



generative gestaltung
.grundlagen digitaler gestaltungsprozesse
computational architecture
.möglichkeiten und anwendung in der architektur

DANIEL NIEB
master-thesis



generative gestaltung
.grundlagen digitaler gestaltungsprozesse
computational architecture
.möglichkeiten und anwendung in der architektur

DANIEL NIEB
master-thesis

2012

generative gestaltung
computational architecture

Entstanden im Rahmen einer Master-Thesis an der
Hochschule Karlsruhe | University of Applied Sciences
bei Prof. Dipl. Ing. Meissner, Prof. Dipl. Ing. Günster

Satz, Layout, Grafiken, Texte und alle
Abbildungen sind, sofern keine Quelle
angegeben ist, von mir erstellt worden.

© Daniel Nieb, Juni 2012

0. einführung

- . einleitung | motivation . II
- . zieldefinition . IV
- . methodik . V

1. kontext | basiswissen

- theoretischer kontext
 - . begriffserklärung | definition . 2
 - . von wcomputerized zu >> computational design . 3
- technischer kontext
 - . programmierung . 6
 - . programmiersprache . 7
 - . geometrische, parametrische und assoziative modelle . 8
 - . parametrische und algorithmische gestaltungsprozesse . 10

2. methodischer kontext | entwurfswerkzeuge

- . agentensysteme . 14
- . attraktoren . 16
- . stochastische suchverfahren . 17
- . zelluläre automaten . 18
- . fraktale . 20
- . morphing . 21
- . genetische algorithmen . 22

3. software | entwurfsumgebung

- . re-aktiv | co-aktiv | pro-aktiv . 27
- . programme . 29
- . rhinoceros 3d | grasshopper . 30
- . maya – mel | python . 34
- . processing . 38

4. anwendung | ein veränderter entwurfsprozess?

- . der generative entwurfsprozess
 - . vorgehen und unterschiede . 44
 - . top-down und bottom-up . 47
- . beispiele
 - . kaisersrot . 48
 - . olympiastadion peking . 48
 - . wohn- und gewerbebesiedlung kalkbreite zürich . 50
 - . aperiodic symmetries . 52
 - . hyphae lamp . 54

5. entwurf | auswertung des eigenen entwurfsprozesses

- . entwurfseinführung
 - . standort
 - . makroebene | stadt . 58
 - . mikroebene | städtebauliche situation . 60
 - . verkehrsanbindung . 62
 - . öpvn . 64
 - . nutzung
 - . makroebene | hauptnutzungsschwerpunkte . 66
 - . mikroebene | umfeldnutzung . 68
 - . nutzungskonzept . 70
- . entwurfsprozess
 - . phase 1 | analyse . 72
 - . phase 2 | konzeptidee . 74
 - . phase 3 | algorithmus . 76
 - . phase 4 | auswertung . 77
 - . phase 5 | optimierung . 78
 - . phase 6 | bewertung . 79
 - . phase 7 | entwurf . 80

6. reflexion

- . chancen für die architektur . 84
- . fazit . 85
- . ausblick . 87

- anhang

- . quellenverzeichnis . 90
- . abbildungsverzeichnis . 92
- . danke . 93

vorwort

„Thus, architects wishing to use this new tool must not only become hackers (so that they can create the code needed to bring extensive and intensive aspects together) but also be able "to hack" biology, thermodynamics, mathematics, and other areas of science to tap into the necessary resources.“

Manuel De Landa

Mit dieser Arbeit bewege ich mich zwischen unterschiedlichen wissenschaftlichen Disziplinen und verlasse in vielen Bereichen das mir vertraute Terrain der Architektur.

Um die Thematik einfach und verständlich aufbereiten zu können, verlangt es zunächst einen tiefen Einstieg in verschiedene Disziplinen, wie Informatik, Mathematik oder auch Biologie und Physik. Da ich im Rahmen dieser Arbeit auf diesen Wissensgebieten nur als Laie agieren kann, arbeite ich selektiv und beschränke mich auf die mir wesentlich erscheinenden Teilgebiete.

Dabei versuche ich stets fachspezifische Termini weitestgehend zu vermeiden bzw. falls nötig zu erläutern und die Brücke zu architektonischen Fragestellungen zu schlagen.

einleitung | motivation

Zunehmend erlangen Bauwerke in freien Formen und aufsehenerregende Fassadenstrukturen Aufmerksamkeit sowohl in der Architekturfachwelt als auch der breiten Öffentlichkeit.

Diese avantgardistisch anmutenden, oft mit dem Titel »digitale Architektur« versehenen Entwürfe zeigen zwar die gegenwärtige Leistungsfähigkeit von Computern und Software in der Architektur, der dahinter stehende Entwurfsprozess hat sich jedoch nur in relativ wenigen Fällen grundlegend geändert.

Früher von Hand mit Stift und Reißbrett, Schere und Kleber oder Hammer und Meißel ausgeführten Techniken zur Darstellung werden lediglich virtualisiert und durch Software nachgeahmt.

Dadurch ist sicherlich vieles komfortabler und effizienter geworden und ermöglichte erst eine Erweiterung der architektonischen Formensprache, wie wir sie heute erleben, die bestehende Vorgehensweise von Entwerfen und Planen wird allerdings immer noch nur selten überdacht, sondern lediglich »computerisiert«. Parallel hat sich ein zweiter Ansatz herausgebildet, der nicht neu ist, dem aber in den letzten Jahren immer mehr Aufmerksamkeit geschenkt wird: die Generative Gestaltung oder Computational Design, also das parametrische und algorithmische Entwerfen.

Häufig zu unrecht von einer einseitigen, formalen Betrachtung geprägt, eröffnet dieser alternative Ansatz völlig neue Möglichkeiten, die das Potential des Computers zu nutzen vermag:

nämlich durch informationstechnologische Erfassung, Bearbeitung und Nutzung verschiedenster Parameter die sinnvollsten Lösung im Hinblick auf eine Problemstellung zu entwickeln.

Den Computer als Assistent, ob beim Entwurf oder der Darstellung, war bereits während meines Studiums essentieller Bestandteil. Ständig versuchte ich neue Programme zu erkunden und bei der Arbeit einzusetzen. Während des Verlaufs meiner Studienzeit wurde Computational Design in der zeitgenössischen Architektur populärer und begann auch immer stärker mein Interesse zu wecken. »Wie funktioniert das?« war eine Frage, die mich ständig begleitete und der ich mit dieser Arbeit auf den Grund gehen möchte.

Ich denke vor dem Hintergrund zunehmender Digitalisierung aller Lebensbereiche der heutigen Gesellschaft wird die Architektur nicht ausgeschlossen bleiben. Ich möchte das als Chance begreifen und versuchen mich auf diesem Gebiet weiterzuentwickeln.

"Der Beruf des Architekten ist eine abenteuerliche Tätigkeit: Ein Grenzberuf in der Schweben zwischen Kunst und Wissenschaft, auf dem Grat zwischen Erfindung und Gedächtnis, zwischen dem Mut zur Modernität und echter Achtung der Tradition."

Renzo Piano

zieldefinition

Ziel dieser Arbeit ist es eine solide Grundlage dafür zu schaffen, wie dieser Gestaltungsprozess genutzt werden kann, generelle Möglichkeiten einer Anwendung von Generativer Gestaltung (Computational Design) im architektonischen Bereich aufzuzeigen, gewonnene Erkenntnisse als Architekt in einem praktischen Teil zu verarbeiten und als Entwurfsmethodik darzustellen.

Damit wird versucht ein Einstieg in das Themenfeld zu ermöglichen, Basiswissen für eine weitere Vertiefung zu schaffen und somit die Disziplin des Computational Design in der Architektur weiter voran zu treiben.

Im Mittelpunkt stehen dabei folgenden Fragen:

Was ist mit Computational Design oder mit parametrischem und algorithmischem Entwerfen in der Architektur gemeint?

Welche Möglichkeiten bietet diese Technik für den Entwurf? Wie funktioniert sie?

Verändert sich der Entwurfsprozess dadurch grundlegend und wo liegen die Unterschiede zu klassischen Entwurfsmethoden?

methodik

Einarbeitung in Programmiersprachen, insbesondere Processing, Java, VVVV, Visual Basic

Einarbeitung in Programme Rhinoceros 3D Grasshopper, Cinema 4D, Maya, 3ds Max

Quellenanalysen zu den Themengebieten: Generative Gestaltung, parametrisches- algorithmisches Entwerfen, Code generierte Architektur

Projektanalysen in den Gestaltungsgebieten: Grafik I Design I Kunst I Architektur

Der erste Teil dieser Arbeit wird kurz und selektiv auf den theoretischen Kontext des Themengebietes dieser Master-Thesis eingehen.

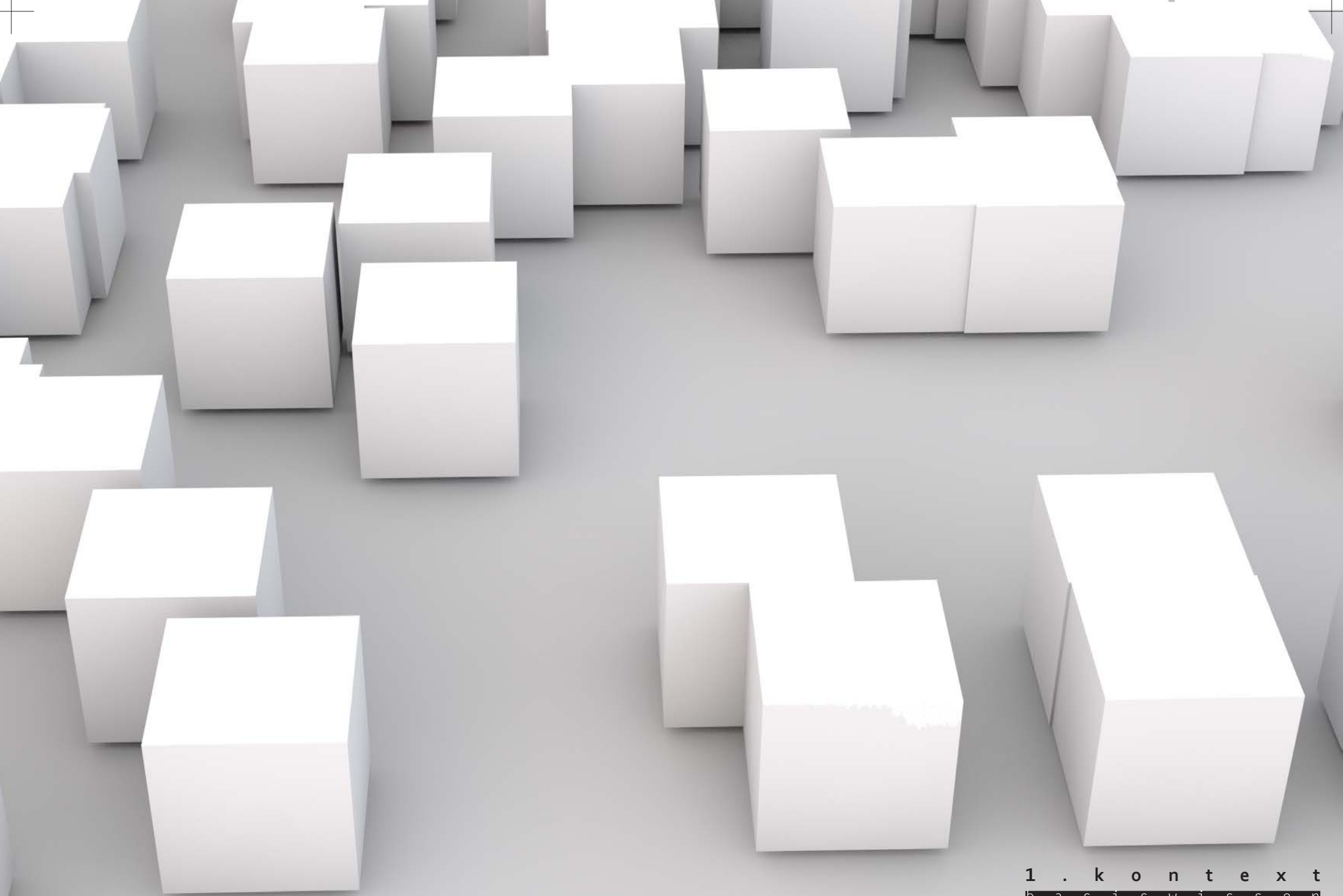
Darin wird zunächst der Begriff Computational Design geklärt, einen Abriss dessen Geschichte gegeben und sich mit den technischen Grundlagen auseinandergesetzt.

Im zweiten Teil soll an Hand einer Zusammenstellung von für den architektonischen Entwurfsprozess interessant scheinende Methoden und deren Algorithmen vorgestellt und die Brauchbarkeit und Einsetzbarkeit hinsichtlich architektonischer Fragestellungen untersucht werden.

Im dritten Teil wird die Rolle der Software dargestellt. Dabei werden exemplarisch drei Programme vorgestellt.

Im vierten wird der Frage nach dem veränderten Entwurfsprozess nachgegangen und durch eine Projektsammlung gezeigt, welche Möglichkeiten sich hinter generativer Entwurfsprozesse verstecken.

Im fünften Teil wird am konkreten Beispiel des Master-Thesis Entwurfes der Entwurfsprozess analysiert und ausgewertet. Dabei wird die Herangehensweise in einzelne Phasen aufgespalten und erklärt.



1 . k o n t e x t
b a s i s w i s s e n

begriffserklärung I definition

Die generative Gestaltung (engl. Computational Design) bezeichnet eine Methodik, wie Gestaltung (Bild, Objekt, Klang) durch Abarbeiten einer prozessualen Erfindung, das heißt eines vom Gestalter geschaffenen Regelwerkes, erzeugt werden kann. Dieses wird üblicherweise in Form von Programmcodes erzeugt und durch Computerprogramme ausgeführt.

Eine Idee wird also nicht mehr »von Hand« geschaffen, sondern anhand „definierter, regelbasierter Prozeduren und parametrisch beschriebener Verknüpfungen“ (1) umgesetzt.

Die Folge ist, dass eine solche Prozedur nicht nur beispielsweise ein einzelnes Bild erzeugen kann, sondern dass durch Veränderung von Parametern ganze Bilderwelten entstehen.

