, and that, sooner or later, comes to an end* (Berlinski, 1999)

   Algorithms allow us to deal with highly complex problems and a huge amount of data. Once the process is finished, the question is what to do with that enormous amount of output.

6. Design Space Exploration

   The design space is a collection of all the solutions calculated during the optimization process (Chien & Flemming, 2002). In many cases, that can be several thousands or even up to several millions of outputs. In order for the designer to being able to navigate, evaluate and choose design

options, a convenient way of navigation through and visualisation of the options has to be provided.

In the following chapters, basic knowledge about these requirements will be provided as well as a discussion about how they will be dealt with in the presented tool.

2.2 Notes on parametric design

In order to enable an algorithm to generate design options that respect all the important constraints set by the designer beforehand, the design constraints have to be translated into a language a computer can "understand". As the human mind tends to think and design in objects like walls, roofs, neighbourhoods etc. rather than in figures and algebraic formulas, the trend to do the latter is comparably new to architecture and began to be increasingly popular with the computer becoming the main designing tool of architects. Representing architecture in terms of numbers and relations instead of objects is what we call *Parametric Design*. Parametric representation can be called a precondition for algorithms to optimize design.

Many of today's common CAD applications provide tools for coded input, though there are two major types that have to be distinguished:

1. BIM software, that mostly allows to create and manipulate library elements via code, which can be picked by the user and inserted in the design. In the BIM case, the model itself is not parametric, but many of the elements used in the model are. Though, the code for those elements is mostly not provided by the user but by the developer, the user's interaction is based on manipulating the elements' variables only. BIM has been getting increasingly popular and accepted in the everyday architecture business, as data exchange between the various parties involved in the design process (such as structural engineers or contractors) is very convenient. Common examples of BIM software are *Autodesk's Revit* or *Graphisoft's Archicad* (Salim & Burry, 2010).

2. Fully parametrical tools that are based on *associative-geometry* use parametric relations between elements like points, curves, surfaces and solids to express design. In that case, the users write the code themselves mostly in a node based way. The two main products offering this approach are *Bentley's GenerativeComponents* and *McNeels* plugin *Grasshopper* for *Rhino* (Salim & Burry, 2010).

As BIM does not offer proper ways for the designer to create a parametric model individually, the use of the term *parametric design* will not include BIM in this thesis. However in future, integration of a proper parametric design tool as well as the possibility of optimization in BIM software is desirable and even quite likely. *Graphisoft* recently started a collaboration with *McNeel* to integrate *Rhino* and *Grasshopper* into *Archicad,* for example (graphisoft.com, 2017).

As mentioned, in parametric design, the designer has to develop relationships between the ways objects connect. The result is a script (in written code or visual nodes) that allows the designer to easily change and edit parameters and obtain design options in a quick and convenient way (Woodbury, Gün, Peters, & Sheikholeslami, 2010). Parametric design requires designers to learn how to express their design ideas in a relation based way and it increases the complexity of both the representation as well as the design interface. It requires knowledge about graph and node compilations as well as an advanced understanding of algebraic and geometrical rules (Aish & Woodbury, 2005).

But the advantages of the described efforts of parameterization can be very rewarding, as the ability to adapt and change the design very quickly and easily enables the designer to test various alternative design options and choose the best one fitting the specific task and context. On a meta-level, the act of expressing design ideas in code improves the designer's contextual understanding, which reaches beyond visual or graphical representation and thus increases the designer's creativity (Salim & Burry, 2010). Furthermore, parametric design can be used for form finding processes and it is able to deal with very complex geometries, which could not be handled in conventional design approaches (Aish & Woodbury, 2005). Combined with rapid prototyping and cnc machining, it is capable to produce an almost unlimited amount of individually prefabricated parts, which can realize the most complex geometries when assembled. The latter fact constitutes some kind of a revolution in architecture, as without parameterization, the realization of buildings has always been dependent on components that could be fabricated in mass production processes. In other words, parametric design freed architecture from its dependency on mass production, which makes Schuhmacher, one of the leading practitioners of parametric design, call out not only a "post-Fordism era", but even a new style of architecture, which is putting an end to "a series of relatively short-lived architectural episodes that included Postmodernism, Deconstructivism and Minimalism" (Schumacher, 2009).

Undoubtedly, the curved and futuristic design that Schuhmacher is postulating and praising as "the elegance of ordered complexity and the sense of seamless fluidity " (Schumacher, 2009) has become a successful trend in the past decade of architecture and those shapes are what many people believe parametric design is solely about.

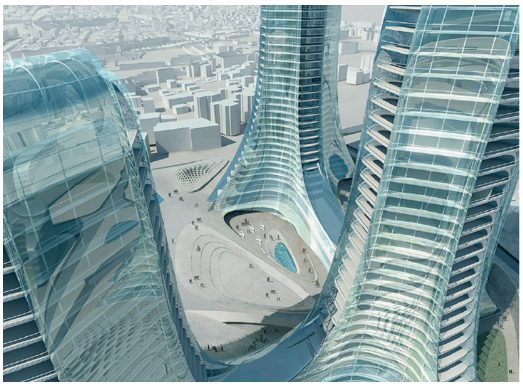


Figure 5: Parametric design by Zaha Hadid architects (Schumacher, 2009)

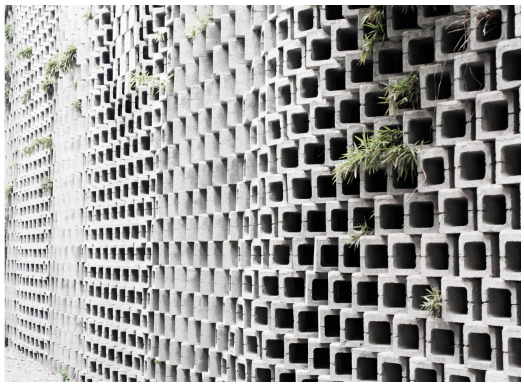


Figure 6: Parametric design by Archiunion

But the notion, that design can actually be generated by data is promising as well as problematic at the same time. It is promising, because design becomes independent from the designer's taste and intuition, it follows the rules of mathematics, statistics and nature instead of arguable subjective opinions. But as mentioned in the introduction, many experts agree, that the attempt to automate design completely is predestined to failure, as the computer is able to deal with huge amounts of data, but is unable to grasp abstract phenomena like aesthetics or psychology. Architecture solely done by machines will be missing some important components (Kourkoutas, 2007), at least at the current state of computer development. Thus, a division of the design work between the human being and the computer is regarded as the most promising way of designing today (Elezkurtaj & Franck, 2002).

The term parametric design has many facets. The eye catching futuristic designs à la Schuhmacher (see Figure 5) are contained as well as more subtle aesthetic approaches (see Figure 6), but also very rational, engineering based branches are part of it. In the end, any kind of project can be expressed in parametric terms, even the most regular and straightforward projects such as done by some radical modernists. Parametric design must not be equated with curvy futuristic shapes. For a post-parametric era, Krause, Derix & Gamlesaeter predict a building process, in which parameters not only define the shape of some geometry but the whole process by a vivid exchange of knowledge and data between all stakeholders. All parts of the building process are interconnected and influence each other, an approach quite similar to where BIM is aiming at (Krause, Derix, & Gamlesaeter, 2011).

For this thesis though, parameterization is necessary to allow an algorithm to optimize design ideas already set by the designer beforehand, no matter if the design uses a very expressive language or just some simple orthogonal geometries. It will not be used to generate shapes from scratch in order to avoid design lacking important non-tangible components.

2.3 Algorithmic design optimization in architecture

> *It is no exaggeration to say that optimization is everywhere, from engineering design to business planning and from the routing of the Internet to holiday planning. In almost all these activities, we are trying to achieve certain objectives or to optimize something such as profit, quality and time.*
>
> (Yang, 2010)

The attempt of architects and engineers to optimize buildings reaches back to the 1950s and was mainly focussing on optimizing floor plan layouts. A successful approach was the so-called *System Layout Planning* (SPL), a procedural method to optimize mostly plant and production facilities layouts, with the aim to minimize ways and costs. Until the 1970s, the field of layout optimization remained an engineers' discipline only, the first architectural research on that topic was done in 1972 by Mitchell and Dillon (Lobos & Donath, 2010).

In the beginnings of algorithmic approaches, in order to obtain a solution, constraints and objectives had to be drastically simplified. Due to that simplification, the solutions obtained were mostly inappropriate and required further manual modification to fulfil their purposes (Yang, 2010).

Examples are, amongst others, the *Design Problem Solver* by Pfefferkorn and a bit later the *SEED* software developed in Carnegie Mellon. The *Design Problem Solver* focussed on the layout of furniture and equipment, but was meant to be characteristic for similar applications like architecture or urban planning. It was restricted to 2-dimensional use, it was slow and limited in scope, but still it was able to deal with multi-criteria and constraints like orientation, position, or view (Pfefferkorn, 1972).

Based on Flemming's *LOOS*, the *SEED* software was able to generate rectangular floor plan layouts containing multiple constraints such as privacy, natural lighting and accessibility. The fact that SEED was able to deal with a higher complexity, provide a higher level of interactivity and even include circulation spaces made it a good basis for further software development, which should be finally put into practice in real projects, so much for the idea (Liggett, 2000).

In 2002, Elezkurtaj and Franck presented an approach to optimize floor plan layouts aided by Evolution Strategy (ES) and Genetic Algorithm (GA) (the terms will be explained later in this chapter), where the user can interactively view and select design options suggested by the system (see Figure 7). The authors stress the importance of interaction with the designer, as they believe that aspects like aesthetics cannot be dealt with by the computer, only by the designer. An important factor for successful interaction is a very fast calculation, that is why the algorithms especially focus on speed. Another interesting notion is the inclusion of randomness aiming for surprising results. Things that one

cannot foresee are the essence of creative processes like architecture, they claim (Elezkurtaj & Franck, 2002). By doing so, they clearly distinguish themselves from the optimization approaches mainly done by engineers, whose only goal was to reach the very best quantitative result.

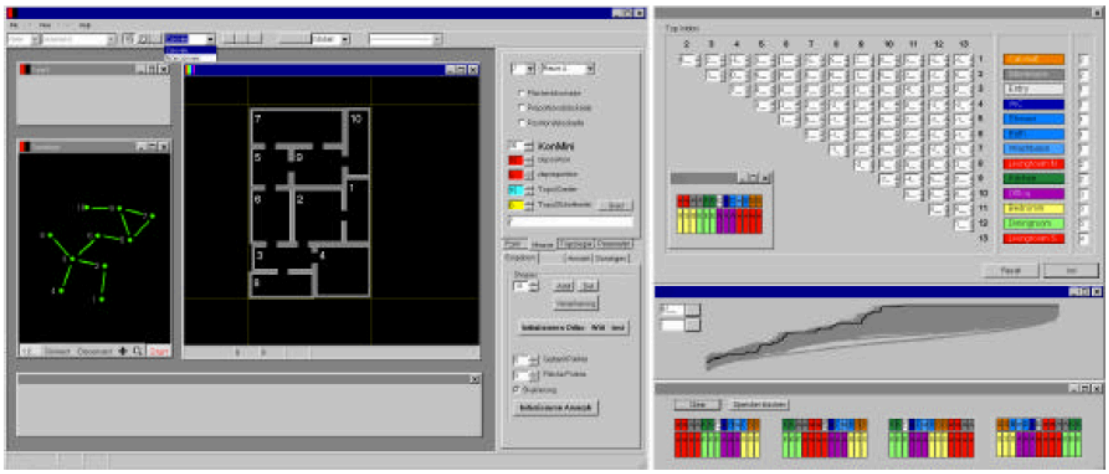


Figure 7: System Supporting Floor Plan Design (Elezkurtaj & Franck, 2002)

But despite over 60 years of effort in developing proper optimization approaches, in today's architectural practice algorithmic optimization virtually plays almost no role. Though, there have been two attempts to integrate automated optimization in commercial CAD applications for architects. One was *Alberti*, published in 1998 by *AcadGraph*, which was working with architectural inputs like room names and stories and the relation between them. It resulted in hundreds of suggested floor plan variations, from which the designer could choose the best fitting one, but still in most of the cases there was a manual modification necessary. In the end, the commercial success and the acceptance of the product failed, so its distribution was finally stopped. The other attempt was done by the more famous company *Nemetschek*, which in 2004 included a so called *Space Planning Tool* in one of their versions of *Vectorworks*. Similar to *Alberti*, rooms, names and sizes had to be defined, the software then suggested layout options that were generated automatically. Already in the next version of *Vectorworks*, the *Space Planning Tool* was dismissed again (Lobos & Donath, 2010).

Today we can state, that the inclusion of algorithmic optimization has failed so far. Lobos & Donath suggest some possible explanations for that phenomenon, which can be added by Ligget's explanations uttered in 2000, but which are still valid today:

- The interface of the programs is not user friendly enough
- There is a lack of support for an iterative design process (Liggett, 2000)
- There is a lack of knowledge about the architectural design process amongst the developers
- Architects are not used to the way of parametric thinking that is required for optimization (Lobos & Donath, 2010)

The unimportance of scripting features in CAD applications was also demonstrated by a study investigating preferences of architects concerning their software, as shown in Figure 8. The question asked was “What are the 3 factors that most influence the choice of software you use”, scripting features were ranked last with less than 1% of approval (Horvat, Dubois, Snow, & Wall, 2011). But especially with computers getting more and more powerful, parametric applications that are knocking at architecture offices’ doors and a young generation of architects who knows how to make use of those applications, the inclusion of algorithmic optimization might soon become a common design tool.

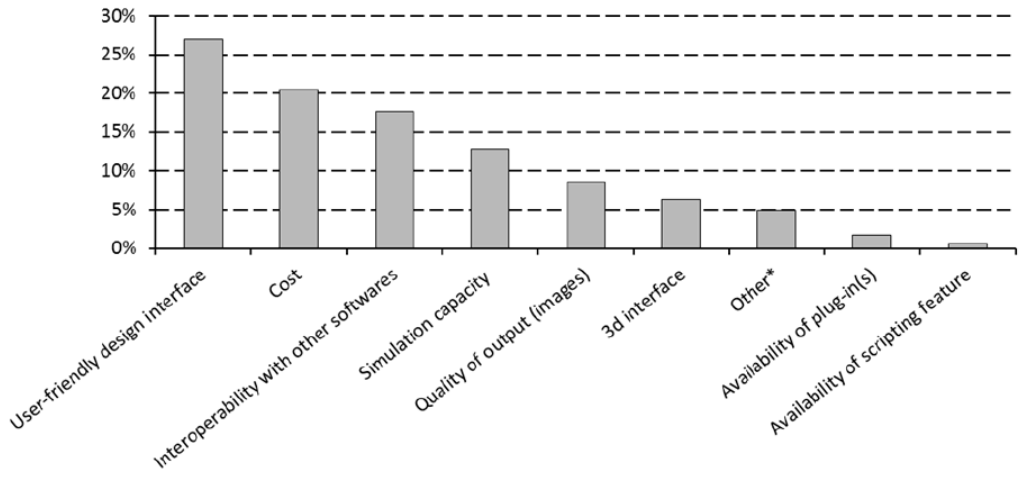


Figure 8: Most influential factors for software choice of architects (Horvat, Dubois, Snow, & Wall, 2011)

2.3.1 Metaheuristic generation of design options

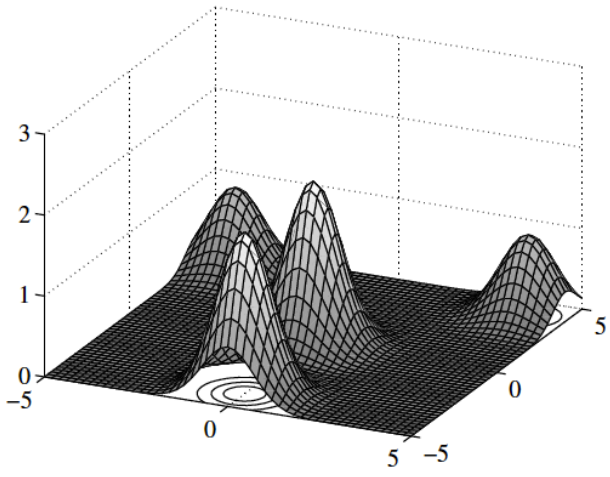


Figure 9: 2-dimensional phase space (Yang, 2010)

Solution finding by the *trial and error* method has been a successful approach ever since mankind has been trying to solve problems. Since World War II, people began to use algorithms in order to solve problems of high complexity, which is today known as *metaheuristics*. Inspired by nature, many different kinds of algorithmic approaches have been developed so far (Yang, 2010).

All of the optimization algorithms have in common, that they try to search for a defined maximum within a *phase space* (the collection of all possible solutions) (Rutten, 2013). A phase space of two variables can be displayed as shown in Figure 9. The two variables are represented by the x and y axis, the corresponding result is displayed in the third dimension. It is a simple version of a phase space that can be displayed in a so called fitness landscape. As the third dimension is used for the level of fitness, only two-dimensional problems can be displayed that way. Many optimization problems are way more complex and have way more variables than just two, so their fitness landscape is abstract and inappropriate for being displayed, as their dimension of complexity exceeds the dimensions available in space (x,y,z axis). In the shown example, the phase space has one global maximum and three local maxima. Algorithms differ in their strategy, how to find the global maximum, and how to avoid getting stuck in one of the local maxima. Generally we can categorize algorithms in

- *deterministic* (no use of randomness) vs. *stochastic* (use of randomness)
  randomness can help to avoid getting stuck in local maxima, most of the currently successful algorithms use randomness for that reason

- *non-gradient* (no use of derivative of the formula) vs. *gradient-based*
  gradient-based algorithms work well for smooth unimodal problems, non-gradient algorithms are to prefer when there is discontinuity in the objective function

- *trajectory-based* (only one search agent) vs. *population-based* (many search agents) (Yang, 2010)

For comparably easy calculations of single-variate problems, gradient based algorithms without randomness work quite well, but for more complex cases such as multivariate optimization, non-gradient based algorithms using stochastic approaches are to prefer (Ciftcioglu & Bittermann, 2008).

As metaheuristics are, as the name reveals, heuristic, there is no guarantee for a successful solution. In many cases, they deliver quite satisfying results, though the iterative *trial and error* approach mostly does not succeed finding the absolute maximum, only a close approximation (Rutten, 2013).

2.3.2 Genetic algorithm

Genetic algorithms use evolutionary strategies to find the global maximum. They are stochastic- and non-gradient- and population-based. The way they work is quite similar to Darwin's rule *Survival of the Fittest* (Derix, 2015). In a first step, the fitness function has to be defined. Then, the first generation of genomes is randomly created (Figure 10(1)). According to their fitness (defined by the fitness function), weaker agents are eliminated, whereas the fittest survive and are selected for mating with other survivors and recombining their genes (2). Through that recombination of genes, also called crossover, the new generation is generated, which again is tested for its fitness (3). That procedure is repeated many times, while every generation is usually approaching the global optimum more and more (4). Similar to real life, also mutations are implemented, with the aim to avoid getting stuck at local optima (Elezkurtaj & Franck, 2002). The genetic algorithm delivers quite good results also in early optimization stages (Rutten, 2013). They are quite easy to handle and can be applied on a huge variety of different problems (Rutten, 2014).

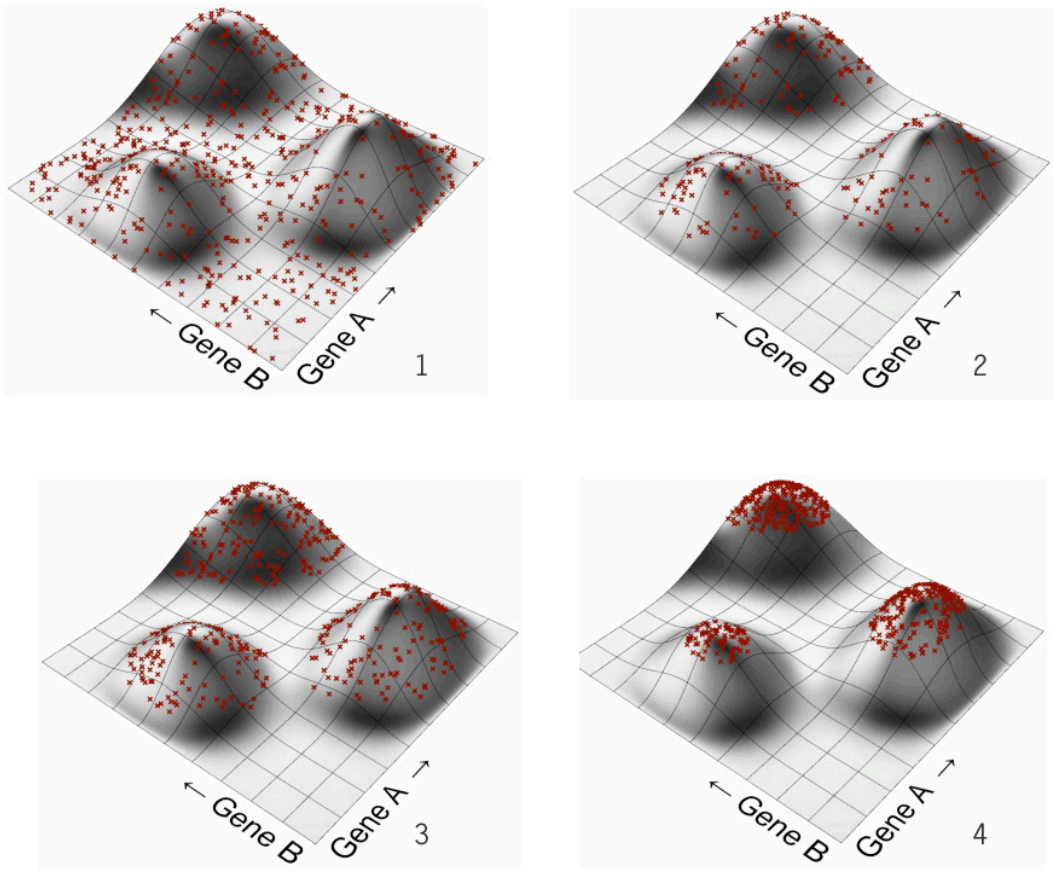


Figure 10: Stages of genetic optimization approaches (grasshopper3d.com)

2.3.3 Simulated annealing algorithm

Simulated annealing is a method inspired by metal processing in the 1980s. It works similar to the crystalline matrix formation of molten metal, which is in its cooling phase at a minimum energy level. The goal is to obtain an even crystal size. It is a stochastic-, gradient- and trajectory- based approach (Yang, 2010). In simulating annealing, a solver starts on a random position in the landscape and, similar to a ball dropping the floor, jumps to its next position randomly. Every new position is analysed by a couple of equations that deliver information about the fitness of the position. If the fitness is accepted, maybe because it is better than the previous one but not necessarily, the solver repeats its procedure from the new position, if it is not accepted, it will try to find new locations starting from the current position (Rutten, 2013) (see Figure 11). That course of action follows the logic of a Markov chain, which adds the necessary randomness to the path of the agent (Yang, 2010). With every new attempt, the jumps will become smaller (similar to the ball loosing its energy) and the criteria for accepting new locations will become stricter, until the global maximum is found. Hence, the lifetime of the solver can be divided into two stages. In the first stage, its jumps are reaching far in order to find the best plateau. Once it is found, smaller jumps are looking for better solutions in the close surrounding on the plateau (Rutten, 2013). The simulated annealing algorithm performs very well in navigating through the solution landscape, which is why the probability to really approximate the global maximum or minimum is high (Rutten, 2013).

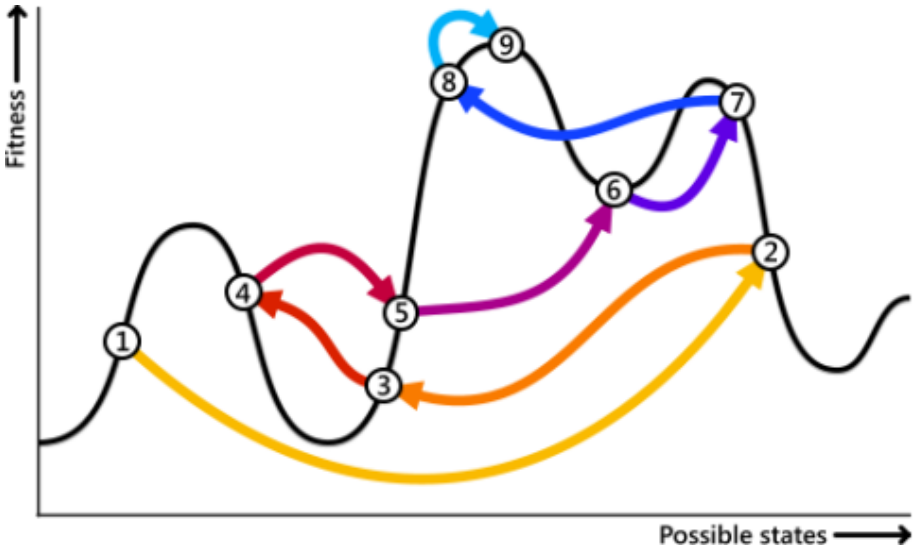


Figure 11: The process of finding the maximum by simulated annealing approaches (Rutten, 2011)

2.4 Evaluation of design options

Algorithmic generation of design options is useless unless there is an efficient way to view, evaluate and compare them and choose the best fitting one. Without an interactive evaluation tool, the designer will lose a lot of time and waste plenty of resources on the selection process; his frustration will grow while going through the numerous outputs. In the end, he might not even be faster than creating a couple of selected options manually. The situation could be compared to analogue and digital photography. As analogue film is not a cheap resource, photographers using the analogue method have to think very carefully about which situation or which object is worth spending a shot on. The advantage of the digital approach is, amongst others, that taking a picture does not cost a thing and that no time has to be spent on developing the pictures. The amount of pictures photographers with analogue cameras take will be considerably less compared to those using digital ones. Still there are some professional photographers preferring the analogue approach because they claim, that choosing and selecting from the sheer amount of digital pictures takes more time than doing it the old way. This example shows, that a proper method of choosing and selecting from numerous digital outputs is crucial for an efficient working process.

2.4.1 Design space exploration

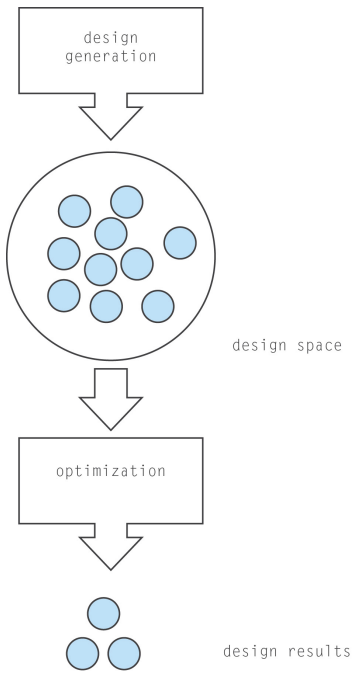


Figure 12: Generation and elimination of design options (Ji, 2012)

Design space, as mentioned before, can be described as a collection of all created solutions (Kang, Jackson, & Schulte, 2010). It must not be mixed up with the *phase space*, which is including all possible solutions. Hence, the *design space* is always part of the *phase space* and always smaller or equal to it (Rutten, 2013). *Design space exploration* refers to the "activity of discovering and evaluating design alternatives" (Kang, Jackson, & Schulte, 2010). *Optimization* is, amongst *rapid prototyping* and *system integration*, an example where *design space evaluation* (DSE) is applied. In the field of optimization, the purpose of DSE is to provide a convenient selection process for users (see Figure 12). That can be achieved by eliminating low score solutions and collecting high score solutions for evaluation and further studies through the designer. (Kang, Jackson, & Schulte, 2010).

Howsoever the design space is organised in detail, it is important for the user to keep his orientation in the design space in order to being able to find and go back to earlier states. Orientation in the design space can be loosely based on website navigation, which includes the commands: *go to, history, view*, and *search*. Those commands enable users to travel in the design space, to review their previous steps, to view their current location in relation to their surrounding and to look for locations of potential interest (Chien & Flemming, 2002).

Another interesting input for navigation design can be obtained from cognitive studies on humans' spatial orientation abilities and strategies, where scientists distinguish three different levels of spatial knowledge: *landmark, procedural* as well as *survey* knowledge. The latter consists of topological information, *procedural* is referring to sequential actions that constitute a route and *landmark* describes the ability to perceive outstanding objects in the environment that serve as reference points. For human orientation, *landmark* knowledge is the most important one (Chien & Flemming, 2002).

2.4.2 Visualization of the design space

Crucial for successful design space navigation is a proper and appealing visualization of results. According to Chien & Flemming, five visualization techniques can be distinguished:

1. Traditional methods, including tools like tables and graphs
2. Nodes and links, used to represent hierarchical or network illustrations, where nodes can be used to typify data sets, which are connected via links.
3. Multiscale views, which can be changed in scale in order to focus on detailed information when zoomed in
4. Perspective views, showing 3-dimensional data and objects
5. Memory palace, is a mnemonic mental-only architectural space containing items in order for a better remembrance (Chien & Flemming, 2002).

As learned from Liggett, one of the reasons for a refusal of optimization tools by the field of architecture could be the lack of a user friendly interface, so a well done and easily understandable visualization of the design options and their evaluation can be the key to a successful optimization tool for architecture, especially as architects are used to a quite visual way of communication (Mahmoodi, 2001).

2.4.3 Iterative design goal approximation

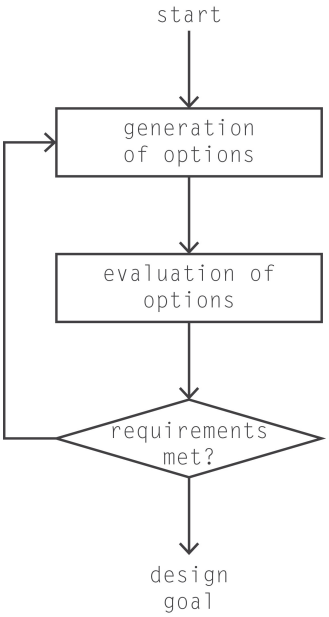


Figure 13: Iterative optimization process (Ji, 2012)

Exploring the design space for the first time, the user might find potential for further improvement of the solutions by adjusting some of the constraints or variables and running the optimization a second time. Depending on the success of the second run and also on the time available in the design process, that procedure can be repeated several times, until the designer's goals are finally satisfied (see Figure 13). That means, that DSE is not only useful for selecting options, but also for evaluation and improving the inputs. In Figure 14, a possible optimization process is displayed. The designer decomposed the design task *A* in 5 sub-problems (*A1,A2,A4,A5*) as described in chapter 1.1. After revising the provided solutions in the design space S1, the user was not satisfied with the results, so the constraints of *A5* to *A5'* needed to be changed in order to obtain another design space *S2* (in the example, the term *solution space* is used instead of *design space*). The user repeats that routine until an appealing design space could eventually be found (Chien & Flemming, 2002).

An optimization process itself is thus in most cases not a linear but an iterative approach. Similar to the way metaheuristics work described in chapter 2.3.1, also the optimization process is improving by application of the *trial and error* method, which in this case is not done by the computer but by the user. By improving the constraints after every run, the desired goal is approached.

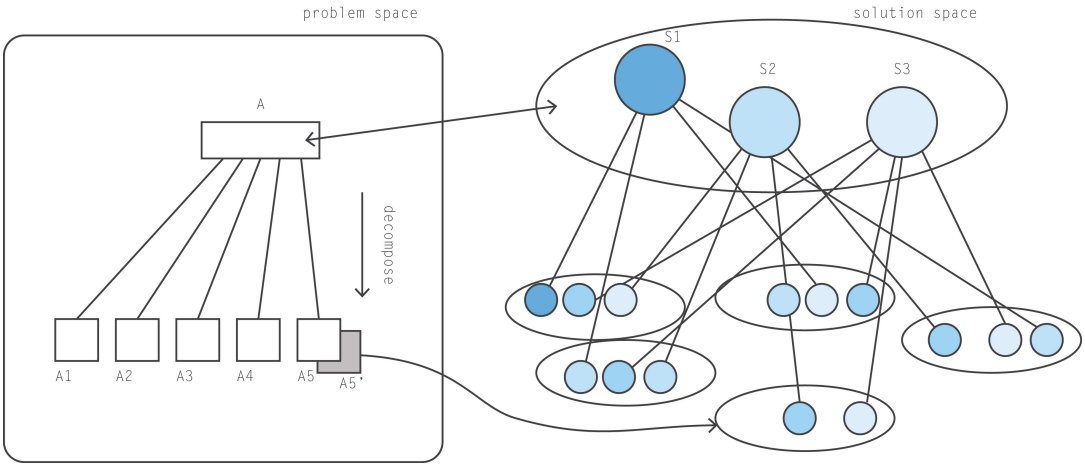


Figure 14: Decomposition of sub-problems (Yezioro, 2009)

2.4.4 Level of detail

The same process can even be extended to a more detailed scale of the project. For example, as soon as a satisfying solution for the massing of the buildings has been found, the designer can “zoom in” and care about smaller recesses and jutties in the façade, where couple of optimization runs can be done as described before. After finding a proper solution for the façade as well, the user could either “zoom out” again and revise if the massing solution is still performing well with the adjusted façade, or if that is unnecessary, the user can further “zoom in” and focus on window openings for example.

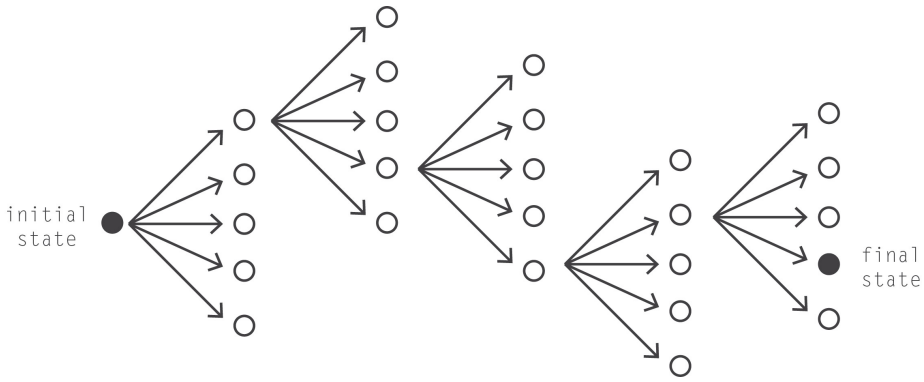


Figure 15: Increasing Level of Detail (Ji, 2012)

The crucial factor for that process in a real projects is time, so the example described is an ideal example where time doesn’t play any role. But with the advance of computer performance, such an extensive process could become reality in the usual design process one day.

2.5 Solar building performance simulation in architecture

As the evaluation tools of this thesis are mainly based on solar aspects, a brief overview is given about the use of solar aspects in architecture, its relevance and current state of acceptance amongst architects.

2.5.1 Relevance of performance simulation

Every year, the amount of solar energy hitting our earth’s surface is twice as much as we could obtain by consuming all non-renewable energy sources at once (see Figure 16). Therefore, the potential to be independent from non-renewable sources is given, but instead of tapping that potential, the vast majority of that solar energy is left unused and we face problems like climate change and air pollution. The *International Energy Agency* has been mentioning

four main reasons for that dissipation, one of them is called *architectural (aesthetic) factors* (Kanters, 2011), indicating that architects have a great potential, but also a great responsibility to reduce global energy consumption by a smarter and more conscious way of designing.

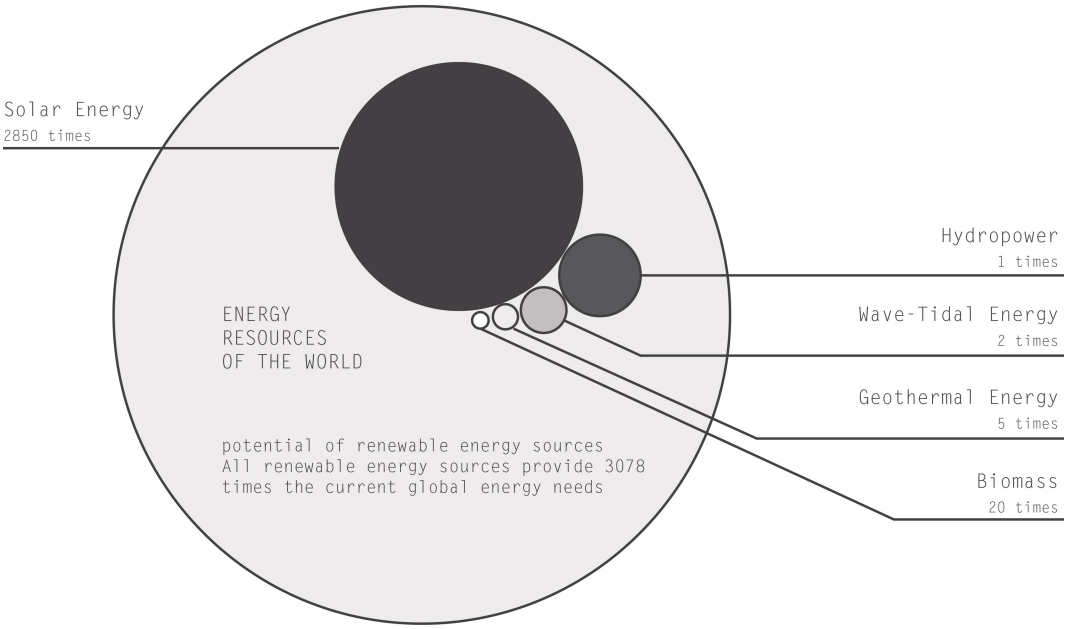


Figure 16: Energy resources of the world (Horvat & Wall, 2012)

2.5.2 Active and passive solar design

Solar energy can be used in two ways, actively and passively.

Active elements are devices designed for the purpose of transforming solar energy into heat or electric power; examples are thermal collectors and photovoltaic (Biermayr, 2013). As active solar tools are not going to be dealt with in this thesis, an extended description will not be provided here.