(2) Folglich sind diese Prozesse grundsätzlich dadurch gekennzeichnet, dass sie emergent, also ergebnisoffen sind. (3)

(1) Siehe: Achim Menges: Architektonische Form- und Materialwerdung am Übergang von Computer Aided zu Computational Design, in Detail „Analog und Digital“, 50. Serie 2010, 5, S.421

(2) Schmidt-Friderichs GmbH (o.J.): „Generative Gestaltung“. <http://www.generative-gestaltung.de/> [Stand:21.04.2012. Zugriff 21.04.2012, 10:12 MESZ]

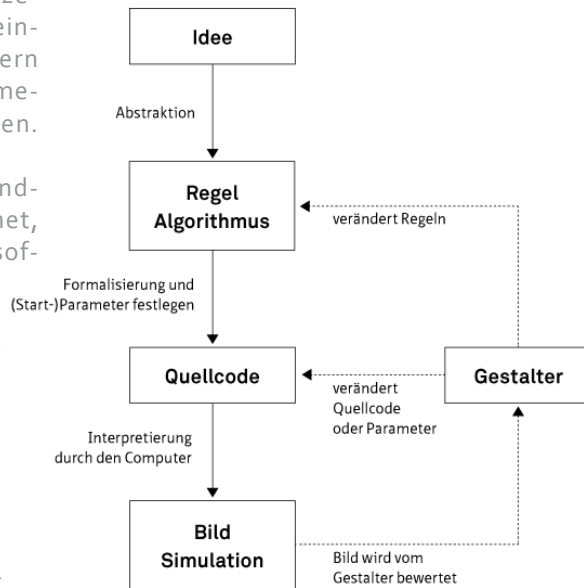
(3) Nikolaus Kuhnert, Anh-Linh Ngo: Entwerfen im digitalen Zeitalter, in Editorial ARCH+ 189 „Entwurfsmuster“, Oktober 2008, S.6-9

Kein Autor (o.J.): „Generative Gestaltung“. <http://de.wikipedia.org/wiki/Generative_Gestaltung> [Stand:22.01.2012. Zugriff 20.04.2012, 18:41 MESZ]

Kein Autor (o.J.): „Generative Kunst“. <http://de.wikipedia.org/wiki/Generative_Kunst> [Stand:12.09.2010. Zugriff 20.04.2012, 18:43 MESZ]

Wesentlich ist beim Formulieren dieser Regelwerke die Gestaltungsidee in die abstrakte Logik eines Programmcodes zu übersetzen. Dabei ist es erforderlich, dass die Idee zur Gestaltung explizit ausgedrückt wird und dadurch auch immer gegenwärtig bleibt. Demnach kann ein häufiges Missverständnis, dass der Gestalter, also Architekt, bei dieser Entwurfsmethode vom Computer ersetzt wird, hier ausgeräumt werden.

Vielmehr wird nämlich die Rolle des Gestalters gestärkt, da er sich von Beginn an der Problemstellung bewusst sein muss und anstelle des Nutzers digitaler Prozesse zu deren Entwickler wird.



Copyright Hartmut Bohnacker, Julia Laub, Benedikt Groß, Claudius Lazzaroni (2009) Buch „Generative Gestaltung“, www.generative-gestaltung.de

von computerized zu >> computational design

Um die Rolle des Computers in der generativen Gestaltung zu verstehen, muss zunächst eine Unterscheidung in der Art der Computernutzung vollzogen werden.

Im englischen Sprachgebrauch wird dabei zwischen »Computerization« und »Computation« unterschieden.

„Computation is a term that differs from, but is often confused with, computerization. While computation is the procedure of calculating, i.e. determining something by mathematical or logical methods, computerization is the act of entering, processing, or storing information in a computer or a computer system. Computerization is about automation, mechanization, digitization, and conversion. Generally, it involves the digitization of entities or processes that are preconceived, predetermined, and well defined. In contrast, computation is about the exploration of indeterminate, vague, unclear, and often ill-defined processes; because of its exploratory nature, computation aims at emulating or extending the human intellect. It is about rationalization, reasoning, logic, algorithm, deduction, induction, extrapolation, exploration, and estimation.“

Siehe: Terzidis, Kostas. Algorithmic architecture (2006). Oxford: Architectural Press S. XI

»Computerization« wird als Vorgang der Eingabe, Aufbereitung oder des Speicherns von Informationen in einem Computer definiert, während »Computation« einen Prozess der Berechnung, also der Bestimmung durch mathematische oder logische Methoden beschreibt.

Die zentrale Bedeutung von Computational Design liegt also nicht im reinen Einsatz eines Computers, sondern vielmehr im Prozesshaften und Informationsbasierten.

Demnach ist die heutige Computernutzung in der Architektur, also der Einsatz von CAD-Programmen (Computer-aided design), überwiegend des Computerized Design zuzuordnen.

Die ersten CAD-Programme arbeiteten jedoch noch in einer anderen Weise. Sie waren zwar auch Systeme zur Speicherung und Visualisierung von geometrischen Informationen. Die Informationen bzw. der visuelle Output wurde jedoch durch Programmiersprachen erzeugt. (4)

Ende der 1960er Jahre wurden dann Grafikprogramme zur Erstellung von Zeichnungen programmiert. Zunächst noch vorwiegend im Flugzeugbau und nur von wenigen Architekten benutzt, fand bereits in den 1980er Jahren ein Umdenken statt. Zwar wurde immer noch am Reißbrett gezeichnet, jedoch drängte sich die neue Technologie immer mehr ins Bewusstsein und deren Benutzer wurden weniger als Utopisten denn

als Spezialisten verstanden.

Als in den 1990er Jahren CAD-Software flächendeckend in die Architekturpraxis Einzug hielt, änderte sich auch schnell deren Funktionsweise und wurde zunehmend zur Computerization genutzt. Das frühere Reißbrett wurde digitalisiert und der Arbeitsplatz des Architekten an den Computer verlegt.

Aufgrund der Einführung einer grafischen Benutzeroberfläche konnten diese Programme einer breiteren Masse zugänglich gemacht werden. Obwohl die Programme so ohne Kenntnisse von Programmiersprachen benutzt werden konnten, wurden gleichzeitig die allgemeinen regelbasierten Prozeduren auf vorgefertigte Werkzeuge reduziert (5).

Demnach erlaubten noch zu Beginn der 1990er Jahren die meisten CAD-Programme ausschließlich den Entwurf orthogonaler Objekte (6).

Im Jahr 2001 schrieben Dörte Kuhlmann und Heimo Schimek in ihrem Buch *Cybertecture*:

„Zu erkennen, dass Kreativität durch Softwarelimitierung eingeschränkt werden sollte, war für einige Architekten wahrscheinlich mit ein Grund sich dieser Arbeitsweise zu entziehen und andere Methoden auszuprobieren, die die Verwendung »artfremder« Software notwendig machten.“

Siehe: Dörte Kuhlmann, Heimo Schimek (Hrsg.) (2001): *Cybertecture*. Die 4. Dimension in der Architektur. Wien: Löcker

Einige Architekten versuchten sich also ein Stück weiter von den Fesseln der Software zu befreien und begannen die fachfremde Anwendung von Programmen aus der Film- und Computerspielindustrie, sowie dem Industrie- und Produktdesign. Das eröffnete der Architektur neue formale Welten und die Zeit der »Blob-Architektur«, wie sie von Greg Lynn erstmals bezeichnet wurde, war angebrochen.

Die entstandenen Entwürfe haben allerdings in der Mehrzahl noch nichts mit generativer Gestaltung, also mit Computation, gemein, sondern sind auch in der Computerization verankert. Es wurden hier lediglich analoge Modellieretechniken (wie beispielsweise Kneten) digital nachgeahmt.

Ein Stufe weiter ging man mit der Entwicklung von CAAD-Programmen (Computer-aided architectural design) und dem bauteilorientierten Modellieren BIM (Building Information Modeling). Hier werden sämtliche Gebäudedaten digital erfasst, kombiniert und vernetzt. Das heißt, alle relevanten Bauteile, wie zum Beispiel Wände, Türen oder Fenster, werden zueinander in Abhängigkeit gebracht. Über Parameter lassen sich deren Abmessungen kontrollieren und verändern oder alle wichtigen Eigenschaften, wie zum Beispiel Dämmwert, Material oder Kosten, zuweisen. Ziel dieses Ansatzes ist es durch Zusammenfassen von vielen bauwerksrelevanten Informationen den Planungsprozess zu optimieren und zu

beschleunigen. Allerdings muss auch hier wieder auf die vorgefertigten digitalen Werkzeuge zurückgegriffen werden. Damit wird das Wesen des kreativen Arbeitens, nämlich fortwährend alternative Wege zu suchen und innovative Konzepte zu entwickeln, durch Softwarelimitierung erschwert oder gar verhindert. (7)

„Wie häufig in der Baugeschichte ergibt sich jedoch eine gewisse Überschneidungsphase, in der neue Technologien zunächst mehrheitlich als Erweiterung der herkömmlichen Praxis genutzt werden. Parallel dazu entwickeln sich neuartige Ansätze, die den tatsächlichen Potenzialen der Technologie entsprechen und so beginnen, die Praxis selbst zu verändern“ (8)

Neben diesen Entwicklungen entstanden in den 1990er-Jahren im Rahmen von experimentell ausgerichteten Universitäten bereits weltweit Initiativen zum Computational Design, welche die »tatsächlichen Potenziale der Technologie« zu nutzen vermag.

Erst in den letzten Jahren ist allerdings ein Trend zu erkennen, der durch zahlreiche Veröffentlichungen wie beispielsweise »Design by Numbers« von John Maeda (2001) oder »Algorithmic architecture« von Kostas Terzidis (2006) bestärkt, Programmierung als grundlegende gestalterische Methode versteht. Derzeit werden auch in Deutschland Programmierkurse zunehmend in die

Ausbildung von Architekten integriert und neue spezialisierte Institute, wie »Informatik in der Architektur« an der Bauhaus-Universität Weimar oder »Computational Design« an der Universität Stuttgart, entstehen. (9) Zuletzt bezeichnet Patrick Schumacher diesen »Parametrismus« als neuen »Internationalen Style« der zeitgenössischen Architektur. (10)

(4) Vgl.: John Frazer (1995): *An Evolutionary Architecture*. London: Architectural Association Publications, S.55.

(5) Vgl.: Philip Belesky (2010): »Ghost in the Machines: Parametric architecture and the philosophy of giles deleuze« <<http://www.manifoldblog.com/?p=116>> [Stand: 14.02.2010. Zugriff 10.04.2012, 16:45 MESZ]

(6) Vgl.: Svenia Schneider: »digitale Form im architektonischen Entwurfs- und Arbeitsprozess zu Beginn des 21. Jahrhunderts in: Caroline Y. Robertson-von Trotha, Robert Hauser (Hg.) (2011): *Neues Erbe. Aspekte, Perspektiven und Konsequenzen der digitalen Überlieferung*. Karlsruhe: KIT Scientific Publishing S.247-257

(7) Kein Autor (o.J.): »Building Information Modeling«. <http://de.wikipedia.org/wiki/Building_Information_Modeling> [Stand: 21.03.2012. Zugriff 22.04.2012, 10:31 MESZ]

(8) Siehe: Achim Menges: *Architektonische Form- und Materialwerdung am Übergang von Computer Aided zu Computational Design*, in *Detail* »Analog und Digital«, 50. Serie 2010, 5, S.420

(9) Reinhard König (2012): »Entwicklung des parametrischen und algorithmischen Entwerfens«. <<http://entwurforschung.de/entwicklung-des-parametrischen-und-algorithmischen-entwerfens/>> [Stand: 09.04.2012. Zugriff 22.04.2012, 11:00 MESZ]

(10) Siehe: Patrick Schumacher: *Parametrismus. Der neue internationale Style*, in: *ARCH+* Nr.195 »Istanbul wird grün«, November 2009 S.106-113

Da eine oberflächliche und abstrakte Behandlung für das Verständnis von Funktionsweisen und Möglichkeiten der Programme nicht ausreichend ist, ist zunächst eine Auseinandersetzung mit den technischen Grundlagen notwendig. Dabei soll nur ein grundsätzliches Basiswissen vermittelt werden, welches zum Verständnis notwendig ist. Es wird daher im Folgenden selektiv und nicht vollständig auf den technischen Kontext eingegangen.

computerprogrammierung

Das Wort Programmierung leitet sich vom griechischen πρόγραμμα »Vorschrift« ab (11). In diesem Sinn wird es auch in der Computerprogrammierung verstanden: als Vorschriften oder Regeln, die ein Computer ausführt.

Ein Computer besteht im Grunde aus einer Vielzahl von elektronischen Schaltern, die ein- oder ausgeschaltet werden können. Durch verschiedene Kombinationen dieser Schalter kann erreicht werden, dass ein Vorgang ausgeführt wird. Bei der Programmierung geht es nun darum, diese Schalter geschickt so zu kombinieren, dass der Computer solch einen gewünschten Vorgang ausführt. Die Umsetzung (Implementierung) des Vorgangs (Softwareentwurf) in ein Programm setzt voraus zu wissen, welche Kombination von Schaltern nötig ist und wie dies dem Computer mitgeteilt werden kann. An diesem Punkt wird eine Kommunikationsform zwischen Mensch und »Maschine« notwendig: die Programmiersprache. (12)

(11) Siehe Bibliographisches Institut GmbH (o.J): „Bedeutung; Rechtschreibung: Programm“. <<http://www.duden.de/rechtschreibung/Programm>> [Stand: 2012. Zugriff 25.04.2012, 10:55 MESZ]

(12) Microsoft (2007): „Grundlagen: Wie Programmierung funktioniert?“ <[http://msdn.microsoft.com/de-de/library/ms172579\(v=vs.90\).aspx](http://msdn.microsoft.com/de-de/library/ms172579(v=vs.90).aspx)> [Stand: 2012. Zugriff 25.04.2012, 11:12 MESZ]

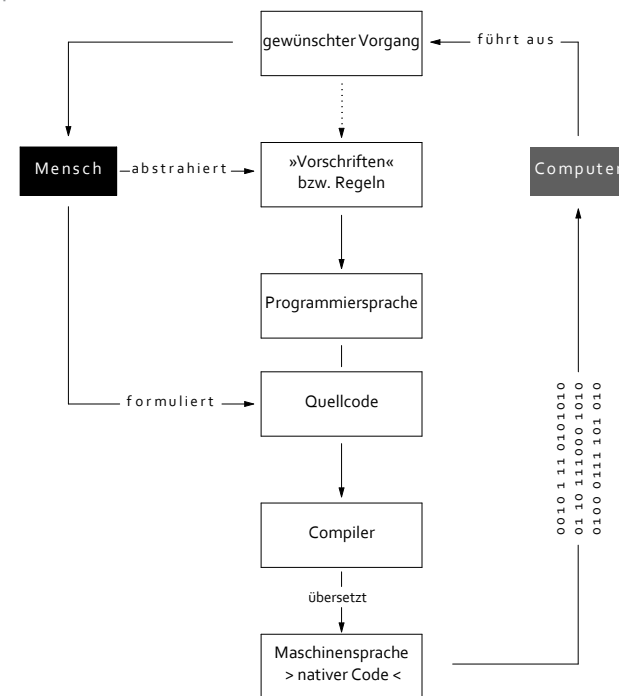
programmiersprache

Nach der Deutschen Industrie-Norm (DIN) 44300, Teil 4 versteht man unter einer Programmiersprache, „eine zum Abfassen von Computerprogrammen geschaffene Sprache.“

Menschen drücken sich mithilfe von Sprache aus, die viele unterschiedliche Zeichen besitzt. Computer verwenden eine sehr einfache Sprache, die nur aus 0 und 1 besteht. 0 bedeutet »aus«, und 1 bedeutet »an«. Der Versuch, mit einem Computer in seiner eigenen Sprache zu sprechen, ähnelt dem Versuch, mit einem Freund per Morsezeichen zu kommunizieren. Es ist möglich, aber kompliziert. (12)

So wie sich verschiedene menschliche Sprachen entwickelt haben, wurden auch unterschiedliche »Computersprachen« in Anbetracht des Einsatzbereiches und im Spannungsfeld zwischen Bequemlichkeit, Verständlichkeit und Übersetzbarkeit entwickelt. Je näher eine Programmiersprache am Einsatzbereich ist, desto bequemer und leichter ist sie dort anzuwenden, desto ungeeigneter ist sie in anderen Bereichen. Dabei werden analog der menschlichen Sprache auch Dialekte entwickelt, das heißt Abwandlungen der höheren Programmiersprachen, die stark auf ein gewissen Einsatzbereiche spezialisierte sind.