Passive solar tools are actually not really tools, but rather guidelines, how constructive components such as windows and walls can be installed in order to use solar power in a smart and efficient way. The following guidelines are rough indications, they can differ from location to location, which is why exact climate data is required for each specific site. Passive solar guidelines include:

- Orientation of the building. Heuristics showed that large glazed openings in the façade facing the equator could heat up the building in cold winter conditions when the sun angle is flat. In hot summer conditions, when the sun angle is steep, shading fins on top of the window can prevent the

sun to penetrate the window and overheat the room. Openings in the west and east façade have bad influence on the energy efficiency and are to avoid or kept as small as possible therefore. Furthermore, heat protection glazing can, to a certain degree, control the amount of heat that is transmitted through the glass.

- Building proportions. A low ratio of external wall area in relation to the building volume causes low transmission heat losses in winter.

- Thermal mass, such as massive concrete walls or slabs can absorb heat, which will be returned slowly to the environment. Used in the right way, it will return its energy during cold nights and vice versa cool the space during warm days (Yezioro, 2009).

2.5.3 Current discourse on performance simulation

Studies show that there is a huge discrepancy in the architects' attitude towards solar design. On the one hand, around 80% of architects in well-developed countries appreciate the use of solar tools in their design; on the other hand the same amount (80%) stated that they *occasionally, rarely or never* use those tools (Horvat, Dubois, Snow, & Wall, 2011). Hence there are many attempts to push architects to further improve the sustainability of their designs, like the *Energy Performance Certificate* in middle Europe (Weeber, Sahner, & Bosch-Lewandowski, 2007) or the *European Directive 2010/31/UE* demanding all new built houses in the European Union to be *nearly Zero Energy Buildings* by 2020. The *California Public Utility Commission* is planning to do the same for houses in the U.S. in close future (Horvat & Wall, 2012). Besides legal standards there are also countless awards and competitions for sustainable architecture and design. So we cannot say that the field of architecture is missing awareness for the problem. But then why solar design is not put into practice to a much larger extent?

There is reason to suspect that if architects show good intentions but they do not proceed to put them into action, either they are not being honest or they lack the skills or tools to implement solar design. Looking at the study done by Horvat, Dubois, Snow & Wall again, the latter can be assumed (see Figure 17). The question to the shown results was: "Are there any barriers to your use of available tools related to architectural integration of solar design? (please, select all that apply)".

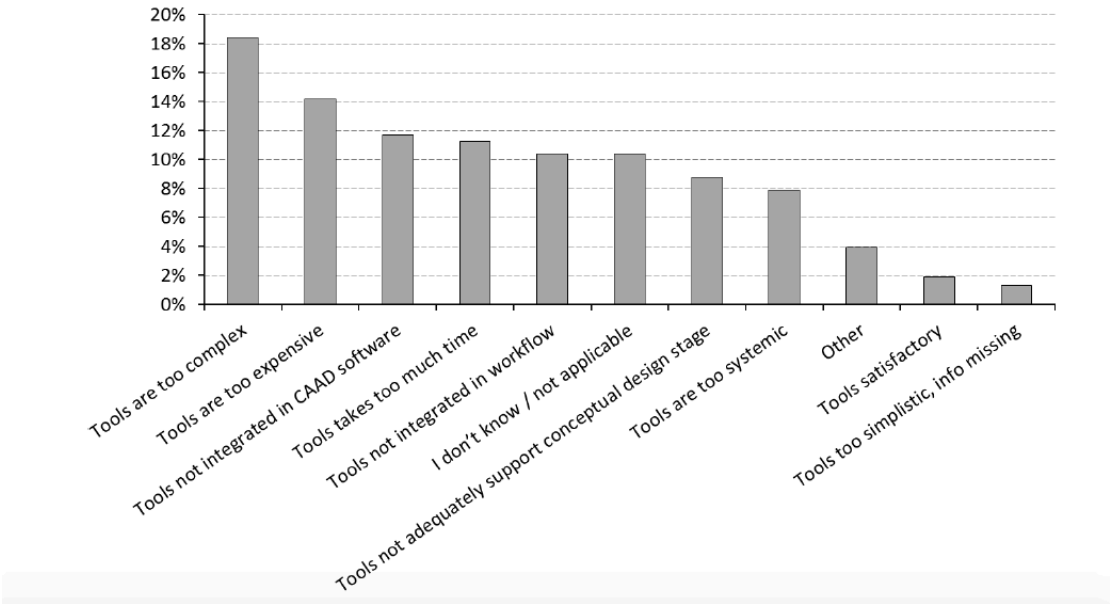


Figure 17: Barriers for software use in architectural field (Horvat, Dubois, Snow, & Wall, 2011)

From a software based perspective the main barriers turned out to be similar to the ones, that lead to a denial of parametric and scripting tools, with main reason being a high complexity. Besides the price, the other main reasons can be summarized as "not matching the architect's design process" (not integrated in CAAD software, too time consuming, not integrated in workflow) (Horvat, Dubois, Snow, & Wall, 2011)

So software-wise, applications would be desirable that integrate powerful solar simulation tools in the usual design process. The attempts done so far by companies like *Autodesk* or *Graphisoft*, which included simple sun light studies in *Revit* respectively *Archicad*, are definitely heading in the right direction, but still lack a lot of detail and proficiency (Horvat & Wall, 2012). So until today, there is no well-accepted all-in-one CAD application available on the market, with the consequence, that solar simulations require exporting the design from the modelling software and importing it into professional simulation software. That process is time consuming and can be accompanied by problems, as different applications are mostly using different floating-point arithmetic and tolerances, which have to be translated. Studies show, that around 3% of a project's costs are caused by software interoperability problems (Salim & Burry, 2010). Moreover, the process of importing and exporting prevents a spontaneous response of the user to results presented by the simulation software, which would increase the designers motivation to quickly test and compare design options and hence to develop a better understanding on the parameters' influence on building performance (Roudsari, Pak, & Smith, 2013). All those software handicaps actually reflect quite well the architects' complaints about insufficiency in complexity, affordability and operability in terms of time and convenient workflows (Roudsari, Pak, & Smith, 2013).

Looking at problems of the architects' side, one has to describe their skills with solar energy tools as *poor* or *very poor*. This may be regarded as a reciprocal effect to the insufficiency of software. As architects do not use the software, their skills are understandably poor. That could be overcome by involving solar energy experts in the design process, but especially in the early design stages, which can be considered the most crucial for a successful implementation of solar energy tools, rarely any architect collaborates with energy experts. At later stages, important design decisions have already been taken and fixed, so there is little space for implementing solar tools (Horvat & Wall, 2012). The main reasons why architects still include simulation tools or energy experts in late design stages is either because they want to test whether the design meets law regulations or because they try to label their design as energy efficient for marketing reasons (Bleil De Souza & Knight, 2007).

Moreover, communication between simulation experts and designers still lacks knowledge and understanding for the counterpart, as experts display their suggestions mainly as pure results in terms of figures without relating those figures to concrete design parameters. That makes it difficult for architects to develop an understanding for consequences that design decisions have on energy efficiency (Krause, Derix, & Gamlesaeter, 2011). The same phenomenon can also be observed in the way that simulation software is presenting its results, which are well understandable for experts, but not appealing to architects, who generally prefer a visual representation of results to pure numbers and tables (Roudsari, Pak, & Smith, 2013). That may explain, why many architects describe available tools as too complex. An increased knowledge of experts and architects about the counterpart's work and hence a better collaboration are necessary to face challenges of our time (Bleil De Souza & Knight, 2007).

3 Building Performance Evaluation Tool (BPET)

In this chapter, a tool will be developed based on the findings of the literature review. The goal is trying to avoid common mistakes that became apparent by studying the history of similar approaches, to come up with a tool that could be used in everyday architecture business. It will enable the designer to generate design options of an already defined design approach and test all of these options for their performance concerning four criteria, which are as mentioned before, *solar radiation*, *shaded areas*, *light incidence* and *view sheds*. After that, the designer will have the opportunity to review the best results assisted by visualized data and make an according choice.

In order to test the tool it will be applied on three projects. The first one is a students' project done by Ivan Matas and the author in 2015, which revealed the potential such a tool could have on the architects' design process. This thesis can be understood as a further development of the tool used in 2015, which lacked scientific foundation as well as practical maturation, still it helped as assistance in a very complex design situation.

3.1 Findings and consequences of literature review

The study of literature has rendered visible some deficits algorithmic optimization tools for architecture have shown so far. In order to come up with an approach how those tools could be improved in order to being implemented in the architects' working routine, some points seem to be of special importance:

- Successful tools have to be of moderate complexity, which does not overcharge a lay person's capacity of understanding a topic the person is not familiar with. Even if the tools do not fulfil the highest level state of the art calculations of building physicists, they can help to improve the building performance. Little optimization is better than no optimization at all, when no one is using the tool. As soon as architects are more familiar with the topic, the complexity could be increased.
- Moreover, the tools have to be embedded in common CAD software in order to ensure a smooth workflow. Exporting into external software is no option.
- The tools have to provide an easy understandable and aesthetically appealing interface if they want to be accepted in a very graphically marked profession such as architecture. Results should never be

presented as plain figures but should always be accompanied by graphical representations.

– A high degree of interaction between the designer and the algorithm is key for a successful implementation. The architect should keep control over every single aspect of his design at any time of the design process. He needs to have a distinct understanding what the algorithm is doing with the design in order to use it in the right way.

Besides those important aspects that literature review has revealed, there is one problem with the way optimization algorithms work, that was not addressed in any of the studied works, but which could be one of the main reasons, why algorithmic optimization and architecture still have not been successfully united, yet.

3.1.1 Discussion on limitations of optimization algorithms

Many architecture optimization approaches (that are using the computer) are using optimization algorithms. The reason for that may be obvious at the first glance: why not use a powerful tool that had been successfully developed throughout the last 70 years and which aims to provide the most efficient solution.

But a closer look reveals that we face significant problems when applying those algorithms to architecture, which leads to the conclusion, that for most cases architecture and optimization algorithms do not fit together very well. The following example will explain why.

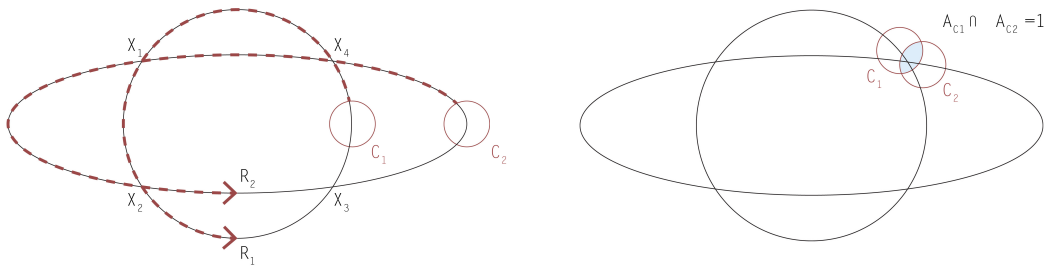


Figure 18: Algorithmic solver test

Figure 18 on the left is showing a very simple optimization problem, created to test the behaviour of different algorithms. There are two small circles C_1 and C_2 that can move on their rails R_1 and R_2 respectively. The intersection points of R_1 and R_2 may be called X_1, X_2, X_3 and X_4. As C_1 and C_2 are moving on their rails, at

some point they will intersect, the optimization task is to obtain an intersection area $A_{C1} \cap A_{C2} = 1$, as shown in Figure 18 on the right. Now, from pure logic, we can say that the solution shown is not the only possible solution. If we move C_1 downwards on R_1 and keep C_2 steady there has to be another intersection fulfilling the equation $A_{C1} \cap A_{C2} = 1$. If we do it the other way round and keep C_1 steady while moving C_2 on R_2 to the right we will obtain another valid solution and if we move both we will obtain a forth one. That procedure is not only valid for the shown intersection point X_4, but also for the other intersections X_1, X_2, X_3. In total, we will end up with 16 possible equal solutions fulfilling the equation.

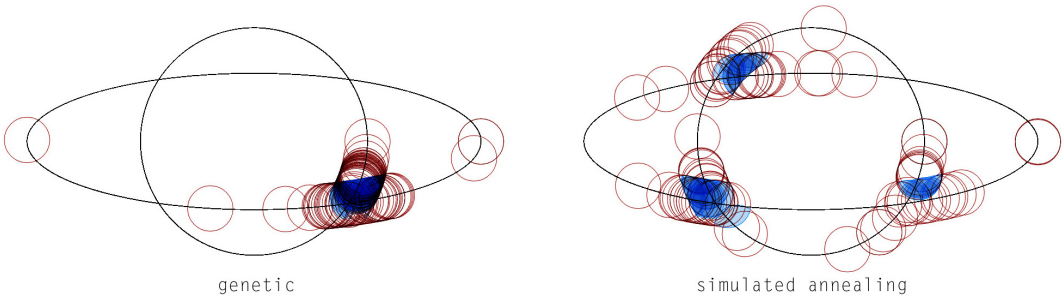


Figure 19: Algorithmic solver test

Figure 19 is showing the results obtained by the *genetic solver* on the left side and the *simulated annealing solver* on the right side. The diagrams actually show 50 solutions like the one displayed in Figure 18 (each single diagram containing only two red circles and the blue area of overlapping if existent), but overlaid in one representation to visualize the focal points of the optimization process. We can see the positions of the small circles throughout the simulation in red as well as their overlapping in blue. Because of the overlapping, the darker the blue, the more circles overlapped in that area. The simulation was executed with the *Galapagos* component for the *Grasshopper* software, which contains both solver solutions.

The genetic solver as well as the simulated annealing solver both delivered a very good result concerning the demanded value, both found a solution where $A_{C1} \cap A_{C2} = 0.9998$, so they only missed the perfect solution for *0,0002*. If we take a look at the distribution of the solution, their characteristics described in the chapters before become apparent.

The genetic solver, as soon as it found some results at intersection X_3, focused on that area without leaving again and tried to optimize the result to its maximum. The circles in the far periphery of X_3 are from the beginning of the search process, when it was trying to find some result at all, as no overlapping of circles always results in $A_{C1} \cap A_{C2} = 0$.

The simulated annealing solver proved its quality of discovering the landscape quite well; we can find high concentration of results at intersection X_1 and X_2 as

well as lower concentration at X_3. Only at X_4 the solver was not able to find any result at all.

From the engineer's point of view, the search process was very successful. The aberration of *0.0002* from the search goal 1 is negligible in practice. But as mentioned, there are 16 solutions that fulfilled the search goal equally well. If we neglect that there are actually 4 at every intersection, which are location wise very similar, we still have 4 very different locations at the intersections (X_1, X_2, X_3 and X_4). The genetic solver denied us 3 of them, only showing the one at X_3. The simulated annealing did better, but still denied us the one at X_4.

For the engineer it does not matter at which location he reaches the desired goal, but for an architect it might actually matter a lot. Assuming the simulation was a site plan optimization, where the best position for placing the building is where the result is 1 ($A_{C1} \cap A_{C2} = 1$), so exactly what the optimization was looking for. Obviously, the site would have a specific environment, which is not the same at every intersection. From an architect's point of view it would be desirable to be presented all the four possible solutions, so that he can pick the perfect simulation result as well as the perfect location concerning the surrounding. Furthermore, for an architect, who has to include also the subjective, non-tangible aspects, the choice where to place the building will probably be a compromise between the optimization result and the other factors. So in the end, maybe not the position that scored the best (*0.99*) will be chosen, but a position with a score of only *0.95*, but which satisfies all the other aspects much better.

In other worlds, for an architect, who has to respect many different factors in his design, it is not so relevant to chose the one and only best result provided by the optimization (as the engineer would do), but a *good* result may be already sufficient if it delivers the overarching best solution.

Assumingly, the loss of control over the optimization process, which became apparent in the shown example, is what makes many architects sceptical and is one of the reasons leading to the denial of optimization tools in general.

3.1.3 A proposal for undirected randomness

What all optimization algorithms have in common is that they are target-oriented. They have been designed for the sole purpose of finding the best result. Yang is comparing their proceeding to treasure hunting. In an unknown landscape, they try to find the treasure hidden on the highest peak of the mountain as fast and efficient as possible, they are not interested in the landscape (Yang, 2010). That's what they have been programmed for. For architecture, which is not only about efficiency but also about aesthetics that may not be the right approach. Maybe a naïve, less efficient treasure hunter, who wants to find the treasure, but is interested in the beauty of the landscape that is surrounding him as well, fits much better to the concept of architecture. A mix of eagerness to find the treasure combined with a *the journey is the reward* attitude.

What those metaphoric words with a bit of irony want to express is that a random generation of design options, which is not target driven, might match the way architects work much better. It is actually quite similar to what *BIG Architects* show in their comic-book *yes is more* and what they call *excess and selection* (see Figure 21) (Ingels, et al., 2009): A huge amount of design options, from which the architect has to choose, with the difference, that a computer can create incomparably more options and at the same time provide critical data like energy performance for all of those options. Combined with rapid prototyping, a comparably fast translation of the virtual options into real physical models can easily be achieved (Kang, Jackson, & Schulte, 2010).

The difference between ordinary optimization approaches using optimization algorithms and the *undirected randomness* approach becomes even more apparent when looking at the distribution of solutions in the phase space as shown by Figure 20. The amount of solutions is identical in both approaches, but the genetic solver clearly shows a concentration of results at the global maximum, while local maxima are ignored. Undirected randomness can include those local maxima by an undirected distribution of solutions.

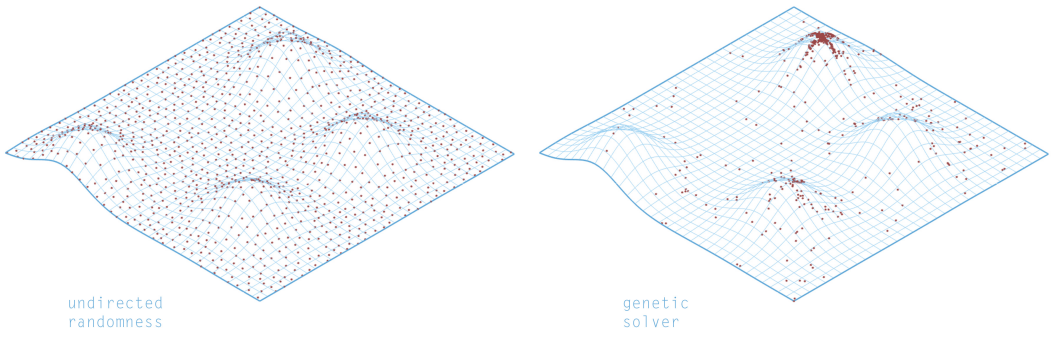


Figure 20: Coverage of undirected randomness vs. genetic solver

Figure 21: BIG. Excess and selection (Ingels, et al., 2009)

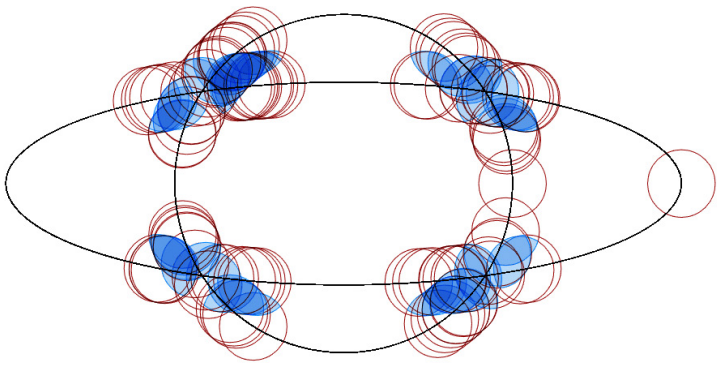


Figure 22: Algorithmic solver test

If that notion of undirected randomness is applied on the previous test example, a much more balanced distribution of solutions can be obtained (see Figure 22). We can see, that all the intersection points manifest a balanced concentration of solutions. The best result in terms of maximum value is less than those of the optimization algorithms, but 0.9986 compared to 0.9998 is almost negligible, especially if, as mentioned before, architectural optimisation is not about finding the absolute maximum at any cost.

Recalling Figure 12 on page 21, where Ji is displaying the optimization process by abstracting the four steps of *design generation -> design space -> optimization –> optimization result*, the function of optimization can be interpreted as eliminating design options of the design space in order to display only the best left-overs (Ji, 2012). In that case, the big difference between the common algorithmic optimization approach and the approach proposed by me is the following: In the common way, the algorithm is the one which optimizes, which is eliminating the "bad" results, whereas in my proposal the "optimizer" is the designer, assisted by data provided by the algorithm. The advantage is obvious: the designer is able not only to make a selection according to a ranking of data, he is also able to choose options fulfilling non-tangible criteria, which the algorithm is not.

Moreover, the suggested approach of undirected randomness has two more advantages.

1. Creativity: A higher potential for surprising results, as postulated by Elezkurtaj & Franck, and thus an increased creativity can be achieved, because its diversity in the created results is higher (Elezkurtaj & Franck, 2002).
2. As the undirected randomness does not have to use any metaheuristic calculations, it saves calculation time and thus can be faster than common optimization algorithms. For interaction with the user as well as for a successful implementation in the architectural design process, speed is crucial (Elezkurtaj & Franck, 2002).

The suggested approach is an important component in the attempt to balance engineering and architecture. Factors that can be optimized should be optimized, but always in respect to non-tangible factors. Aspects like aesthetics or arts are crucial for architecture, though they can never be subject to optimization. So coming back to Yang's statement earlier in this thesis, when he says: "Optimization is everywhere" (Yang, 2010), I have to contradict. Optimization is not everywhere!

3.2 Software used in thesis

The basic platform for the BPET is *Rhinoceros* by *McNeel*. As that software application is very common in the field of architecture, the acceptance of architects to also use the following plugins is comparably high, which is an important precondition for a successful implementation of an optimization tool. *Rhinoceros* is a nurbs based modelling application used by various fields of design (rhino3d.com, 2017).

3.2.1 Grasshopper for Rhino

Grasshopper is a parametric modelling plugin for the *Rhino* software. The developers describe it as “a graphical algorithm editor” which “requires no knowledge of programming or scripting, but still allows designers to build form generators from the simple to the awe-inspiring” (grasshopper3d.com). The way code is represented in *Grasshopper* resembles the object-oriented programming paradigm, where the parametric model is symbolically represented by node components (Salim & Burry, 2010). The way the components connect and are fed with instances of variables constitute a current state of the model, which can be viewed in the *Rhino* viewport in 2d- as well as in 3d-representation. In that process the user can define which of the nodes shall be displayed in the viewport and which shall be turned off, as viewing all nodes simultaneously will lead to an overload of information in the viewport, prohibiting the user to focus on the essential node representations. Different node types symbolize different geometrical objects that can be displayed, such as points, curves, surfaces or solids. Other types of components, such as numerical inputs, Boolean operators, mathematical expressions, information panels and other abstract contents obviously do not have a visual representation in the viewport (Aish & Woodbury, 2005).

The use of such node based scripting tools like *Grasshopper* encourages the designer to think about his design from an additional perspective. Besides the view in common 2d- and 3d representations, the more diagrammatic view of the nodes connected to a graph may lead to a better understanding of how the whole design connects together and where there are dependencies of objects or groups. Especially in early design stages, grasshopper can be used to generate multiple design options rapidly and contribute to a better understanding of the design task (Salim & Burry, 2010).

Parametric instances can be “baked” and that way being converted into regular Rhino geometry, which can be further manipulated manually (Akos & Parsons, 2014). Furthermore, *Grasshopper* provides the possibility to easily use and integrate various plugins for innumerable purposes, also user scripted nodes can be created in different programming languages. Thus, it can be considered as basic platform for many different other applications, such as physical

simulations, engineering calculations, geometrical optimization and many more (food4rhino.com, 2017).

Grasshopper natively even contains two algorithmic optimization solvers (genetic and simulated annealing), which are combined in the *Galapagos* component, other kinds of solvers are available as plugins. For reasons mentioned in chapter 3.1.2, the solvers will not be used in this thesis. Nevertheless, depending on the specific project or optimization task an optimization conducted by one of the solvers could be reasonable, e.g. if one option has already been chosen from the approach suggested, a final optimization with very strict constraints, which does not change the design significantly could aim for ultimate performance. But as time matters and architecture does not have to go for the absolute maximum of performance anyway, especially not in multivariate optimization, those solvers will not be used in the suggested approach.

Today, *Grasshopper* is one of the most common parametric tools used by designers, so it has the great potential to avoid the problems described in chapter 2.5.3 that architects are complaining about the circumstance, that simulation tools are not included in the software they use (Roudsari, Pak, & Smith, 2013). As a consequence, *Grasshopper* has been chosen as a platform, where the *Building Performance Evaluation Tool* will be based on.

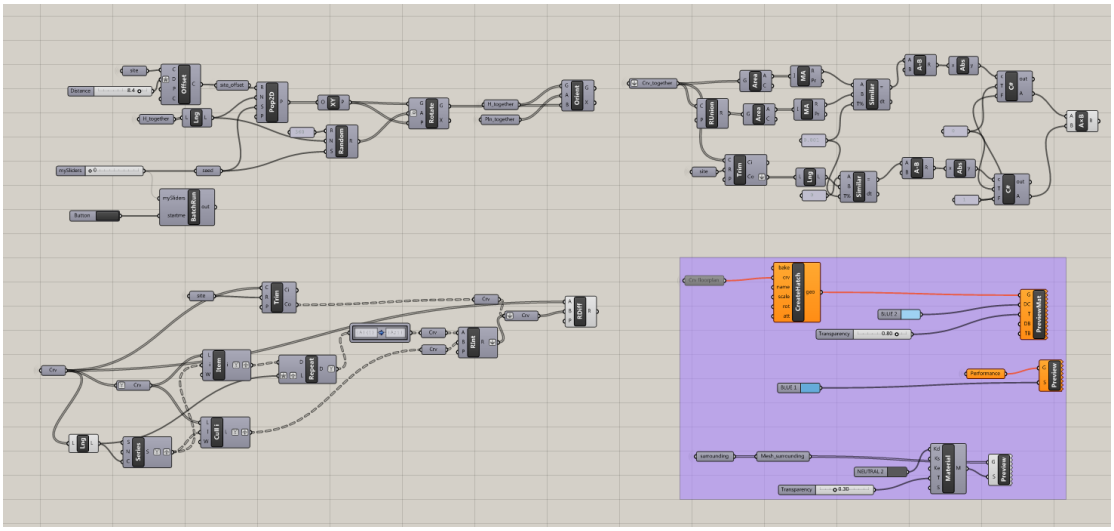


Figure 23: Grasshopper's node based interface

3.2.2 Ladybug for Grasshopper

Ladybug is an open-source plugin for *Grasshopper*, especially programmed for architects to simulate environmental behaviour of their design options. It is aiming for assistance of the architect in initial design stages. Results are presented in an easily understandable way in 2d- as well as in 3d-views in the rhino viewport. For obtaining accurate results based on the location of the

project, ladybug uses standard *EnergyPlus* weather files (.epw files), which can be downloaded from the *EnergyPlus* website, that offers weather files for approximately 2100 locations worldwide (energyplus.net/weather). The general features of *Ladybug* cover radiation studies, orientation studies, sun path and optimum solar form finding using multi-objective optimization.

The developers of *Ladybug* mention that they were aware of the software problems described in chapter 2.5.3, which motivated them to develop a tool that is embedded in a well-accepted parametric design platform and is capable of simulating all relevant components that influence the energy performance of a building in real time. Another emphasis was put on a visual based presentation of results that is understandable even for users with limited knowledge about the topic.

While *Ladybug* is focusing on early design stages, *Honeybee*, established by the same developers, can be regarded as the continuation of *Ladybug* for later design stages, when materials, window positions or room sizes have been specified. The interoperability of the two programs is given, so the work in *Grasshopper* can just be continued without any export and import (Roudsari, Pak, & Smith, 2013). Moreover, Honeybee can be considered as hub to other simulation tools if needed (see Figure 25).

Although there are a couple of other similar energy software solutions available for *Grasshopper*, the *Ladybug* product family was chosen for this thesis as it provides the most overarching supply of simulation applications. Figure 24 is showing an overview over applications available for *Grasshopper* and their limitations. Note, that the chart was published by the *Ladybug* developers and cannot be considered as objective therefore.

PROCESSES		ANALYSIS TOOLS				
		Ladybug	Heliotrope	Geco	Gerilla	Diva-for-Rhino
Climate Analysis	Analysis	✓				
	Visualization	✓	✓**			
Massing/Orientation Study		✓		✓		✓
Daylighting Study		✓		✓		✓
Energy Modeling		✓			✓	✓*

Figure 24: Overview over energy simulation software available for Grasshopper (Roudsari, Pak, & Smith, 2013)

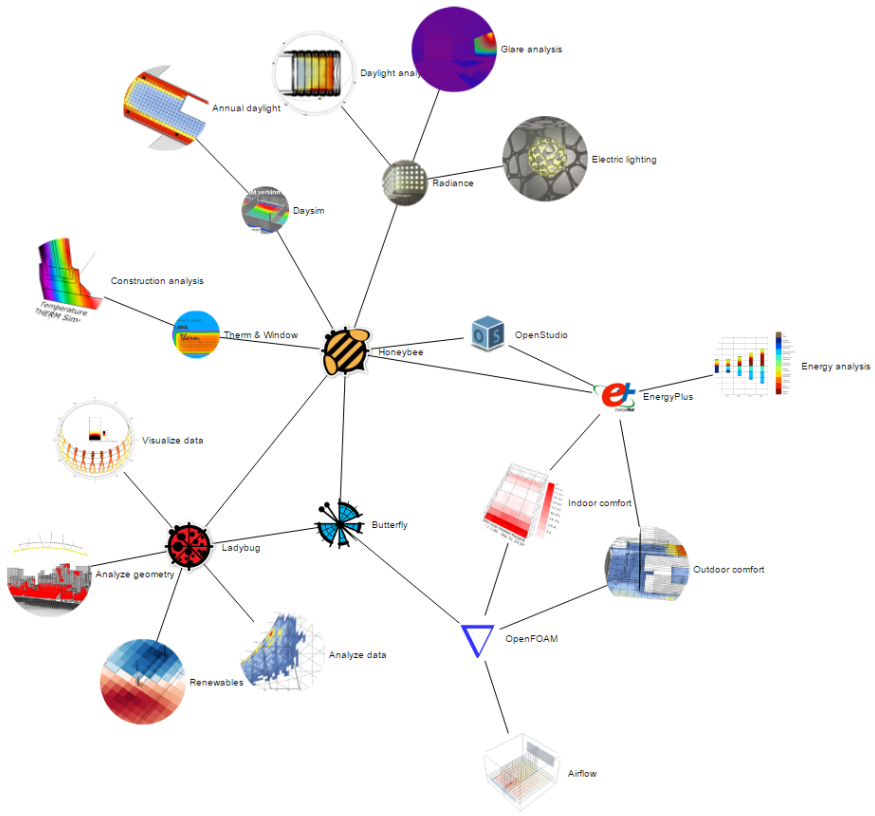


Figure 25: The Ladybug product family and its relations (Roudsari, Pak, & Smith, 2013)

3.3 Generation of design options

For the generation of design options, a parametric model has to be provided by the user, which contains all necessary constraints and variables. For that purpose, the user has to reflect carefully which parts of the design can be changed within a given range and which parts have to remain constant. The requirements and characteristics of such a parametric model largely depend on the specific project and the optimization goals, which is why the parameterization process cannot be standardized. As mentioned before, not every project may be appropriate for optimization, if the design idea does not allow any variations.

The design options will be created by an algorithm using the undirected randomness approach introduced in chapter 3.1.2. The way it is applied is project specific, but in any case a range for variables has to be defined, from which the algorithm can pick values randomly. The random values are obtained from the *Grasshopper random* component, which is creating pseudo-random numbers within a user-defined range. The attribute *pseudo* is referring to the fact, that those values are not truly random in a mathematical sense, as their creation is always deterministic, meaning that the same inputs in the random

component will always return the exactly same outputs. Different sets of random values can be obtained by changing the so-called *seed* value (grasshopper3d.com). The fact that the values are pseudo-random is in the case of this thesis very helpful, as by entering a specific seed value later on again, the exactly same values and hence, the same design option can be regenerated. For the generation of design options, every time the algorithm changes the seed value, new random values are created and that way a new design option is born. So in the end, the amount of random seed values is equivalent to the amount of options generated. As soon as the algorithm has generated one option, it will test the option for the criteria that the designer applied beforehand.

3.4 Design evaluation criteria

As mentioned in chapter 2.1, only tangible criteria can be optimized by an algorithm. Precondition for that is a numerical expression of the criterion. As there is no such thing as an indisputable approach for the conversion of a criterion into numbers, there is constant discussion amongst scientist and experts on how to improve or enlarge calculations to approach an exact display of those criteria. In many cases, the reality is by far too complex for an all-embracing method, which is why simplified assumptions have to be used. In the field of energy efficient architecture that can be considered as such a complex case, there are numerous approaches on how to calculate the efficiency of a building; the discrepancy of different approaches regarding the result is remarkable (Sofic, 2009).

For a simulation tool applied in early design stages, where many of the required specifications are for a very sophisticated calculation method are missing, simplified assumptions have to be made. The result thus cannot be regarded as a valid statement on the building's final energy efficiency fulfilling all the state of art criteria of building physics' calculations. It is much rather showing the designer a direction with greater potentials for an efficient design. Hence, units such as kWh used in that early design stage do not allow any concrete statement about the future energy consumption but have to be considered as abstract quantities with the sole purpose of comparing and evaluating design options. For this reason, units are deliberately not used in this thesis; all calculated values for the design options have to be understood as factors rather than results.

3.4.1 Passive Solar architecture

All passive solar architecture calculations that will be applied in this thesis will be based on weather files that can be downloaded at the *EnergyPlus* website for various locations worldwide (energyplus.net/weather).

3.4.1.1 Solar Radiation

The amount of solar radiation received by a specific area is dependent on the site's location, on its angle towards the sun, on the season as well as on the actual atmospheric conditions. As the latter is changing constantly, weather files contain an average value of radiation for a specific location. Given any specified geometry, we are able to calculate the total radiation on that geometry for a chosen period of time with the help of simulation software such as *Ladybug*.

In order to make statements on how the received radiation will affect the energy consumption of a building throughout a year, we have to distinguish four different cases depending on the specific location of the project and its corresponding climate conditions.

1. In locations where the temperature throughout a year never reaches a level that requires cooling a space, only heating demand has to be considered. In that case the higher the radiation is during the heating period, the higher is the amount of energy that can be saved.

2. In locations the opposite is the case, where the temperature never reaches a level that requires heating, only the cooling demand throughout the cooling period has to be considered. The higher the radiation is during that period, the more energy needs to be used to compensate that radiation in order to cool the room.

3. In locations where both heating and cooling is required, the two seasonal cases as described in chapter 2.5.2 have to be considered. For the heating period, a high radiation in order to heat up indoor spaces is wanted, though a high radiation in summer will lead to increased energy consumption for cooling the space. Hence, the total annual amount of radiation does not allow any prediction about energy efficiency, which is why the two cases have to be evaluated separately.

4. In locations where neither heating nor cooling is required, the calculation of radiation can be considered obsolete as it has no influence on energy efficiency.

The calculation for cases 1 and 2 are simple, as the higher or the lower, respectively, the radiation result for a specific design option, the better. So the value obtained from the *Ladybug radiation analysis* component can be used without further calculation. For case 3 a way has to be found, how to value and compare the two cases in order to come up with one overall annual value. An exact calculation of how that case of radiation is influencing energy efficiency is highly complex and could be the only topic of another thesis easily. Especially as important factors such as material choices or energy concept are still missing in the early stage, it is only possible to point out design solutions that show a high potential for being energy efficient. But as the early design stages are the most influential ones on performance behaviour, the choice of a design option that shows good potential is already worth a lot, as it can be improved to a very high performance in later, more detailed design stages.

For obtaining a value that allows comparing different design options, the following simplified way of calculation based on heating degree days (HDD) and cooling degree days (CDD) is chosen:

The phases of cooling demand and the phases of heating demand throughout a year will be separated by using a so called base temperature T_b, a value which differs from location to location and which is set by the state regularities. For Austria, that value is *12.0°* (Sofic, 2009), for Germany it is *15.0°* (energie-lexikon.info), for Canada *18.0°* (Newsham & Donnelly, 2013) and for most of the U.S. states it is *18.3°* (Sailor, 2001). As there is no European standard for the calculation of the cooling degree days, many studies apply the U.S. standard with $T_b = 18.3°$. For this thesis, though, the calculation will be mainly based on a study conducted in 2011 in Germany, which adjusted the calculation process to more realistic European user behaviour. As two of the example projects on which the tool will be applied on later are located in Vienna, Austria and a similar user behaviour of German and Austrian users can be assumed, that calculation method can be considered the most appropriate one. It is using a value for $T_b = 19°$ and applying a heating threshold of $T < 12.0°$ and a cooling threshold of $T > 19.0°$, meaning days with mean temperature $T < 12.0°$ are counted as heating days and days with $T > 19.0°$ are counted as cooling days (Olonscheck, Holsten, & Kropp, 2011).

The heating degree days for each month can be calculated with the following equation:

$$HDD_m = \sum_{d=1}^{n_d} (T_b - T) \quad for\ T < 12.0°$$

whereas for the cooling degree days the formula is

$$CDD_m = \sum_{d=1}^{n_d} (T - T_b) \quad for\ T > 19.0°$$

where

HDD_m	=	Heating degree days for the specific month
CDD_m	=	Cooling degree days for the specific month
n_d	=	the number of days in a particular month
T	=	the mean daily temperature
T_b	=	base temperature

The annual heating/cooling degree days (HDD_a;CDD_a) will be obtained by calculating the sum of the monthly results, so

$$HDD_a = \sum_{d=m}^{12} (HDD_m)$$

and

$$CDD_a = \sum_{d=m}^{12} (CDD_m)$$

where

HDD_m	=	Heating degree days for the specific month
CDD_m	=	Cooling degree days for the specific month
HDD_a	=	Heating degree days for the whole year
CDD_a	=	Cooling degree days for the whole year

HHD and CDD are good evidence indicating the heating as well as the cooling period for a specific location combined with a quantitative weighting of the required heating or cooling demand, respectively (Sailor, 2001).

Taking Vienna, Austria as an example, Table 1 is showing the correspondent monthly mean temperatures (BMWFW, 2017) in C°.

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
0.67	2.34	6.36	11.32	15.83	19.35	21.4	20.83	17.04	11.21	5.59	1.84

Table 1: Monthly mean temperatures in C° for Vienna, Austria

By inserting those values into the formula, the results shown in Table 2 are obtained. The heating period for Vienna is according to the calculation from October to April and the cooling period from June to August. May and September do not require heating or cooling. The total heating degree days sum up to HDD_a = 2832.7, whereas the cooling degree days are HDD_a = 141.6. The data already suggests, that in the climate of Vienna, heating expenses contribute much more to the energy consumption than cooling does.

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
568.2	466.5	391.8	230.4	0.0	10.5	74.4	56.7	0.0	241.5	402.3	532.0

Table 2: Monthly HDDs (red) and CDDs (blue) for Vienna, Austria

According to the calculated heating and cooling periods the solar gains for each period have to be calculated separately. That will be done with the *Ladybug* radiation analysis component, which is providing the total amount of radiation on the test geometry in the defined test period in kWh. For test geometry only window openings will be considered. If window positions are still unknown, walls for potential windows can be used as well, but the result will be less precise. Solar transmission gains through walls and roofs will be neglected, as a proper insulation is assumed to being installed in later design stages anyway.

A common equation for an approximation of the heating energy demand Q_h is

$$Q_h = 0{,}024 * f * HDD_a * (H_T + H_V) - \eta * (Q_S + Q_I)$$

where

Q_h	=	Heating energy demand
f	=	factor for inclusion of a night setback of the heating system temperature = 0.95 (kh/d)
HDD_a	=	Heating degree days for the whole year
H_T	=	Transmission heat losses
H_V	=	Heat ventilation losses
η	=	factor for inclusion of the utilization factor of internal and solar heat gains
Q_S	=	Usable solar heat gains (constant value)
Q_I	=	Usable internal heat gains (constant value)

Expressed in words, that equation transforms the HDD_a multiplied by transmission losses (H_T+H_V) and subtracted the solar gains from it (Q_S+Q_I) into the heating energy demand in kWh/a. But as for that equation a lot of unknown factors are demanded, such as U-values for building components or user behaviour, for an early design stage the equation is too complex and prohibiting a fast and intuitive evaluation of design options. So in regard of a user-friendly design tool, a more simplified, but less accurate calculation will be applied. In that case, the HDD_a respectively the CDD_a will just be considered as factors, which quantify energy demand in abstract terms and value the radiation result delivered by *Ladybug*. So for example a very high CDD_a will put more importance to the weighting of the summer radiation result, a low one will diminish it. As stated before, in the Vienna case, the CDD_a is comparably low, so the potential of saving cooling energy is way less than the potential of saving heating energy. The following equation is trying to include that weighting and helping to obtain a more accurate statement about energy saving potential. However, the radiation value Q then will be a virtual value only valid for comparing the design options.

$$Q = \left(\frac{HDD_a}{HDD_a + CDD_a} \times Q_H \right) - \left(\frac{CDD_a}{HDD_a + CDD_a} \times Q_C \right)$$

where

Q	=	validated total radiation value
HDD_a	=	Heating degree days for the whole year
CDD_a	=	Cooling degree days for the whole year
Q_H	=	heating period solar radiation result provided by simulation
Q_C	=	cooling period solar radiation result provided by simulation

As in the cooling period high solar radiation is contrarious to low energy consumption, it will be subtracted from the heating period solar radiation. The equation can be considered as a compromise between a very accurate computation and a well working application in early design stages. The higher the value is for a design option, the higher its potential for being energy efficient.

As the window position might not be fixed or even considered in early design stages, two different scenarios have to be inspected. In case the window positions are known, the calculation will be executed as described before for every given window position, whereas if the window positions are unknown, surfaces of potential window positions (such as external walls) have to be provided. The result returned by the calculation can be used as hint for energy efficient window positioning. Before defining surfaces of potential windows, considerations about good window orientations according to passive solar principles as described in chapter 2.5.2 are helpful.

The area that is tested for its radiation will be subdivided in smaller subareas, each of them tested for its specific radiation individually and resulting in one single value. The sum of all the values is the total radiation on the test area for the defined period. The higher the number of subdivisions, the more precise is the result, but the longer takes its calculation. As the calculation of radiation is very complex and time consuming, a well considered decision is highly recommended.

For the calculations of solar radiation, the following specifications have to be made by the user:

input

- < Parametric model containing first design decisions
- < Position of windows as surface *or:* Area for potential windows as surface
- < Surrounding neighbourhood

variables

- Heating period
- Cooling period
- Subdivision u and v of radiation mesh

output

- > Q = validated total radiation result

3.4.1.2 Shaded areas

Shaded areas are in a negative correlation to solar radiation, the more shaded an area is, the less is its solar radiation. Still there are some good reasons to include both solar radiation and shading.

- Shading studies are used as a common tool in urban planning, as shading effects of buildings on their environment can be visualized.
- Solar radiation studies are focusing on heat effects in indoor spaces, hence its calculation is for most cases only reasonable for building openings such as windows or for efficient positioning of photovoltaic cells.
- While radiation studies mainly examine effects on energy potential, shadow studies more subjectively focus on aspects like spatial qualities. Studies show, that shading can have both positive as well as negative consequences on humans' health (Rehan & Islam, 2015).

For obtaining a shading result for each design option, an area has to be defined to test for shading. That defined area will be subdivided in rectangular subareas. Those subareas will be considered as shaded, if within the test hour its area is covered by shadows by more than 50%. The result will be a binary value μ, where shaded = 0, not shaded = 1. Figure 26 (1) is showing such a result for all subareas for 3pm local time. By accumulating those results for every hour of the day, the area will be evaluated by a value S_A, where $0 < S_A < 24$. (24 would be an extreme case only possible in summer in far north or far south regions in the world, where an area could potentially receive 24h of sunlight).

$$S_A = \sum_{i=1}^{24} (\mu_i)$$

where

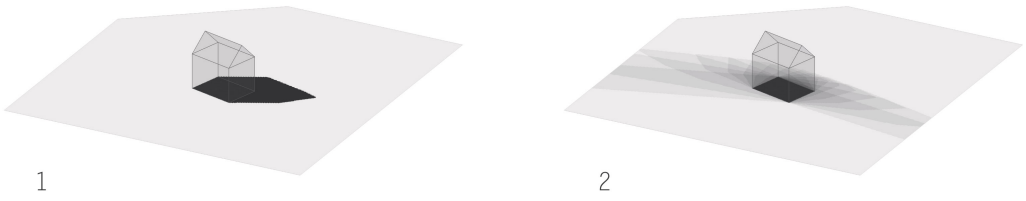
S_A = value for a specific subarea on the specific test day
μ = binary factor for shaded = 0; not shaded = 1

By another accumulation of each performance of all the subareas S_a, the value S_d will be obtained.

$$S_D = \sum S_A$$

where

S_D = value for the whole tested area on the specific test day
S_A = value for a specific subarea on the specific test day



1 2

Figure 26: Shading result for one hour and overlaid for the whole day

Figure 26 (2) is showing the final result for one possible design option graphically, where all the hours of a day are overlaid.

Except for locations at the equator, the shading result differs for every day in the year, with a minimum of shaded areas when the sun angle is the steepest (June 21st) and a maximum when the angle is the lowest (December 21st, for the southern hemisphere vice versa). In order to obtain a fair result, both the two extreme cases will be calculated and summed up into one final value

$$S = \frac{S_{d\,dec} + S_{d\,jun}}{2}$$

where

S	=	final shading result
$S_{d\,dec}$	=	shading result on December 21st
$S_{d\,jun}$	=	shading result on June 21st

S will be finally used to compare the design options. In that approach, as little shading as possible is desired, so the bigger the value for S, the better the option. There might be rare cases, where a maximum of shading is wanted, in that case the values have to be inverted.

When using multivariate optimization with shading as well as solar radiation as criteria, it is recommended to not use shading analysis on the windows of the building, as that is already analysed by the radiation. Because of the negative correlation of the two criteria, the shading result will make the radiation result futile and vice versa, so in the end there will be an ambiguous total result. Instead, shading analysis can very well be used for on-site outdoor areas and neighbouring areas and buildings.

For the calculations of shading, the following specifications have to be made by the user:

input

- < Parametric model containing first design decisions
- < Area or areas or geometry that shall be analysed for shading

variables

- Day or days considered for calculation (default: Dec 21st ; Jun 21st)
- Subdivision u and v of geometry that shall be analysed

output

- > S = Sum of all non shaded areas for 24h on the defined test days

3.4.1.3 Light incidence

While solar radiation and shading are criteria that are mostly dependent on direct solar light, light incidence focuses on the diffuse (indirect) radiation, which in architecture has always been one of the major design criteria, as it is the natural source to light up rooms. Since the invention of electrical lighting, there are other ways to have bright rooms even without connection to exterior light, but solar light is still the preferred light source, not only for energy saving reasons, but much more for psychological and health reasons. That is why, in many countries, there are regularities that specify the amount of diffuse sunlight that has to be available in a room.

In Austria, the rule to regulate light incidence is called OIB-300.3-005/07-001. According to that, taking the vertically lowest and horizontally centred point of a window and drawing a virtual line from that point 45° degrees perpendicular to the window towards the sky, that line must not be blocked by any obstacle such as a building on the opposite side. If that line is blocked, also another such line within a range of horizontally 60° may fulfil that criterion in order to satisfy the regulation (see Figure 27) (Österreichisches Institut für Bautechnik, 2007).

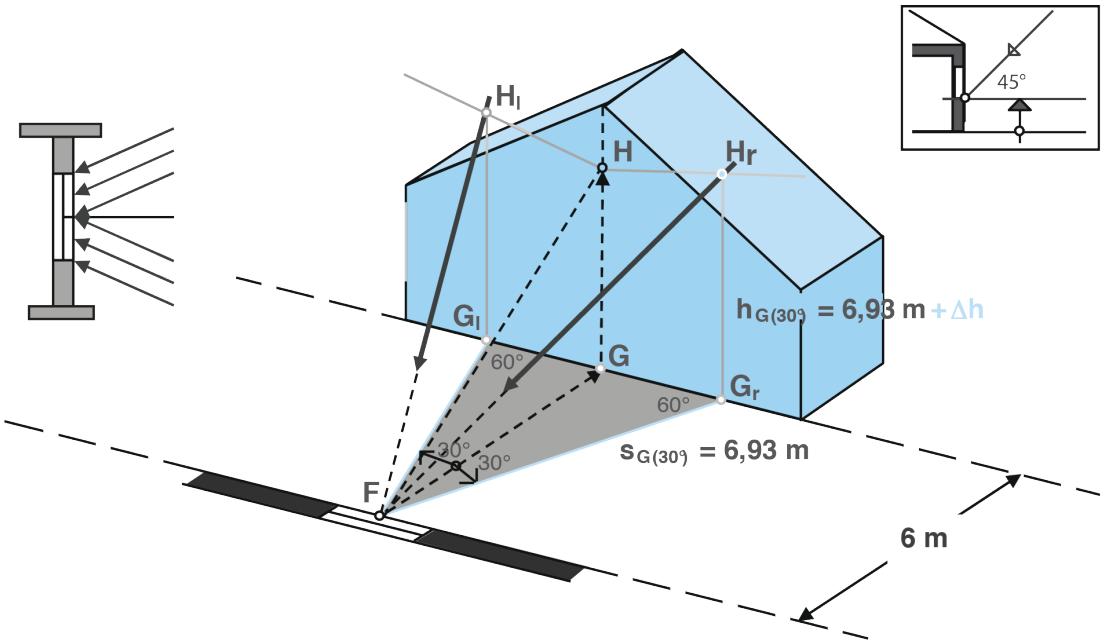


Figure 27: Light incidence regularities for Austria (Österreichisches Institut für Bautechnik, 2007)

The value for comparing the design options will be calculated by creating a couple of 45° vectors as described within the horizontal 60° angle. If all of those lines intersect an obstacle of the surrounding, the checked location will be treated as invalid for a window position, hence the room cannot be used as a proper room. If any of the lines has no intersection, the condition can be considered as fulfilled and the location as valid for a room window. For invalid conditions, the location will be treated as $L_i = 0$, for valid condition as $L_i = 1$. The sum of all locations together provides the final value indicating how many valid locations the design option contains.

$$L = \sum_{n=1}^{n_l} (L_i)$$

where

L	=	final value showing amount of valid locations
n_l	=	number of total locations
L_i	=	Location result at specific location i

The values suggested are obtained by the Austrian standard, but can be manipulated by the user and adjusted to other regulations or demands easily. As the window position might not be fixed or even considered in early design stages, two different scenarios have to be taken into account.

In case the window positions are known, the calculation will be executed as described before for every given window position (Figure 28 (1)).

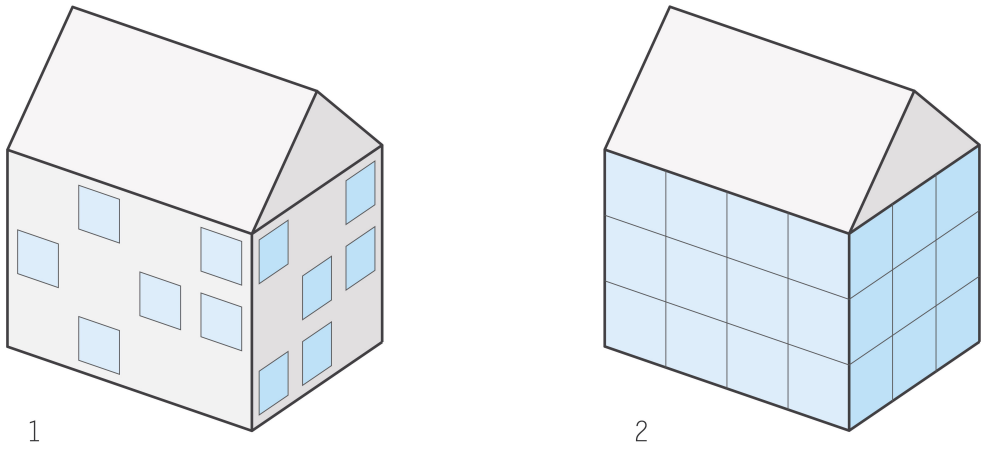


Figure 28: Approach for calculation of known and unknown window positions

In case the window positions are unknown, surfaces of potential window positions (such as external walls) have to be provided. Those surfaces will be subdivided into a rectangular grid defined by the user. A vertical grid size matching the stories of the building is recommended. For every subarea of the subdivided grid, the same calculation procedure as described before will be applied, thus showing potential window positions and their quantity for every design option (2). A high number of subdivisions will increase the calculation time, that's why the subdivision process has to be well considered.

For the calculations of light incidence, the following specifications have to be made by the user:

input

- < Parametric model containing first design decisions
- < Position of windows as surface *or:* Area for potential windows as surface
- < Surrounding neighbourhood

variables

- Vertical angle (default: 45°)
- Horizontal angle range (default: 60°)
- *If no windows given:* subdivision u and v of potential window area

output

- > L = Number of valid locations concerning light incidence

3.4.2 Further project relevant criteria

Even though the thesis' topic actually focuses on solar optimization mainly, view sheds are included to test proper multivariate optimization, as the solar criteria are correlated to each other to a certain extend. In order to have an independent criterion, view sheds are included, which could be accompanied by any other tangible factor in future use, if considered important for optimization. View sheds were specifically chosen to represent multivariate criteria, since they were an important criterion for the design of the *ppag* example project.

3.4.2.1 View sheds

In ordinary language, the view is mostly evaluated according to its aesthetic value. We appreciate hill peaks, observation decks, towers etc. for their beautiful and amazing view. But it is actually not the view itself, that is considered beautiful, but much rather the objects that can be perceived through that view. However, the more objects can be perceived and being related to each other in one view, the more impressive is the sensation of a view. So two components seem to determine the value of a view: The beauty of the perceived objects on the one hand and the area that can be overlooked on the other hand. The former is a purely subjective, qualitative factor, the latter an objective, quantifiable one. As in optimization we can only deal with tangible factors that can be expressed in a quantitative way, we have to optimize views according to the area they overlook.

In science, a view shed is referred to as isovist, a "set of all points visible from a given vantage point in space and with respect to an environment" (Benedikt, 1979). Isovists focus on environmental perception of visible, opaque surfaces in space, excluding translucent or highly reflective surfaces such as glass, mirrors or the sky. Figure 29 is showing three different isovists in the same space depending on the different vantage points in that space and the boundary in shape of a surrounding circle.

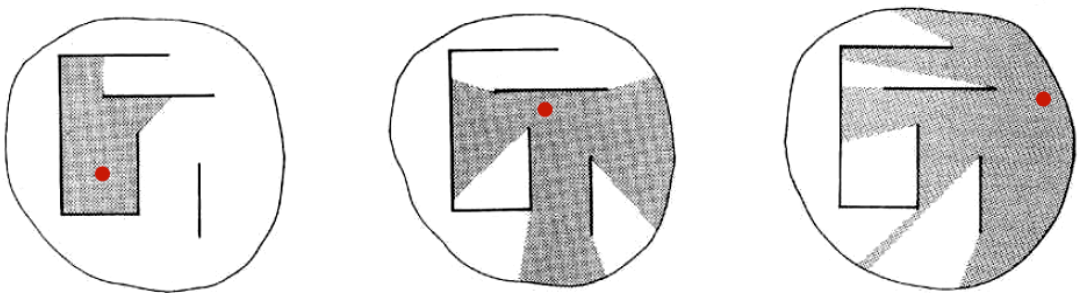


Figure 29: Isovists according to different vantage points in space (Benedikt, 1979)

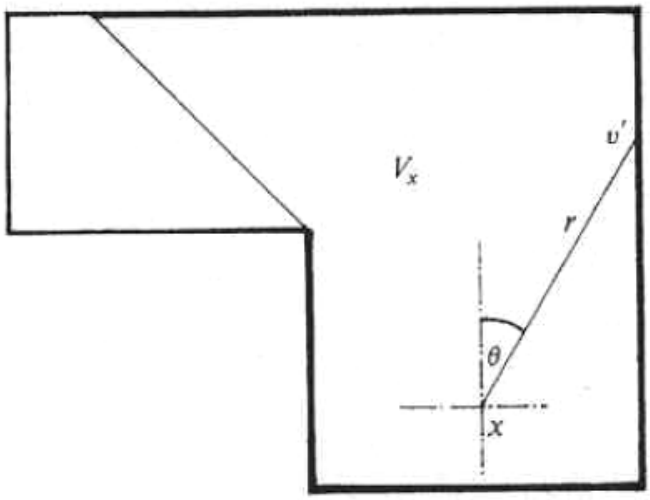


Figure 30: Calculation of isovists (Benedikt, 1979)

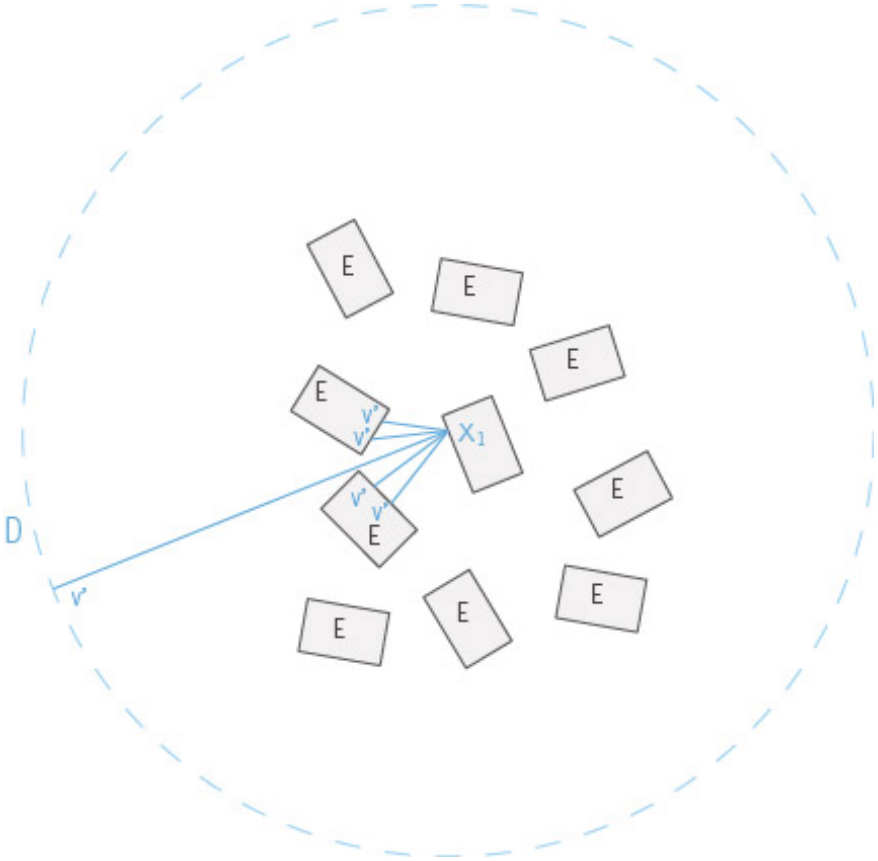


Figure 31: Calculation approach of view sheds for the presented tool

The quantity of an isovist can be expressed by calculating the area or it can be approximated by the sum of lengths of a set of radial lines *r* within a certain field of view *θ* connecting the vantage point *x* and the intersection points with surfaces *v'* (see Figure 30) (Benedikt, 1979).

For this thesis, the latter method will be applied, as for the computer the time needed to calculate areas is much longer than computing and adding the lengths of several lines. As shown in Figure 31, a range *j* of horizontal radial lines r_{ij} is converging in the vantage point x_i , which can be assumed as the window position. The angle θ of the range is in this case 60°, following the same logic as for the light incidence described in chapter 3.4.1.3. The lengths of the lines r_i are defined by the first intersection point (v') of r_i with the environmental surfaces *E* or the virtual boundary circle *D*. The sum V_i of the lengths d_{ij} of the lines r_{ij} could be regarded as a final result to measure the view shed of x_i, but there are two problems with lines that have no intersection with the surrounding buildings (surfaces *E*).

- First, as their length would be infinite without being stopped by the virtual boundary circle *D*, the accumulated lengths V_i of x_i would be infinite for all x_i, which have at least one line r_{ij} that is not intersecting with neighbouring buildings (surfaces *E*). In less mathematical words, the value of all window positions with at least one unblocked view would numerically approximate infinity. Hence all windows that do not have at least one unblocked view, would have a concrete value, which, compared to infinite values, would approximate 0. Consequently the solution space would consist of only binary values, either showing the value of 0 for all windows with blocked views or the value of infinity for all the others.

- In order to avoid that, the boundary circle *D* is introduced, stopping lines at a certain point. But the question is, which value to assume for the radius of *D*, as it is a virtual boundary. The larger the radius, the more emphasis is put to unblocked views, the smaller, the less is the difference between blocked and unblocked views. Moreover, the circle is stopping the lengths of views abruptly, which does not correspond to the behaviour of views in real life, which gradually fade out towards the horizon.

In order to attenuate the described effect, the square roots of the lengths V_i of the lines r_{ij} will be calculated. That will fade out the influence of far distance views gradually and provide a more balanced result as shown in Figure 32

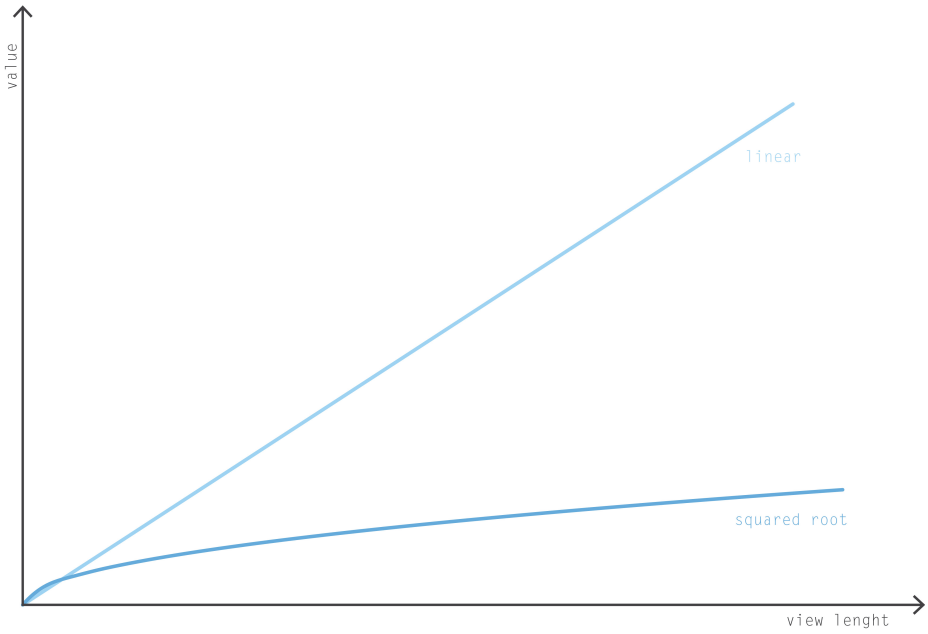


Figure 32: Comparison of linear and squared root graphs

The equation for the result V_i will thus be the following

$$V_i = \sum_{n=1}^{j} \left(\sqrt{d_{ij}} \right)$$

where

V_i	=	factor at location i
j	=	number of range of radial horizontal lines
d_{ij}	=	lengths of radial horizontal line at location i

in order to calculate the final factor for the design option, the equation will be

$$V = \sum_{n=1}^{i} (V_i)$$

where

V	=	final factor for all locations
i	=	number of locations analysed for views
V_i	=	factor at location i

As described before, sometimes it is a specific kind of object such as a famous building that makes a view special in a qualitative way. For even respecting that case in the optimization, for every line r_{ij} hitting that special object (building) the specific length d_{ij} will be multiplied by a factor with a certain value. For a hotel in Paris e.g. which wants to put some emphasis on viewing the Eiffel Tower, that could be the way to proceed. As no such special eye catcher is in the surrounding of the example projects, that case will not be represented in the tool.

As the window position might not be fixed or even considered in early design stages, two different scenarios have to be considered.

In case the window positions are known, the calculation will be executed as described before for every given window position.

In case the window positions are unknown, surfaces of potential window positions (such as external walls) have to be provided. Those surfaces will be subdivided into a rectangular grid defined by the user. A vertical grid size matching the stories of the building is recommended. For every subarea of the subdivided grid, the same calculation procedure as described before will be applied, thus showing potential window positions promising good views. A high number of subdivisions will increase the calculation time, which is why the subdivision process has to be well considered.

For the calculations of light incidence, the following specifications have to be made by the user:

input

- < Parametric model containing first design decisions
- < Position of windows as surface *or:* Area for potential windows as surface
- < Surrounding neighbourhood

variables

- Horizontal angle range (default: 60°)
- Number of horizontal radial lines within set range (default: 5)
- Radius of boundary circle
- *If no windows given:* subdivision u and v of potential window area

output

- > V = squared root of the total length of views

3.5 Design space exploration

Before BPET can be applied on a design, the designer has to provide a parametrical model that incorporates all of the designers' ideas so far and that allows modification done by the algorithm according to the set constraints. After that, a decision has to be made, which of the evaluation criteria should be applied, which depends on the specific direction to which the designer wants the design to be optimized. Single-variate optimization is easier to execute and faster to calculate, but multivariate optimization has the potential to optimize the model in a more holistic way.

After the choice of the appropriate optimization criteria, the algorithm can be started. However, before starting it to go for a high number of design options, a quick test with just two or three options is recommended in order to avoid thousands of invalid design options and a lot of wasted time because of some wrong settings. The higher the number of generated options, the higher the probability to find the very best results, but the longer is the calculation time as well. That is why for the first run a lower number of generated options is recommended. In later runs the result can still be optimized with improved settings.

After one run is finished, the output will be a list of data containing the index of the design options and the corresponding performance for each criterion. So if all optimization criteria are applied, four lists will be generated. Based on those lists, the user can start to explore the results. For doing that, he will be assisted by some tools in order to easily view and evaluate the best options.

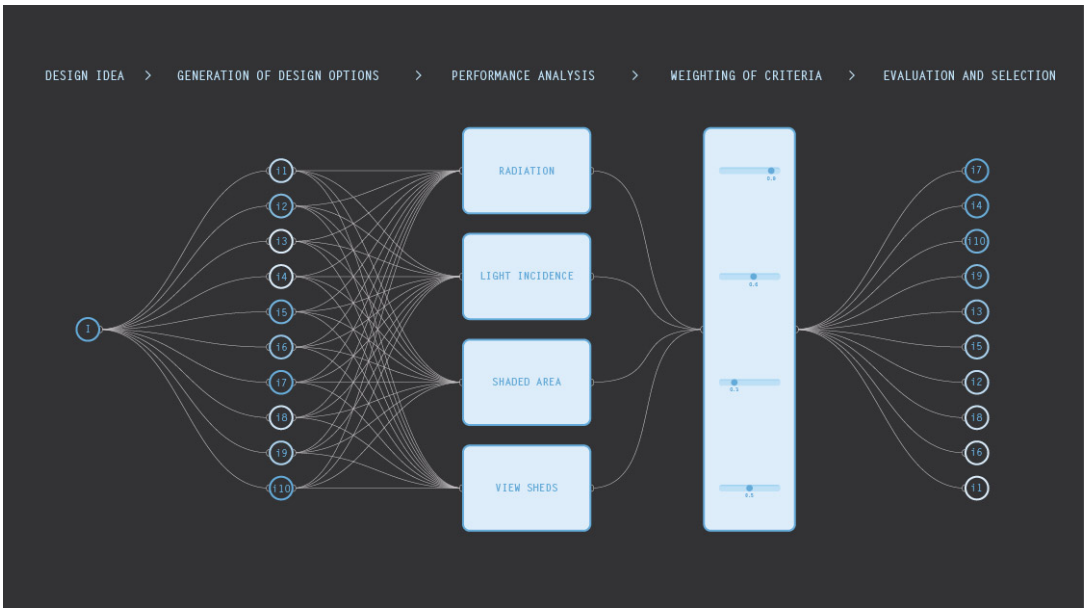


Figure 33: The optimization process as suggested in this thesis

Figure 33 displays the suggested optimization process graphically. Starting from the design idea, alternative options of that idea are created. The four data lists created in that process are represented by the four rectangles. After weighting the four criteria according to specific design requirements, a ranking is shown, enabling the user to view the best solutions only and dismissing bad solutions.

3.5.1 Weighting of criteria by user

A weighting of criteria is important, as not all of the four criteria might be equally relevant for the design. For weighting the criteria, the user can define their importance on a range between 0 and 1, where 0 is ignoring the criterion completely and 1 is putting the maximum of importance to it. However, those values are always in relation to the other values, so if the user sets 0.1 for all of the four criteria, obviously all of them will be treated equally. That is also true for any other value, which is applied to all the four criteria equally. If the user wants to stress radiation for example, she might rate it with 1.0. Consequently, all the other values should be less, let's assume light incidence to be 0.6, shaded areas 0.3 and view sheds to be 0.5. But how can shaded areas e.g. be directly compared to radiation and be regarded as one third of the importance of radiation, although their factors and their virtual units are different. For a correct comparison like that, the results have to be mapped into the same range. In order to do so, the following equation can be used.

$$R_{Ci} = \frac{U_{Ci}}{U_{C\,high}}$$

where

R_{Ci} = relative result for the design option *i* of criterion *C*
U_{Ci} = value for the design option *i* of criterion *C*
$U_{C\,high}$ = highest value U_i for *C*

The equation will relate every single value U_{Ci} to the highest value $U_{C\,high}$ returned by the algorithm. Hence, the related maximum score according to that equation will be 1 for the highest result; all the other results will be part of a range 0 to 1. That formula is valid for cases, where the highest result in terms of numbers is considered the best result, as it is the case for light incidence e.g., where a maximum amount of sufficient light incidence is the goal.

In cases where it is the opposite, for example if the user wants the maximum of shading on a certain surface, the equation has to be

$$R_{Ci} = 1 + (1 - \frac{U_{Ci}}{U_{C\ low}})$$

where

R_{Ci}	=	relative result for the design option *i* of criterion *C*
U_{Ci}	=	value for the design option *i* of criterion *C*
$U_{C\ low}$	=	lowest value U_i for *C*

As after the use of that equations, all results for the four criteria are ranged within 0 and 1, they are now ready for being weighted and ranked accordingly.

The equation used for the weighting will be

$$R_{total} = \frac{\omega_Q * R_{Qi} + \omega_S * R_{Si} + \omega_L * R_{Li} + \omega_V * R_{Vi}}{\omega_Q + \omega_S + \omega_L + \omega_V}$$

where

R_{total}	=	final result used for ranking
Q	=	Solar radiation
S	=	Shading
L	=	Light incidence
V	=	View sheds
ω	=	weighting factor defined by user

For the final result, again, the scores will be part of the range 0 to 1, so that the user can classify the score just by seeing the plain number, as it resembles a percentage based representation, where 100% is defined as the maximum achievable. In most multivariate optimization cases though, even the first ranked option does not achieve the value 1.0, as that value requires, that one option achieved the best score in every single criterion applied.

Besides the results for the design options, another important value will be provided to the user for a better evaluation of the results, which will describe the range of results in relation to the maximum result. The value will show the user to what extend the different design options can influence the performance. It can be described as a factor for the potential of optimization. The value will be called *significance factor* (SF) in this thesis. It can be calculated as

$$SF = \frac{C_{high} - C_{low}}{C_{high}}$$

where:

SF = significance factor for criterion C
C_{high} = highest result for C
C_{low} = lowest result for C

The significance factor is always within the range 0 to 1. A high value will indicate that there is a huge difference in the results of the design options and hence, that the potential for optimization is high, a small one will indicate the opposite.

An example shall explain the use of the SF. Assuming the analysis of solar radiation has brought forth values within a range between 55 and 100, hence the SF is 0.45. The analysis for view sheds, however, brought forth values within a range between 99 and 100 and a resulting SF of 0.01. In case of radiation, it can make a huge difference, which option is chosen, as the best option is performing almost twice as well as the worst, whereas in the case of view sheds, that difference is comparably small, it makes little difference which option to choose. The SF shows which potential each of the criteria has, which can have influences on the weighting through the user.

Moreover, as manipulation of a list of data is very fast compared to computing geometry and the corresponding results of the design options, the user will be able to play with the four criteria-weighting-sliders intuitively and see, what influence their manipulation has on the ranking in real-time.

To sum up, after the generation of design options the user will be provided the following data

- The SF for each criterion used, describing the potential that the criterion has.
- A ranking of the design options according to the user's weighting.
- A table showing the total results and the specific values that each option scored in the specific criteria.

3.5.2 Visualisation and choice of results

So far, only data has been produced. For a user-friendly evaluation of the results, a proper way of visualizing the data has to be provided. In the BPET, four different types of graphic tools will assist the user.

1. Bar charts and tables

   The first tool provides the user with two kinds of bar charts accompanied by the table of the top results. The first bar chart is sorted by the index of the design options, the order in which they have been created (which is identical with the seed value for the *Grasshopper random* component creating the options). The height of the bar is defined by the final result R_{total} of the design option. The second one is ranking the design options according to their result, so the best results are shown first, the worst last, as displayed in Figure 36. In both cases, the index of the option is displayed underneath the bar, the result on the top of the bar.