Die Programmiersprache stellt da eine einfachere Kommunikationsform zwischen Mensch und Computer bereit. Als Quelltext oder Quellcode verfasst, ist es eine Übersetzung der computereigenen Sprache (Maschinensprache oder auch nativer Code genannt) in eine für Menschen besser verständliche, leichter erlernbare Sprache um dem Computer Anweisungen zu geben.



(Einige Beispiele dazu werden im Kapitel 3 vorgestellt)

Die Anweisungen des Quellcodes einer bestimmten Programmiersprache wird von einem dafür speziellen Programm (Compiler) in die Maschinsprache übersetzt. (13) Das bedeutet, dass der Anwender einer Programmiersprache nicht wissen muss, was der Computer macht oder wie er es macht. Er muss lediglich wissen, wie die Programmiersprache funktioniert. Entsprechend muss der Anwender eines Programmes, nicht dessen Quellcode verstehen können, sondern lediglich wissen, wie das Programm funktioniert. Daraus kann abgeleitet werden, dass die puren Anwender eines Programmes an die vom Programmierer festgelegten Funktionen und Rahmenbedingungen gebunden sind. Hingegen sind die Anwender einer Programmiersprache, also die Programmierer selbst, gänzlich frei in ihren Ideen.

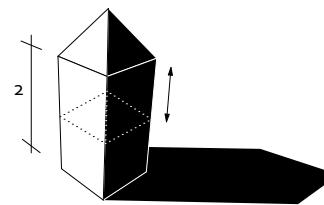
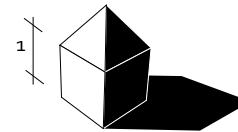
(13) Kein Autor (o.J.): „Programmierung“. <<http://de.wikipedia.org/wiki/Programmierung>> [Stand: 14.04.2012. Zugriff 25.04.2012, 11:11 MESZ]

geometrische, parametrische und assoziative modelle

Zum Verständnis der computerbasierten Entwurfsmethoden ist es wichtig zu wissen, wie physikalische Objekte im Computer abgebildet werden. Als ein geometrisches Modell wird dabei die computergestützte Beschreibung der Form geometrischer Elemente (wie Punkten, Kanten, Flächen, Volumen) bezeichnet. Diese werden durch ihre Eigenschaften, wie zum Beispiel Koordinaten im Raum, datentechnisch repräsentiert. Ein Linie kann so beispielsweise als Verbindung zweier Punkte beschrieben werden; die Punkte wiederum als x- | y-Koordinaten in der Fläche oder x- | y- | z-Koordinaten im Raum. Diese Elementei-

genschaften sind zur Darstellung im Computer unbedingt notwendig und können auf dieser Ebene als Parameterwerte angesehen werden. Aus dieser Sicht ist jedes geometrische Modell auch ein parametrisches Modell.

Wird ein geometrisches Modell so erstellt, dass zwischen den geometrischen Elementen bzw. deren Parametern Abhängigkeiten (Assoziativität) bestehen, spricht man von einem assoziativen Modell. Bei dieser Technik wirkt sich eine Veränderung eines Parameters auf alle assoziierten Parameter aus. Stellt man sich beispielsweise eine Pyramide auf einem Quader mit der Höhe 1 vor, kann bei einem assoziativen Modell nun die Position im Raum der Pyramide mit den Parametern der Höhe des Quaders verknüpft werden.



Damit wird durch Änderung der Höhe des Quaders von 1 auf 2 auch die Lage der Pyramide im Raum neu definiert und geändert.

Schwierigkeiten bereitet bei assoziativen Modellen eine netzartige Assoziativität, bei der sich Elemente in einer zirkulären Abhängigkeit zueinander befinden.

Eine solche Vernetzung ist aufgrund der damit einhergehenden Komplexität schwer zu handhaben und erfordert bestimmte Kontrollmechanismen, welche die Bedingungen der Interaktion zwischen zwei Elementen genau definieren. Eine solche Definition ist mittels Algorithmen anzugeben, weshalb netzartig aufgebaute assoziative Modelle um spezielle Methoden ergänzt werden müssen. Aus der dargestellten Unterscheidung zwischen geometrischen und assoziativen Modellen ergeben sich verschiedene Möglichkeiten für deren Einsatz im architektonischen Entwurfsprozess. Es wird unterschieden zwischen Methoden des parametrischen und algorithmischen Entwurfs. (14)

(14) Vgl.: Reinhard König (2012): „Entwicklung des parametrischen und algorithmischen Entwurfs“. <<http://entwurforschung.de/entwicklung-des-parametrischen-und-algorithmischen-entwurfs/>> [Stand: 09.04.2012. Zugriff 22.04.2012, 11:00 MESZ]

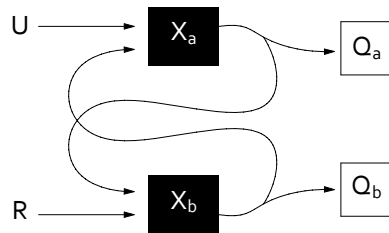
parametrische und algorithmische gestaltungsprozesse

Beim parametrischen Gestaltungsprozess werden zunächst Objekte durch ihre Parameter definiert. Im Weiteren können mittels assoziativen Verknüpfungen dieser Objekte komplizierte Abhängigkeiten erfasst und daraus die Gestalt generiert werden. In einem nächsten Schritt wird es möglich durch Veränderung von Parametern auf einfache und schnelle Weise aus der selben Anzahl von Parametern nicht nur ein Objekt, sondern eine Reihe von Variationen des selben Typus zu generieren und weiter zu optimieren.

Wesentlicher Unterschied zum herkömmlichen computerbasierten Gestaltungsprozess liegt dabei im Vorgehen wie geometrische Modelle erstellt werden: Sie werden nicht zeichnerisch, sondern mathematisch erfasst. Die Parameter sind dabei bestimmte Variablen der Geometrie wie x - | y -Koordinaten, Distanzen oder Radien, die über definierte Abhängigkeiten miteinander in Beziehung stehen (15). Eine Geometrie wird also nicht vom Gestalter entworfen und ist demnach nicht durch ihre Gestalt definiert. Vielmehr ist sie durch die vom Gestalter entworfenen Regeln und Abhängigkeiten, den sogenannte Algorithmen, beschrieben.

Im Unterschied zum parametrischen werden jedoch beim algorithmischen Gestaltungsprozess netzartige Para-

meterverknüpfungen berücksichtigt. Das heißt, dass es zu Interaktionen und Rückkoppelungen zwischen Objekten kommen kann. Diese Prozesse spielen bei komplexen Systemen eine wichtige Rolle.



„Ein Algorithmus ist eine aus endlich vielen Schritten bestehende eindeutige Handlungsvorschrift zur Lösung eines Problems oder einer Klasse von Problemen.“ (16)

Im Kontext dieser Arbeit werden Algorithmen als ein Computerprozedere zur Lösung von Problemstellungen verstanden. Sie erfordern Abstrahierung, Generalisierung, Ableitung und strukturierte Logik.

Ein Algorithmus kann beispielsweise mit den Anweisungen eines Kochrezeptes verglichen werden. Die Abfolgen von »Einkaufen der Zutaten, diese vorzubereiten, diese zu kombinieren, sie zu kochen und zu servieren« sind algorithmische Schritte der Essenzubereitung.

Theoretisch ist ein Algorithmus also die Abstraktion eines Prozesses und bietet ein Muster an, welches zur Ausführung der gewünschten Aufgabe führt.

Zum Beispiel wäre der Algorithmus zur Zubereitung von Bratkartoffeln aus folgenden Schritten aufgebaut:

1. Schälen
2. Abkochen
3. Zerkleinern
4. Braten
5. Servieren



Würden diese Schritte durchmischt, einer hinzugefügt oder weggelassen werden, entstünde ein alternatives Rezept, welches auch als Aufgabe die Zubereitung von Bratkartoffeln hätte. Das Resultat könnte jedoch besser, gleich oder schlechter als die ursprüngliche Absicht sein. Indessen können Abänderungen, Wahllosigkeit oder andere Zwischenfälle im Prozess vielleicht zu einer neuen Lösung führen. Keine, die man im Voraus in Betracht zog, aber die möglicherweise hilft, die Problemstellung besser zu verstehen oder eine mögliche Lösung zu finden.

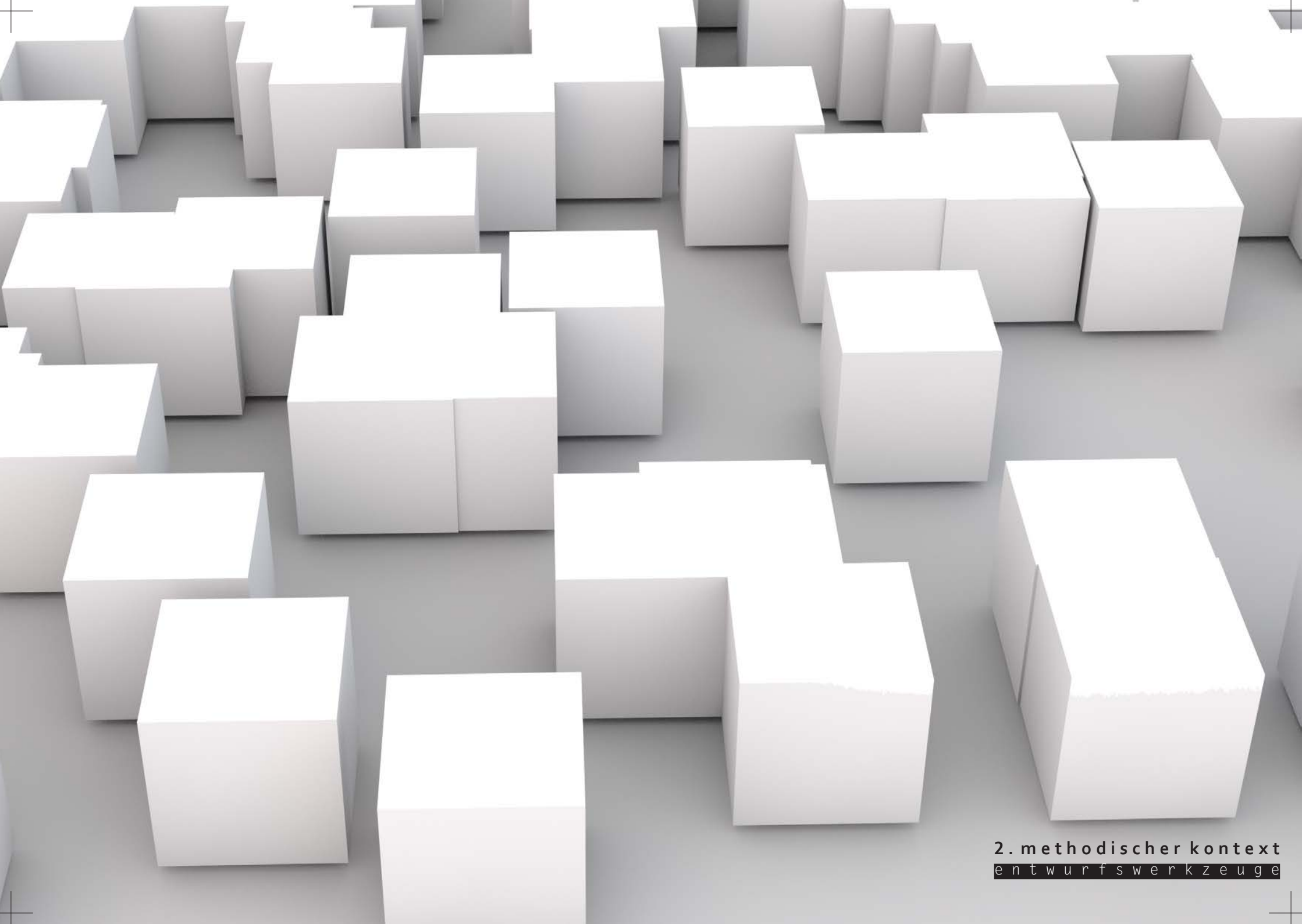
Während beim Entwerfen der meisten Algorithmen der Lösungsweg schon bekannt ist, gibt es jedoch einige Problemstellungen, deren Lösung noch unbekannt oder nur vage beschrieben werden kann. In diesem Fall können Algorithmen dazu benutzt werden, mögliche Wege zu erkunden, die vielleicht auf eine Lösung führen. (17)

Im nächsten Kapitel werden einige Entwurfswerkzeuge und deren Algorithmen vorgestellt und erklärt.

(15) Vgl.: Reinhard König (2012): „Entwicklung des parametrischen und algorithmischen Entwerfens“. <<http://entwurforschung.de/entwicklung-des-parametrischen-und-algorithmischen-entwerfens/>> [Stand: 09.04.2012. Zugriff 22.04.2012, 11:00 MESZ]

(16) Siehe: Kein Autor (0.J);„Algorithmus“ . <<http://de.wikipedia.org/wiki/Algorithmus>> [Stand: 24.04.2012. Zugriff 27.04.2012, 13:04 MESZ]

(17) Vgl.: Terzidis, Kostas. Algorithmic architecture (2006). Oxford: Architectural Press. S.15; 65-66



2. methodischer kontext
entwurfswerkzeuge

In diesem Kapitel werden für den architektonischen Entwurfsprozess interessant scheinende Methoden und deren Algorithmen vorgestellt und erklärt.

Nach 5 Sek.



30 Sek.



60 Sek.



...

Der Weg eines einfachen Agenten

agentensysteme

Ein Agent ist eine Einheit, die sich nach verschiedenen Verhaltensregeln frei und autonom bewegen kann.

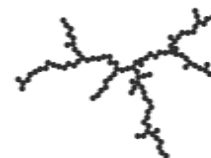
Ein einfaches Beispiel für einen Agenten ist ein Kreis, der sich in acht Richtungen fortbewegen kann. Nach jedem Schritt bewegt er sich weiter in eine zufällig gewählte Richtung. Nebenstehende Grafik ist mittels eines solchen Agenten entstanden.

Ein Agentensystem besteht aus mehreren gleichartigen oder unterschiedlich spezialisierten, selbstständig handelnden Einheiten. Dabei ist es möglich, dass einzelnen Einheiten interagieren und kollektiv in der Lage sind, ein Problem zu lösen.

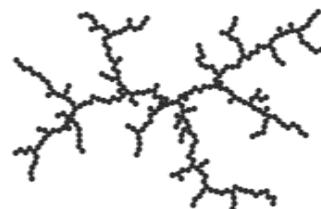
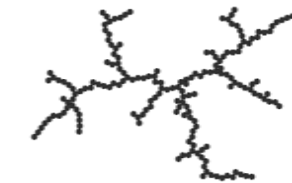
Anwendung können diese Systeme in der Simulation von Bewegung in Gebäuden (z.B. Evakuierungsverhalten) finden. (18)

(18) Oliver Fritz: Programmiertes Entwerfen, in ARCH+ Nr.189 „Entwurfsmuster“, Oktober 2008, S.60-61

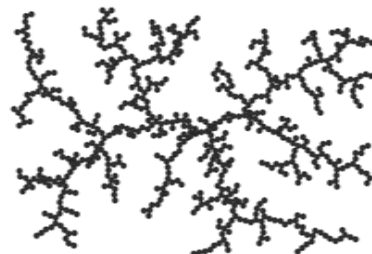
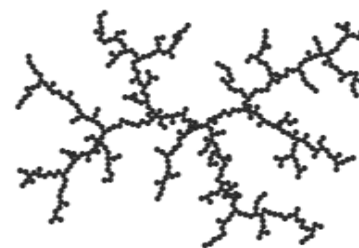
Nach 10 Sek.