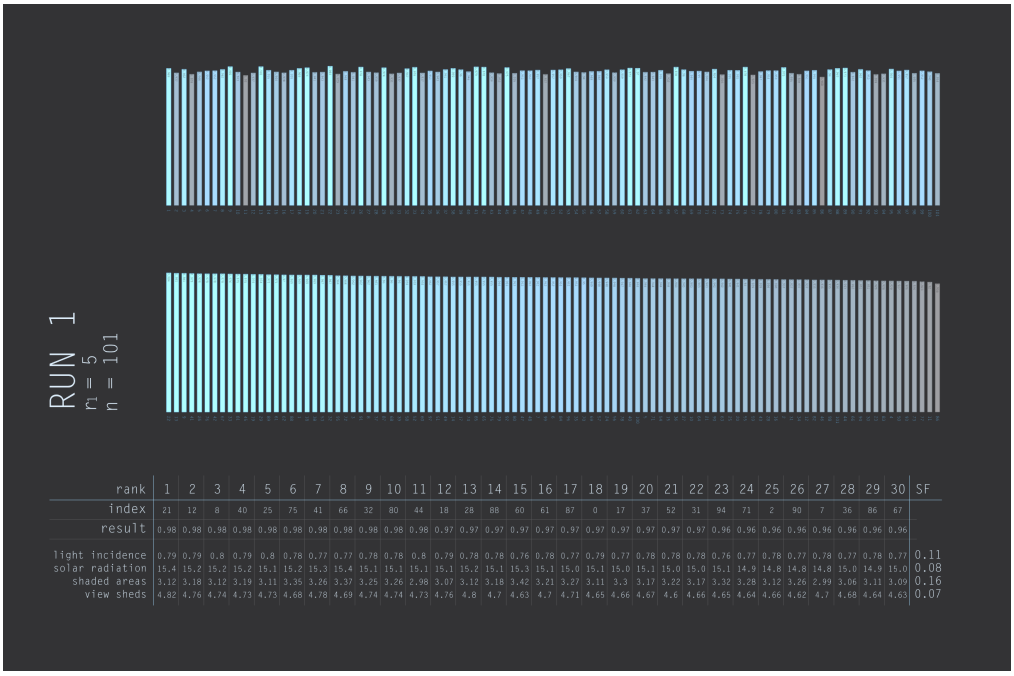


Figure 34: Performance chart

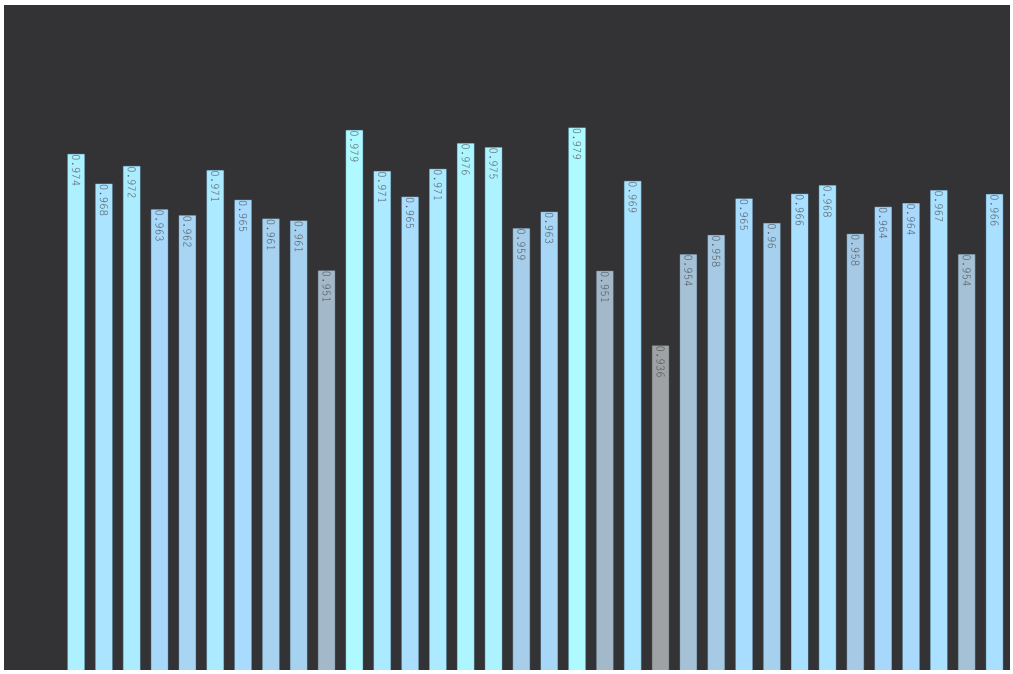


Figure 35: Bar chart sorted by index

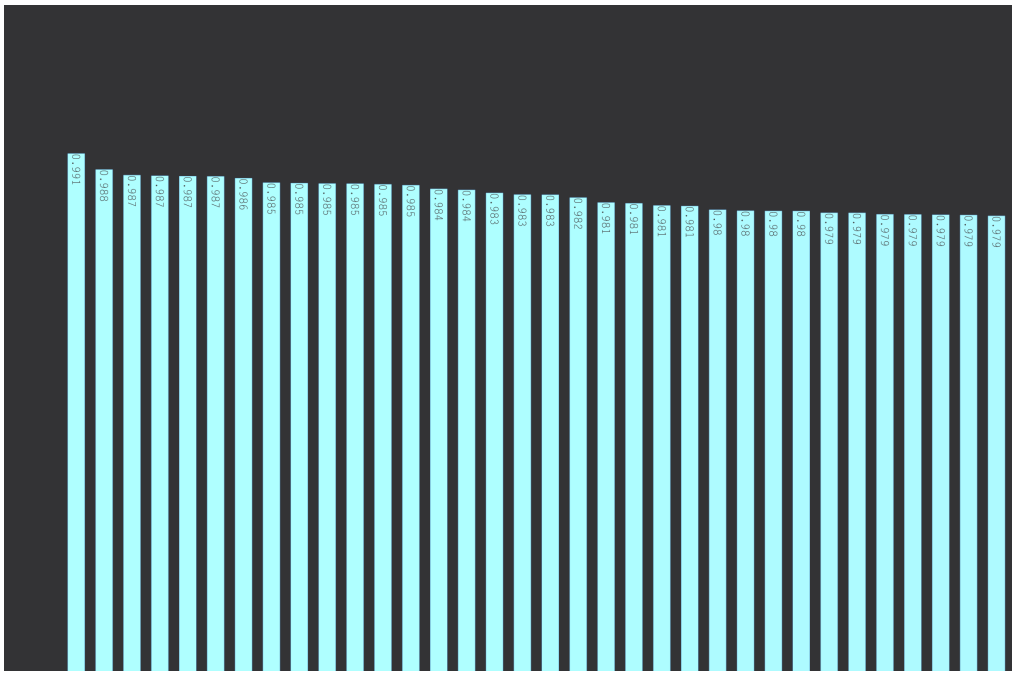


Figure 36: Bar chart sorted by rank

The table is sorted according to the ranks of the options. It contains information about the final result R_{total}, and underneath the values for the specific criteria. As the total result R_{total} is always a relative value, comparing the performance of the particular option to the best performances achieved in the particular run, it cannot be used for comparison with other runs, unless the results of the runs are not joined into one table (which is usually not the case in the BPET). So for a quick

comparison of options of different runs, the specific results for the criteria can be viewed, as those are absolute numbers. On the very right of the table, the *Significance Value* SF can be viewed for each criterion.

2. Radar chart

   For more graphical information about the strengths and the weaknesses of the viewed design option concerning its performance, a so-called radar chart will be displayed additionally (see Figure 37). The radar chart is a tool to visualize multivariate data, its axes are radial from its centre. The higher the corresponding value, the larger is its distance to the centre. By connecting all scores in a radial manner, a polygon is created. The area of the polygon is correlated to the overall (unvalued) performance of the design option. The radar chart is a powerful tool to quickly compare the performance of different options and their characteristics (1).

   As the radar chart is only applicable on data with at least three variables, it will be substituted by a bar chart for two or one-dimensional cases (2).

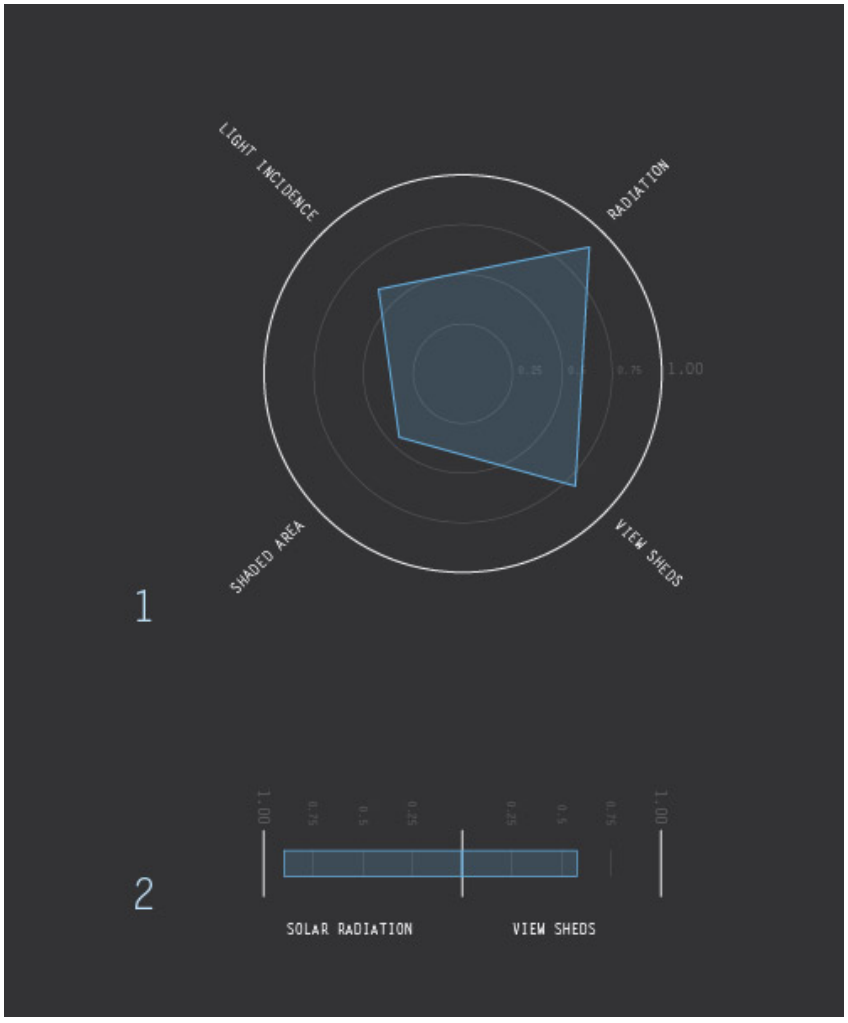


Figure 37: radar chart and bar chart

3. Geometry of design option

   For architects, a crucial tool will be the 2d and 3d graphical representation of the design options, as it delivers important information about spatial qualities. A specific option will be shown in the *Rhino* viewport as *Grasshopper* representation (see Figure 38). The user can conveniently examine the option in a conventional way for designers. For further manipulation of an option, it can be *baked* (transformed into regular *Rhino* geometry) and treated with every tool available in *Rhino*. For each option shown, the corresponding total result, its rank, its index, its radar chart respectively bar chart as well as the run, selected values for variables and the selected weightings of criteria will be displayed additionally for a better orientation in the design space and an immediate connection with the result.

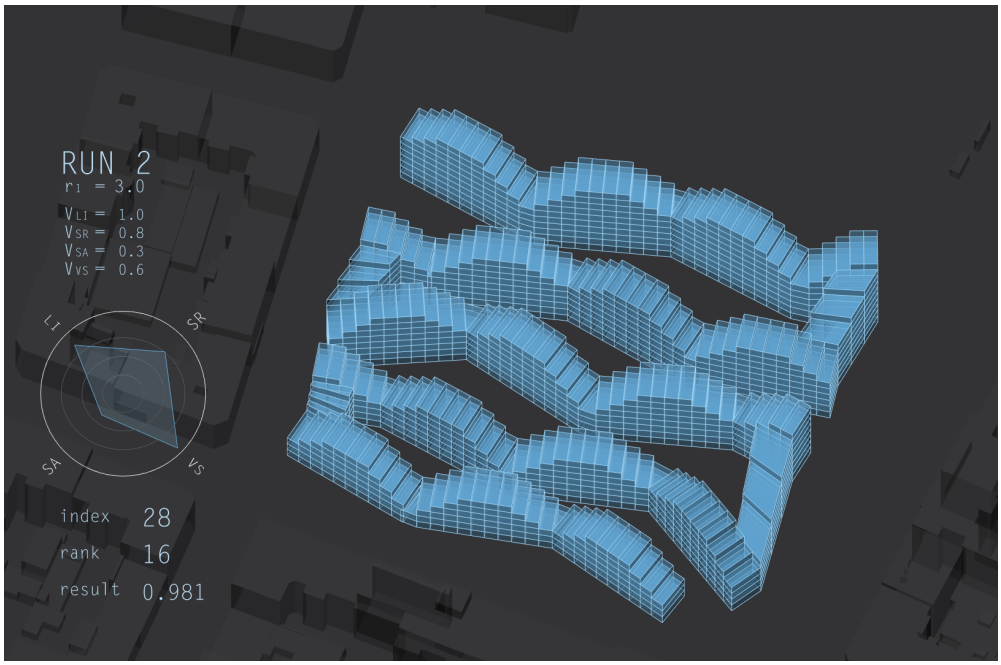


Figure 38: Display of geometry in 3d view

4. Graphical results of criteria on the geometry

   For an even more detailed view on how the design options behave concerning the four criteria, the performance can be displayed graphically on the 3d model. Those graphics will visualize the performance behaviour of different parts of the model. They will enable the designer to determine critical parts of the design or suggest locations that could be used for windows e.g.

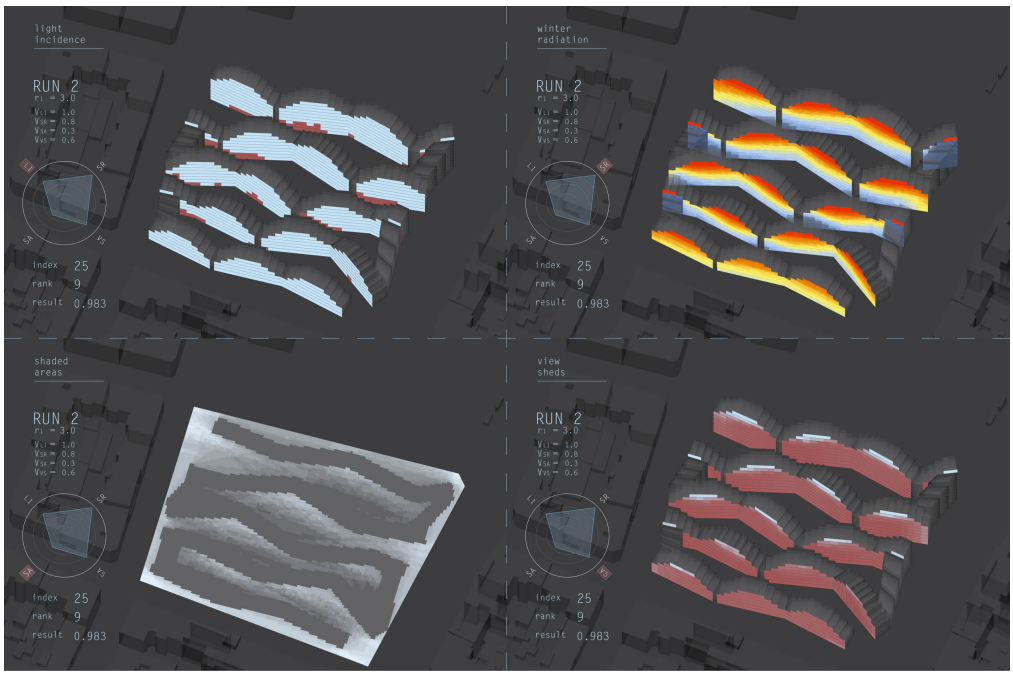


Figure 39: Visual performance illustration

3.5.3 Design Space Evaluation and Level of Detail

Once the design options have been evaluated, the user might be either satisfied with the result and pick one option, or there might be the desire for further steps to optimize the design with modified variables or at a smaller scale.

In case of modified variables, detailed information for specific locations in the design given by the visualization tool is able to point out critical parts of the design. By adjusting the variables accordingly, an improved result can be obtained in a further optimization run. For that purpose it is important, that all settings of variables for a specific optimization run are noted down, saved and linked to the corresponding set of data that will be produced, in order to being able to go back to previously created options.

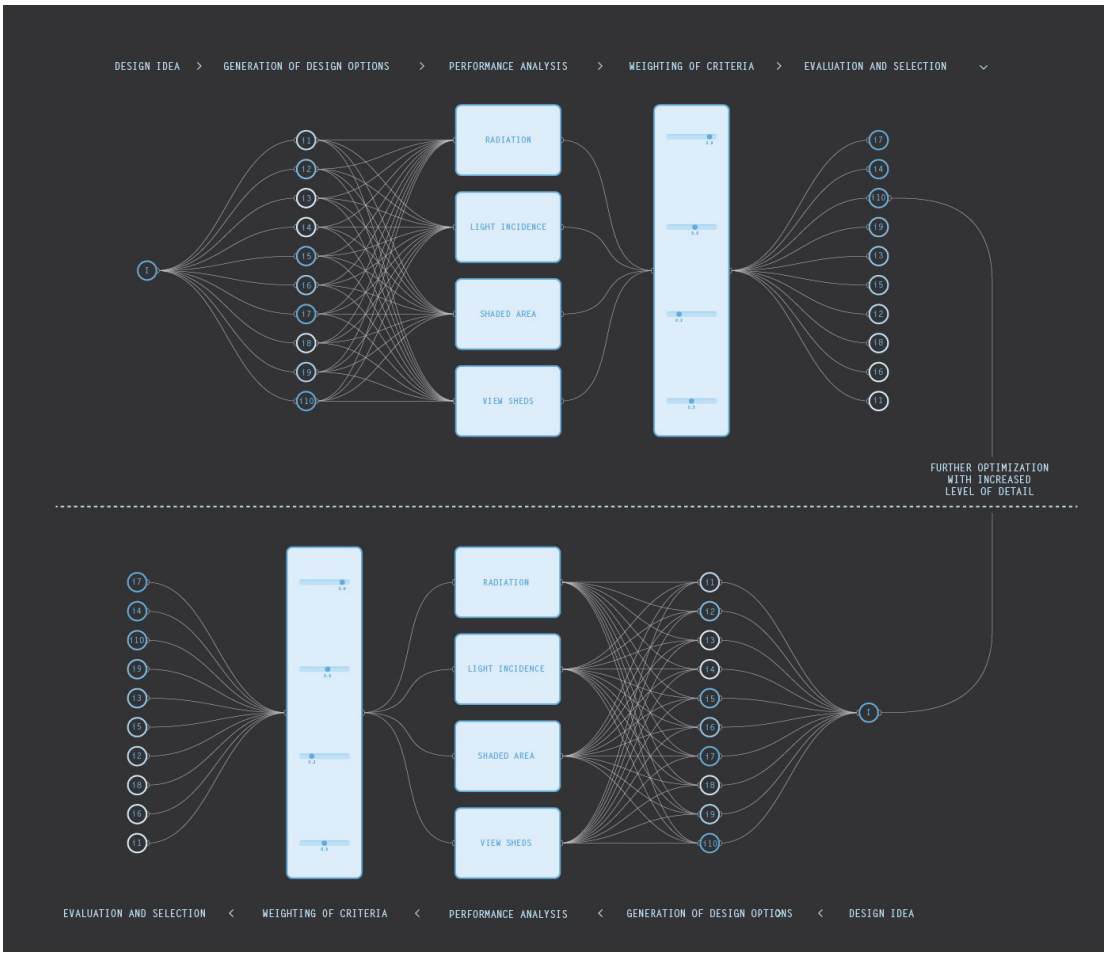


Figure 40: Optimization approach with increased level of detail (LOD)

Once a satisfying solution is found, the user is also able to increase the level of detail. In Figure 40 the user picked an option of the first run, which was focusing on a rough massing of the design. Now the chosen massing is modelled parametrically in more detail, and the process of optimization can be repeated as described before. Also for that case it is important to note and save settings and options in order to being able to go back. That process can be repeated endlessly, the limiting variable though in that case is above all time.

4 Case studies

In order to test the BPET, it will be applied on three different projects, which all have a high potential for optimization, as many spatially equivalent alternatives are possible, but the corresponding behaviours concerning light and views are very complex. In order to test the influence of different climate conditions as well, two very different locations have been selected, two projects are located in Vienna, Austria, and the third one is in Singapore.

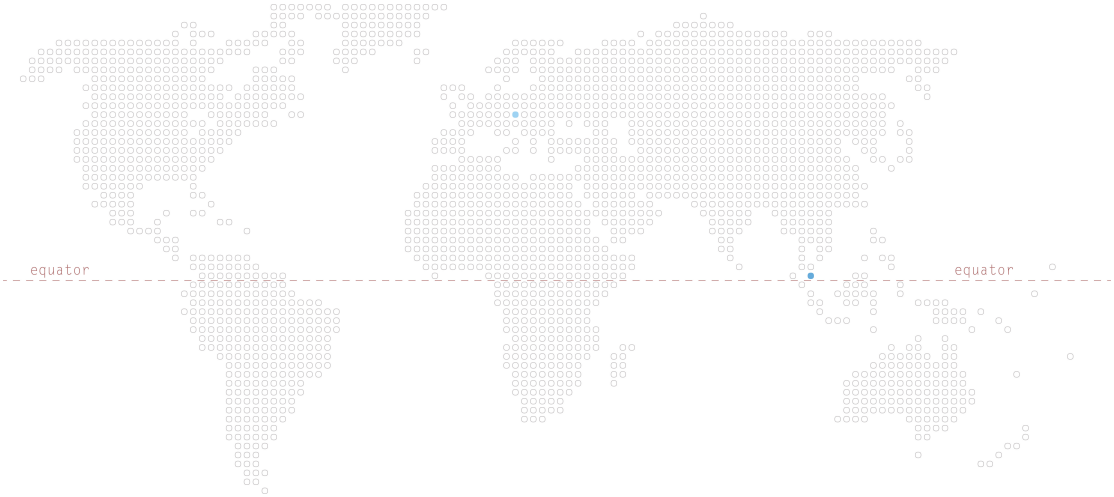


Figure 41: World map showing Vienna and Singapore

Vienna is located in the so-called *temperate climate zone* (webquest.hawaii.edu, 2017). The climate in Vienna is characterized as transition climate, which has maritime influences from the west and continental influences from the east. The continental influences cause cold winters and hot summers (wien.gv.at, 2017). As described in chapter 3.4.1.1, Vienna has a cooling as well as a heating period, though the majority of the room conditioning energy demand has to be spent on heating. The heating period is from October to April, the cooling period from June to August.

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
568.2	466.5	391.8	230.4	0.0	10.5	74.4	56.7	0.0	241.5	402.3	532.0

Table 3: Monthly HHDs (red) and CDDs (blue) for Vienna, Austria

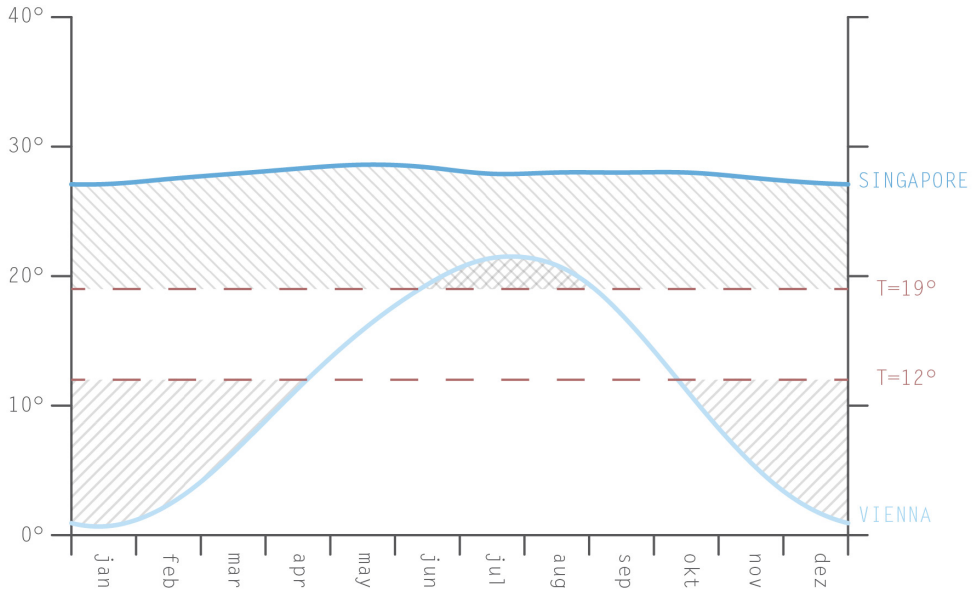


Figure 42: Mean temperatures for Vienna and Singapore including HDD and CDD

When looking at the diagram of the two locations in Figure 42 the drastic difference between temperatures in Vienna and Singapore becomes obvious. Singapore, which is located at the sea and close the equator, shows the characteristics of a tropical climate region, which is high temperatures and no seasons (webquest.hawaii.edu, 2017). Singapore has a high cooling demand throughout the whole year, which is shown by the hatched areas in Figure 42, indicating the HDD or the CDD, respectively. This means that for Singapore, the radiation amount for the whole year has to be calculated, options with a lower radiation are to prefer. Although in this case they are not really relevant, Table 4 is showing CDDs for Singapore.

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
251.1	238.0	275.9	279.0	297.6	282.0	275.9	279.0	270.0	270.0	258.0	254.2

Table 4: CDDs for Singapore (sdwebx.worldbank.org, 2017)

Two of the presented projects are already built, which is why the approach of this tool application is to assume being in an early design stage and to respect all the project specific characteristics, so that similar and wholesome design options to the built one can be generated.

4.1 up and down by Ivan Matas and Fabian Hübner

The project was done by Ivan Matas and the author within the framework of a design studio at the Technical University of Vienna in 2015. The task was to design a mixed-use hybrid building containing residential use as well as other functions defined by the designer and to provide a high amount of private outdoor spaces such as terraces and balconies. A high density was to be achieved by a mainly horizontal distribution of the buildings on a roughly 9,000 m^2 site, located in the 10th Viennese district (see Figure 43).

Figure 43: South view and site plan

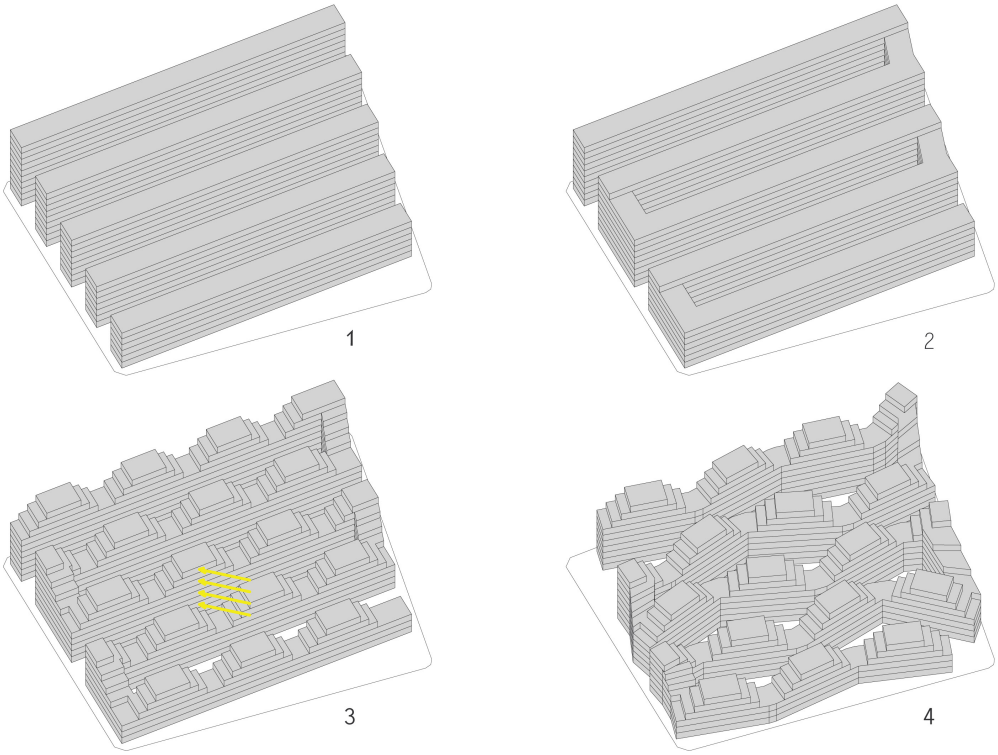


Figure 44: The design concept

The approach in the first step was to fill the site with five rows, each of them increasing its height for one floor compared to the previous one (see Figure 44). As all of the rows were facing south and the height increases from south to north, they were designed to receive a maximum of possible sunlight from the south (1). By connecting the rows alternately at east and west, one continuous line was created (2). In the next step, the height of the rows was increased and decreased in a way, that there are three valleys and three peaks in each row and hence, terraces could be created. The peaks were shifted in every second row, so that a valley in the first row was followed by a peak in the second row and so on, in order to improve sun incidence and views to the south (3). As the rows were all straight and parallel to each other, the decision was to rotate every section (consisting of a peak in the middle and valleys at each side), so that view relations could be improved and the public spaces in between the rows become more dynamic (4). That decision made us realize that the way how to rotate the segments was actually a very complex problem, as every rotation influenced the shape of all adjacent yards and the distance to rows in front and behind the segment. Furthermore, there were no indications from the environment or any other reasons that made a specific constellation more plausible than others. So we came up with the idea to generate a high number of design options, compare them with each other and choose one. And as light incidence is a very critical issue in high-density settlements, the generated options were also analysed for light incidence by the algorithm. The foundation of this thesis was born.

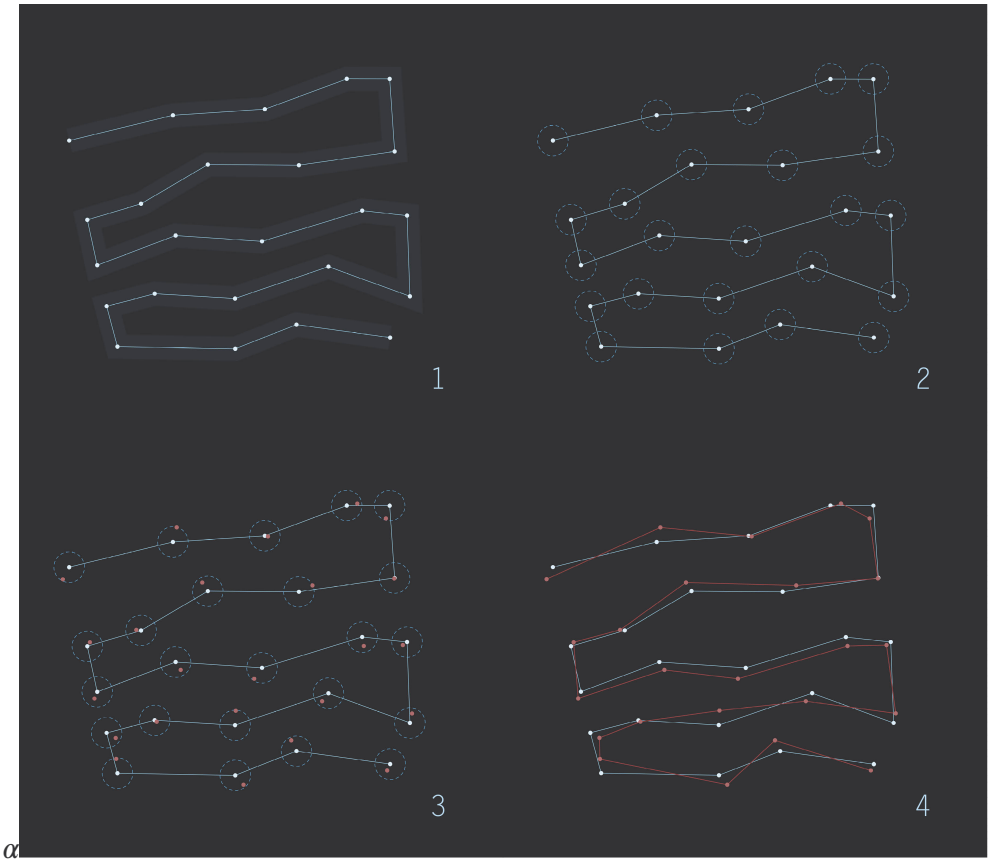


Figure 45: Parameterization of the design approach

The parametric model for the design separates the segments in order to modify their position as a whole. The modification is actually based on a polyline that is connecting 24 manually set points and that were considered as a good starting point for design alternatives (see Figure 45). The polyline is following the shape of the initial design in floor plan (1). Taking the 24 points as a centre, a radius *r* is defined within which newly generated points can be potentially placed (2). New points were created, by moving the initial points for a random distance *d* within the range $0 < d < r$, and a direction rotated randomly around the initial points with the angle α which is part of the range $0 < \alpha < 360$ (3). By connecting those new points, a new direction for the segments is set (4) and a new design option created. The dimensions of the segments remain unchanged by the process. By changing the seed value for the random movement of the points, new constellations can be created by the algorithm, controlled by the constraint of radius *r*. That procedure assures, that the design ideas mentioned before remain untouched by the algorithm while only the rotations of the segments can be modified by it. That way, we as designers do not lose control over our design, instead it enables us to test many options in a very complex situation and moreover, to optimize them concerning their performance.

Considering the guidelines for passive solar architecture, maximum window openings were planned for the south facades, windows to all other directions were to avoid for solar radiation reasons but also for reasons of maximal intimacy in that very dense arrangement. So for the analysis of solar radiation as well as light incidence and view sheds the south facades are considered only. For shading, the public areas of the site are object of analysis.

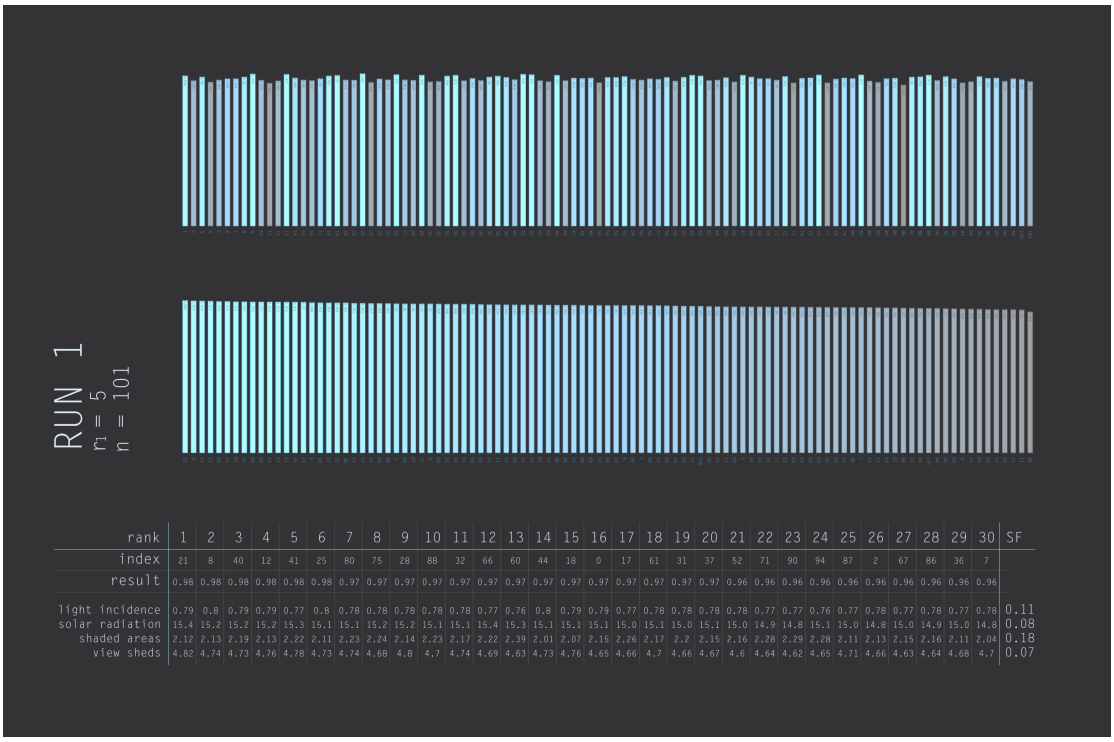
In the first run of optimization, the following settings are defined:

Inputs run 1

amount n	=	101
radius r	=	5.00 m
$V_{\text{light incidence}}$	=	1.00
$V_{\text{solar radiation}}$	=	0.80
$V_{\text{shaded areas}}$	=	0.30
$V_{\text{view sheds}}$	=	0.60

V is the weighting of criteria. For weighting, first priority was given to light incidence, as it is very important to supply the flats with a sufficient amount of light in that dense constellation. The more spaces can be used as adequate rooms, the more efficient is the design and the lower can be the price for each flat. Also solar radiation is stressed in the evaluation, as it can assure high energy efficiency. View sheds and shaded areas are considered less important for the design. In the first run, the amount *n* of calculated options was only 101

in order to have a quick test of results. It is 101 instead of 100, because lists in *Grasshopper* always start from 0, so the range from 0 to 100 contains 101 numbers. The calculation time was around 350 minutes.



	rank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	SF
	index	21	8	40	12	41	25	80	75	28	88	32	66	60	44	18	0	17	61	31	37	52	71	90	94	87	2	67	86	36	7	
	result	0.98	0.98	0.98	0.98	0.98	0.98	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	
light incidence		0.79	0.8	0.79	0.79	0.77	0.8	0.78	0.78	0.78	0.78	0.78	0.77	0.76	0.8	0.79	0.79	0.77	0.78	0.78	0.78	0.78	0.77	0.77	0.76	0.77	0.78	0.77	0.78	0.77	0.78	0.11
solar radiation		15.4	15.2	15.2	15.2	15.3	15.1	15.1	15.2	15.2	15.1	15.1	15.4	15.3	15.1	15.1	15.1	15.0	15.1	15.0	15.1	15.0	14.9	14.8	15.1	15.0	14.8	15.0	14.9	15.0	14.8	0.08
shaded areas		2.12	2.13	2.19	2.13	2.22	2.11	2.23	2.24	2.14	2.23	2.17	2.22	2.39	2.01	2.07	2.15	2.26	2.17	2.2	2.15	2.16	2.28	2.29	2.28	2.11	2.13	2.15	2.16	2.11	2.04	0.18
view sheds		4.82	4.74	4.73	4.76	4.78	4.73	4.74	4.68	4.8	4.7	4.74	4.69	4.63	4.73	4.76	4.65	4.66	4.7	4.66	4.67	4.6	4.64	4.62	4.65	4.71	4.66	4.63	4.64	4.68	4.7	0.07

Figure 46: Performance chart run 1

Figure 46 shows the result of the first run in numbers. The SF value indicates a high variety of results for shaded areas. The lowest variety occurred for view sheds, meaning that its potential for optimization is the lowest. Even though the SF value is the highest for the criterion that was weighted the lowest, the weighting of the criteria will remain unchanged, as the most important criteria for the design (light incidence and solar radiation) still shows a good significance.

The values themselves do not have a very high explanatory power without looking at the corresponding geometry that has been produced and that can be viewed in 2d as well as in 3d. The best 30 results have been visualized and can be evaluated according to their spatial qualities as well as to their performance (see Figure 47).

Although those are the best options performance wise, some problems can be found when looking at the floor plan qualities. Spatially, a balanced distribution of useable, not too narrow yards and a smooth continuous zigzag shape of the massing are desirable, constellations as shown in Figure 48 are unsatisfactory.

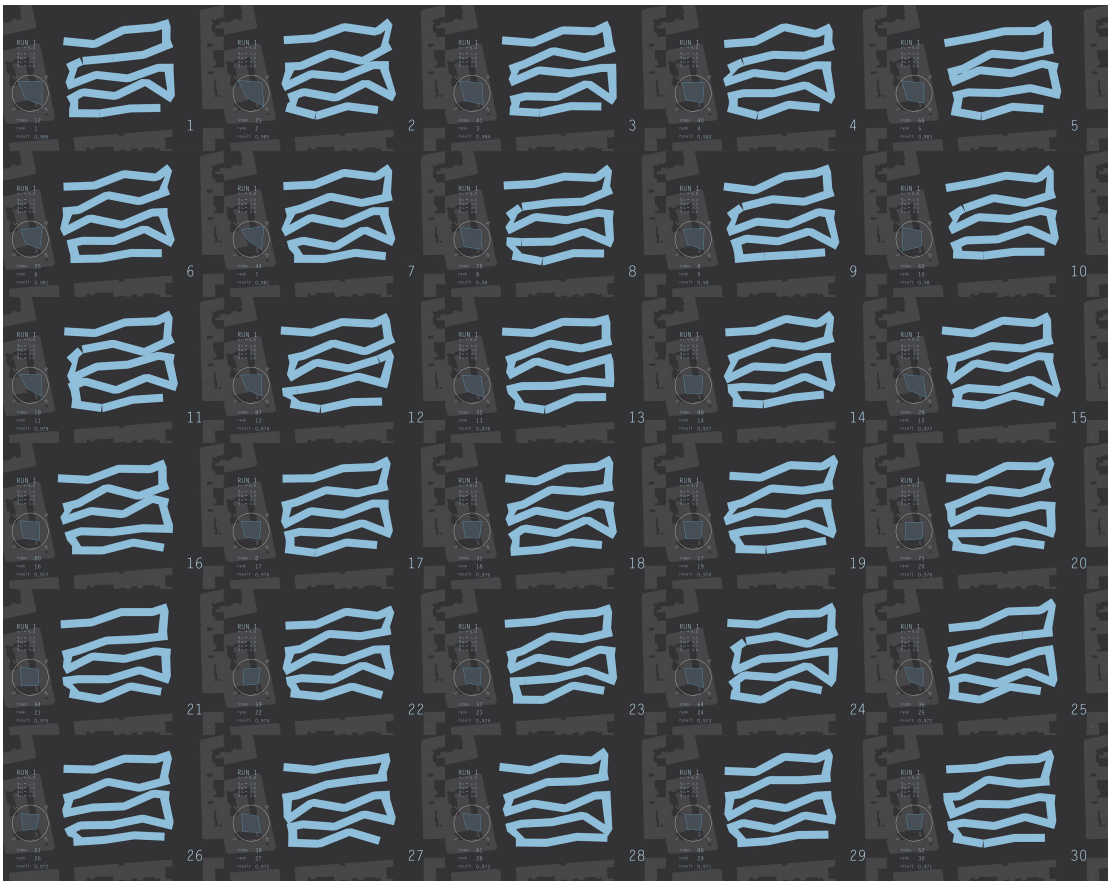


Figure 47: Floor plan options run 1, top 30

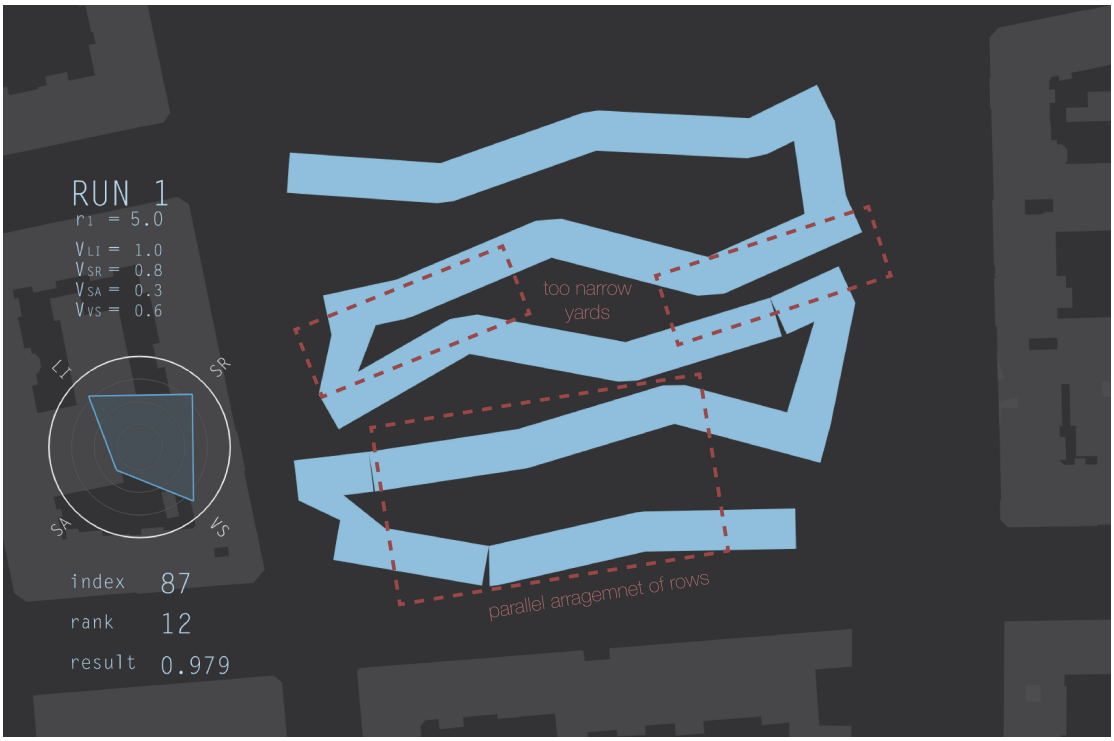


Figure 48: Unwanted constellations

Eventually, the results of run 1 are not satisfactory, the value of 5.0 m, which has been chosen for radius *r* is probably to high, so a second run with a lower radius has to be executed. The weighting of the criteria as well as the amount *n* remain unchanged, so the following inputs are set, returning the results shown in Figure 49 and Figure 50

Inputs run 2

amount n	=	101
radius r	=	3.00 m
$V_{\text{light incidence}}$	=	1.00
$V_{\text{solar radiation}}$	=	0.80
$V_{\text{shaded areas}}$	=	0.30
$V_{\text{view sheds}}$	=	0.60

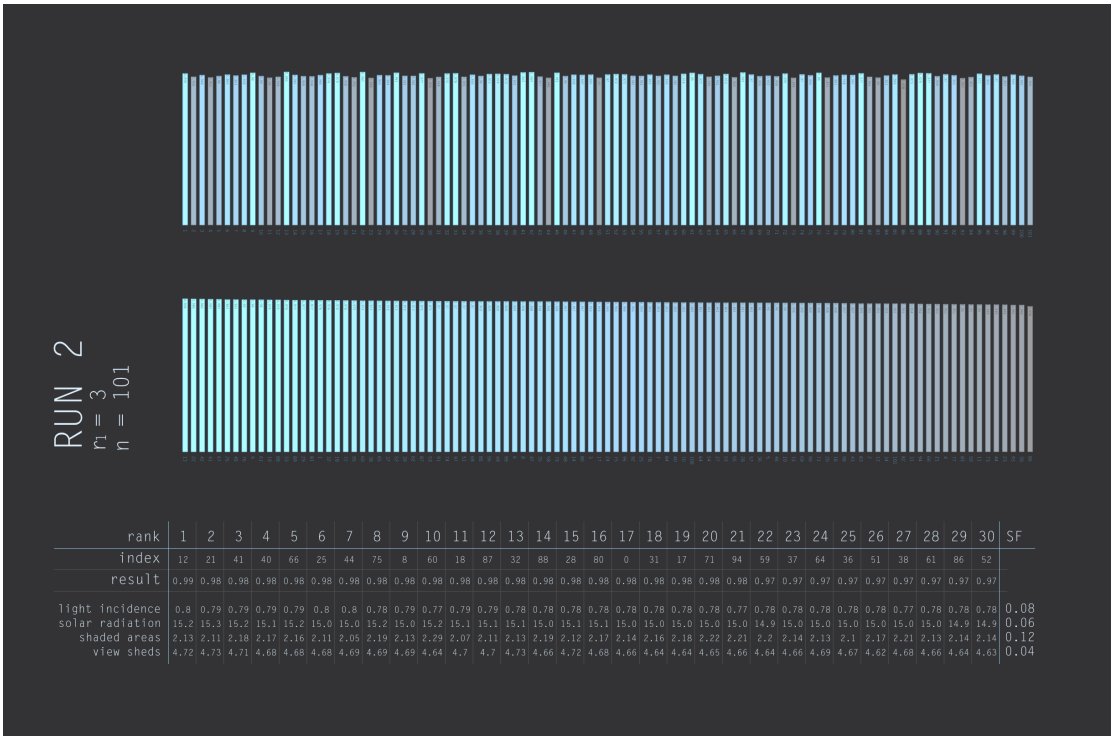


Figure 49: Performance chart run 2

The options created in run 2 are more similar to each other because of the decreased radius, which also results in lower SF values. But from a spatial point of view, the results can be regarded as superior compared to those of run 1. The option with index 25 (ranked 6th) however seems to perform best considering spatial as well as performance criteria (see Figure 51), and is therefore inspected more precisely.

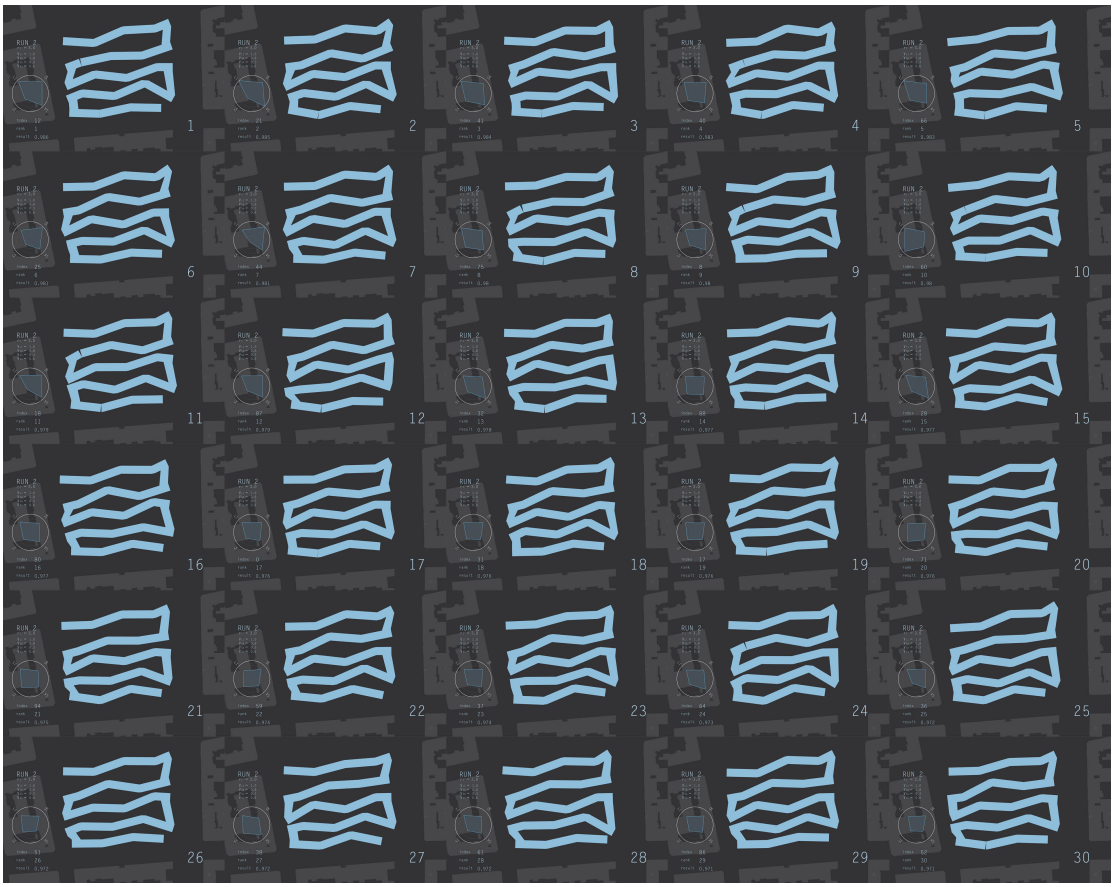


Figure 50: Floor plan options run 2, top 30

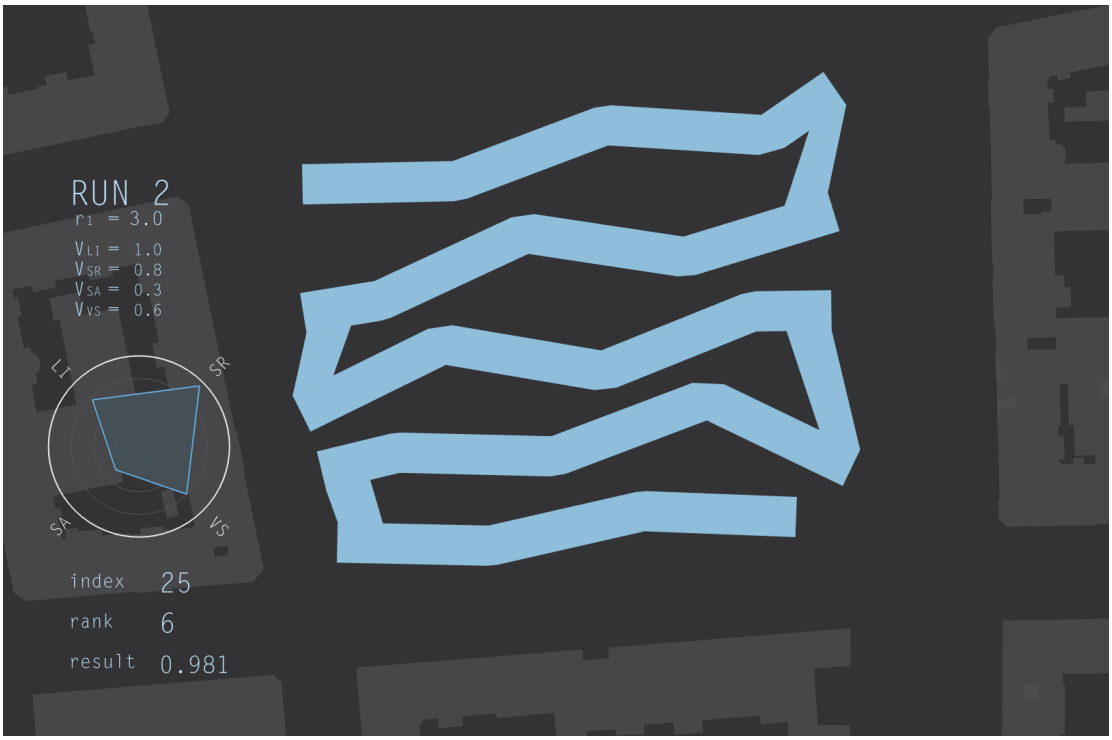


Figure 51: Run 2, option ranked 6th, floor plan

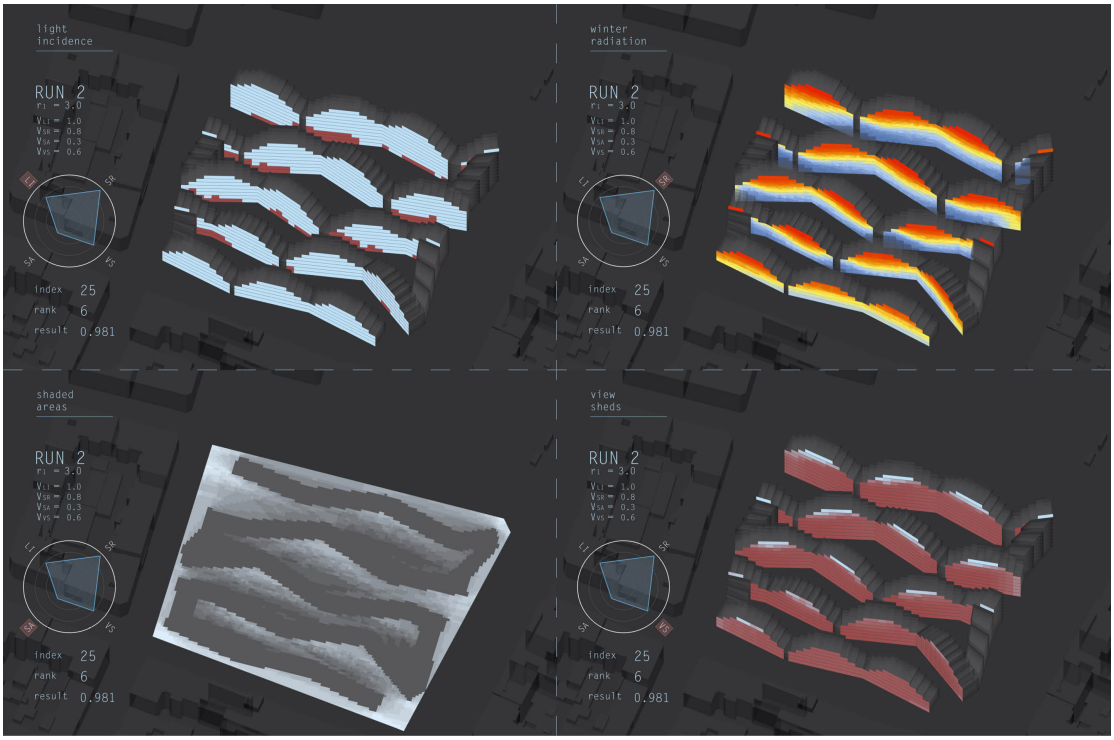


Figure 52: Visual performance illustration for option ranked 6th in run 2

Especially when looking at the light incidence (see Figure 52) one can see the critical areas of option 25, which do not receive enough light, yet. Also the winter case radiation, which is supposed to be the higher the better, indicates that narrow parts of the design do not receive enough radiation. So in order to continue with an advanced optimization, the generated points, which define index 25, are *baked* to *Rhino* geometry, so that they can be moved manually. That way, I as a designer can interfere in the optimization process and improve the design according to the information provided by the performance illustrations. The result of the manual improvement is shown in Figure 53, which now is taken as starting point for the main optimisation run. For that major run, 1001 options are generated with the radius r = 3.00 m, as that radius already produced satisfying results in run 2. The calculation time of run 3 was approximately 3,500 minutes, which is a bit more than 58 hours.

Inputs run 3

amount n	=	1001
radius r	=	3.00 m
$V_{\text{light incidence}}$	=	1.00
$V_{\text{solar radiation}}$	=	0.80
$V_{\text{shaded areas}}$	=	0.30
$V_{\text{view sheds}}$	=	0.60

Figure 53: Floor plan after manual manipulation

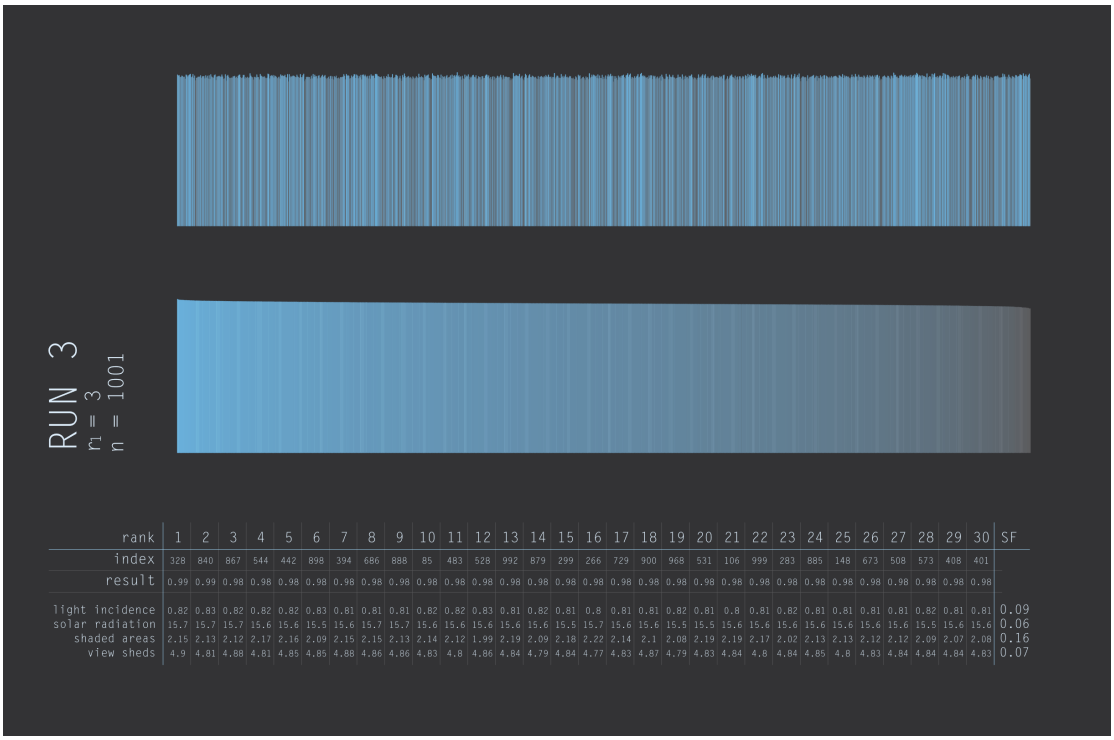


Figure 54: Performance chart run 3

Although the radius *r* is the same in run 2 and run 3, the significance values of run 3 are higher, not only because the points of the starting option were changed, but also because higher amounts of options usually will result in higher SFs, as more extreme values can be found. Also the specific results for the four

criteria improved compared to run 3, which indicates a successful run. Spatially, the run produced a lot of suitable floor plans. In my option as designer, option indexed 879 (ranked 14th) is the most successful compromise between very good floor plan behaviour and excellent performance (see Figure 55). The shape is smooth and the sizes of the yards are comparably large, the segments are rotated in a way, that most of them are not parallel to segments of neighbouring rows. Even though I actually wanted to avoid a rather straight row like the second row in option 879, by seeing the result and the relation of that straight row to the other rows, I changed my opinion and think, that it even makes the design more harmonic, which is certainly an entirely subjective perception (see Figure 56). But this case demonstrates the strength of generating options with a randomized approach, as it broadens the designer's mind of possible solutions by also creating unintended options.

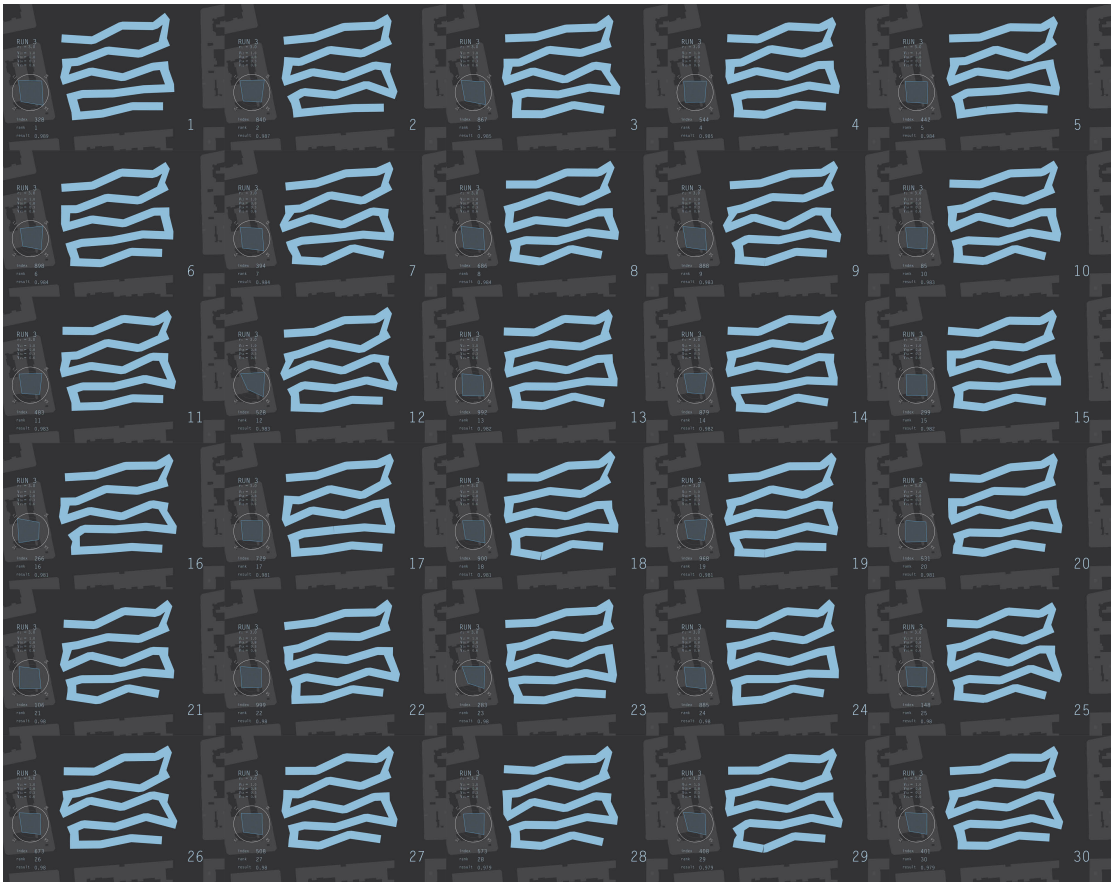


Figure 55: Floor plan options run 3, top 30

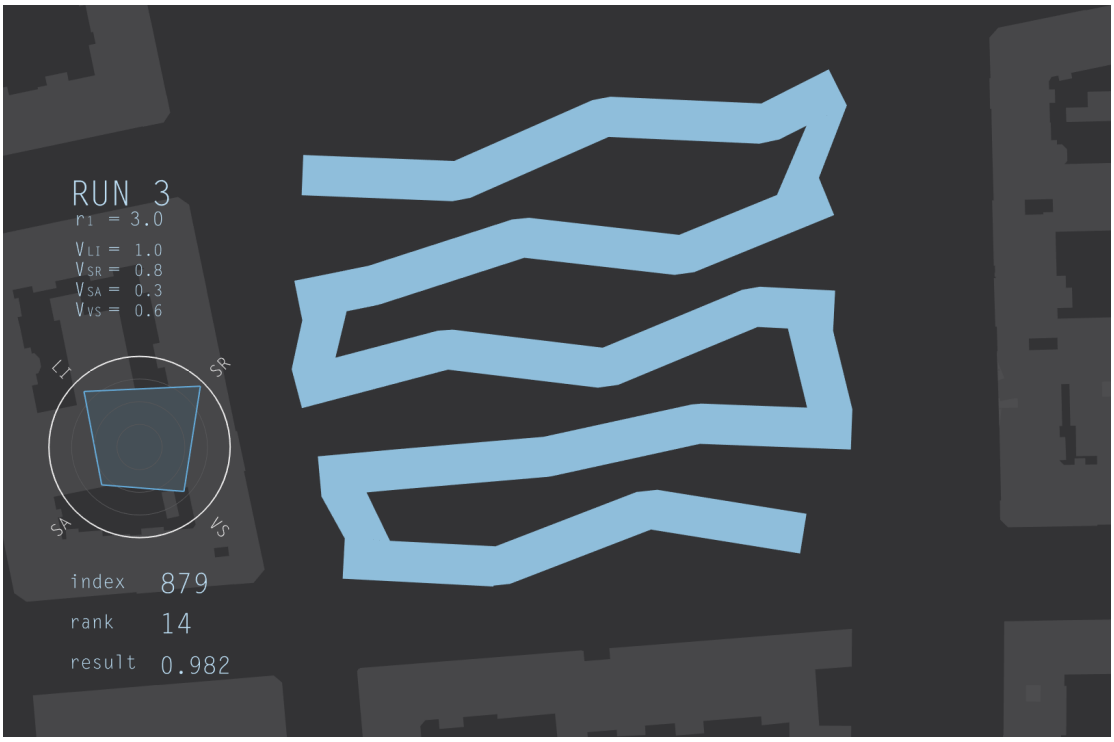


Figure 56: Run 3, Chosen option ranked 14th, floor plan

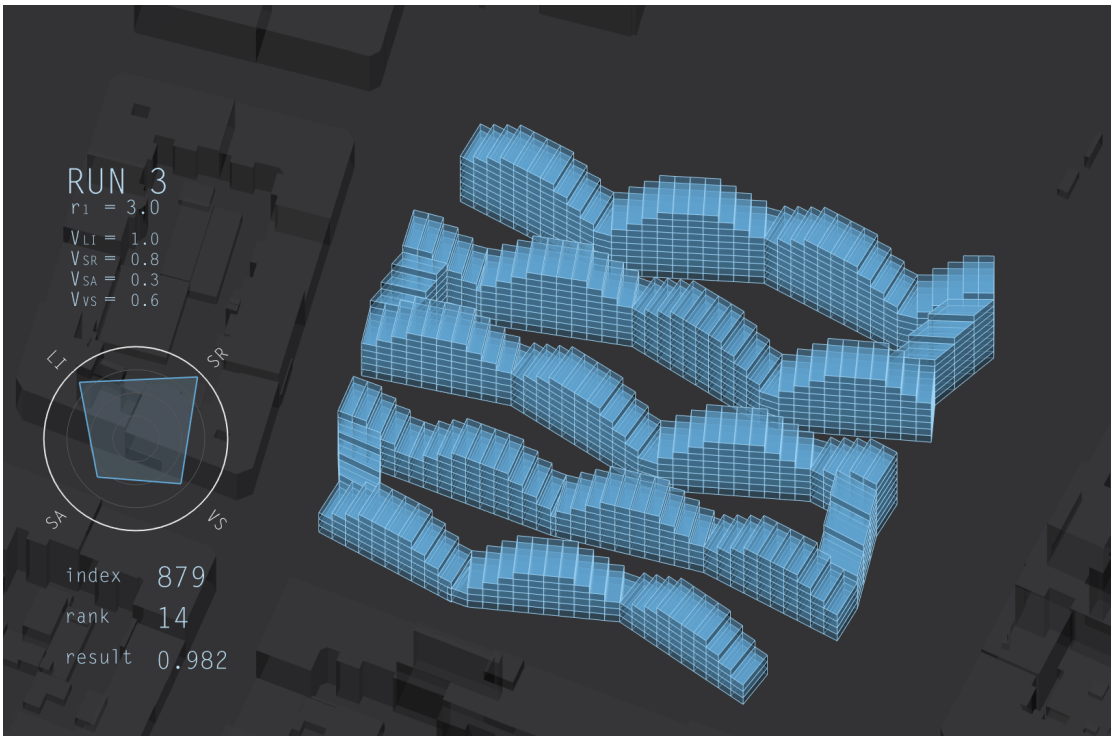


Figure 57: Run 3, chosen option ranked 14th, axonometric view

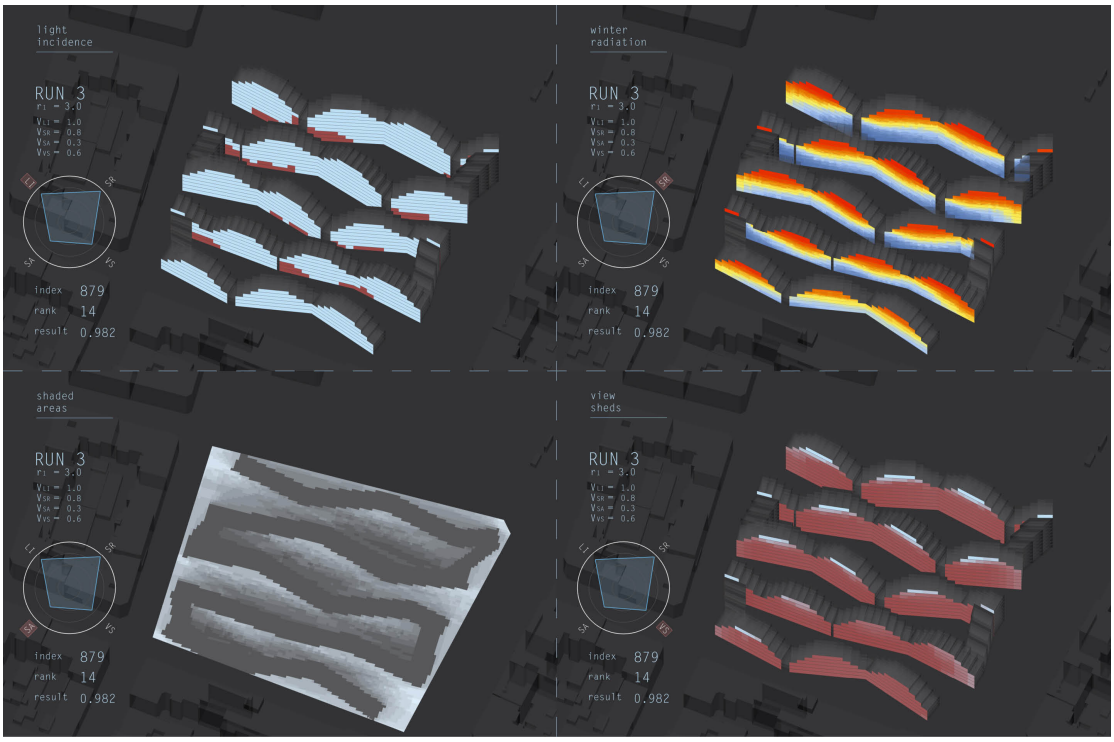


Figure 58: Visual performance illustration for option ranked 14th in run 3

Obviously, further runs could be used to further optimize the result, but as time is limited in design processes, and the optimization of massing already took quite long, option 879 is chosen for the further workflow. The optimization helped to provide a superior massing layout and improved its performance compared to the initial design, which was used as starting point for optimization. For the total result, the chosen option shows a 3.72% improvement compared to the initial design, its light incidence is improved by 5.46% and solar radiation by 4.12% (see Figure 59). Interestingly, the initial option performs better in shaded areas, but as shaded areas were weighted lowest, their influence on the final score is comparably low.

With around 3-5%, the improvements' quantity may seem little at the first glance, but one has to consider that those improvements are achieved simply by arranging the massing in an efficient way; no further energy measurements have been applied so far. From that point of view, 5% improvements obtained just by a superior arrangement of massing are quite remarkable. Though, one has to notice that those percentages cannot be taken too serious in their exact quantity at this early design stage, as a lot of other aspects in the further design process determine the final performance. But those figures show options that have the potential for the best performance in the end and are very important, as massing is hard to change later on, but has a high influence on the performance.

improvment to	initial option	worst option
total result	3.72%	5.29%
light incidence	5.46%	6.92%
solar radiation	4.12%	5.59%
shaded areas	-2.07%	3.95%
view sheds	3.33%	3.2%

Figure 59: Improvements run 3

As the facades in the early stages were considered flat for the definition of massing, they do not contain any exterior spaces like balconies or terraces. But as mentioned in the design task description, one of the design goals was to create those exterior spaces. So in order to fulfil those goals and to make the façade more interesting, the idea was to have jutties and recesses in the building envelope (see Figure 60). In the first step, when the façade was flat, the general depth of the segments was 8.0m. Increasing or decreasing the depth for every unit independently can create a result similar to the design sketch. The chosen minimal depth is defined by 7.0m, the maximal depth by 9.0m. In order to make sure that jutties have a minimal depth that can still be used for outdoor spaces on top of the jutty, only three possible depths of the units are allowed, which are 7.0m, 8.0m and 9.0m. As the back of the unit remains at its initial position and hence the back façade remains flat, two possible depths of balconies or terraces can be obtained, which are 1.0m and 2.0m (see Figure 61). Unusable jutties of just 15 cm for example are avoided that way.

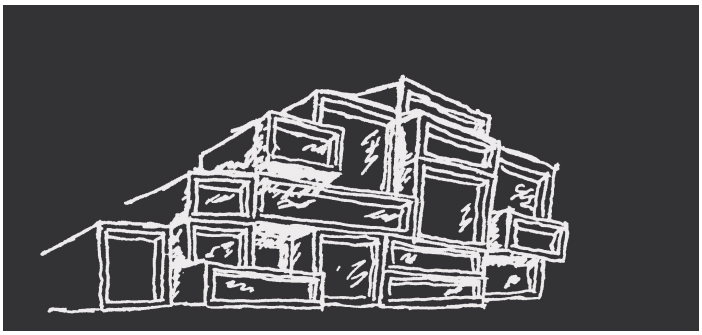


Figure 60: Conceptual hand sketch of facade

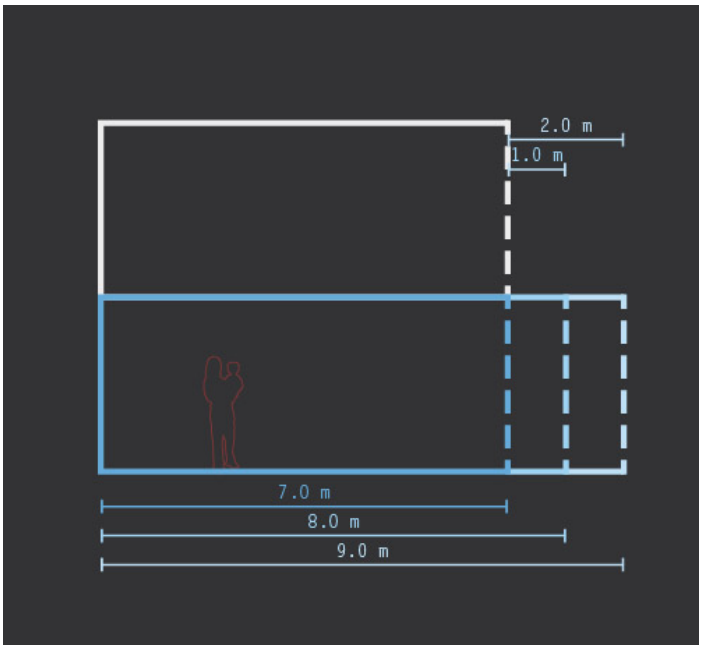


Figure 61: Different depth of jutties

As there are plenty of units and hence a very high number of possibilities how to organize the depths of jutties, a manual way of designing those jutties and recesses is very complex and time consuming. Moreover, the manipulation will affect the light incidence and the radiation hitting the windows. So again, this design task is predestined for algorithmic optimization. As shading and views are not so much affected by the design change, this optimization run will only analyse light incidence and solar radiation. The amount n of options will be 201, as the options are quite similar to each other and thus, not a very high amount of options is needed. As a continuation of the design process, the massing will be obtained from the chosen option of run 3 (indexed 879). The calculation time takes around 660 minutes.