30 Sek.



60 Sek.



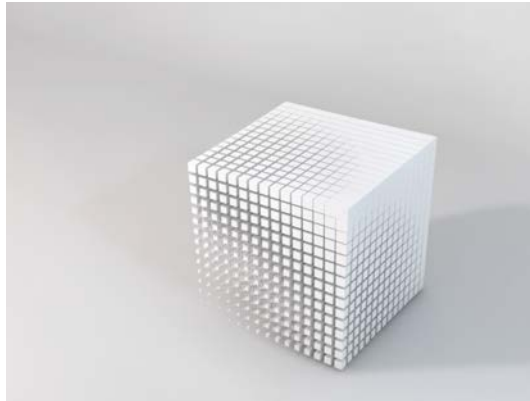
Wachstumsstruktur

Mittels Agenten können auch Wachstumsprozesse beschrieben oder Verdichtungsstrukturen hergestellt werden. Ein Algorithmus für eine einfache Wachstumsstruktur wäre beispielsweise:

Erzeuge ein Kreis und lass ihn sich so lange fortbewegen bis er auf ein feststehendes Objekt trifft. Dann stoppe ihn und erzeuge wieder ein Kreis mit den gleichen Eigenschaften.

attraktoren

Attraktoren sind virtuelle Magnete, also gedachte - und programmierte - Punkte im Raum, die andere Objekte verändern können. Beispielsweise anziehen, abstoßen oder aber auch verformen (z.B. skalieren). Ähnlich wie bei einem Magneten haben Attraktoren auf Objekte, die sich in ihrer Nähe befinden, starken Einfluss, auf entferntere wenig oder keinen. Attraktoren stellen eine wichtige generative Methode dar, da sie Grenzen erzeugen und Elemente verformen beziehungsweise allgemein verändern können.



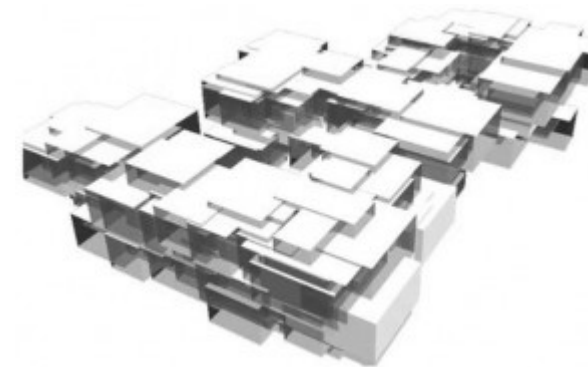
In diesem System aus Würfeln und einem einzelnen Attraktorpunkt wirkt sich die Entfernung zum Attraktor auf die Größe der Würfel aus. Je näher die Würfel dem Attraktor kommen, desto stärker werden sie skaliert. Natürlich kann so ein System auch aus mehreren Attraktoren oder auch unterschiedlichen Objekten bestehen.

stochastische suchverfahren

Das Wort »Stochastik« stammt aus dem griechischen stochastike (téchnē) und bedeutet »zum Erraten gehörend(e Kunst)«. Im Duden wird es wie folgt definiert:

„Teilgebiet der Statistik, das sich mit der Untersuchung vom Zufall abhängiger Ereignisse und Prozesse befasst“

Ein stochastisches Suchverfahren wird im Kontext der generativen Gestaltung als ein Algorithmus verstanden, der zufällig einen Bereich absucht bis er eine bestimmte Bedingung vorfindet.



Wohnungsbauprojekt, welches das stochastische Suchverfahren zur Bestimmung der Lage der Wohneinheiten nutzt (23)

Stellt man sich folgende Problemstellung vor: eine Reihe von Wohneinheiten soll in einem definierten Baufeld verteilt werden, sodass sie sich nicht überschneiden, aber dennoch alle in die Grenzen des Baufeldes passen,

kann ein stochastisches Suchverfahren angewandt werden, um eine Lösung zu generieren. Der Algorithmus dazu wäre beispielsweise wie folgt aufgebaut:

1. Platziere eine Wohneinheit an eine beliebige Position innerhalb des Baufeldes.
2. Vergleiche sie mit allen vorherigen Wohneinheiten, falls es eine Überschneidung gibt beginne von vorne, ansonsten platziere sie und beginne dann von vorne, solange bis keine Wohneinheiten mehr zu platzieren sind.

Natürlich kann dieser Algorithmus auch modifiziert werden, sodass zum Beispiel nur Überschneidungen zugelassen werden, indem die Logik umgekehrt wird. Die Bedingungen für eine Platzierung können aber auch so komplex werden, dass sie verschiedene architektonische Anforderungen erfüllen (z.B. öffentlicher Raum, Sonneneinstrahlung, Abstandsflächen, Erschließung etc.) (23)

(23) Vgl.: Terzidis, Kostas. Algorithmic architecture (2006). Oxford: Architectural Press. S.88

zelluläre automaten

Ein zellulärer Automat ist eine regelmäßige Anordnung von Zellen. Meist wird ein rechteckiges Gitternetz aus identischen Zellen verwendet. Dabei kann jede Zelle eine endliche Zahl von Zuständen annehmen und hat eine begrenzte Zahl von Nachbarzellen, mit der sie in Wechselwirkung steht und die sie beeinflussen können. Die Zustände der einzelnen Zellen und das dadurch generierte Muster des gesamten zellulären Automaten ändert sich in einzelnen Schritten, welche durch sogenannte Zellübergangsregel(n) bestimmt werden, die gleichwohl für alle Zellen gelten. Nach jedem Schritt werden die Regel(n) auf alle Zellen des Automaten angewandt und eine neue Generation entsteht.

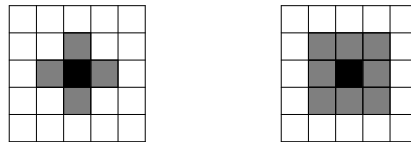
Die Bestandteile eines zellulären Automaten, nämlich zum einen die Zellen und zum anderen die Übergangsregeln zur Berechnung des nächsten Zustandes einer Zelle, sind sehr einfach strukturiert, bringen aber komplexe Muster und Systeme hervor. (20)

Der Grundgedanke eines zellulären Automaten ist, ein komplexes System nicht mit komplexen mathematischen Gleichungen zu beschreiben, sondern die Komplexität aus einfachen Wechselwirkungen von Individuen, die einfache Regeln befolgen, entstehen zu lassen. (21)

Dabei sind vier Eigenschaften zu unterscheiden:

Erstens, die Geometrie der Zellanordnung. Eine 1-dimensionale Anordnung, in denen die Zellen in einer Reihe liegen, eine 2-dimensionale Anordnung von beispielsweise quadratischen Zellen, wie etwa bei einem Schachbrett oder z.B. hexagonalen Zellstrukturen. Auch sind 3- oder gar 4-dimensionale Anordnungen denkbar, aber nicht in jedem Falle anschaulich darstellbar.

Zweitens muss festgelegt werden, welche Nachbarn eine Zelle haben soll. Zum Beispiel in einem Quadratgitter könnte man vier Zellen als Nachbarn (Neumann-Nachbarschaft) angeben, aber auch acht Zellen (Moore-Nachbarschaft) als Nachbarn sind möglich.



Die dritte Kenngröße eines zellulären Automaten ist die Zahl der Zustände pro Zelle. Am einfachsten sind Automaten mit nur zwei Zuständen zu handhaben, die man als 0 oder 1 oder aber »sichtbar« und »unsichtbar« interpretieren kann.

Als viertes sind die Übergangsregeln zu nennen. Die große Anzahl möglicher zellulärer Automaten entspringt ihrem großen Variantenreichtum an



Zelluläre Automaten mit 2-dimensionalem Gitternetz und Würfeln als Zellen

Ihre Struktur und ihr Verhalten kann aber auch bei architektonischen, landschaftlichen oder städtebaulichen Problemstellungen nützlich sein. Von ländlicher Ansiedelung und sozialen Wechselwirkungen zu Materialverhalten und Luftzirkulation können zelluläre Automaten interessante Interpretationen von urbanen und architektonischen Phänomenen liefern. (21) Außerdem bietet die Fähigkeit Muster zu generieren eine vielseitige formale Anwendung und Erweiterung architektonischer Gestaltung.

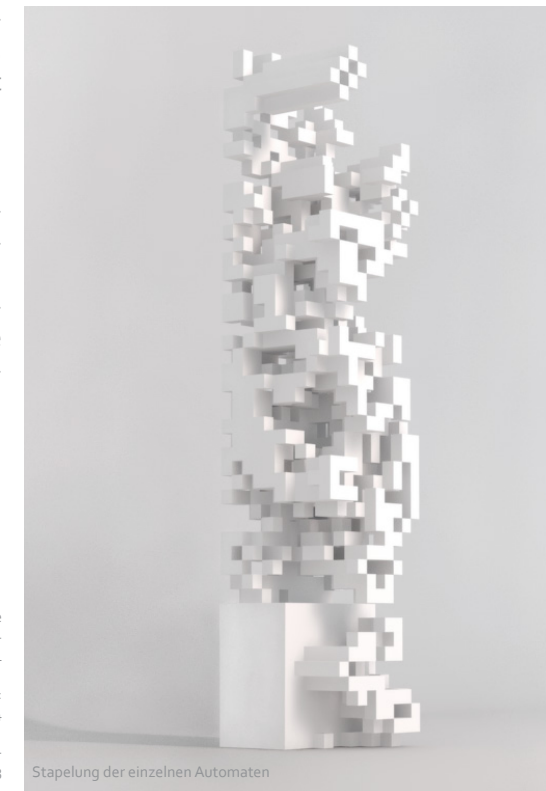
Regeln, die man für das "Zusammenleben" der Zellen angeben könnte. Wenn eine Zelle z.B. k Zustände hat und jeweils n Nachbarn, dann kann man k hoch k hoch n mögliche Regeln formulieren. Bei einer Mooreschen Nachbarschaft und jeweils 2 Zuständen sind es so schon 10 hoch 77 Regeln. (22)

Zelluläre Automaten wurden ursprünglich entwickelt um organische selbst-eretzende Systeme zu beschreiben.

(20) Vgl.: Reinhard König, Christian Bauriedel: „Computergenerierte Stadtstrukturen“. Diplomarbeit an der Techn. Universität Kaiserslautern, 2004. S.51

(21) Vgl.: Terzidis, Kostas. Algorithmic architecture (2006). Oxford: Architectural Press. S.94

(22) Hans-Georg Beckmann: „Zelluläre Automaten“. Virtuelle Lehrerbildung im Fach Informatik in Niedersachsen, 2003



Stapelung der einzelnen Automaten

fraktale

Ein Fraktal ist ein natürliches oder künstliches bzw. geometrisches Objekt, welches durch rekursive oder iterative Operationen ein sich wiederholendes Muster generiert. Diese Muster weisen eine unendliche Detailtiefe auf und besitzen zum Teil eine selbstähnliche Struktur, die nach unterschiedlichen Iterationsstufen auftritt. Das ist beispielsweise der Fall, wenn ein Objekt aus vielen verkleinerten Objekten seiner selbst besteht. Der Begriff Fraktal wurde 1975 von Benoît Mandelbrot geprägt und stammt vom lateinischen fractus »gebrochen« (von lateinisch frangere »in Stücke (zer-)brechen«).

Der Algorithmus für beispielsweise den »Menger-Schwamm« setzt sich iterativ aus folgenden Schritten zusammen:

1. Ausgangslage ist ein Würfel. 2. Zerlege nun jede Oberfläche des Würfels (bzw. jede der Teilwürfel) in neun Quadrate und generiere daraus $27 = 3 \cdot 3 \cdot 3$ Teilwürfel. 3. Lösche jeden Würfel in der Mitte jeder Oberfläche, sowie den Würfel im inneren des großen Würfels. 4. Wende Schritt 1-3 auf jeden verbleibenden kleineren Würfel an. (19)

Durch ihren Formenreichtum besitzen Fraktale einen großen ästhetischen Reiz, der bereits die Kunstform der Fraktalkunst hervorbrachte und formal auch im architektonischen Gestaltungsprozess eingesetzt werden kann.

(19) Vgl.: Kein Autor (o.J): „Menger-Schwamm“. <<http://www.mathetreff-online.de/mathelexikon/m/menger-schwamm>> [Stand: 08.09.2011. Zugriff 28.04.2012, 10:40 MESZ]



Menger-Schwamm 1. bis 4. Iterationsstufe

morphing

Morphing oder auch Hybridisierung beschreibt einen Prozess der Transformation eines Startobjektes in ein Endobjekt. Dabei verändert ein Objekt kontinuierlich seine Gestalt, um die Gestalt eines anderen Objektes anzunehmen.

Dieser Prozess besteht grundsätzlich aus der Auswahl von zwei Objekten und den Zwischen-Transformationschritten. Das erste Objekt transformiert sich dann schrittweise in das zweite.

Anwendung findet Morphing vor allem in der Film- und Werbeindustrie. In der Architektur könnte der Nutzen einer solchen Operation nicht im Vorgang selbst liegen, also der erfolgreichen Transformation in das Endobjekt. Der Reiz geht vielmehr von den dazwischen liegenden Zuständen aus, die das Startobjekt durchläuft. Architektonisches Morphing kann demnach die strukturelle Integrität zweier Objekte bewahren. Ein Würfel wird beispielsweise allmählich in eine Pyramide transformiert. Dabei existieren zwar immer zwei Objekte, das Start und Endobjekt, man kann es jedoch auch als nur ein Objekt ansehen, welches von einem Zustand in einen anderen wechselt. Dieses Objekt kombiniert Charakteristika von beiden

Ursprungsobjekten, die in die Transformation involviert sind und kann als Hybrid verstanden werden, welcher aus der Topologie des einen und der Geometrie eines anderen Objektes zusammengesetzt ist. (24)



(24) Vgl.: Terzidis, Kostas. Algorithmic architecture (2006). Oxford: Architectural Press. S.97-100

genetische algorithmen

Genetische Algorithmen sind Optimierung- und Suchverfahren, die sich an die Grundprinzipien der biologischen Evolution anlehnen. Sie finden meist Verwendung bei komplexen Systemen, in denen es nicht nur einen Lösungsweg gibt.

Daher geht es in Optimierungsverfahren häufig nicht darum die beste Lösung zu finden, sondern nur eine Reihe von guten bzw. optimalen Lösungen, auch als Lösungsraum bezeichnet.

In diesem Zusammenhang bedeutet optimal, eine Zielfunktion (auch Bewertungs- oder Fitnessfunktion) zu maximieren oder zu minimieren. Dabei hat die Zielfunktion die Aufgabe eine mögliche Lösung mit dem Lösungsraum zu vergleichen und zu qualifizieren. Gute Lösungen bekommen dabei einen höheren Wert als schlechte. Gesucht wird demnach die Lösung bei der die Werte der Zielfunktion am höchsten sind.