Inputs run 4

amount n	=	201
J	=	7.00 m ; 8.00m ; 9.00m
$V_{light\ incidence}$	=	1.00
$V_{solar\ radiation}$	=	0.80

J is the possible depth of jutties, it can take only three different values as described before. By randomly changing those depths, different constellations of jutties can be created by the algorithm. The weighting of criteria still remains the same as in the runs before.

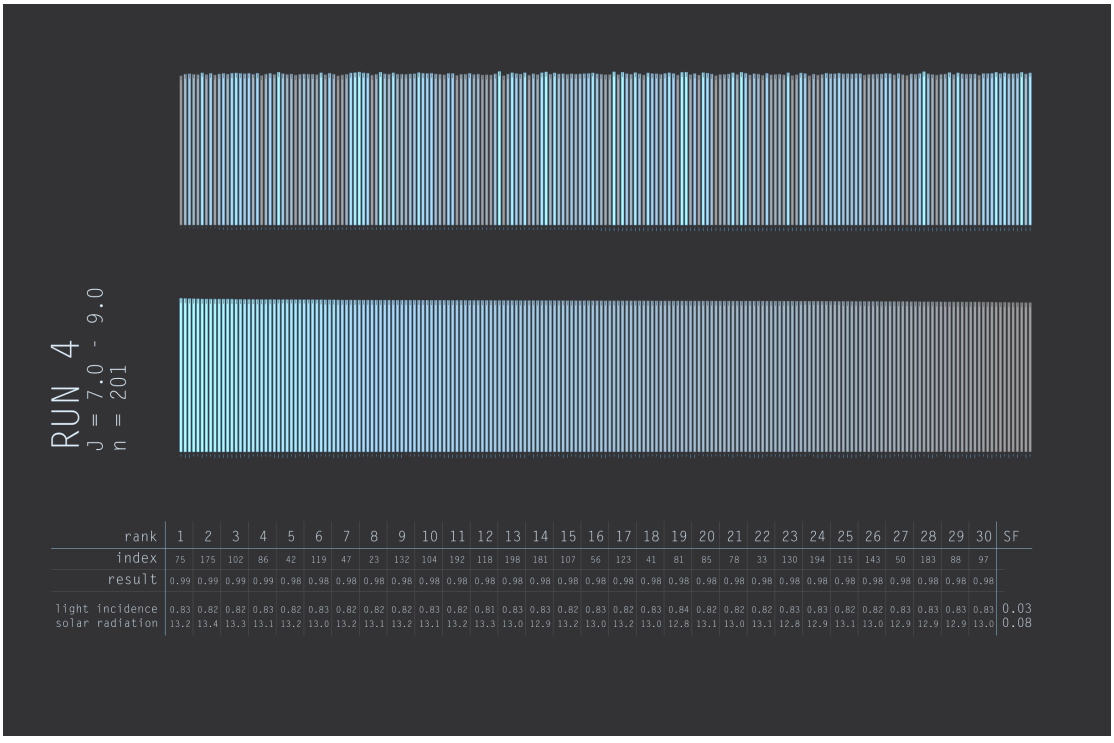


Figure 62: Performance chart run 4

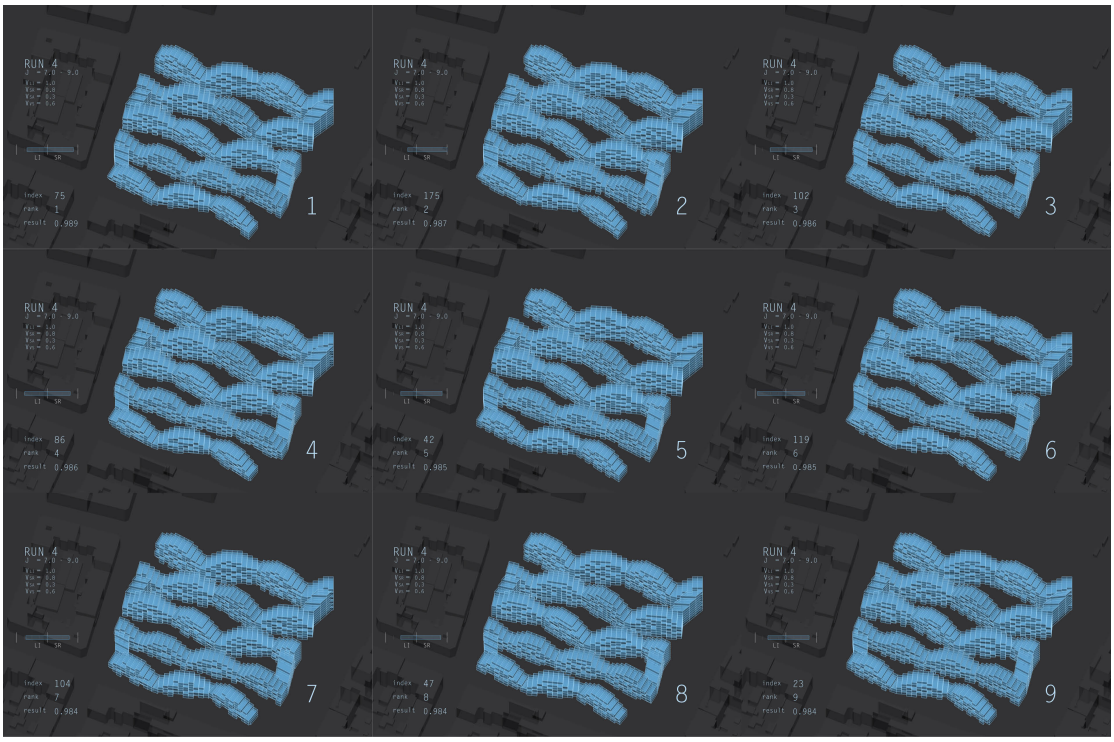


Figure 63: Jutty options run 4, top 9

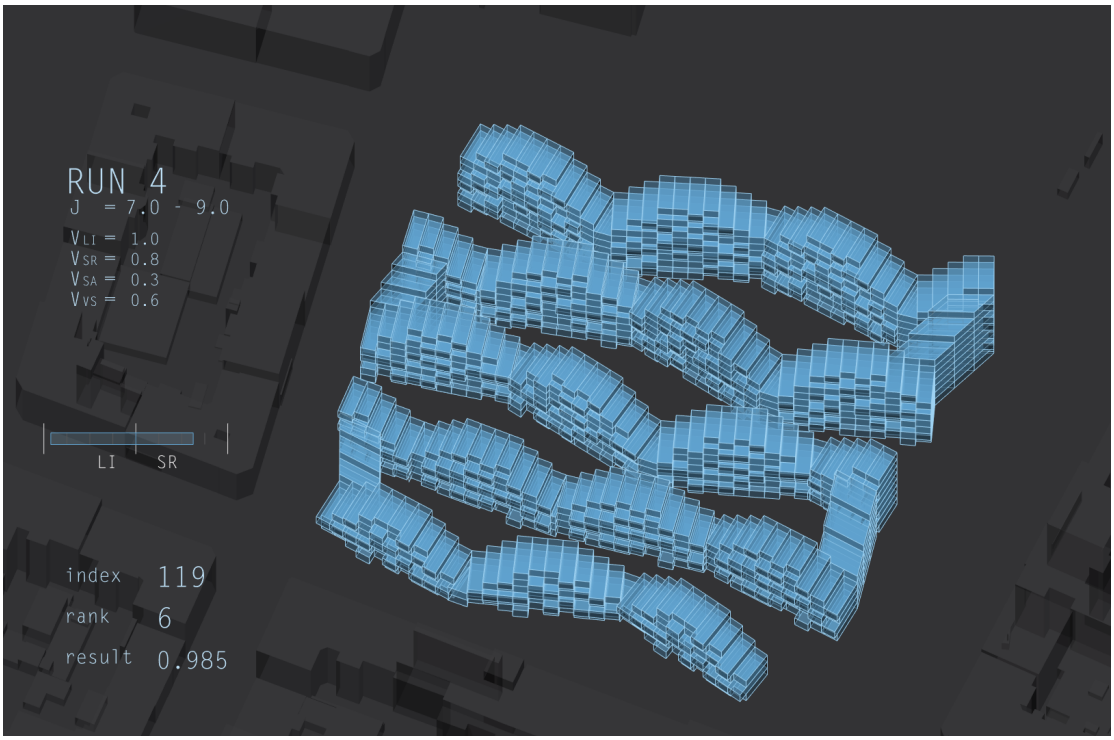


Figure 64: Run 4, chosen option ranked 6th, axonometric view

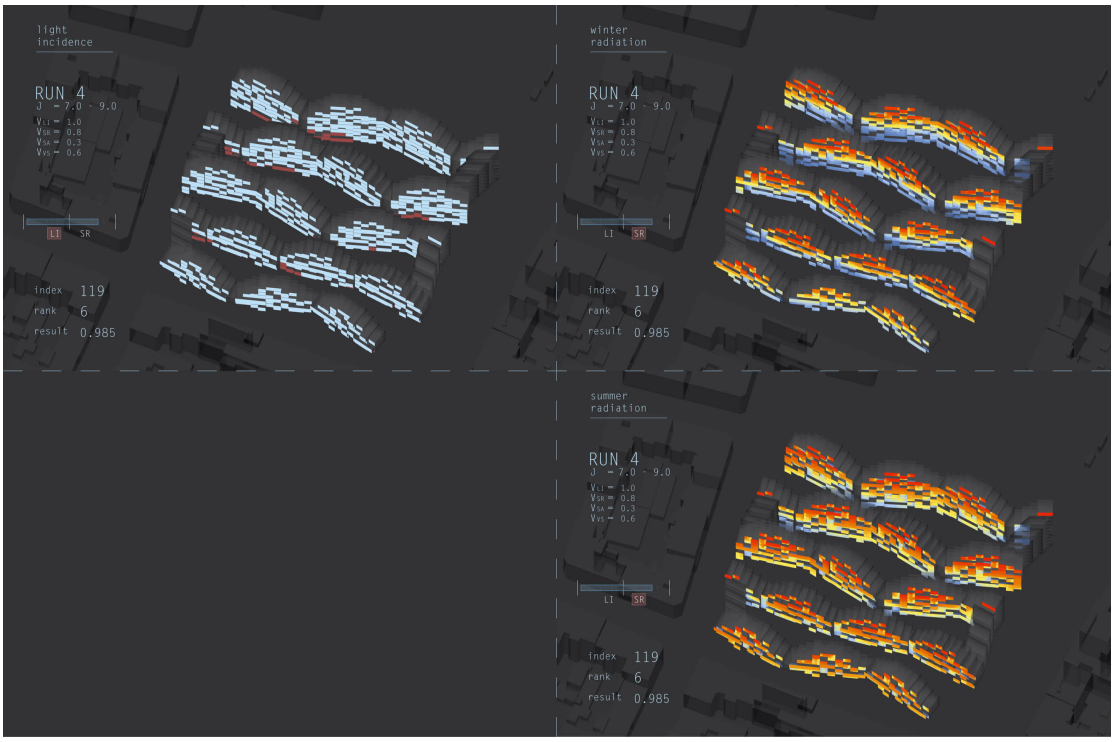


Figure 65: Visual performance illustration for option ranked 6th in run 4

What is striking when looking at the results in Figure 62 and especially the radar charts in Figure 63 is that there are no options, which have a top result in both criteria. So those which show the best result in one criterion only are not amongst the top total results, which indicates a negative correlation between the two criteria. For aesthetical reasons in terms of a balanced distribution of jutties and recesses, my choice is option 119, which is ranked 6th (see Figure 64). That option shows very good results in terms of light incidence, but it is lacking a top result for solar radiation. Nevertheless, in my opinion that option is performing best considering its overall performance.

Comparing that option to the chosen option from run 3 with the flat façade, the jutty option was able to further improve the light incidence by 1.08%. However, the performance for solar radiation was diminished by 8.33%. That loss of performance is not a specific problem of the chosen option, but all the options of run 4 perform much worse than those with the flat façade of run 3, in average by 8.42%. This can be explained by the shading effect of the jutties on windows underneath them, which results in lower total radiation on the façade (see Figure 65). Before executing the simulation, I expected the opposite, as I thought that the jutties would reduce the radiation in summer a lot, but leave them at a similar level in winter, because of the low sun angle. So the total result (less summer radiation, but same winter radiation) should have been better. But obviously, the savings of the summer case cannot regain the losses in winter, as in Vienna the heating period has a much higher impact than the cooling period. In other, hotter locations though, the jutties could indeed have a positive effect on solar radiation performance. Also smaller jutties, which are adapted to the exact sun angle could have such an effect. That case shows, that theoretical knowledge

about passive solar architecture is basic for its application, but it is not enough without testing and simulating a specific case.

As a designer, I have to evaluate that situation and decide whether the loss of energy efficiency is worth changing or even dropping the design idea of jutties. Decreasing the size of the jutties could improve the radiation performance, but at the same time it will make the resulting balconies unusable, so that is not an option. Also cancelling the jutties is not an option, as the facades would become too boring, and moreover, the yards might even have an acoustic problem with two flat facades on both sides (simulating this could be subject of a future add-on to the BPET). So in the end, aesthetical and other functional reasons have to overrule the importance of solar radiation. The optimized chosen option in run 4 still has a satisfying solar radiation behaviour compared to other results of run 4. And moreover, by optimizing the massing, at least 4% energy consumption could have been potentially saved only by choosing that specific constellation.

At that point, the optimization process of the project can be considered as successful and it can be terminated. Obviously, further optimization would probably bring some little improvements. Going back to the massing stage and test different massing options with jutties instead of plain facades could be one reasonable approach. But the additional effort probably would not justify the little improvements that are still possible. In later design stages though, smaller details could become object of optimization, when decisions about materials or room sizes and positions are made.

4.2 slim city by ppag architects

The project *slim city* by *ppag architects* is located in a noteworthy urban planning project in the suburbs of Vienna, called *Seestadt Aspern*. As the name indicates, a 50,000 m^2 artificial lake constitutes the centre of the new settlement, which is still in its completion that stretches over several stages and years. Eventually, the *Seestadt Aspern* will be home of 20,000 inhabitants and working place for another 20,000 people. Unlike other common suburban settlements, the *Seestadt* tries to implement downtown characteristics such as high density and building heights, good infrastructure and a well-established public transport system (aspern-seestadt.at, 2017). Those characteristics make it special as suburban planning project and interesting for architects to experiment with new urbanism.

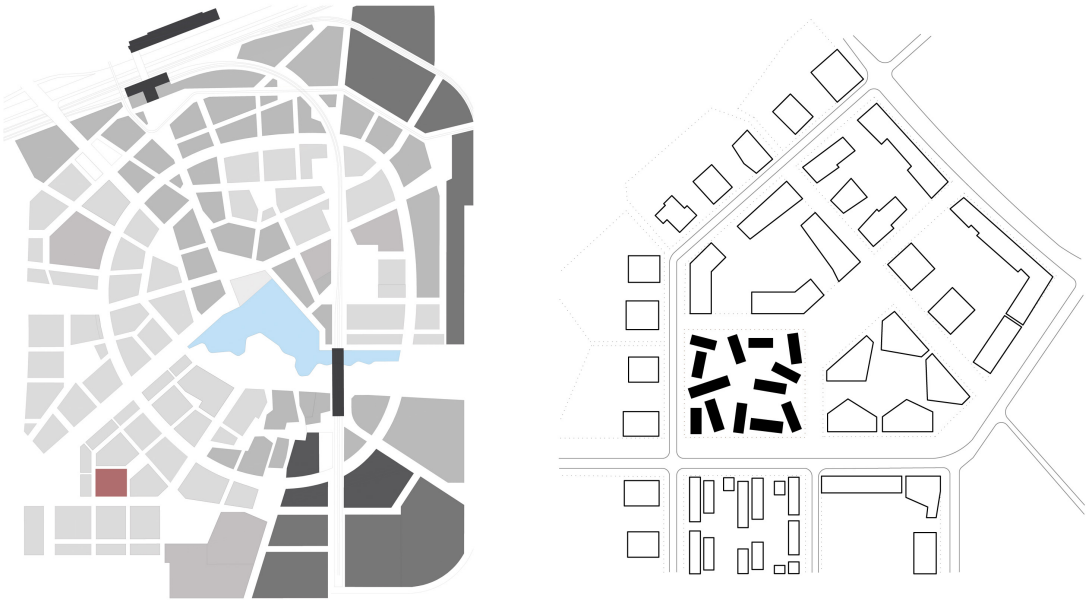


Figure 66: Site plans (ppag architects, 2015)

Figure 67: Slim city (ppag.at)

Figure 68: Slim city (derstandard.at)

One of the already completed projects is *slim city*, which can be regarded as a new approach to orient buildings in an environment independent from car traffic and strict ensemble regularities. The irregular distribution of buildings brings forth a sequence of differently sized squares, which can be used by public as well as private parties (ppag architects, 2015). As described in the introduction, this new approach of distributing buildings independent from almost any environmental context constitutes a new way of architectural freedom, but is accompanied by a new dimension of complexity as well. To handle that complex situation, *ppag* used an algorithm to define window positions and sizes according to light incidence and views.

For this thesis, the project is of high relevance, as it is a perfect example for reasonable use of algorithmic optimization, which helps to improve and justify the finally selected design option. So it is an excellent occasion to test the BPET on a real project. As the architects had diverse reasons to choose the final design option, which go beyond the portfolio of the BPET and are partly non tangible as well, the application of BPET on the *slim city* project is not aiming at criticising the architects' decisions, but testing the tool in a relevant project under real environmental conditions. Therefore, constraints have to be set which might not match the original constraints of the architects. The thesis will try to enter the design process in an early design stage, where the concept of a random appearance of building distribution, the number of buildings and the building heights are already defined. Later project characteristics such as balconies, accessibility and floor plans will not be considered in detail in this thesis.

For the optimization as usual, the first step is to translate the design into a parametric model. Therefore, the building blocks of the original *ppag* design are formulated as rectangles that cover the measurements of the 13 buildings in floor plan. Those rectangles are supposed to be randomly spread within the site to create design alternatives. In order to avoid unnecessarily long computation time, the first step tests options for their validity only in 2d without applying any of the BPET criteria for evaluation. Valid solutions are options, where the buildings neither intersect each other nor the site border. Figure 69 is showing the original *ppag* layout on the left (1), which is obviously a valid solution, and a solution on the right, which contains intersections of the buildings themselves and with the site border and is therefore invalid (2).

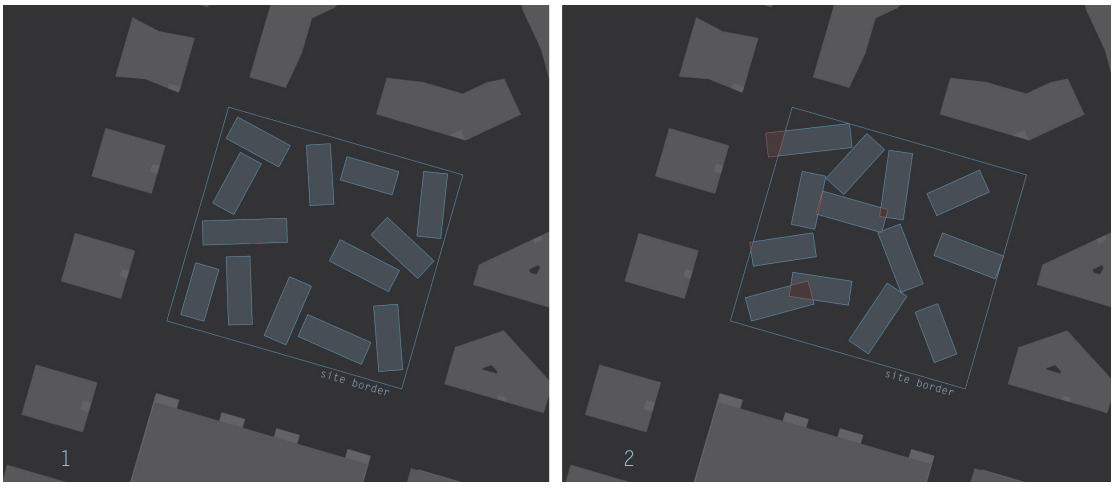


Figure 69: Valid and invalid solutions

In order to realize a completely random distribution, for each building a random point within the site as well as a random angle *a* (*0<a<360*) are defined, which constitute the building's position in a corresponding option. However, that method didn't show any success, as the calculation of 50,000 different options hasn't returned a single valid option. The probability for valid options seems to be too low with those loose constraints (see Figure 70).

Figure 70: Invalid solutions with random approach

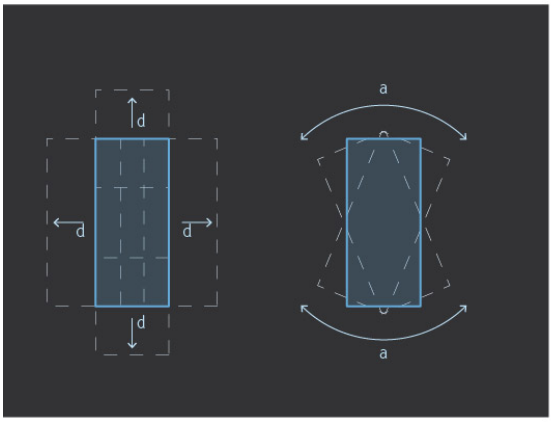


Figure 71: Constraints for movement

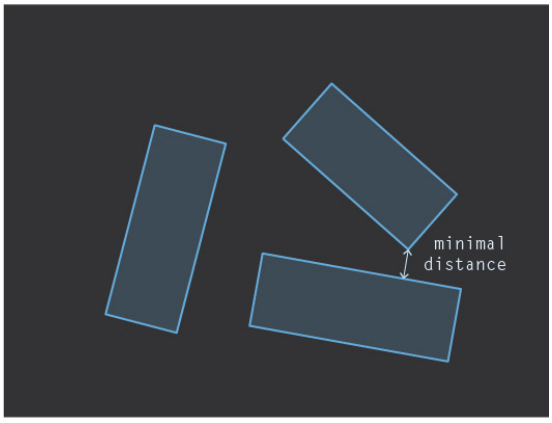


Figure 72: Minimal distance

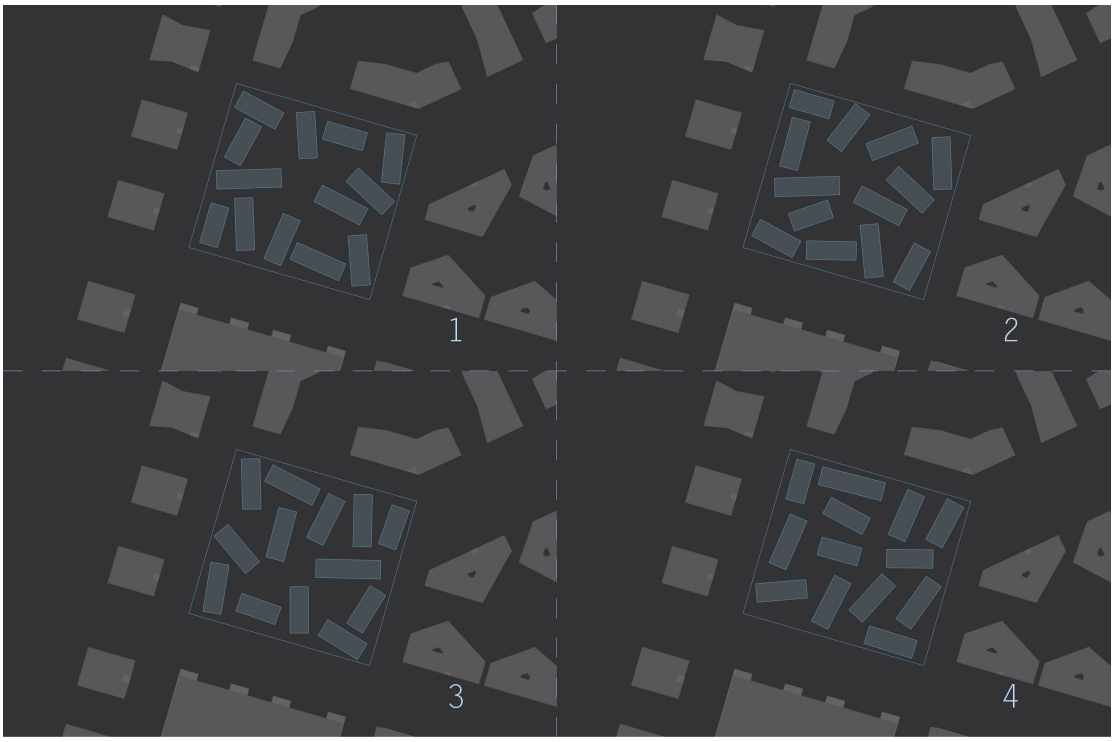


Figure 73: Four initial states

In order to solve that problem, the constraints have to be tightened. So an initial state of a manually set distribution has to be provided, from which design options can be generated within a certain range of constraints. A distance *d* is describing the range, within which the building can be moved, and an angle *a* is defining the range, within which the building can be rotated (see Figure 71 and Figure 72). As in that computation method the solutions will show similarities with the initial state, which are the bigger, the tighter the constraints are set, four initial states are defined manually to increase the diversity of solutions. One of the four initial states is the original *ppag* layout (see Figure 73 (1)). On the basis of those four constellations, 100,000 solutions are calculated with a distance d = 3.0m and an angle a = 60°. The huge amount of 100,000 calculations is necessary, as only 3.19% turned out to be valid solutions, which corresponds to

3178 valid options. Moreover, valid solutions are by its definition options that do not have any intersections, which does not mean that buildings cannot be very close to each other without intersecting. For that reason, the minimal distance of buildings is calculated for each option and options are ranked according to that minimal distance, from the highest to the lowest distance. That way, only the best solutions concerning minimal distance can be considered for the optimization with applied evaluation criteria. Looking at the 1000 best ranked results for minimal distance, it turned out that only 2.0% were derived from the original *ppag* starting constellation (see Figure 73 (1)), 7.4% from constellation 2 (2), 32.6% from constellation 3 (3) and the majority of 58.0% from constellation 4 (4).

The calculation of run 1 took approximately 870 minutes and was executed with the following inputs.

Inputs run 1

amount n	=	501
distance d	=	3.00 m
angle a	=	60°
$V_{\text{light incidence}}$	=	0.70
$V_{\text{solar radiation}}$	=	0.50
$V_{\text{view sheds}}$	=	1.00

The weighting of criteria follows the maxims defined by the architects' design, which was optimized for views. That is why *view sheds* are weighted highest, *light incidence* influenced the window sizes, so the second importance is given to it. As no window positions are defined, yet, all of the buildings' walls are analysed for the evaluation criteria.

What is striking when looking at the results is, that within the top 30 only 1 options is obtained from original option 4, none are obtained from original options 1 and 2 and 29 are obtained from original option 3 (see Figure 75). The highest potential for optimization has the *light incidence* criterion with a SF of 0.11. As almost all options are derived from original option 3, the spatial differences are not very extreme, but still there are important differences between them. The most convincing option in my opinion is the option ranked 23rd with index 239, as it creates a large longitudinal major yard in the centre, which connects almost all buildings (see Figure 76). The distribution of buildings is more or less even and it establishes a nice degree of porosity towards the site's surrounding, which ensures a balanced level of semi privacy within the site. Moreover, similar to the real project, that option has the potential of combining buildings to clusters that share the same staircases. None of the buildings is having a parallel neighbour, which improves the view relations and creates more interesting yards.