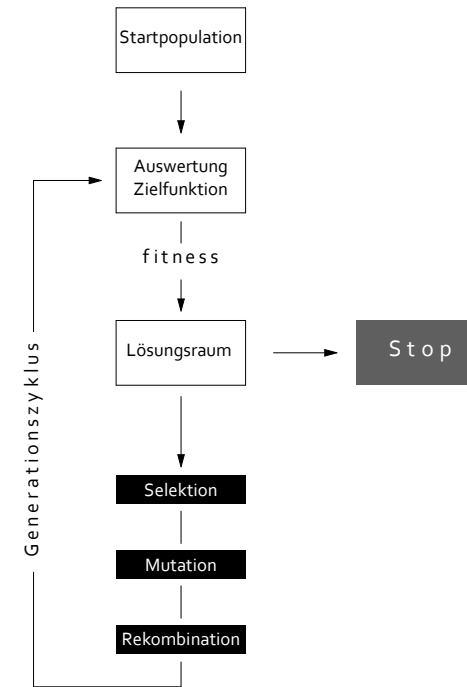
Genetische Algorithmen verwenden nun Prinzipien aus der Evolutionstheorie um ständig neue Lösungen zu generieren, die dann anhand der Zielfunktion auf Maximalwerte getestet werden können. Grundlegende Idee dabei ist, so lange Lösungsvorschläge zu produzieren bis eine ausreichend gute Lösung gefunden wurde. In diesem Prozess finden die Evolutionsmechanismen, Selektion, Mutation und Rekombination statt.

Zunächst erzeugt der Algorithmus eine Startpopulation, also eine erste Generation von Lösungsvorschlägen mit zufälligen Merkmalen.

Im ersten Schritt wird dann mit Hilfe der Zielfunktion deren »Fitness« überprüft und daraus eine Überlebenswahrscheinlichkeit errechnet. Diese wird für die Selektion benötigt. Im nächsten Schritt wählt der Algorithmus zufällig diejenigen Lösungsvorschläge aus, die die nächste Generation bilden und rekombiniert diese. Dabei spielt die Überlebenswahrscheinlichkeit eine große Rolle. Sie gibt die Wahrscheinlichkeit an mit der ein Lösungsvorschlag ausgewählt wird.

Die daraus neu entstandene Generation von Lösungsvorschlägen wird dann im letzten Schritt mittels Mutation verändert, um neue und insbesondere fittere Lösungen zu erhalten. Dabei verändert die Mutation zufällig ein oder mehrere Gene, also Merkmale des Lösungsvorschlages.

Die drei Schritte Mutation, Selektion und Rekombination können beliebig oft angewandt werden. Mit jeder Generation können Änderungen auftreten, die bessere oder aber schlechtere Lösungen enthalten. Nach einer gewissen Zahl von Generationen pendelt sich das System im Idealfall bei einer bestimmten Qualität von Lösung ein und verändert sich kaum noch, dann ist die Lösung mit der größten Fitness der Population gefunden. (25)



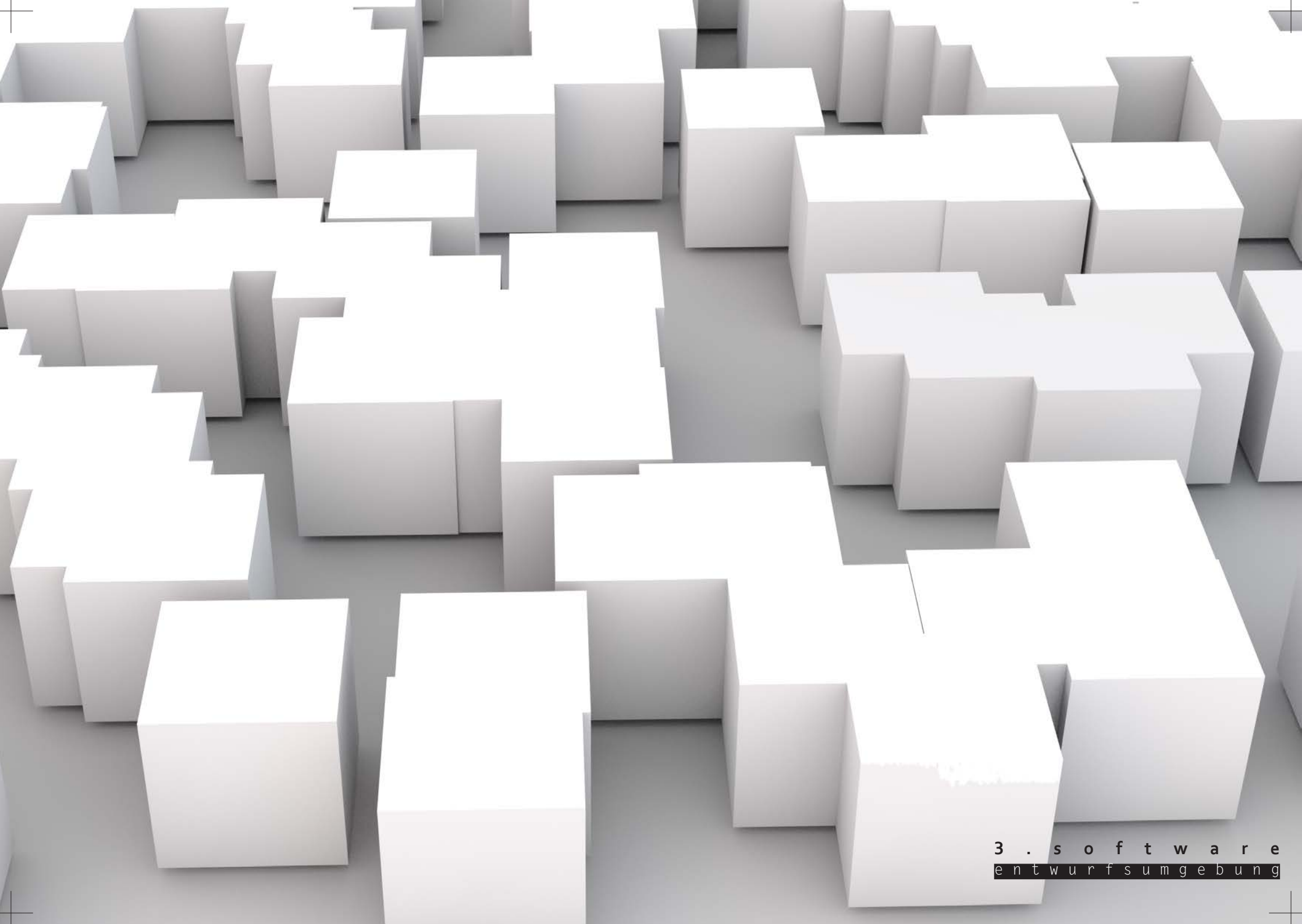
„survival of the fittest“

Herbert Spencer

Dieser Prozess kann auf eine Vielzahl von architektonischen Problemstellungen angewandt werden, insbesondere da er ohne großes Wissen über die Struktur des Problems umgesetzt werden kann, und da die Menge an zulässigen Lösungen und die Definition der Zielfunktion schon alle nötigen Informationen enthalten. Das bedeutet, es lassen sich viele unterschiedliche Parameter berücksichtigen über deren Zusammenspiel nicht viel Wissen notwendig ist. Dies macht genetische Algorithmen sehr flexibel und für die Architektur interessant: Ähnlich des evolutionären Prozesses der biologischen Morphogenese (Entstehung der Form von Lebewesen), welche die Form, Material und Struktur und deren Leistungsfähigkeit unter Einwirkung zahlreicher Kräfte vereinigt, geht es auch im Architektorentwurf um die „kreative Synthese vielfältiger Faktoren zur Definition von Organisation, Raum und Form.“ (26)

(25) Vgl.: Steffen Harbich: „Einführung genetischer Algorithmen mit Anwendungsbeispiel“. E-Paper der Universität Magdeburg. 2007
Siehe auch: Kein Autor (o.J.): „Evolutionärer Algorithmus“. <http://de.wikipedia.org/wiki/Evolutionärer_Algorithmus> [Stand: 21.04.2012. Zugriff 28.04.2012, 18:10 MESZ]

(26) Stefan Gruber: Von der Formfindung zur performativen Struktur, in ARCH+ Nr.189 „Entwurfsmuster“, Oktober 2008, S.117



3 . s o f t w a r e
e n t w u r f s u m g e b u n g

„If you're using someone else's software,
you're living in someone else's dream.“

John Maeda

Grundsätzlich kann man sagen, dass nicht jedes Computertool für jede Anwendung und jeden Anwender gleich gut geeignet scheint. Es gibt Unterschiede im Einsatzbereich, der Bequemlichkeit und Handhabung oder der Komplexität.

Mittlerweile herrscht eine undurchschaubare Dichte und Vielzahl von CAD-Systemen, die speziell auf Architekten zugeschnitten, jedoch nicht zwingend zum parametrischen oder gar algorithmischen Entwerfen geeignet sind. Außerdem scheinen Programme fachfremder Disziplinen, wie zum Beispiel Grafikdesigns oder der Filmindustrie, interessant und ein näherer Blick lohnenswert. Neu ist auch, dass sich Programme entwickelt haben, welche die Grenze zwischen Entwicklern von Programmen und Anwendern von Programmen verwischen. Das heißt, dass Entwickler Programme nutzen und auch Anwender selbst Programme schreiben. Das bedeutet keineswegs, dass man sich als Laie das Fachwissen eines Programmierers ohne weiteres aneignen kann, aber dass es möglich ist, ohne spezielle Ausbildung, innerhalb kurzer Zeit anzufangen, Programme zu schreiben.

Weder die Programme noch die Strategien damit zu arbeiten sind neu, aber die Benutzeroberflächen sind soweit vereinfacht worden, dass Programmieren für Gestalter zugänglich wurde.

Im Folgenden soll geklärt werden welche Rolle Programme beziehungsweise Programmiersprachen in der generativen Gestaltung einnehmen. Dabei wird zunächst generell der Einsatz von Computerprogrammen beleuchtet, um dann exemplarisch drei Programme näher zu betrachten: Ein bereits teilweise etabliertes Programm unter Architekten und vor allem Architekturstudenten (Grasshopper), ein Programm aus der Film- und Computerspielindustrie (Autodesk Maya, insbesondere die programminterne Sprache MEL), sowie ein Tool aus dem Bereich des Mediendesigns (Processing).

re-aktiv | co-aktiv | pro-aktiv

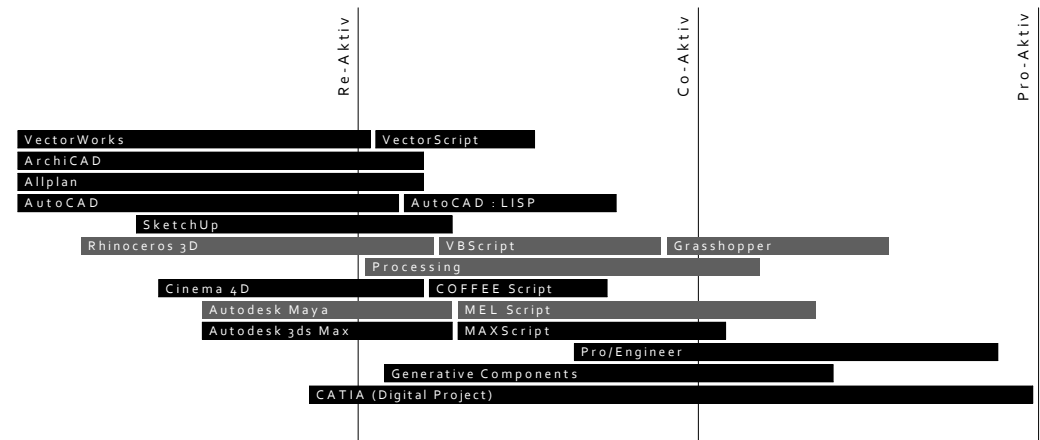
Generell kann man Computerprogramme zur architektonischen Gestaltung in drei Kategorien aufteilen. Ein wichtiges Kriterium zur Unterscheidung bietet dabei der Grad der Benutzer-Interaktivität und damit die Visualisierung der Veränderungen in Echtzeit.

Zur ersten Kategorie der »re-aktiven« Programme gehören vor allem CAD-Systeme, die Architekten zur Anfertigung ihrer 2D- und 3D-Zeichnungen gebrauchen. Diese Programme basieren auf Vektoren bzw. geometrischen Modellen, die in einer Zeichenoberfläche mittels Mauseingabe indirekt oder über die numerische Angabe von Eigenschaftswerten direkt eingegeben werden und über geringe oder keine Möglichkeit der parametrischen Verknüpfung verfügen.

Diese Art der Computernutzung ist der Computerization zuzuordnen und ist zwar für Architekten ein geeignetes und einfach zu bedienendes Werkzeug, spielt jedoch in der generativen Gestaltung und im Rahmen dieser Arbeit eine untergeordnete Rolle.

Allerdings können viele dieser CAD-Systeme durch ihre zugrunde liegenden Programmiersprachen »ferngesteuert«, also direkt bedient werden, und ihre Funktionalität wird demnach erweitert, sodass sie bedingt die Funktionalität der zweiten Kategorie, der »co-aktiven« Programmen besitzen (beispielsweise AutoCAD mit AutoCAD-LIPS oder VectorWorks mit VectorScript).

Co-aktive Programme verfügen über einen Quelltext-Editor, der Eingriffe in die Terminologie des Programms erlaubt. Hier verwischen die Grenzen zwischen Entwicklern und Nut-



zern von Programmen. So kann auf parametrische Verknüpfungen zugegriffen werden, die sich dort auch editieren lassen. Des Weiteren lassen sich hier auch automatisierte Abläufe oder Algorithmen selbst erstellen. Diese zweite Gruppe eignet sich also zum algorithmischen und parametrischen Entwerfen. Voraussetzung dafür ist allerdings Fachkenntnis der programminternen Programmiersprache, welche Architekten in der Regel zunächst nicht beherrschen.

Die dritte Kategorie stellen die »proaktiven« Programme dar. Das sind vollparametrische und auf Attributen basierende Systeme, bei denen schon bei der Eingabe Verknüpfungen erstellt werden können. Ein wesentlicher Vorteil ist die freie Definition von Eigenschaften und generativen Algorithmen, welche diese Programme simultan visualisieren und so den Prozess leiten und strukturieren.⁽²⁷⁾ Die verschiedenen Parameter haben eine interaktive und wechselseitige Beziehung, welche nicht mit dem herkömmlichen Ursachen-Wirkungs-Prinzip, sondern eher mit einem Reiz-Reaktions-Modell erklärbar sind („stimulus-responsive mechanism“ wie es Patrik Schumacher nennt). Der Unterschied lässt sich gut am Beispiel des Unterschiedes zwischen dem Treten eines Balls und dem Treten eines Hundes erkennen. Während man einen Ball tritt, gibt es nur eine Ursache, das Treten des Balls, und dessen Wirkung, der Ball nimmt eine

andere Position ein. Tritt man jedoch einen Hund, hängt die Reaktion von mehreren Faktoren ab und kann dadurch nicht so einfach vorhergesagt werden. ⁽²⁸⁾

⁽²⁷⁾ Vgl. Martin Schroth: Digitale Formgenerierung, in ARCH+ Nr.189 „Entwurfsmuster“, Oktober 2008, S.124-125

⁽²⁸⁾ Vgl. Patrik Schumacher: Autopoietic Elegance in: MRGD Morphe, RIEA (Research Institute of Experimental Architecture), 2008

programme

„Es gibt wesentlich mehr Benutzer als Hersteller von Werkzeugen. Dieses Ungleichgewicht gründet auf den doch sehr unterschiedlichen Fähigkeiten, die man braucht, um ein Werkzeug zu benutzen, im Vergleich zu denen, die man braucht, um ein Werkzeug herzustellen. Um ein Werkzeug zu benutzen, braucht man nichts weiter zu tun, als mit der Maus auf einem bestimmten Punkt zu klicken. Um ein neues Werkzeug zu kreieren, muss man auf jeden Fall die geheimnisvolle Kunst des Programmierens beherrschen.“

John Maeda (2004): Creative Code. Ästhetik und Programmierung am MIT Media Lab. Boston: Birkhäuser Verlag S.113

Während es 2004 noch unbedingt notwendig war die „geheimnisvolle Kunst des Programmierens“ zu beherrschen, ist es heute, acht Jahre später, bereits möglich, ohne weitreichende Programmierkenntnisse Algorithmen und Programmroutinen zu erstellen. Die Werkzeuge dazu sind frei zugänglich, manche sogar Open-Source-Projekte. Darunter finden sich auch einige, die speziell für Gestalter entwickelt wurden und sich deshalb sehr gut für Architekten eignen. Hier werden kurz die wichtigsten vorgestellt:

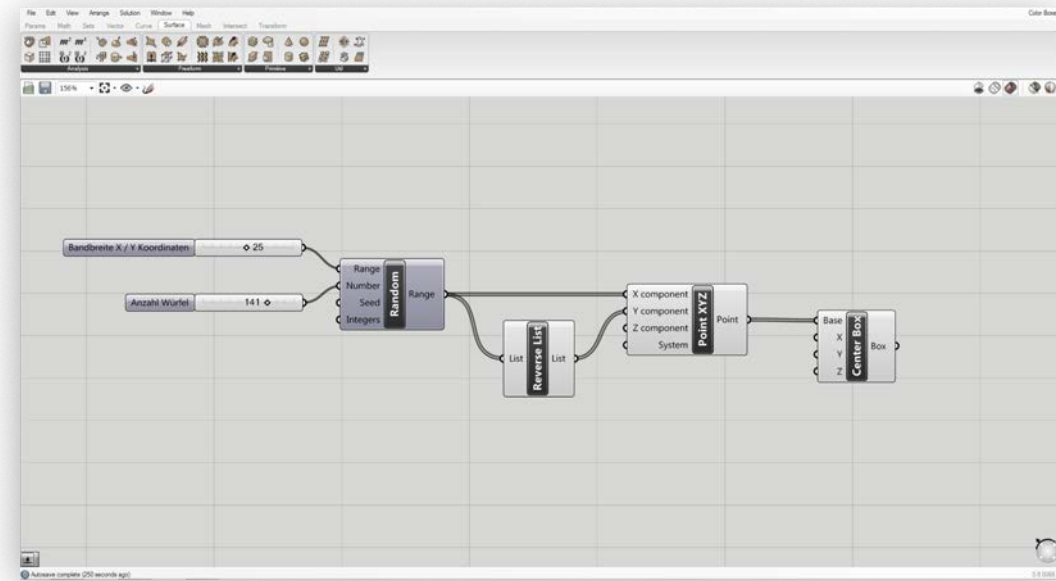
rhinoceros 3d - grasshopper

„For designers who are exploring new shapes using generative algorithms, Grasshopper® is a graphical algorithm editor tightly integrated with Rhino's 3-D modeling tools. Unlike RhinoScript, Grasshopper requires no knowledge of programming or scripting, but still allows designers to build form generators from the simple to the awe-inspiring.“



Scott Davidson (2012): „About Grasshopper...“. <<http://www.grasshopper3d.com/>> [Stand: 2012. Zugriff 01.05.2012, 16:06 MESZ]

Grasshopper zählt zu den pro-aktiven Programmen zugeordnet und ist ein sogenanntes Plug-In für das Modellierwerkzeug Rhinoceros 3D, das heißt, eine externe Softwareerweiterung. Die erste Version von Grasshopper, damals noch »Explicit History« genannt, wurde 2007 veröffentlicht und ist seitdem kostenlos erhältlich. Es ist eine Art visuelle Programmiersprache und daher gerade für Anwender mit wenig oder keiner Programmiererfahrung sehr gut geeignet. Es lassen sich komplizierte Systeme und Abhängigkeiten erzeugen und damit eignet es sich hervorragend zum parametrischen und generativen Entwerfen.



Grundidee von Grasshopper ist ein grafischer Editor. Das heißt, es wird hier der Quelltext nicht durch Text sondern mit grafischen Mitteln formuliert. Während Programmiersprachen eine kodierte, abstrakte Version in Form von Wort und Satz von Algorithmen sind, kann man sich den grafische Editor von Grasshopper als eine visuelle Version von Algorithmen vorstellen. Vergleichbar mit Flussdiagrammen sind sie verständlicher und flexibler und von Gestaltern wesentlich besser zu handhaben. (30)

Dennoch können Parameter-Abhängigkeiten zwischen Elementgruppen definiert und diese Abhängigkeiten mit vordefinierten oder selbst erstellten Algorithmen verknüpft werden.

Mit einigem Aufwand ist es auch möglich, zirkuläre Abhängigkeit, also Rückkopplungsprozesse zwischen geometrischen Elementen, festzulegen. Daher eignet sich das Plug-In auch zum algorithmischen Entwerfen.

Nachteil sind zwar die vorgefertigten Werkzeuge, das heißt es liegt eine Softwarelimitierung vor, jedoch lassen die sich ohne weiteres durch die programminterne Programmiersprache »VBScript« erweitern.

In jedem Fall gehört es zu den bedienungsfreundlichsten Systemen, welche es erlauben, mit wenigen Programmierkenntnissen und primär

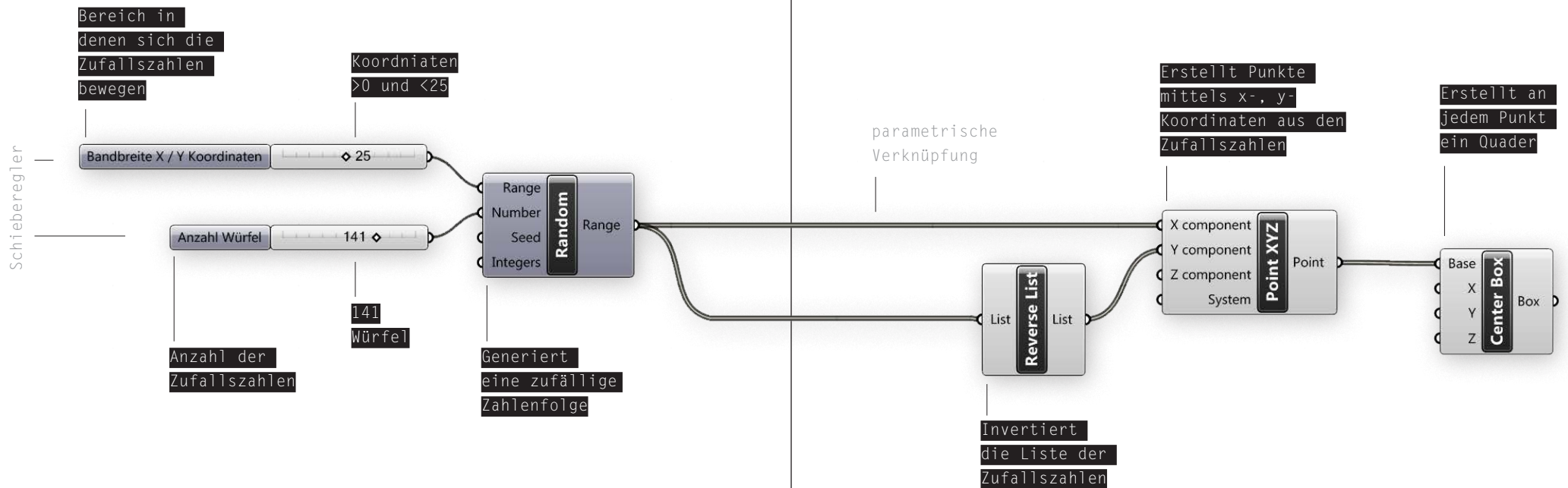
grafischen Mitteln generative Prozesse und Modelle zu erzeugen und ist daher gerade für Architekten interessant.

Nicht zuletzt wegen der intuitiven grafischen Benutzeroberfläche und den vielfältigen Möglichkeiten zum Experimentieren erfreut sich Grasshopper bereits großer Beliebtheit in der Architektur Avantgarde. (31)

(30) Zubin Khabazi: „Generative Algorithms using Grasshopper“ E-Paper von <<http://www.grasshopper3d.com/page/tutorials-1>>
 (31) Reinhard König (2012): „Entwicklung des parametrischen und algorithmischen Entwerfens“. <<http://entwurforschung.de/entwicklung-des-parametrischen-und-algorithmischen-entwerfens/>> [Stand: 09.04.2012. Zugriff 22.04.2012, 11:00 MESZ]



Durch Verschieben des Schiebereglers kann die Anzahl der Würfel und deren Ausdehnung im Koordinatensystem kontrolliert werden.



autodesk maya - mel | python

„Autodesk Maya for Architects and Designers. The award-winning Autodesk® Maya® software is a powerful, integrated 3D modeling, animation, visual effects, and rendering solution with deep roots in the film and games industries. Leading architects, automotive and consumer product designers, and creative advertising professionals also benefit from the unique Maya toolset for exploring, validating, and conveying their designs.

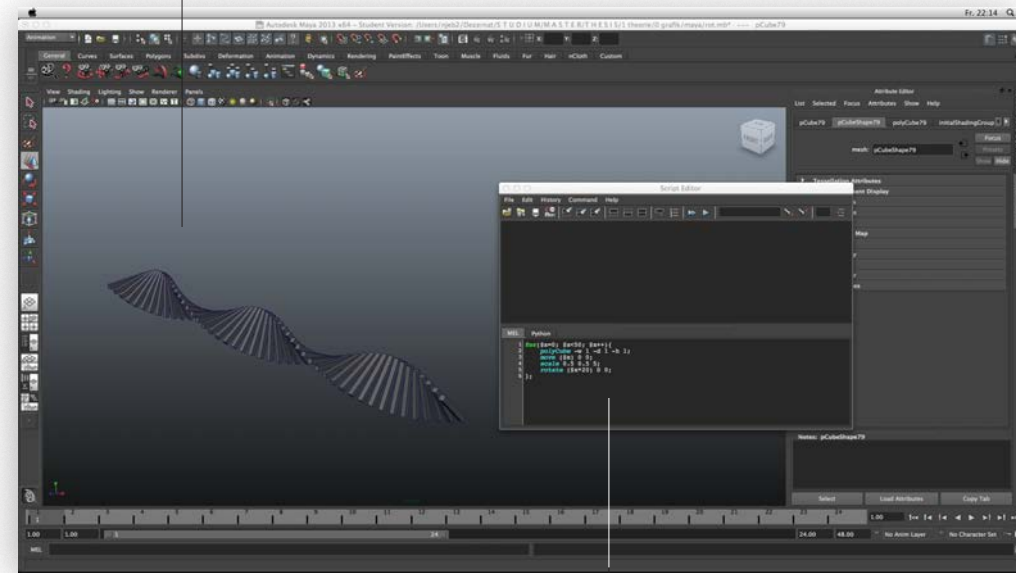
Maya for Architecture

The power of Maya for architects lies in its flexibility for generating architectural forms and shapes. Maya offers a robust platform with extensive modeling toolsets that complement the building information modeling (BIM) workflow during conceptualization. Architects can also develop their own rules in Maya with the powerful, embedded scripting language, Maya Embedded Language (MEL).“



Autodesk, Inc. (2012): „Design Visualization“. <<http://usa.autodesk.com/adsk/servlet/item?siteID=123112&id=13591323>> [Stand: 2012. Zugriff 02.05.2012, 08:23 MESZ]

3D-Modell
Output



Quelltext-
Editor

Die Software Maya ist eine sehr verbreitete 3D-Modellierungs-, Visualisierungs- und Animationssoftware. Sie kommt ursprünglich aus dem Bereich der Film- und Fernsehindustrie, spielt aber auch eine große Rolle bei der Entwicklung von Video- und Computerspielen. Zunehmend findet sie auch in anderen Bereichen, wie der Industriellen Fertigung, des Produktdesigns, in der Entwicklung und Forschung oder aber der Architektur Verwendung.

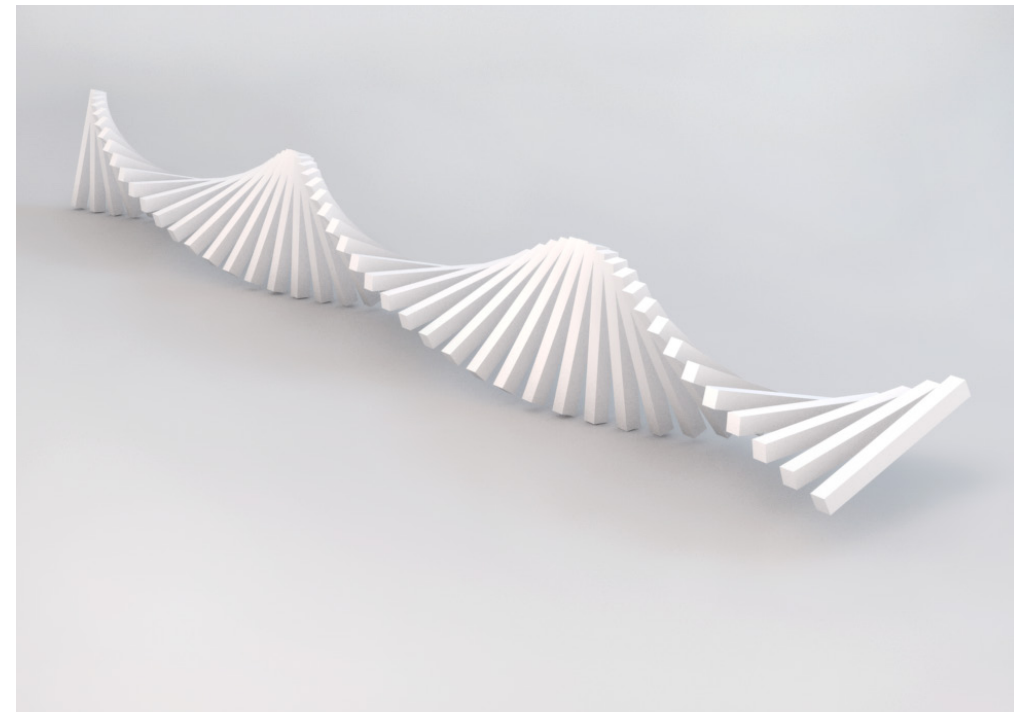
Maya selbst ist ein re-aktives Programm, verfügt aber auch über erweiterte Funktionen, wie Animation, Partikelströme oder physikalische Eigenschaften, wie Gravitation, sodass sich beispielsweise Gas- und Flüssigkeitsbewegungen simulieren lassen (MayaFur, MayaFluids).