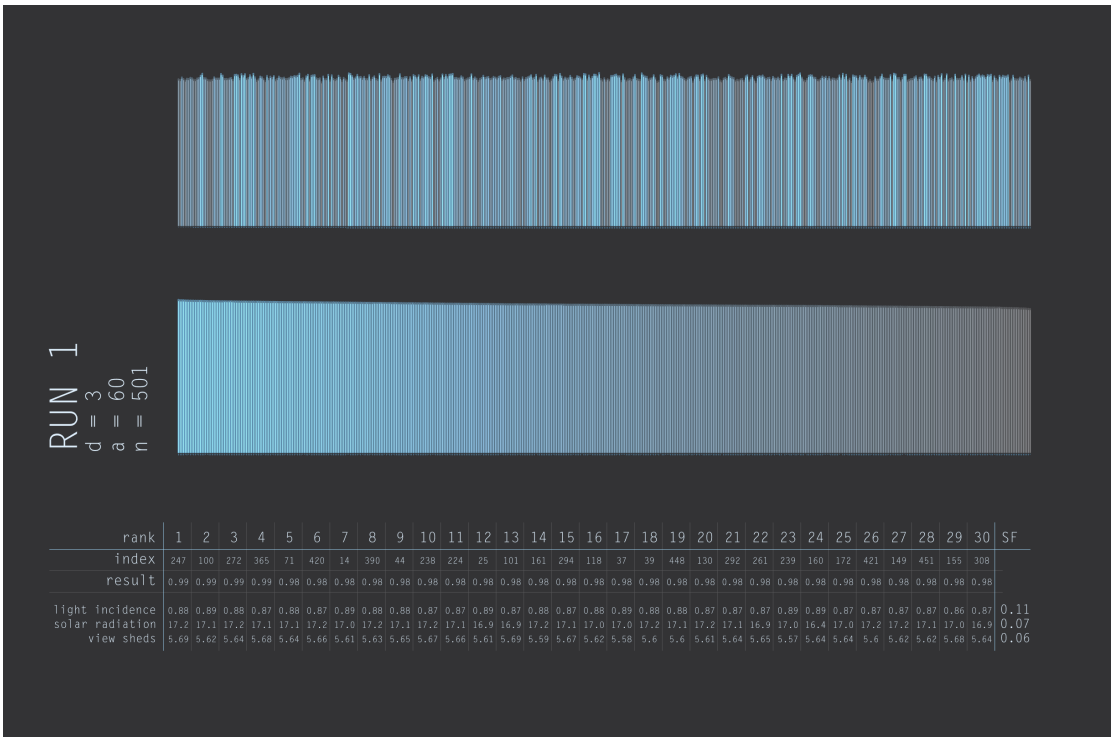


Figure 74: Performance chart run 1

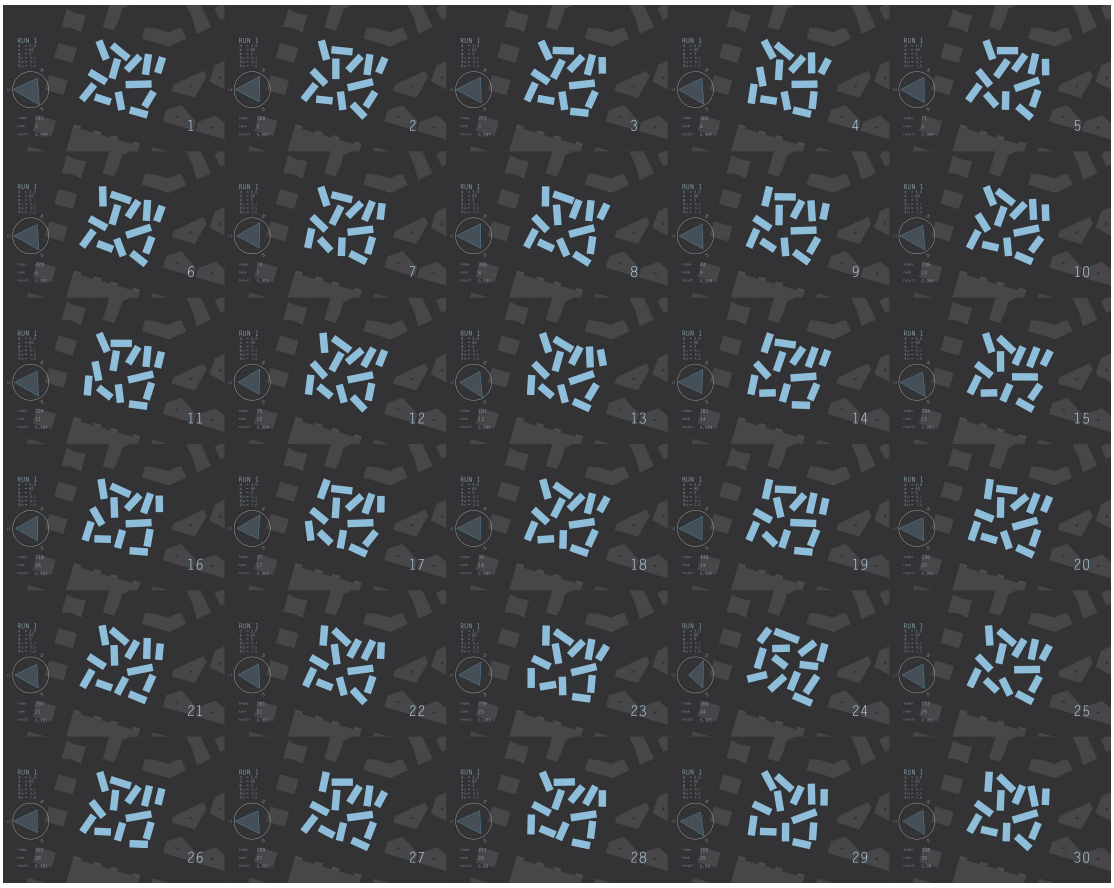


Figure 75: Floor plan options run 1, top 30

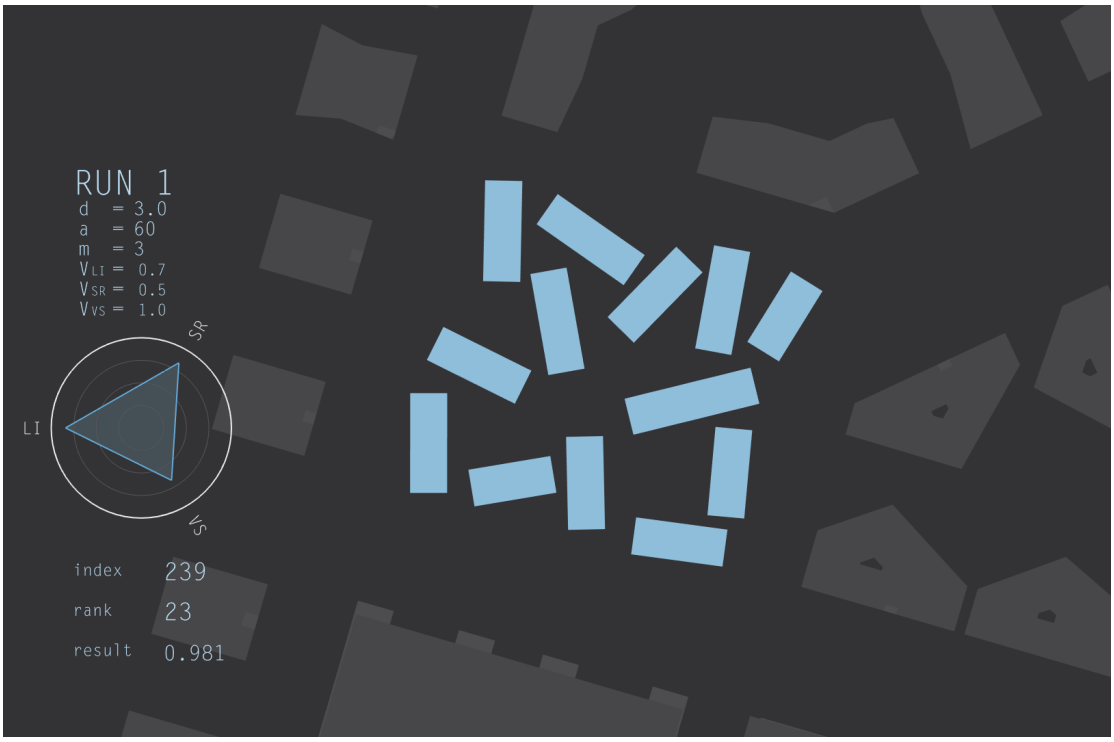


Figure 76: Run 1, chosen option ranked 23rd, floor plan

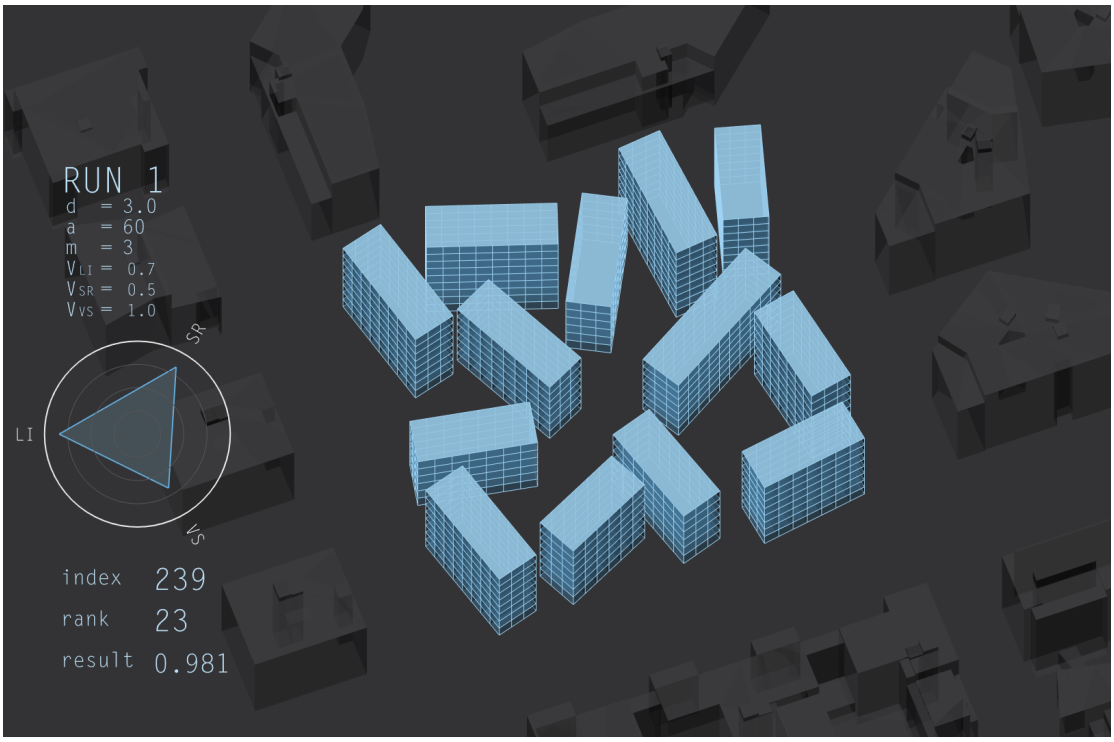


Figure 77: Run 1, chosen option ranked 23rd, axonometric view

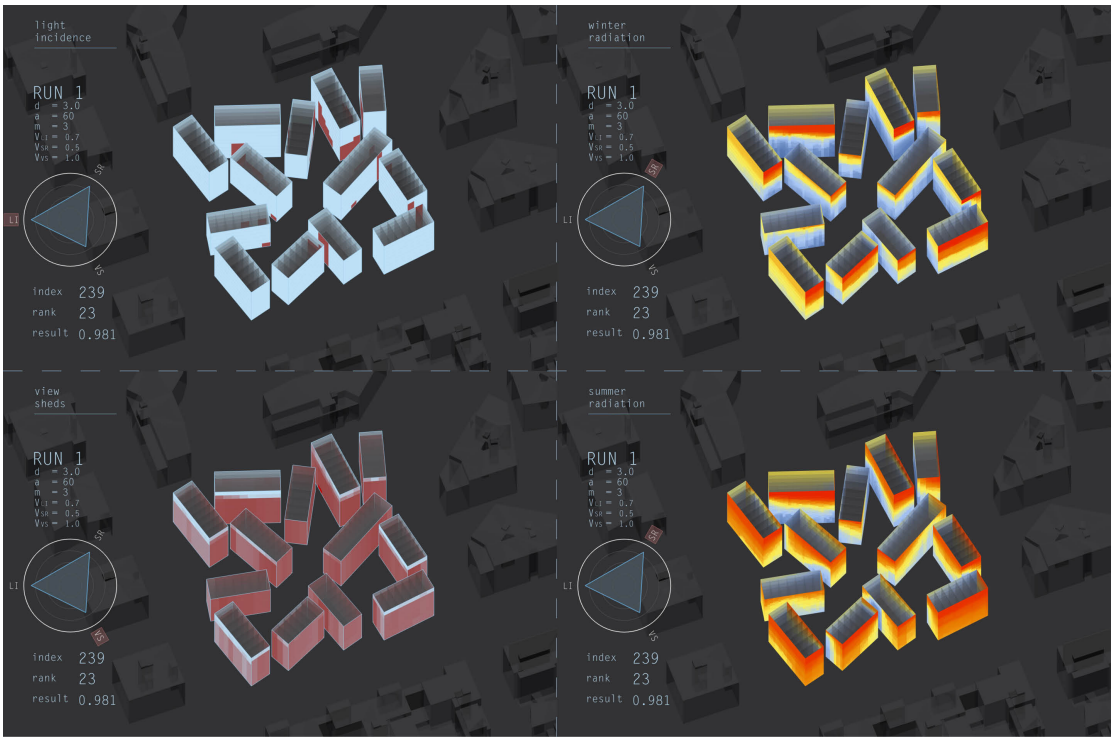


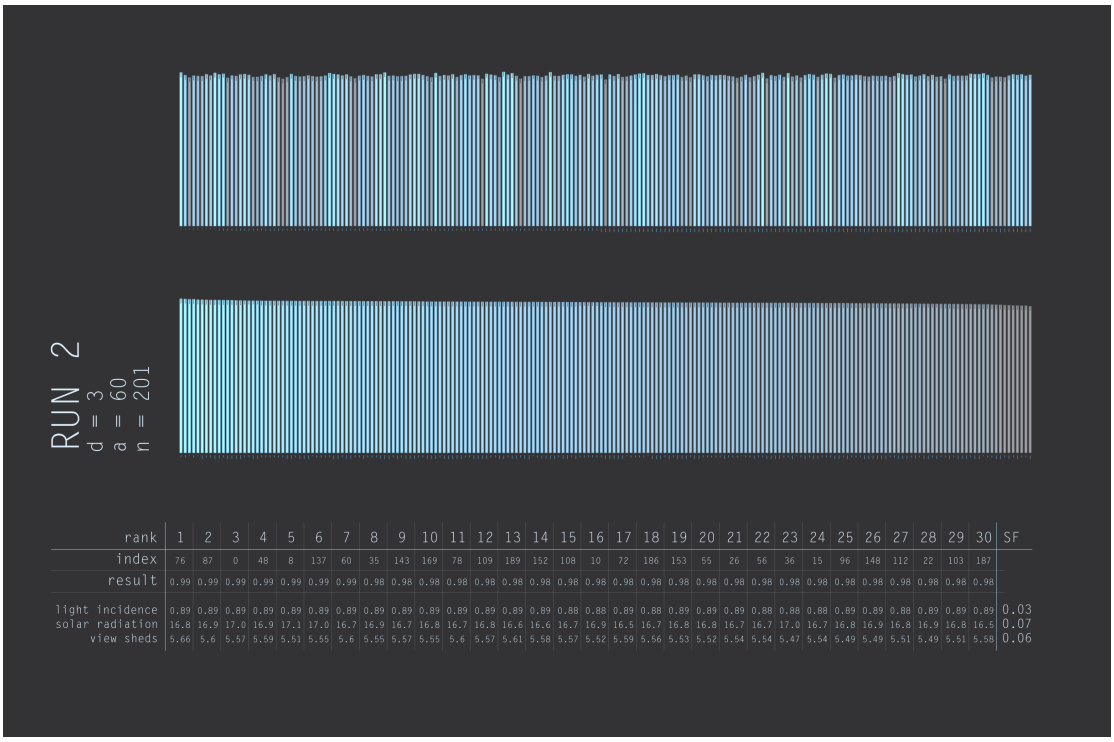
Figure 78: Visual performance illustration for option ranked 23rd in run1

As the heights for each building have been adopted from the original *ppag* design, a different distribution of heights could lead to an even better performance with the same floor plan. So in run 2, different options with different heights are tested with the same variables as in run 1. The range of the heights *h* and the overall volume of the buildings remain the same though, in order to have a fair comparison to the results of run 1 and to ensure, that the density is not changed.

The calculation of run 2 took approximately 570 minutes and was executed with the following inputs.

Inputs run 2

amount n	=	201
heights h	=	18.00 m - 26.50 m
$V_{\text{light incidence}}$	=	0.30
$V_{\text{solar radiation}}$	=	0.70
$V_{\text{view sheds}}$	=	1.00



rank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	SF
index	76	87	0	48	8	137	60	35	143	169	78	109	189	152	108	10	72	186	153	55	26	56	36	15	96	148	112	22	103	187	
result	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	
light incidence	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.89	0.88	0.88	0.89	0.88	0.89	0.89	0.89	0.88	0.88	0.88	0.89	0.89	0.88	0.89	0.89	0.89	0.03
solar radiation	16.8	16.9	17.0	16.9	17.1	17.0	16.7	16.9	16.7	16.8	16.7	16.8	16.6	16.6	16.7	16.9	16.5	16.7	16.8	16.8	16.7	16.7	17.0	16.7	16.8	16.9	16.8	16.9	16.8	16.5	0.07
view sheds	5.66	5.6	5.57	5.59	5.51	5.55	5.6	5.55	5.57	5.55	5.6	5.57	5.61	5.58	5.57	5.52	5.59	5.56	5.53	5.52	5.54	5.54	5.47	5.54	5.49	5.49	5.51	5.49	5.51	5.58	0.06

Figure 79: Performance chart run 2

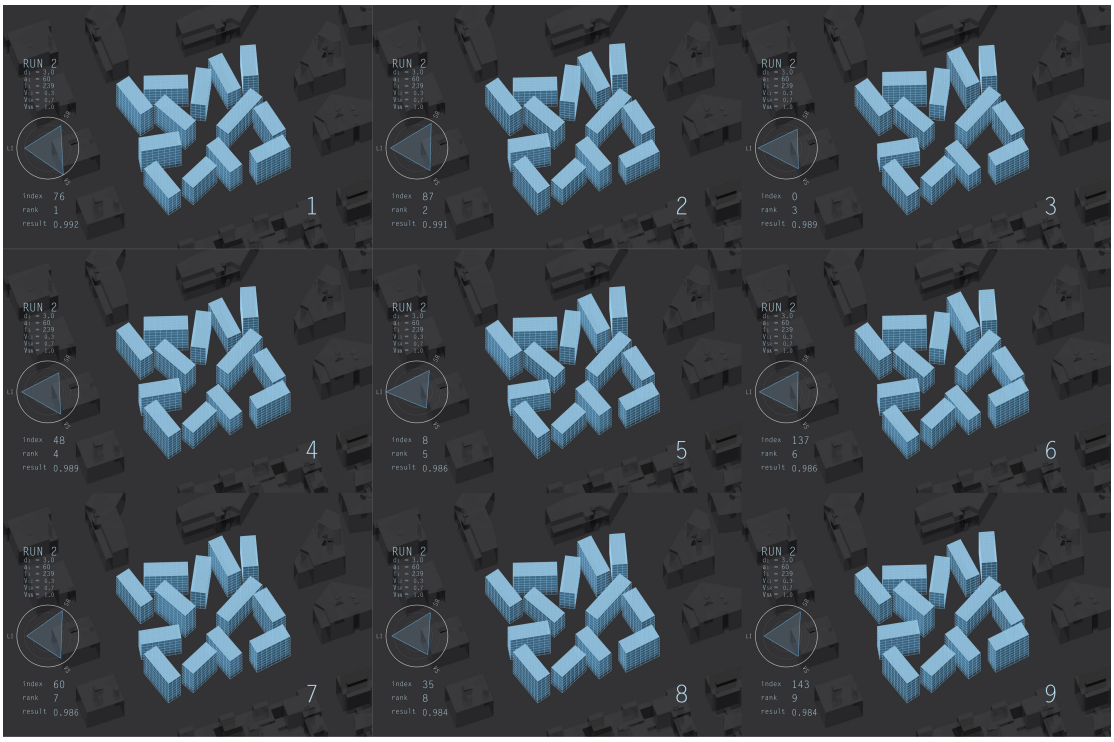


Figure 80: Height options run 2, top 9

When looking at the SF values in Figure 79 we can determine that solar radiation has a higher potential for optimization compared to light incidence, which is why I changed the weighting of the two criteria in favour of solar radiation. View sheds are still the main criterion, so the weighting remains unchanged at 1.0.

Also striking is the fact that index 0, which is exactly the chosen option of run 1, is ranked 3rd, meaning that run 2 only produced two more successful options. Index 0 was coincidentally already very close to the maximum. As the first option does not show any specifically bad spatial behaviour, in that case the first ranked option is chosen, which clearly shows the best value concerning views (see Figure 81).

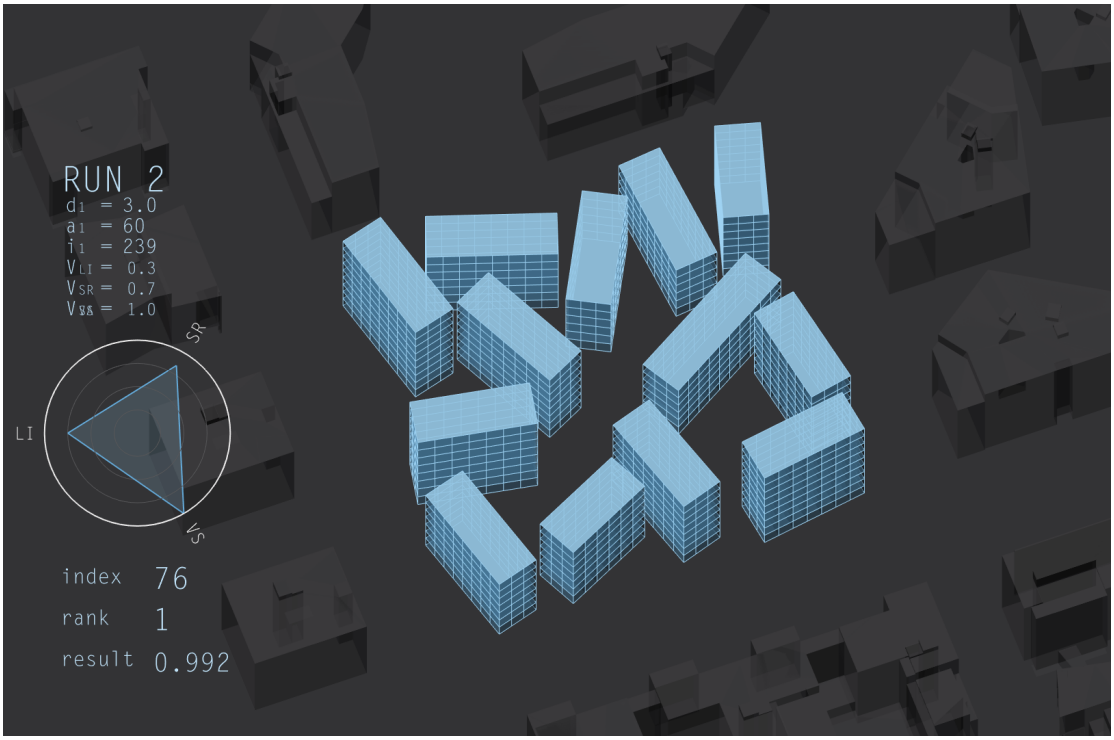


Figure 81: Chosen option ranked 1st, run 2

At this point, the optimization can be regarded as successful, as a satisfying spatial solution has been found, that also convinces with excellent performances. In a further step, windows have to be defined, which can be done with the assistance of the 3d performance illustrations of the buildings. But as the definition of windows is mainly dependent on other important design components such as interior floor plan layout, which is not topic of this thesis, the step of window definition will not be respected here. Anyway, an optimization run with defined window positions could deliver a more precise result than the one at this early stage.

The chosen option was able to improve the initial design layout by *ppag* for 1.91% concerning the total result (see Figure 83). Interestingly, the initial layout performed a little bit better in *solar radiation* compared to the optimized one, even if the difference is only 0.12%. Generally we can state, that the *ppag* layout already provided a very good solution concerning its performance. Only in light incidence the optimization was able to deliver a significantly better result.

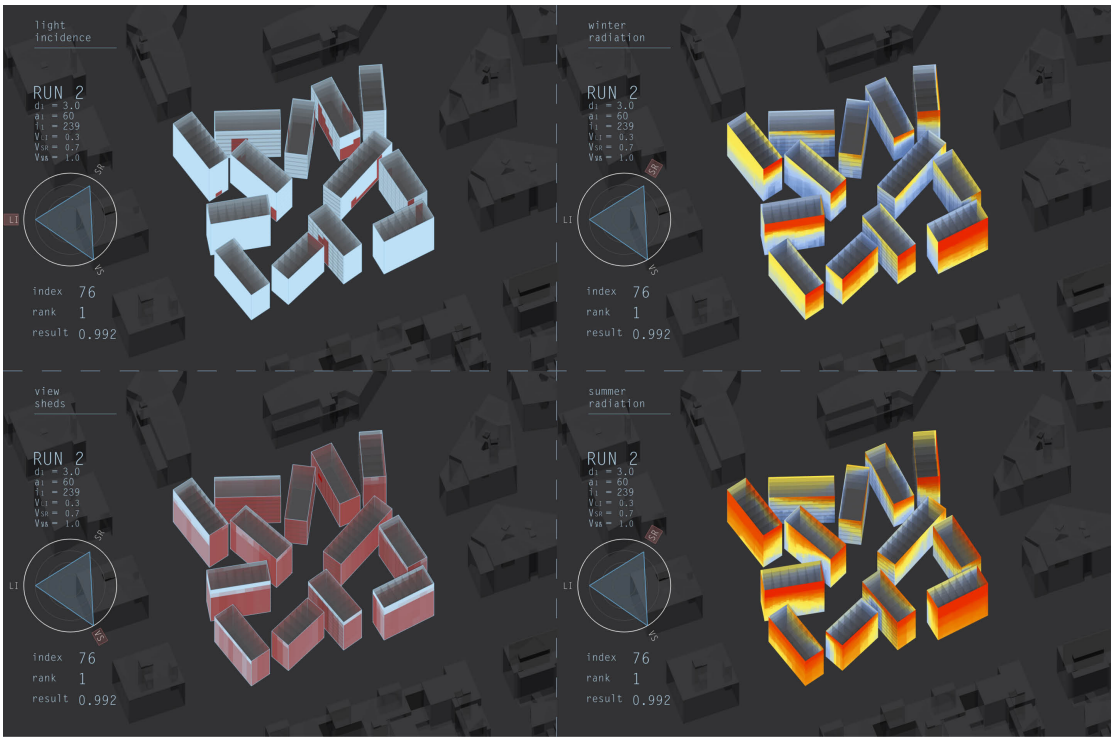


Figure 82: Visual performance illustration for option ranked 1st in run 2

improvment to	initial option	worst option
total result	1.91%	4.89%
light incidence	5.77%	8.03%
solar radiation	-0.12%	3.87%
view sheds	2.24%	4.77%

Figure 83: Improvements run 2

4.3 The interlace by OMA / Büro Ole Scheeren

The project is situated in a comparably green, tropical and low dense environment in Singapore, but because of a rapid on-going densification of Singapore, the building itself completed in 2013 had to achieve a high density. 170,000 m^2 total gross floor area are distributed on an 8 ha site, which results in a plot ratio of 2.1 (australiandesignreview.com, 2017). Unlike the usual local residential typology of vertical towers, the concept of OMA / Ole Scheeren is to stack those volumes in a horizontal way, in order to create green spaces on each on the bar's roofs and to establish a connection between all buildings (see Figure 85). Because of its huge scale and its interconnectivity, the project is also called *vertical village*. The architects claim that the building was optimized according to wind, daylight and solar factors. As a response to the hot tropical climate in Singapore, the massing deliberately supports shading in the courtyards in order to make their use more convenient. For their energy concept, the architects were awarded the *Universal Design Mark Platinum Award* and *Green Mark Gold PLUS Award* from Singapore's Building and Construction Authority (buro-os.com, 2017).

Figure 84: View from balcony and top view (buro-os.com, 2017)

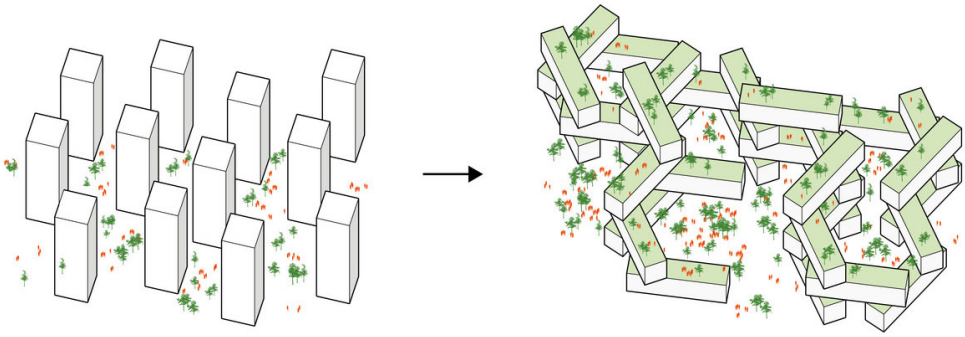


Figure 85: Conceptual diagram (buro-os.com, 2017)

Although the arrangement of the massing is making a random and unforced impression on pictures, the project's layout is actually strictly organized in six equally sized hexagonal courtyards (see Figure 86 (1)). Around them, 31 building blocks are distributed; all of exactly the same dimensions containing six floors. The maximum amount of blocks stacked above each other is four, which results in a top height of 24 floors. For the positioning of blocks in this hexagonal grid it can be found that within the same level, two neighbouring sides of a hexagon both can never be occupied by blocks at the same time. Hence, by looking at the ground floor plan, the underlying hexagonal grid is not easily identifiable (2). As blocks of the level above are never positioned on top of a block below, only bridged connections are possible. Their feasible positions are determined by the blocks below, because every block needs to be supported at both ends by two blocks of the ground floor. Figure 86 (3) is showing all possible positions for blocks of the second level (red) according to the layout of the ground floor (blue). But also in the second level, no hexagonal side can have a neighbouring block, so the amount of blocks has to be reduced until the rule is fulfilled (4). The same procedure is repeated to the top level, while the amount of feasible block positions decreases with each level.

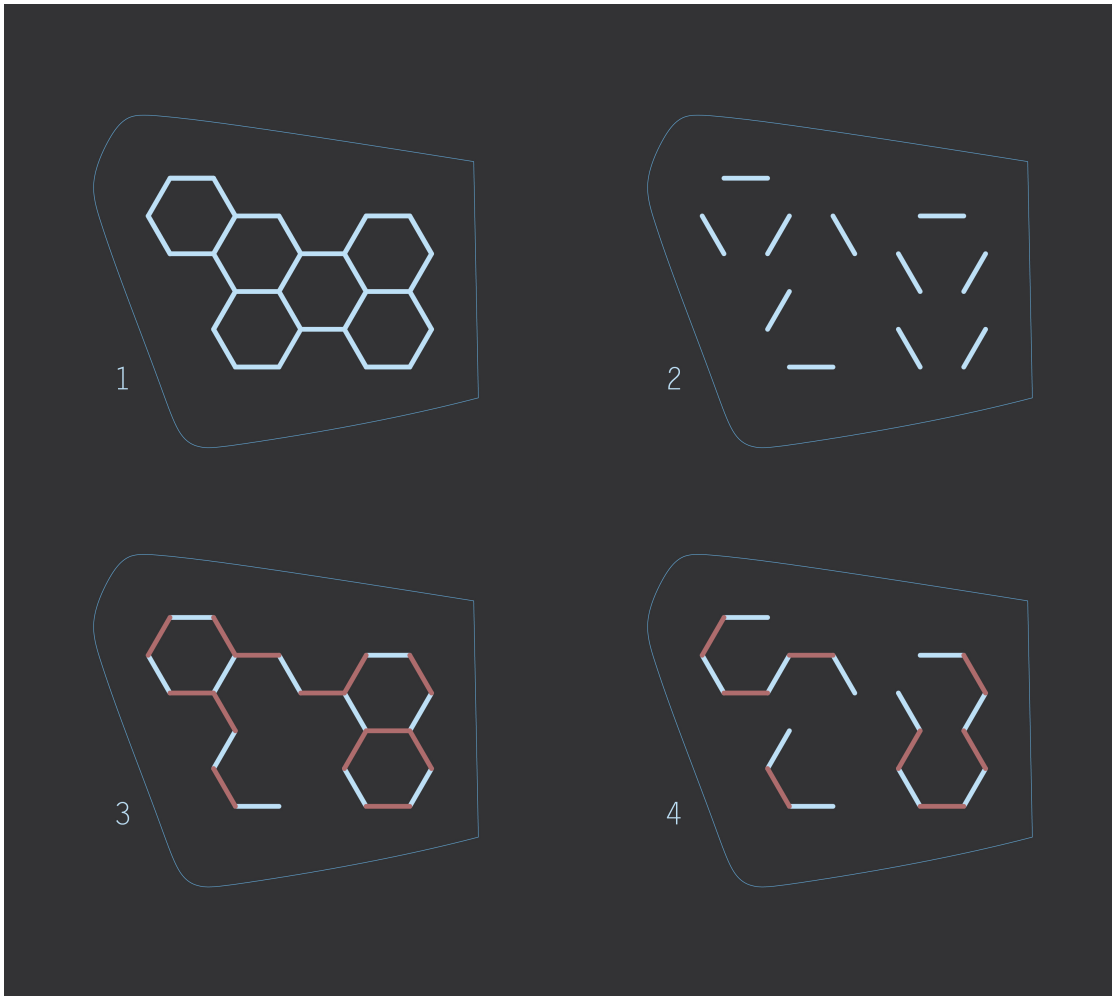


Figure 86: Logic of stacking the blocks

Those rules have to be respected in the parametric model in order to produce equivalent design options to the original design idea. The hexagonal grid defining possible block positions limits the amount of possible constellations, so unlike the examples before, in this case the amount of possible solutions is not infinite, but still very high. In order to increase the variety of solutions, six initial ground floor plan constellations are defined manually (see Figure 87), according to which the blocks above are created randomly following the described rules. The amount of blocks is set to 31 for each option in order to enable a fair comparison of design options. Also the height remains unchanged at a maximum of four levels for each option.

Figure 87: Six initial ground floor plan constellations as basis to generate design options

Figure 88: Connected original layout by OMA and unwanted design option

When looking at the original schematic concept in Figure 85 we can see that all the blocks are connected into one single conglomerate that constitutes the *vertical village*. We can strongly assume that this connection is a crucial component of the design idea, solutions where there are two or more detached

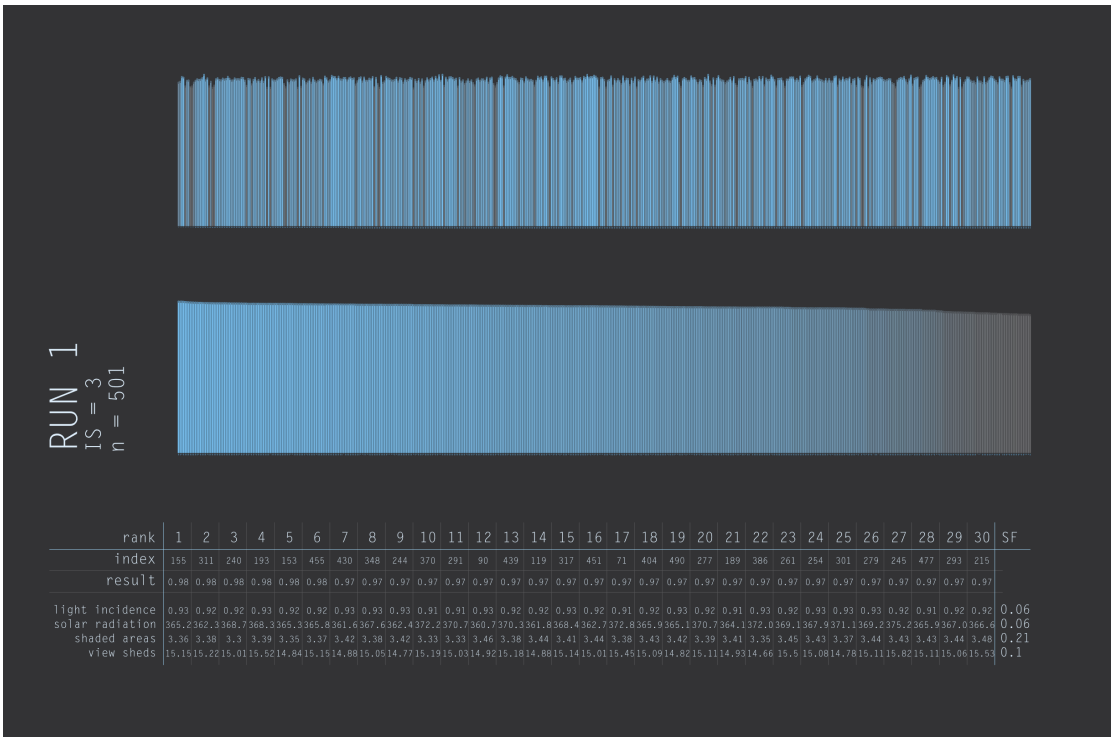
parts of the village have to be regarded as unwanted, but possible according to the constraints defined so far (see Figure 88). In order to dismiss those unwanted options, a pre-run has to be executed that is testing the options for their connectivity.

All the criteria available in the BPET are relevant for a good design option in that project. As described before, the climate of Singapore does not require any heating, instead the cooling demand is high, which is why radiation has to be calculated for the whole year and a low radiation value is regarded as desirable. Shading will be calculated for the yards, as a lot of outdoor activities are provided in those yards, hence lower temperatures through shading are wanted. The other criteria will be dealt with as usual. For the weighting of criteria, maximal stress is put to shading in the courtyards and low solar radiation, which is why both criteria are weighted with 1.0. *View sheds* are weighted lowest, as the surrounding consisting of forest and some few low buildings offers good views anyway, only on the site neighbouring east a high office building was completed recently. Because of assuming an early design stage, the exact window positions are still unknown, although the intention of having large glazing all around the 31 blocks seems deducible, which is why all of the walls are considered as window positions and hence analysed for *solar radiation*, *light incidence* and *view sheds*. As the system of the design approach is determined by the hexagonal grid and the dimensions of the blocks, no further variables can be defined in this example. So run 1 is executed with the following settings, its duration for the calculation of $n = 501$ options is approximately 555 minutes.

Inputs run 1

amount n	=	501
$V_{\text{light incidence}}$	=	0.70
$V_{\text{solar radiation}}$	=	1.00
$V_{\text{shaded areas}}$	=	1.00
$V_{\text{view sheds}}$	=	0.30

After executing run 1, it turned out that different seed values can lead to spatially identical options, which is why duplicate options have to be eliminated before doing the evaluations, so effectively only 382 different options have been generated.



rank	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	SF
index	155	311	240	193	153	455	430	348	244	370	291	90	439	119	317	451	71	404	490	277	189	386	261	254	301	279	245	477	293	215	
result	0.98	0.98	0.98	0.98	0.98	0.98	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	
light incidence	0.93	0.92	0.92	0.93	0.92	0.92	0.93	0.93	0.93	0.91	0.91	0.93	0.92	0.92	0.93	0.92	0.91	0.92	0.93	0.92	0.91	0.93	0.92	0.93	0.93	0.93	0.92	0.91	0.92	0.92	0.06
solar radiation	365.2	362.3	368.7	368.3	365.3	365.8	361.6	367.6	362.4	372.2	370.7	360.7	370.3	361.8	368.4	362.7	372.8	365.9	365.1	370.7	364.1	372.0	369.1	367.9	371.1	369.2	375.2	365.9	367.0	366.6	0.06
shaded areas	3.36	3.38	3.3	3.39	3.35	3.37	3.42	3.38	3.42	3.33	3.33	3.46	3.38	3.44	3.41	3.44	3.38	3.43	3.42	3.39	3.41	3.35	3.45	3.43	3.37	3.44	3.43	3.43	3.44	3.48	0.21
view sheds	15.15	15.22	15.01	15.52	14.84	15.15	14.88	15.05	14.77	15.19	15.03	14.92	15.18	14.88	15.14	15.01	15.45	15.09	14.82	15.11	14.93	14.66	15.5	15.08	14.78	15.11	15.82	15.11	15.06	15.53	0.1

Figure 89: Performance chart run 1

Figure 90: Floor plan options run 1, top 30

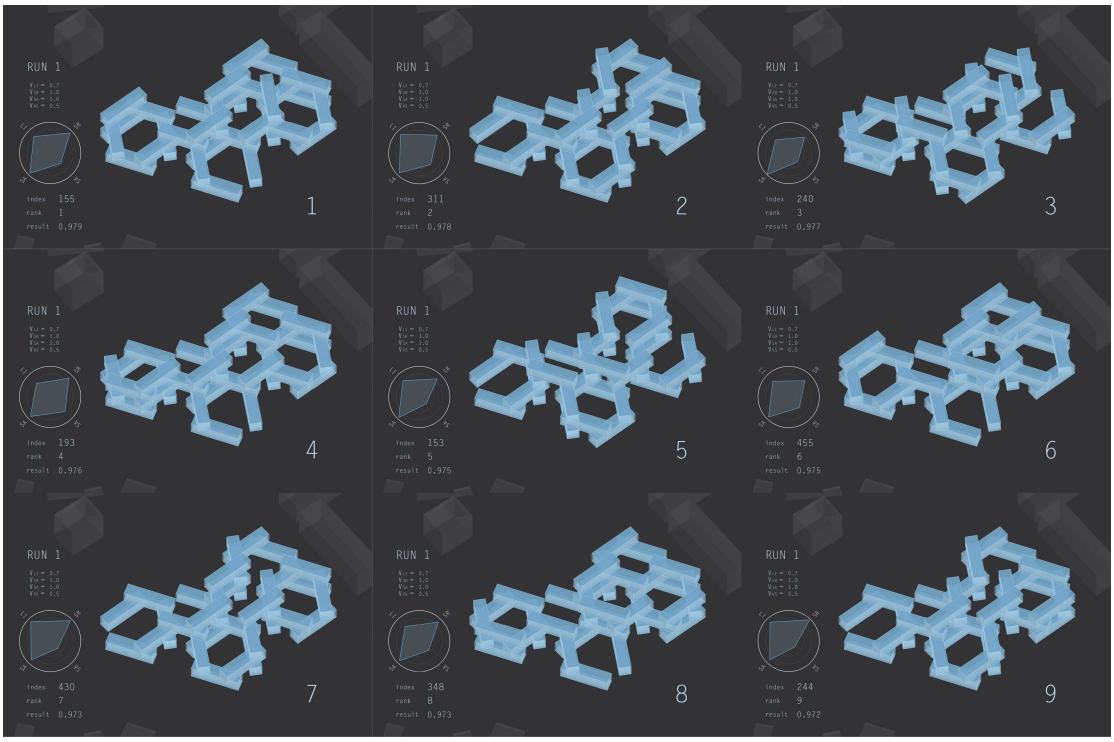


Figure 91: Axonometry options run 1, top 9

Spatially, in my opinion an option is to prefer which has a balanced distribution of blocks, with a slight increase of heights from the west with its low surrounding to the east with the neighbouring ten-storey-high office building. Moreover, a continuity of courtyards is desirable, constellations that enclose courtyards entirely and therefore separate them from other yards are unwanted, such as the 8th ranked option in Figure 90 and Figure 91 is doing. Considering those spatial requirements, option indexed 153, ranked 5th is performing best from my point of view.

What is striking when comparing the radar charts of the best options as shown in Figure 90 and Figure 91 is their similarity. All of them perform pretty well in *shaded areas* as well as in *solar radiation*, the performance in *light incidence* is comparably lower and differs between the options, whereas all of the options perform very badly in *view sheds*. That fact made me look at the worst options, and indeed the worst options show the best results in *view sheds*, whereas the other criteria's performance is bad (see Figure 92). This indicates a negative correlation between *solar radiation* and *shaded areas* on the one hand and *view sheds* on the other hand. As low *solar radiation* and a maximum of *shading* was searched for, constellations similar to fortresses, which enclose the courtyards and hence block a maximum of sunlight, are most suitable to achieve this, whereas open constellations allow a maximum of *views*. This is confirmed by comparing the best to the worst solutions, whereby all of the worst solutions arise from the same initial state, only the distribution of higher-level blocks differs. The more enclosed options are ranked better because of the high weighting of *solar radiation* and *shaded areas*, if *view sheds* were weighted higher, the result would flip. Spatially, the more open options may perform

better, which is why in this situation the designer has to make a decision whether to favour energy efficiency or spatial qualities. There is no guideline, which decision is better, but considering the very high temperatures in Singapore, I as designer would favour energy efficient solutions in this case, which is why option ranked 5th is chosen in this example (see Figure 93, Figure 94, Figure 95).