Neben diesen Möglichkeiten existiert auch eine programminterne Steuerungssprache, MEL (Maya Embedded Language). MEL orientiert sich dabei konzeptionell sowie im Aufbau und Syntax an C und bietet umfangreiche Möglichkeiten zur Programmierung und Automatisierung.

Außerdem steht die Programmiersprache Python zur Verfügung. Dadurch wird die Erweiterbarkeit und Flexibilität nochmals enorm ausgeweitet. Das Schreiben von Programmen ist vor allem für Programmierneulinge attraktiv, da die Python-Syntax sehr verständlich ist. Man kann mit Python auf sämtliche MEL Befehle zugreifen, umgekehrt funktioniert dies ebenfalls. Zusätzlich kann der Funktionsumfang durch frei verfügbare Python Bibliotheken problemlos erweitert werden. Beide Programmiersprachen werden innerhalb von Maya in einem Quelltext-Editor eingegeben. (32)

Als freie Programmiersprache eignet es sich zwar gut zum algorithmischen und parametrischen Entwerfen, erfordert jedoch aufgrund der Komplexität und des speziellen Syntax, die Programmiersprachen besitzen, zur professionellen Anwendung weitreichende Programmiererfahrungen. Daher ist es sicherlich nicht uneingeschränkt für alle Architekten geeignet, sondern bleibt eher Spezialisten vorbehalten.

(32) Vgl. Kein Autor (o.J): „Maya“. <[http://de.wikipedia.org/wiki/Maya_\(Software\)](http://de.wikipedia.org/wiki/Maya_(Software))> [Stand: 17.02.2012. Zugriff 02.05.2012, 09:20 MESZ]



Die Codezeilen innerhalb der geschweiften Klammern werden genau 51 mal ausgeführt. Zu Beginn wird die Variable \$x auf den Wert 0 gesetzt, bei jedem Durchlauf um 1 erhöht (x++) und zwar solange der Wert noch kleiner 50 ist

```
1 for($x=0; $x<50; $x++){
```

Es werden also nach einander 51 Würfel erzeugt,

```
2 polyCube -w 1 -d 1 -h 1;
```

dann um eine Einheit entlang der x-Achse verschoben,

```
3 move ($x) 0 0;
```

entlang der z-Achse um das 5-fach, entlang der x- und y-Achse um die Hälfte skaliert,

```
4 scale 0.5 0.5 5;
```

und schließlich stufenweise um 10° rotiert,

```
5 rotate ($x*10) 0 0;
```

```
6 };
```

processing

„Processing is an open source programming language and environment for people who want to create images, animations, and interactions. Initially developed to serve as a software sketchbook and to teach fundamentals of computer programming within a visual context, Processing also has evolved into a tool for generating finished professional work. Today, there are tens of thousands of students, artists, designers, researchers, and hobbyists who use Processing for learning, prototyping, and production.“



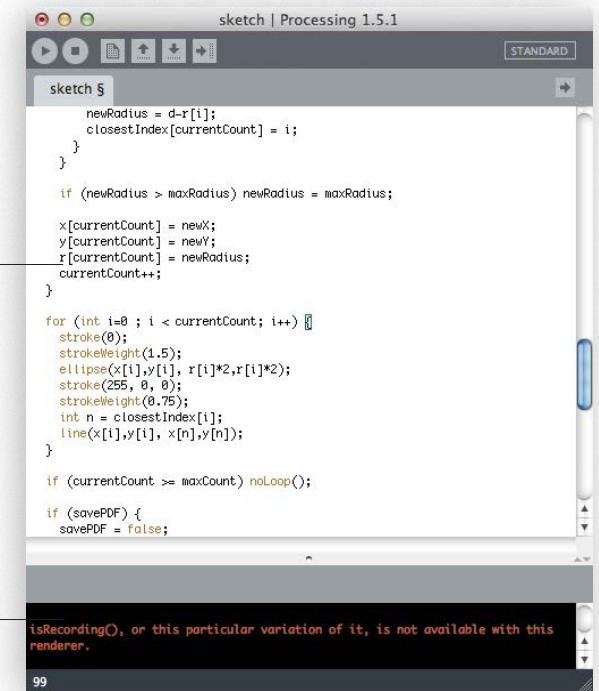
Kein Autor (o.J.): „Processing“. <<http://www.processing.org/>> [Stand: 2012. Zugriff 02.05.2012, 12:44 MESZ]

Processing ist eine speziell für den Gestalter entwickelte und stark auf die Einsatzbereiche Grafik, Simulation und Animation ausgerichtete Programmiersprache. Sie wurde ursprünglich zu Lernzwecken verwendet, erfreut sich aber mittlerweile einer rasch wachsenden Gemeinde von Künstlern, Designern und Programmierern.

Processing hat den Charakter einer stark vereinfachten Version der Programmiersprache Java und richtet sich somit auch an Programmieranfänger. Es lassen sich einfach visuelle Elemente, sowie auch dreidimensionale geometrische Objekte und Interaktionen programmieren. Die pro-

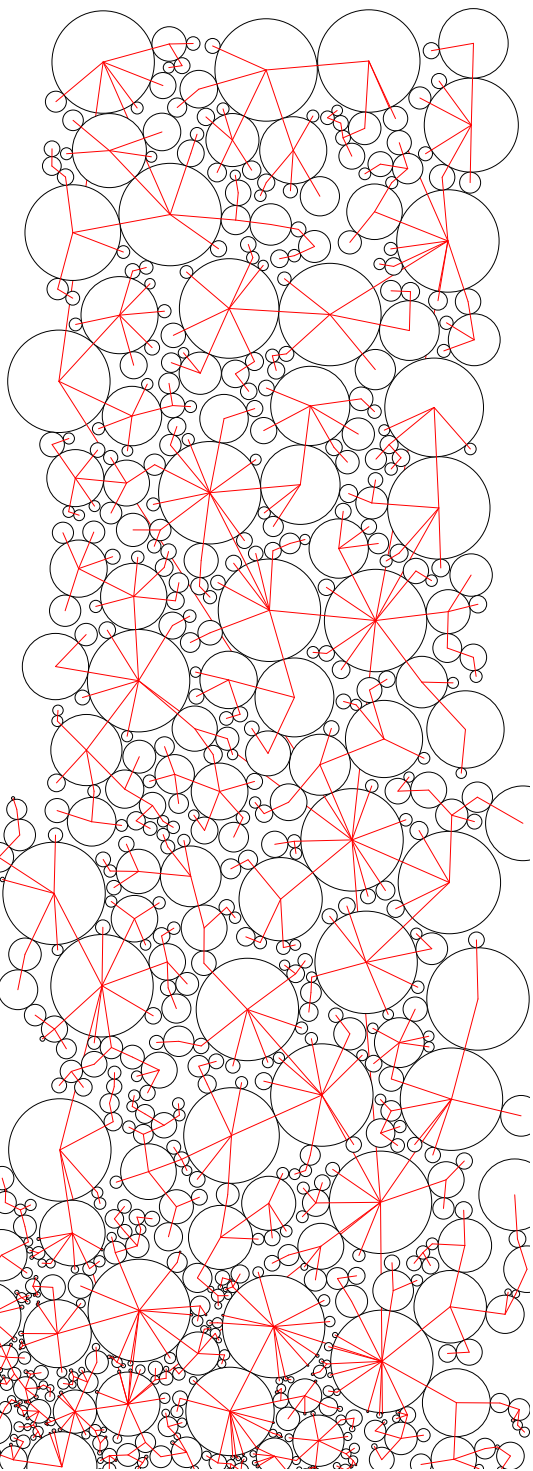
Quelltext-
Editor

Konsole
Ausgabe von
Statusinfor-
mationen



grammierten Ergebnisse lassen sich durch mehrere Schnittstellen wie jpeg, pdf oder aber als 3D-Modell im dxf-Format weiternutzen.

Durch den simplen Aufbau mit einem einzigen Editor-Fenster, in welchem der Quelltext eingegeben wird, und einem Ausgabefenster, in welchem die programmierten Algorithmen ausgeführt werden, ist es schnell zu verstehen und leicht zu erlernen. Im Einsatzfeld der Architektur durchaus noch wenig erprobt, eignet es sich dennoch auch hier für generative Gestaltungsprozesse und zur Simulation architektonischer Problemstellungen.



Zunächst wird die Position und der Radius für einen Kreis erzeugt

Alle vorhandenen Kreise werden dann mit dem neu hinzugekommenen verglichen. Wenn es eine Überschneidung gibt, wenn also die Entfernung kleiner ist als die Summe der beiden Radien, wird die Variable `intersection` auf `true` gesetzt und der aktuelle Durchgang abgebrochen.

Falls jedoch keine Überschneidung vorhanden ist, wird der am nächsten gelegenen Kreis bestimmt. Als Ergebnis wird dessen Index gespeichert.

Der Radius des neuen Kreises wird dann so groß wie möglich gemacht (aber höchstens so groß wie durch `maxRadius` angegeben).

Der so erzeugte Kreis wird nun ins Display gezeichnet. Dabei ist die Linienfarbe 0 (RGB=Schwarz) und die Dicke 1.5

Zusätzlich wird jeder Kreis mit seinem nächsten Nachbarn verbunden. Dabei ist die Linienfarbe rot (RGB=255,0,0) und die Dicke 0.75

```
void draw() {
    ...
    float newX = random(0+maxRadius,width-maxRadius);
    float newY = random(0+maxRadius,height-maxRadius);
    float newR = minRadius;

    boolean intersection = false;
    for(int i=0; i < currentCount; i++) {
        float d = dist(newX,newY, x[i],y[i]);
        if (d < (newR + r[i])) {
            intersection = true;
            break;
        }
    }

    if (intersection == false) {
        float newRadius = width;
        for(int i=0; i < currentCount; i++) {
            float d = dist(newX,newY, x[i],y[i]);
            if (newRadius > d-r[i]) {
                newRadius = d-r[i];
                closestIndex[currentCount] = i;
            }
        }

        if (newRadius > maxRadius) newRadius = maxRadius;
        x[currentCount] = newX;
        y[currentCount] = newY;
        r[currentCount] = newRadius;
        currentCount++;
    }

    for (int i=0 ; i < currentCount; i++) {
        stroke(0);
        strokeWeight(1.5);
        ellipse(x[i],y[i], r[i]*2,r[i]*2);
        stroke(255, 0, 0);
        strokeWeight(0.75);
        int n = closestIndex[i];
        line(x[i],y[i], x[n],y[n]);
    }
}
```



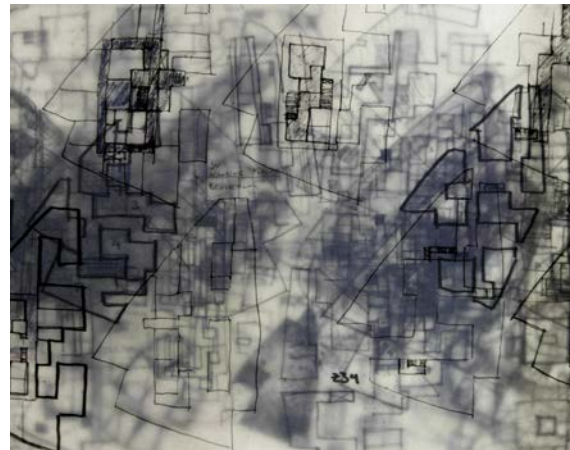
4 . a n w e n d u n g
ein veränderter entwurfsprozess?

vorgehen und unterschiede

Im architektonischen Entwurf gilt es verschiedenste, häufig divergente Interessen und Anforderungen gegeneinander abzuwägen und zu erfüllen. Dazu wird zu Beginn eine konzeptionelle Idee entwickelt und nach Möglichkeit durch alle Maßstäbe verfolgt. Entwerfen ist eine sinnliche, sich in Rückkopplung mit dem Bekannten vollziehende Tätigkeit. Das Arbeiten in Varianten und Denken in Typologien sind dabei traditionelle Mittel.

Analog zum klassischen steht am Anfang des »digitalen« Entwurfsprozesses die konzeptionelle Idee zur Umsetzung und Erfüllung aller Randbedingungen und Anforderungen.

Wesentlicher Unterschied liegt in der weiteren Vorgehensweise: Traditionell wird der Architekt, gelenkt durch Intuition, Erfahrungswerte und persönliche Präferenzen verschiedene Lösungsansätze ausprobieren. Er tritt als Entwerfer einer Gestalt auf, der sich im Idealfall an die übergeordneten Konzepte hält.



Im generativen Entwurfsprozess rückt dieser handwerkliche, intuitive Aspekt zunächst in den Hintergrund, dafür wird Abstraktion und Information zum hauptsächlichen Element. Nicht mehr die Frage, wie Gestalt entworfen wird, sondern wie die Konzeptidee zur Gestaltung abstrahiert wird ist von Bedeutung. Es gilt in dieser Phase das Konzept in ein algorithmisches Regelwerk zu übersetzen.

Dabei ist es erforderlich, dass die Idee zur Gestaltung explizit ausgedrückt wird und demnach durch alle Maßstäbe immer gegenwärtig ist.

Entgegen dem allgemeinen Verständnis, dass der Architekt bei dieser Entwurfsmethode vom Computer ersetzt wird, kann an dieser Stelle behauptet werden, dass seine Rolle gestärkt wird:

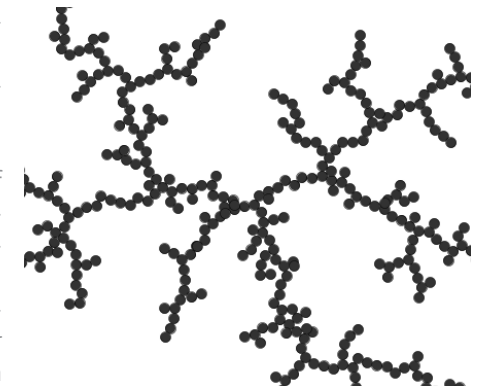
In der nächsten Entwurfsphase generiert der Computer anhand des vom Architekten erstellten Algorithmus Entwurfsvarianten.

In der Architektur gibt es nie nur eine richtige Lösung sondern meist viele ähnlich gute. Es liegt in dieser Phase beim Architekten Potentiale zu erkennen und aus einer Vielzahl an Varianten auszuwählen. Die Ergebnisse sind jedoch nicht als Endprodukt zu sehen, sondern liefern Raum für unterschiedliche Interpretationen und dienen als Grundlage für Verbesserungen.

An dieser Stelle kann entweder auf konventionelle Weise weitergearbeitet und der Entwurf verfeinert werden.

Oder aber es wird die zugrundeliegende Abstraktion modifiziert oder einzelne Parameter im Programm verändert. Durch die Wechselbeziehung zwischen dem vom Computer generierten Ergebnis und der schöpferischen Intuition entwickelt sich das generative System bei jedem Iterationsschritt weiter.

Die konzeptionelle und steuernde Kompetenz bleibt weiterhin beim Architekten. Die Software übernimmt nur die Rolle des stillen Helfers. Klar ist dabei, dass sie nicht zu ermüden ist und den gewünschten Output viel schneller produzieren kann. Interessanter ist jedoch, die Möglichkeiten generativer Prozesse aus dem Blickwinkel der Emergenz zu betrachten: Unter Emergenz wird in diesem Zusammenhang ein unerwartet auftretendes Ergebnis verstanden, wenn also das Zusammenwirken der Einzelteile zu mehr führt als man zunächst annehmen würde. (Ein Beispiel sind Wachstumsstrukturen aus Agenten. Ein sehr einfacher Algorithmus erzeugt unerwartete organische koralienartige Strukturen. Siehe Kapitel 2 Agentensysteme).