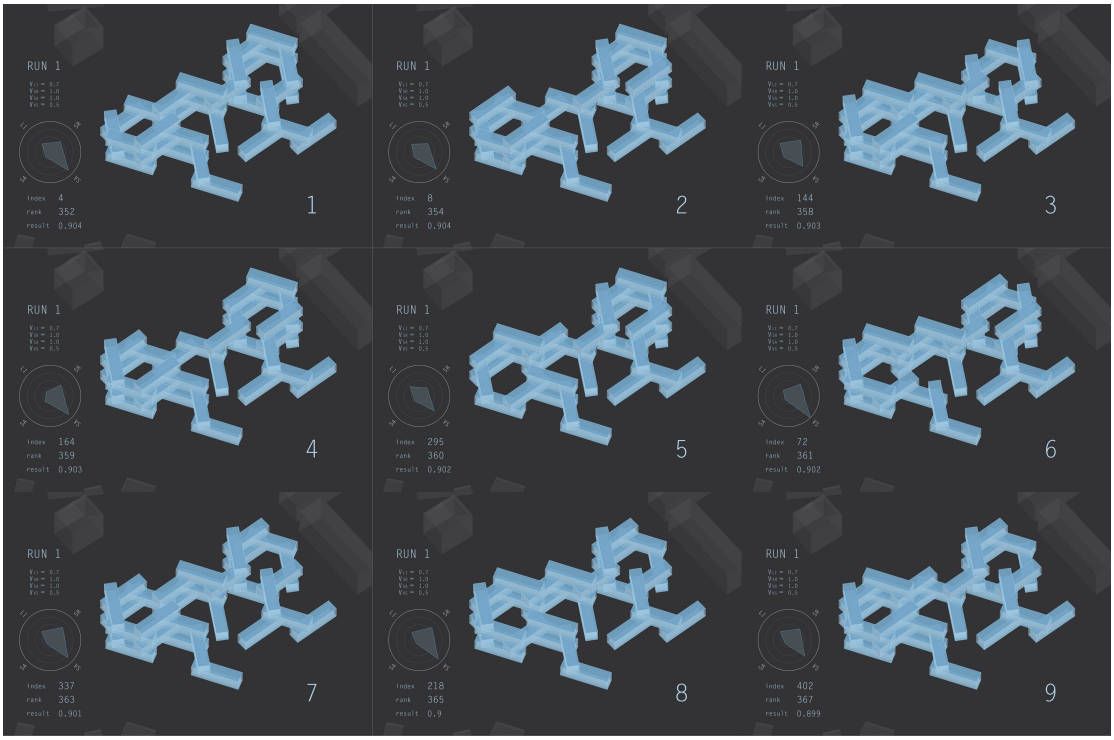
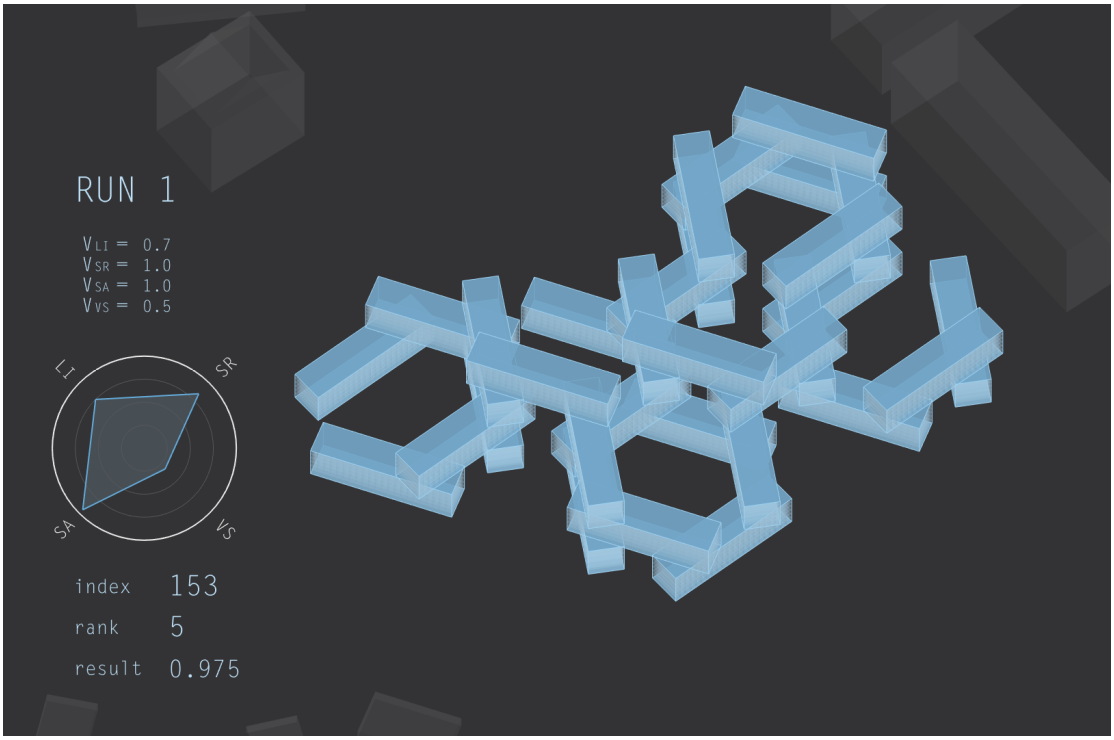


Figure 92: Worst options run 1, axonometric view

Figure 93: Chosen option ranked 5th, floor plan



,

Figure 94: Chosen option ranked 5th, axonometric view

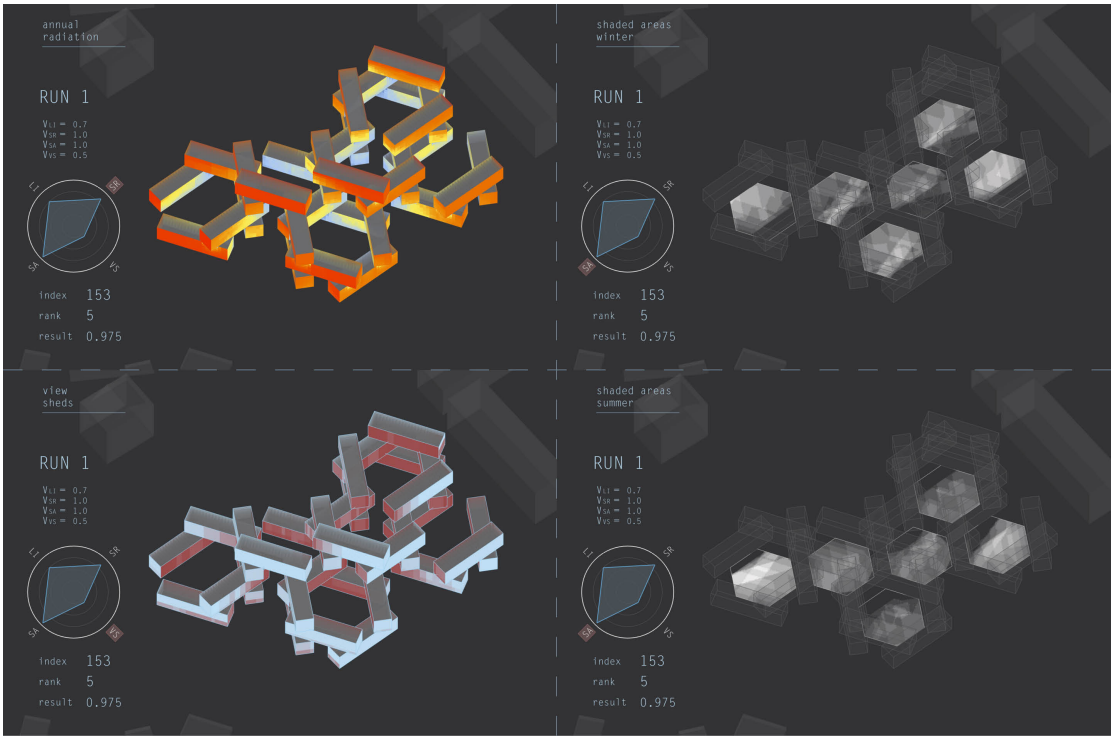
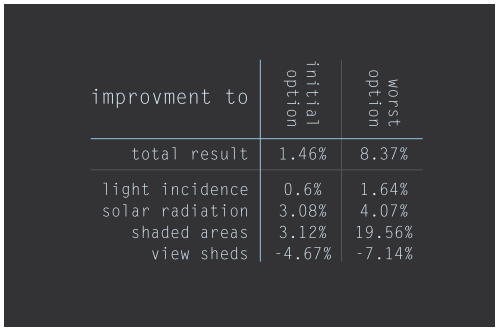


Figure 95: Visual performance illustration for option ranked 5th in run 1

For the specific performances in *solar radiation* and *shaded areas*, characteristics of Singapore's location close to the equator become apparent, there is almost no difference between south and north (see Figure 95). Because of the very steep sun angle, shading of obstacles is less effective than in regions closer to the poles. Still, when looking at the SF value of 0.21 for shaded areas (see Figure 89), the impact of different constellations is enormous. This is also confirmed by looking at the improvements of the chosen option compared to the worst option, where the difference in *shaded areas* is 19.56% (see Figure 96). At the same time we can observe what was discussed before, that there is a negative correlation between *shaded areas* and *view sheds*, so the chosen option is performing 7.14% worse in *view sheds* compared to the last ranked option. When comparing the chosen option to the original design, the differences are minor. Still we can see that the chosen option prefers an energy efficient design, whereas the original design by OMA favours spatial qualities.



improvment to	initial option	worst option
total result	1.46%	8.37%
light incidence	0.6%	1.64%
solar radiation	3.08%	4.07%
shaded areas	3.12%	19.56%
view sheds	-4.67%	-7.14%

Figure 96: Improvements run 1

As in this example there are no variables that can be adjusted in order to further improve the design, only a higher number of calculations could possibly bring forth better options. But as the number of possible constellations is limited anyway, the effort might not be worth calculating more options, so the optimization for this project is terminated after one run.

4.4 Résumé

The Building Performance Evaluation Tool proved its good applicability on architectural projects and was able to produce results, which keep the same architectural qualities as the initial design, but show improved performance properties. In the tested examples, the potential of performance optimization was up to 21% difference between the worst and the best performance, largely dependent on the specific project and the set constraints, although in some cases of negative correlation, deteriorations for some criteria have to be accepted as well. This potential absolutely justifies the use of BPET, if we consider that approximately an extra week of work, which is necessary for the tool application, can improve a building's performance throughout its whole lifetime. In order to meet future energy goals and to help preserving our environment, applications such as the BPET are able to contribute to that significantly.

Comparing the three projects, it can be found that potential improvements are the larger, the bigger the difference of design options is. So in case of *the interlace*, huge differences in the SF values became apparent, due to very different spatial constellations. Moreover, the designer's weighting of design criteria is especially influential in cases of negative correlation between criteria, as the ranking can totally flip when changing the stress of criteria. Such negative or positive correlation, respectively, becomes apparent when looking at the radar charts. In the example of *up and down* and *slim city*, they are very balanced for the best solutions indicating a positive correlation or no correlation, which makes it easier for the designer to choose, as the decision is not an "either, or" decision but a case where a solution can be selected, that performs well in all criteria. For *the interlace* the opposite was the case, the designer had to decide whether the project should stand for an energy efficient approach or an approach focussing on spatial qualities, high performances in both poles were not available. In such a case it might be useful to not only inspect the best-ranked results but also the "worst" results, as those are performing best in the lower weighted criteria. For compromise solutions, also the middle of the spectrum can be interesting for the designer.

General guidelines for positioning of massing or façade design could not be deducted from the examples, as the project's location and its specific climate largely influence the building performance. On the contrary, the case of *up and down* showed, that assumptions made according to guidelines could be misleading without really simulating the design under its specific conditions. What can be deducted from the examples though is that under conditions, where the heating demand is much higher than the cooling demand, open constellations which enable a maximum of sunlight to hit windows perform better concerning solar radiation. In climate conditions as they are the case in Singapore though, which are characterized by a huge cooling demand throughout the whole year, closed constellations, similar to fortresses are able to block a maximum of sunlight and hence to save cooling expenses.

The *ppag* example showed, that a completely random distribution of several buildings on a site in high density conditions is a practically impossible task, as the extremely high probability for invalid (intersecting) solutions in a completely random setting prohibits the generation of valid design options within a reasonably amount of time, as an enormously high number of options had to be produced (in the case of slim city 50,000 random options haven't brought forth a single valid solution). In this case, the computer fails to do what is an easy task for the human mind, which proves the statement of the beginning of this thesis, that architecture solely done by algorithms is an illusion at the current state of computer development. The combination of the human mind setting the initial ideas and constraints and the computer generating numerous design options to that design idea respecting certain constraints turned out to be a powerful combination though.

For the proposed method of *undirected randomization*, as described in chapter 3.1.2, the résumé is quite positive. *Undirected randomness* proved that also a comparably small number of calculated options leads to a usable result. For optimization algorithms such as the genetic solver, a common minimum amount of calculations is around 2500 (Rutten, 2014), which in the case of the *up and down* project would have calculated for approximately 8750 minutes respectively 146 hours, excluding the time needed for the algorithm's native internal calculations.

Moreover, *undirected randomness* provided a balanced variety of different spatial results with very good performance values, from which the user can choose. So actually, in the case of this thesis, the algorithm cannot be called optimization algorithm, as it is not optimizing anything, it is just providing design options with the correspondent performance result. The optimizing part is only represented by the user himself, who is evaluating and choosing the options and in that way optimizing the design. This constitutes a big difference between common optimization approaches and the one presented in this thesis, the BPET.

The advantage of using *undirected randomness* becomes especially apparent in cases where there is a negative correlation of criteria. As described before, in the example of the interlace not only the best options according to the weighting set by the user might be interesting, but also the "worst" options, as they are the best options in other weighting scenarios. And for a compromise solution, also solutions scoring average are relevant. If a common optimization algorithm such as the genetic solver is used, the weighting of criteria has to be done before the generation of options, in order to give the algorithm a definition, how "good" options are defined. As a consequence, the vast majority of generated options would perform well in the highly weighted criteria, options performing average or badly are eliminated in early optimization stages by the algorithm and hence are not available for inspection. Using *undirected randomness* though, the generation of results is executed without any preferences regarding their performance. As a result, we will obtain a balanced variety of different kinds of options and the weighting of criteria can easily be done after the generation of

options, as it is not required for the generation process. Therefore, changing the weighting after the generation will simply change the ranking of options and is easily possible without running the time-consuming generation process once again. So for all possible weightings an equal amount of options can be viewed. This is a very critical factor that allows the designer to keep full control over the design process and assures that no possible option is denied by a target-driven algorithm.

5 Conclusions

This thesis came up with an optimization tool, which had the goal to being prepared for the special demands of architectural optimization. Therefore, an important separation has been made between measurable, tangible criteria and unmeasurable, abstract criteria. The tool only optimizes measurable criteria while leaving the unmeasurable criteria, which have to be defined beforehand, untouched and therefore ensuring, that architectural qualities are not diminished. The way how to define and calculate the available criteria of *solar radiation*, *light incidence*, *shaded areas* and *view sheds* is focussing on its application in an early design stage and can hence be regarded as compromise between highly sophisticated and maximal accurate calculation methods on the one hand and fast and easy computation of inputs, which do not overstrain the user's capabilities, on the other hand. Moreover, the approach of *undirected randomness* has been established and applied on the BPET, which was tested on three example projects. BPET was able to deliver satisfying options to the initial design idea, which follow the same design rules, but perform better concerning the applied criteria. Yet, the BPET is still in its infantile stage and there is potential for improvement. The following points show such potential.

1. Complexity

   BPET so far only focuses on solar aspects combined with view sheds. For an overarching analysis of energy efficiency potential, more aspects have to be included in the calculation, such as aerodynamics in order to compute ventilation potential for example. Though, adding more criteria would increase the complexity and hence the calculation time. Also the way how to calculate the contained criteria could be further improved and optimized with the help of experts in the corresponding fields, which is mostly building physics. However, it would be important to find a compromise between very accurate, but user overcharging approach and a highly simplified, but inaccurate one.

2. Computation Time

   Although the attempt was to simplify calculation methods in order to focus on a short computation time, it still took very long to compute options for evaluation. As a consequence, the amount of generated options was way less than what I actually wanted to calculate. Especially the calculation of solar radiation, which was done with the help of *Ladybug's Radiation Analysis* component was very time consuming, the other criteria could be computed much faster. One approach to solve this would be the use of so-called *supercomputers*, which are used for complex simulations such as weather forecasting and which have a much higher computation capacity than common personal computers.

But as the focus of this thesis was to implement tools such as the BPET into architects' workflows, those tools have to be able to work under real office conditions and therefore, on regular office computers. Exporting the design and handing it over to an external *supercomputer* would be a great barrier for architects to using the tool, and moreover, the process of exporting and importing is always connected to compatibility problems, which can be very time consuming as well. What could be possible though is a similar approach to so-called *renderfarms*, where multiple computers are connected via a network and share the computation task, so all connected machines in an office could be used simultaneously to calculate complex tasks much faster. Anyhow, the approach of this thesis was to use a common personal computer and test, what can be achieved under those low performance conditions. As most offices today own more powerful machines than the one used for this thesis (8GB RAM) and the computer performances are constantly improved by computer industries, computation time will get increasingly less in future anyway.

3. Scope of level of detail

   BPET is designed for early design stages, where massing and building envelope characteristics are defined. More detailed optimization of design components such as circulation spaces, building units or shading devices in later design stages was not part of this thesis, but would be a reasonable addition for a holistic optimization tool, which accompanies the designer through the whole design process. Also including active solar devices such as photovoltaics could enrich the portfolio of BPET in future use.

Besides those specific future potentials of BPET, some more profound problems of algorithmic optimization became obvious while conducting this thesis.

1. Changing climate conditions

   Optimization is mostly done on the basis of current climate conditions (which are actually past climate conditions, as they are obtained by climate data collected in the past years). But as climate conditions are not a stable phenomenon, climate is very likely to change within the lifetime of a building, so its optimization might become less efficient or even obsolete, as shown by a study done by Glassman & Reinhart, which postulates a consideration of future climate scenarios in optimization processes (Glassman & Reinhart, 2013).

2. Lack of objective guidelines for optimization

   Even though optimization always has to be based on quantifiable data, which actually suggests an easy and fair comparison of datasets, the way to interpret this data can vary heavily. As shown in this thesis, there are many different approaches on how to calculate energy efficiency for example, which all return different results and discussion on how to improve those methods is still vivid amongst experts. This demonstrates, that even the quantifiable nature of optimization is in practice a very subjective matter. Even in the example of the BPET itself, there is no formula on how to reasonable value criteria; the weighting is up to the specific taste of the designer.

   This lack of objectiveness in optimization can lead to an undefined and inflationary use of optimization in the field of architecture. It is becoming increasingly popular amongst architects to claim that their project has been optimized for certain criteria without giving exact information on how the optimization has been conducted and what the improvements actually are. So some global standards for optimization would be desirable, which allow for an objective comparison of performance. Especially in the case of competitions, testing all projects under standardized conditions would contribute to a fair procedure instead of unprovable claims that the design has been optimized.

The described problems may raise the question that if there are so many uncertainties and inaccuracies in optimization, is optimization useful at all or just a tool for justifying design? While developing the BPET, this was the question that I also asked myself, but I think we can answer that question with a counterquestion: “Are we making things worse with optimization?” So even if the exact extend of improvement through optimization is not predictable or very subjective, there is a very high probability that the optimization will have positive effects on the final design. For certain, that probability is higher compared to not optimizing at all. For that reason, even the ability to only save a small percentage of resources or improving spatial qualities a little is worth the effort.

Summarized, algorithmic optimization of architecture is not capable of solving all efficiency problems for buildings at once, but it can definitely contribute to improved performances and it should therefore be used to fight the unnecessary consumption of non-renewable resources. The BPET has proven that it is possible to optimize architecture without losing the essentials of design approaches.

In that sense, I want to end this thesis with some words by Louis Kahn uttered almost a century ago, which were back then obviously not related to algorithmic optimization but to the design process in general, but which fit very well to describe the philosophy of the approach in this thesis.

> *A great building must, in my opinion, begin with the unmeasurable, must go through the measurable in the process of design, but must again in the end be unmeasurable.*
>
> Louis Kahn, 1930 (Stöckli, 1992)

References

Literature

Aish, R., & Woodbury, R. (2005). Multi-level interaction in parametric design. In International symposium on smart graphics (pp. 151-162). Springer Berlin Heidelberg.

Akos, G., & Parsons, R. (2014). Foundations. The grasshopper primer third edition. Studio Modelab

Benedikt, M. L. (1979). To take hold of space: isovists and isovist fields. Environment and Planning B: Planning and design, 6(1), 47-65.

Berlinski, David (1999): The advent of the algorithm: The idea that rules the world. Harcour

Biermayr, P., et al. (2013). Innovative Energietechnologien in Österreich-Marktentwicklung 2012. Berichte aus Energie-und Umweltforschung, 17, 2013.

Bleil De Souza, C., & Knight, I. (2007). Thermal performance simulation from an architectural design viewpoint.

BMWFW (Bundesministerium für Wissenschaft, Forschung und Wirtschaft) (2017). Klimadatenrechner. Austria. Retrieved from https://www.bmwfw.gv.at/EnergieUndBergbau/klimadatenrechner/Seiten/zurBerechnung.aspx

Cassel, K. W. (2013). Optimization. Variational Methods with Applications in Science and Engineering, PART III. Cambridge: Cambridge University Press

Chien, S. F., & Flemming, U. (2002). Design space navigation in generative design systems. Automation in Construction, 11(1), 1-22.

Ciftcioglu, O., & Bittermann, M. S. (2008). Solution diversity in multi-objective optimization: A study in virtual reality. In 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence) (pp. 1019-1026). IEEE.

Derix, C. W. (2015). The space of people in architectural computation : a conceptual framework for the design computation of human-centric spatial environments. Wien

Elezkurtaj, T., & Franck, G. (2002). Algorithmic support of creative architectural design. Umbau, 2, 16.

Ferrero, A., Lenta, E., Monetti, V., Fabrizio, E., & Filippi, M. (2015). How to apply building energy performance simulation at the various design stages: a recipes approach. In Proceedings of BS2015: 14th Conference of International Building Performance Simulation Association (pp. 2286-2293).

Glassman, E. J., & Reinhart, C. (2013). Facade optimization using parametric design and future climate scenarios. In Conference of International Building Performance Simulation Association (vol. 13, pp. 1585-1592).

Horvat, M., & Wall, M. (2012). Solar Design of Buildings for Architects: Review of Solar Design Tools. Report T, 41.

Horvat, M., Dubois, M. C., Snow, M., & Wall, M. (2011). International survey about digital tools used by architects for solar design. International Energy Agency Solar Heating and Cooling Programme.

Ingels, B., Ginsberg, E., Pahhota, D., Zahle, D., Johansson, H., Pedersen, A., & Bergman, K. U. (2009). Yes is more: An archicomic on architectural evolution.

Ji Guohua (2012). Parametric Diagram and Perfomative Design. School of Architecture and Urban Planning, Nanjing University

Kang, E., Jackson, E., & Schulte, W. (2010). An approach for effective design space exploration. In Monterey Workshop (pp. 33-54). Springer, Berlin, Heidelberg.

Kanters, J. (2011). Adequacy of current design tools and methods for solar architecture–results of IEA-SHC Task 41's international survey. In Proceedings of PLEA (pp. 65-70).

Kourkoutas, V. (2007). Parametric form finding in contemporary architecture. Na.

Krause, D., Derix, C., & Gamlesaeter, A. (2011). The Virtual Building Simulator: A post-parametric spatial planning environment.

Liggett, R. S. (2000). Automated facilities layout: past, present and future. Automation in construction, 9(2), 197-215.

Lobos, D., Donath, D. (2010). The problem of space layout in architecture: A survey and reflections. Arquiteturarevista, 6(2).

Mahmoodi, Amir Saeid (2001). The design process in architecture - a pedagogic approach using interactive thinking. Leeds

Mayer-Schönberger, V., & Cukier, K. (2013). Big data: A revolution that will transform how we live, work, and think. Houghton Mifflin Harcourt Publishing Company. New York

Newsham, G. R., & Donnelly, C. L. (2013). A model of residential energy end-use in Canada: Using conditional demand analysis to suggest policy options for community energy planners. Energy policy, 59, 133-142.

Olonscheck, M., Holsten, A., & Kropp, J. P. (2011). Heating and cooling energy demand and related emissions of the German residential building stock under climate change. Energy Policy, 39(9), 4795-4806.

Österreichisches Institut für Bautechnik (2007). Erläuternde Bemerkungen zu OIB-Richtlinie 3 „Hygiene, Gesundheit, Umweltschutz“

Pfefferkorn, C. E. (1972). The Design Problem Solver, A System for Designing Equipment or Furniture Layouts. Carnegie-Mellon Univ Pittsburgh pa dept of computer science.

Ppag architects (2015). Seestadt Aspern D8. Slim City. Vienna

Rehan, S. M. T. I., & Islam, K. S. (2015). Analysis of building shadow in urban planning: a review.

Roudsari, M. S., Pak, M., & Smith, A. (2013). Ladybug: a parametric environmental plugin for grasshopper to help designers create an environmentally-conscious design. In Proceedings of the 13th International IBPSA Conference Held in Lyon, France Aug.

Rutten, D. (2013). Galapagos: On the logic and limitations of generic solvers. Architectural Design, 83(2), 132-135.

Rutten, D. (2014). Evolutionary Principles applied to Problem Solving. Retrieved from http://www.grasshopper3d.com/profiles/blogs/evolutionary-principles

Sailor, D. J. (2001). Relating residential and commercial sector electricity loads to climate—evaluating state level sensitivities and vulnerabilities. Energy, 26(7), 645-657.

Salim, F. D., & Burry, J. (2010). Software openness: evaluating parameters of parametric modeling tools to support creativity and multidisciplinary design integration. In International Conference on Computational Science and Its Applications (pp. 483-497). Springer, Berlin, Heidelberg.

Schumacher, P. (2009). Parametricism: A new global style for architecture and urban design. Architectural Design, 79(4), 14-23.

Sofic, M. (2009). Erhöhung der Anwendbarkeit vereinfachter Berechnungsverfahren zur Bestimmung des Heizwärme-und Kühlbedarfs von Gebäuden: als Basis für ein Sicherheitskonzept. Na.

Stöckli, T. (1992). The measurable and the unmeasurable or-from form to design to existance.

Van, T. N., Miyamoto, A., Trigaux, D., & De Troyer, F. (2014). Cost and comfort optimisation for buildings and urban layouts by combining dynamic energy simulations and generic optimisation tools. Eco-Architecture V: Harmonisation between Architecture and Nature, 142, 81.

Weeber, R., Sahner, G., Bosch-Lewandowski, S. (2007). Evaluierung ausgestellter Energieausweise für Wohngebäude nach enev BMVBS-Online-Publikation 01/2011

Woodbury, Robert; Gün, Onur Yüce; Peters, Brady; Sheikholeslami, Mehdi (2010). Elements of parametric design; Routledge, London

Yang, X. S. (2010). Nature-inspired metaheuristic algorithms. Luniver press.

Yezioro, A. (2009). A knowledge based CAAD system for passive solar architecture. Renewable Energy, 34(3), 769-779.

Zeleny, M. (1998). Multiple criteria decision making: eight concepts of optimality. Human Systems Management, 17(2), 97-107.

Websites

http://buro-os.com/the-interlace/
last access 2017-09-08

http://derstandard.at/2000013547519/Die-Angst-vor-der-Stadt
last access 2017-08-30

http://sdwebx.worldbank.org/climateportal/index.cfm?page=country_historical_climate&ThisCCode=SGP
last access 2017-09-08

http://www.dictionary.com/browse/optimization
last access 2017-08-21

http://www.energyplus.net/weather
last access 2017-08-18

http://www.food4rhino.com/
last access 2017-08-23

http://www.graphisoft.com/archicad/rhino-grasshopper/
last access 2017-08-19

http://www.grasshopper3d.com
last access 2017-08-23

http://www.grasshopper3d.com/forum/topics/what-are-random-seed-values
last access 2017-08-23

http://www.webquest.hawaii.edu/kahihi/sciencedictionary/C/climatezone.php
last access 2017-09-08

https://www.aspern-seestadt.at/
last access 2017-08-30

https://www.australiandesignreview.com/architecture/interlace-oma/
last access 2017-09-08

https://www.energie-lexikon.info/heizgradtage.html
last access 2017-08-18

https://www.rhino3d.com/
last access 2017-09-08

https://www.wien.gv.at/statistik/lebensraum/wetter/
last access 2017-09-08

List of figures

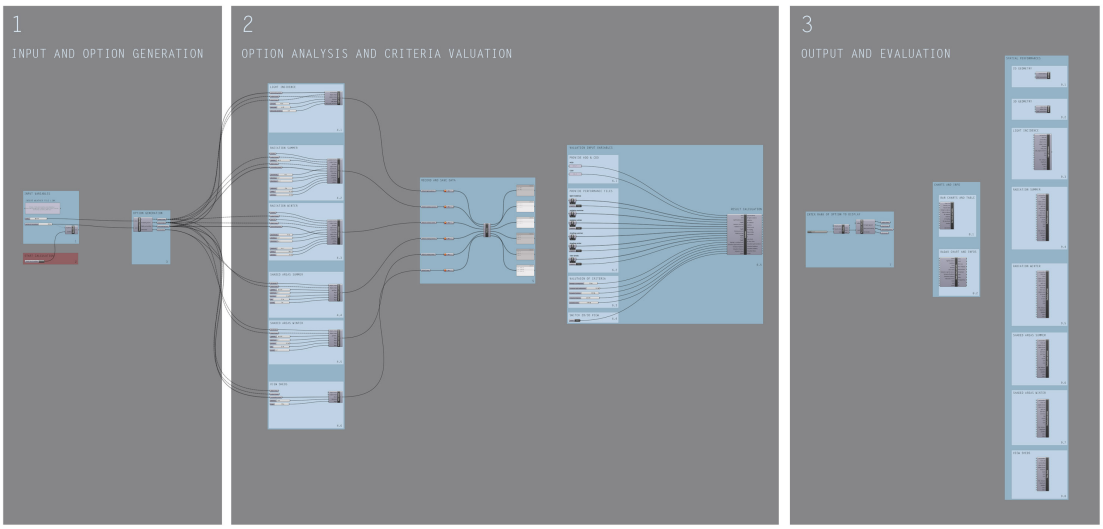
Note: All figures, which don't have any source given are done by the author

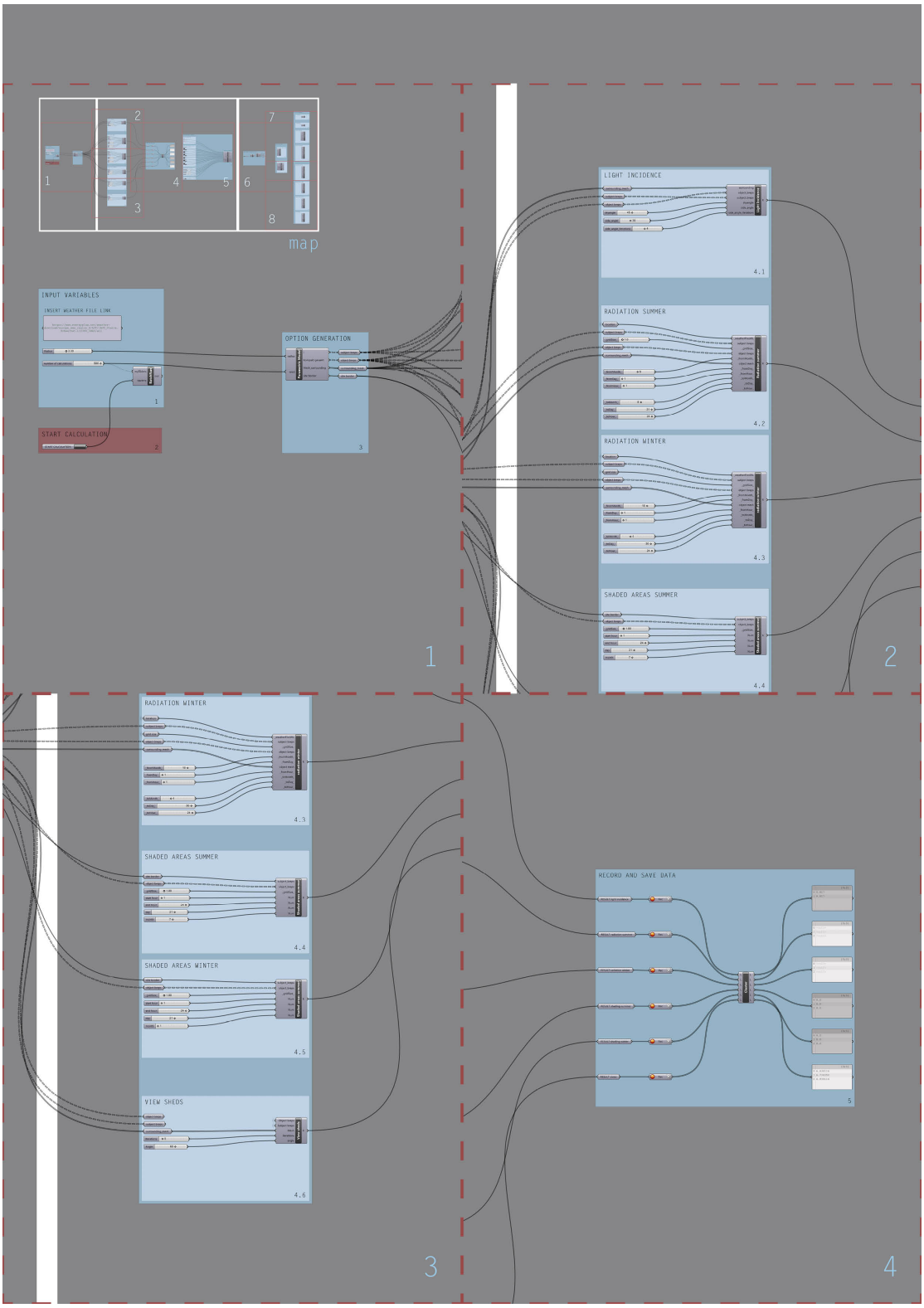
List of abbreviations

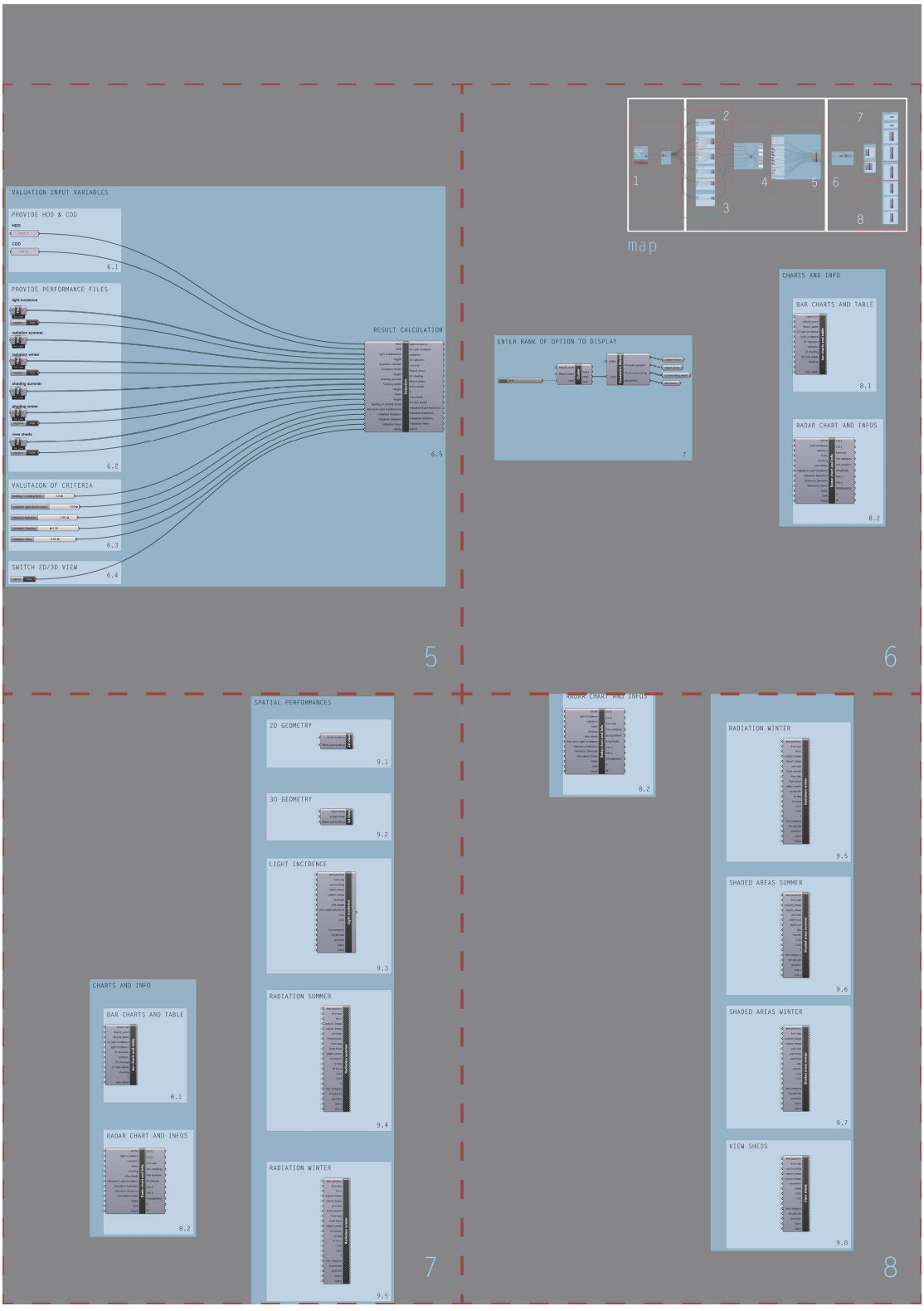
BIM	Building Information Modeling
BPET	Building Performance Evaluation Tool
CDD	Cooling Degree Days
DSE	Design Space Evaluation
HDD	Heating Degree Days
LI	Light incidence
SA	Shaded areas
SF	Significance factor
SR	Solar radiation
VA	View sheds

Appendix

The appendix is showing the grasshopper script for the project *up and down*. Its structure is very similar to the scripts of the other presented projects, with exception of some project specific small differences such as the number of evaluation criteria used. The parametric model obviously is unique for each project. The shown nodes are clusters containing sub-script, which is summarized in those nodes and which is much more complex and spacious than the super script presented here. For that reason it is impossible to show all components of the script.




map
INPUT VARIABLES
INSERT WEATHER FILE LINK
START CALCULATION
OPTION GENERATION
LIGHT INCIDENCE
RADIATION SUMMER
RADIATION WINTER
SHADED AREAS SUMMER
RADIATION WINTER
SHADED AREAS SUMMER
SHADED AREAS WINTER
VIEW SHEDS
RECORD AND SAVE DATA
1
2
3
4


VALUATION INPUT VARIABLES
PROVIDE HDD & CDD
HDD
CDD
6.1
PROVIDE PERFORMANCE FILES
light incidence
radiation summer
radiation winter
shading summer
shading winter
view sheds
6.2
VALUTAION OF CRITERIA
6.3
SWITCH 2D/3D VIEW
6.4
RESULT CALCULATION
6.5
5
map
ENTER RANK OF OPTION TO DISPLAY
7
CHARTS AND INFO
BAR CHARTS AND TABLE
8.1
RADAR CHART AND INFOS
8.2
6
SPATIAL PERFORMANCES
2D GEOMETRY
9.1
3D GEOMETRY
9.2
LIGHT INCIDENCE
9.3
RADIATION SUMMER
9.4
RADIATION WINTER
9.5
CHARTS AND INFO
BAR CHARTS AND TABLE
8.1
RADAR CHART AND INFOS
8.2
7
RADAR CHART AND INFOS
8.2
RADIATION WINTER
9.5
SHADED AREAS SUMMER
9.6
SHADED AREAS WINTER
9.7
VIEW SHEDS
9.8
8