Dabei können neue Lösungswege und Ideen entstehen, die man im Vorhinein nicht in Betracht zog. Insbesondere da der Algorithmus unvoreingenommen und frei von typologischem



Denken ist, wird der Lösungsspielraum grundlegend erweitert.

Die so gefundenen Lösungen zeichnen sich durch eine hohe Integrität aus, da sie direkt aus den projektspezifischen Parametern und eng nach einer Konzeptidee entwickelt wurden.

Der menschliche Faktor kann hier jedoch nicht außer Betracht bleiben. Das logische deterministische Vorgehen eines Algorithmus kann zwar eine gewisse Problemstellung unter Beachtung komplexer Wechselbeziehungen optimieren, ästhetische oder konzeptionelle Faktoren müssen jedoch stets durch den Architekten überwacht werden.

top-down und bottom-up

Im klassischen Architekturentwurf wird »Top-Down«, also von großen sukzessive in kleinere Maßstäbe gearbeitet. Das bedeutet, dass ein Projekt beginnend mit dem Städtebau, über die Grundrissplanung bis hin zu Konstruktionsfragen im Detail entwickelt wird. Die äußere Form wird dabei nach und nach ausgebildet.

Bei dieser Vorgehensweise ist in den ersten Entwurfsschritten meist noch wenig über spätere Problemstellungen, wie innere Funktionszusammenhänge, bauphysikalische Eigenschaften oder konstruktive Systeme bekannt. Demnach ist der klassischen Entwurfsprozess durch ein zyklisches Überarbeiten aller Entwurfskennzeichen geprägt. Dieser Prozess ist daher keineswegs kontinuierlich und linear, sondern unter Umständen sogar regressiv. Das heißt, dass Entscheidungen im frühen Entwurfsstadium später zu schwerwiegenden, wenn auch unvorhersehbaren Konflikten führen können. So ist in der Architekturlehre der Satz » Zum Entwerfen gehört das Verwerfen« integraler Bestandteil.

In der Computational Architecture wird dieses Vorgehen umgekehrt: Die Projekte werden »Bottom-Up«, also vom kleinen zum großen Maßstab entworfen. Den Entwurfsvarianten liegt daher keine übergeordnete Form und komplizierte Strategie zu-

grunde. Stattdessen wird die spätere Gestalt durch die vom Architekten definierten Regeln und Abhängigkeiten, wie beispielsweise die Raumzusammenhänge oder geographische Eigenschaften beschrieben und folgt damit dem von Louis Sullivan propagiertem Prinzip „form follows function“ (siehe dazu im Kapitel Beispiele die Projekte Wohn- und Gewerbesiedlung Kalkbreite oder Aperiodic Symmetries).

„Whether it be the sweeping eagle in his flight, or the open apple-blossom, the toiling work-horse, the blithe swan, the branching oak, the winding stream at its base, the drifting clouds, over all the coursing sun, form ever follows function, and this is the law. Where function does not change form does not change.“

Louis Sullivan

kaisersrot

Kaisersrot ist eine Team aus Architekten, Stadtplanern und Informatikern, die sich zum Ziel gesetzt haben mit Hilfe von generativen Prozessen rationale Entwurfsmethoden zur Lösung komplexer Planungsaufgaben zu entwickeln.

Der Zusammenschluss entstand ursprünglich aus einer Forschergruppe an der Universität Kaiserslautern und dem Büro KCAP in Rotterdam (daher Kaisers-Rot) und forschte an der ETH Zürich an der Professur CAAD von Ludger Hovestadt. 2008 ist aus der Forschergruppe eine finanziell selbsttragende Firma hervorgegangen, die Planungsdienstleistungen zur Lösung architektonischer Fragestellungen anbietet. Dabei arbeiten sie sowohl selbst an praktischen Projekten, als auch in Kooperation mit Architekturbüros, die ihre Dienstleistung benötigen:



Thomas Wallmeyer: Olympiastadion

olympiastadion peking
herzog & de meuron

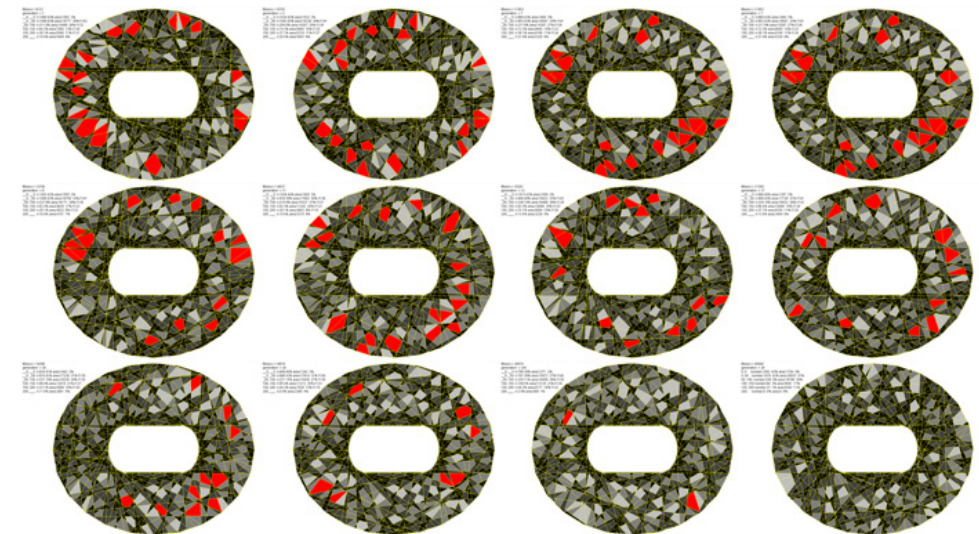
In Kooperation mit Herzog & de Meuron führte das Team Kaisersrot eine Studie zur Optimierung des Dachtragwerkes des Olympiastadions in Peking durch. Ziel dabei war es, jeden der Kastenträger so an die unregelmäßige Form des Stadions anzuschmiegen, dass die Trägerzwischenräume, wegen der Eindeckung mit ETFE-Kissen nicht zu groß oder zu klein werden. Außerdem sollten die Riegel wegen der Durchbiegung nicht zu lang werden und die Durchmesser der Kastenträger nicht zu klein, damit noch Schweißarbeiten im Inneren durchgeführt werden können. Zu Beginn wurden zufällig 50 Varianten mit je 50 Kastenträgern erstellt. Diese bildeten den veränderlichen Gencode, welcher durch Mutation und Rekombination neue Lösungen

hervorbringen sollte. Jede generierte Variante wird an Hand der Kriterien, wie Feldgröße, Trägerdurchmesser, etc. bewertet und selektiert. Die Varianten mit hoher Fitness werden dann gekreuzt um eine neue, insbesondere fittere Generation hervorzubringen. Nach 250 Generationen pendelte sich das System bei einer bestimmten Lösung ein und veränderte sich kaum noch. Auch sind die zu großen und zu kleinen Felder verschwunden.

Die Software macht dies zwar ähnlich, aber ist nicht zu ermüden und viel schneller. Zudem ist sie in der Lage durch den Einsatz von genetischen Algorithmen den Anteil der großen und kleinen Feldern zu optimieren. Die Varianten steigern so ihre Fitness und werden optimiert. Ein Beispiel dafür, wie sich generative Prozesse zur Bewältigung von chaotischen Strukturen einsetzen lassen. (33)

Dies ist ein Paradebeispiel für architektonische Optimierung. Diese Aufgabe lässt sich von Hand kaum bewältigen, da bei Behebung eines Problems, wie beispielsweise eines zu großen Feldes an anderer Stelle sofort neue Probleme entstehen.

(33) Vgl. Markus Braach (2010): „National Beijing Olympic Stadium“ <http://www.kaisersrot.ch/kaisersrot-02/2003_National_Beijing_Olympic_Stadium_%28CN%29.html> [Stand: 21.03.2010. Zugriff 04.06.2012, 12:55 MESZ]



Kaisersrot: Strukturvarianten Olympiastadion

wohn- & gewerbesiedlung kalkbreite zürich slik architekten

In diesem Projekt beschäftigte sich die Kaisersrot Gruppe in Zusammenarbeit mit SLIK Architekten mit der Ausformulierung eines sehr komplexes Raumprogrammes.

Ziel war es dabei die vielschichtigen Beziehungen und Ansprüche der über 35 Nutzungen in ein räumliches Gefüge zu übersetzen.

Dazu wurde ein genetischer Algorithmus verwendet, welcher die Nutzungen auf dem Baugelände anordnet und auf Erfüllung von definierter Kriterien überprüft.

Diese Kriterien sind beispielsweise Belichtung, Erschließung oder funktionale Zusammenhänge und entscheiden über die Qualität des Entwurfs.

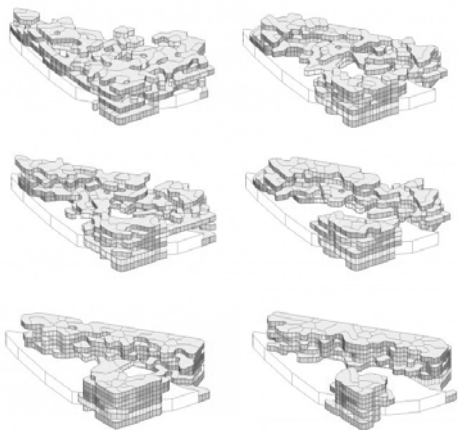


Visualisierung
Kaisersrot: Kalkbreite

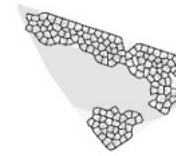
Zitat der Architekten:

„Der Architektur-Automat liefert ein räumliches Gebilde, welches aus dem Raumprogramm unter Berücksichtigung der Form des Grundstückes und der Baugesetze entstanden ist. Diesem Gebilde muss man nun eine Struktur und eine Hülle hinzufügen um es als Architektur bezeichnen zu können. Doch zuerst gilt es, unter einer sehr grossen Menge möglicher Lösungen eine Auswahl zu treffen, was durch das Gewichten der oftmals gegensätzlichen Anforderungen an die Lösung geschieht.“

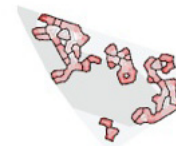
Hat man sich für eine räumliche Konstellation entschieden, müssen konstruktive Ansätze gefunden werden welche die der Lösung innewoh-



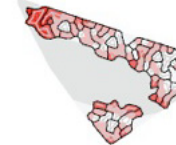
Entwicklungsschritte
Kaisersrot: Kalkbreite



Grundstruktur



Fassadenlänge



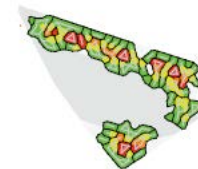
Auskragung



Typen



Lärmbelastung



Belichtung

Bewertungskriterien
Kaisersrot: Kalkbreite

nenden Regeln zu einer tragenden und möglichst einfach realisierbaren Struktur umwandeln. Dabei muss vor allem auch die Haustechnik auf die grosse formale Varianz der einzelnen Gebäudeteile abgestimmt werden. Zuletzt bleibt die Frage der Hülle, bei welcher man sich an Sempers „textile Architektur“ erinnert, gilt es doch für einen gegebenen Körper eine passende Fassade anzufertigen. Hierfür wurde auf parametrische Elementfassaden zurückgegriffen, die automatisch um das Volumen gewickelt werden können und so ein umfangreiches Variantenstudium ermöglichen.“

SLIK-Architekten (o.J.): „Wohn- und Gewerbesiedlung Kalkbreite“
<<http://www.slik.ch/?pid=12>> [Stand: 2012. Zugriff 06.06.2012, 14:53 MESZ]

Das entwickelte Projekt stellt eine Lösung dar, in der sowohl die inneren als auch die äusseren Sachzwänge für sämtliche Raumeinheiten in einem hohen Masse erfüllt sind. Das Gebäudevolumen ist demnach direkt aus dem Raumprogramm, der städtebaulichen und bauphysikalischen Randbedingungen entwickelt. Die Architekten steuerten lediglich durch einige direkte Festsetzungen den Entwurfsprozess.

Die dadurch entstandenen Strukturen sind direkt aus der Funktion abgeleitet und besitzen ihre eigenen räumlichen Qualitäten, die eine Gegenposition zum konventionellen orthogonalen Raster darstellt.

aperiodic symmetries theverymany

Aperiodic Symmetries ist Teil einer Serie von Experimenten der Gruppe THEVERYMANY um Marc Fornes. Hintergrund ist die Auseinandersetzung mit komplexen räumlichen Strukturen mit dem Ziel ein Maximum an Vielfalt mit einem Minimum an geometrischen Elementen zu generieren. Dabei wird die Form über Optimierungs-Algorithmen gefunden.

Der Arbeit liegt RhinoScript (vgl. Kapitel Software) zugrunde. (34)

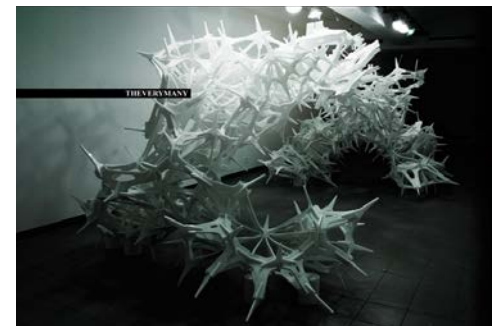


Marc Fornes: 09 Ap. Symmetries.

Bei dieser Installation an der Universität von Calgary fügen sich 757 sternförmige Verbindungen und 883 Formelemente zu einer räumlichen Struktur zusammen mit einer Gesamtgröße von 6.3 x 2.5 x 3 Metern. Die einzelnen Teile bestehen aus Polyethylenplatten und wurden mit einer CNC-Fräse innerhalb von fünf Tagen produziert. CNC-Fräsen erlauben das automatisierte Fräsen von Freiformen. Die Maschinen werden programmiert und erlauben so, eine große Menge an individuell geformten Einzelteilen zu produzieren. (35) Dabei können die geometrischen Daten direkt aus der Software ausgelesen und verwendet werden.



Marc Fornes: 09 Ap. Symmetries.



Marc Fornes: 09 Ap. Symmetries.

(34) Marc Fornes (o.J.): „09 Ap. Symmetries“. <<http://theverymany.com/constructs/09-aperiodic-symmetries/>> Stand: 2012. Zugriff 06.06.2012, 16:58 MESZ]

(35) Hartmut Bohnacker, Benedikt Groß, Julia Laub, Claudius Lazzeroni (Hrsg.) (2009): Generative Gestaltung: Entwerfen. Programmieren. Visualisieren. Mainz: Hermann Schmidt Verlag

hyphae lamp nervous system

Nervous System ist ein generatives Designbüro, welches im Spannungsfeld von Wissenschaft, Kunst und Technologie arbeitet. Die Gründer Jessica Rosenkrantz und Jesse Louis-Rosenberg sind selbst Architekten. Zentral in ihrer Arbeit ist die Auseinandersetzung mit Programmen, welche auf Prozesse und Muster der Natur basieren und einzigartige Kunst, Schmuck und Hauswaren schaffen.



Nervous System: Hyphae Lamp

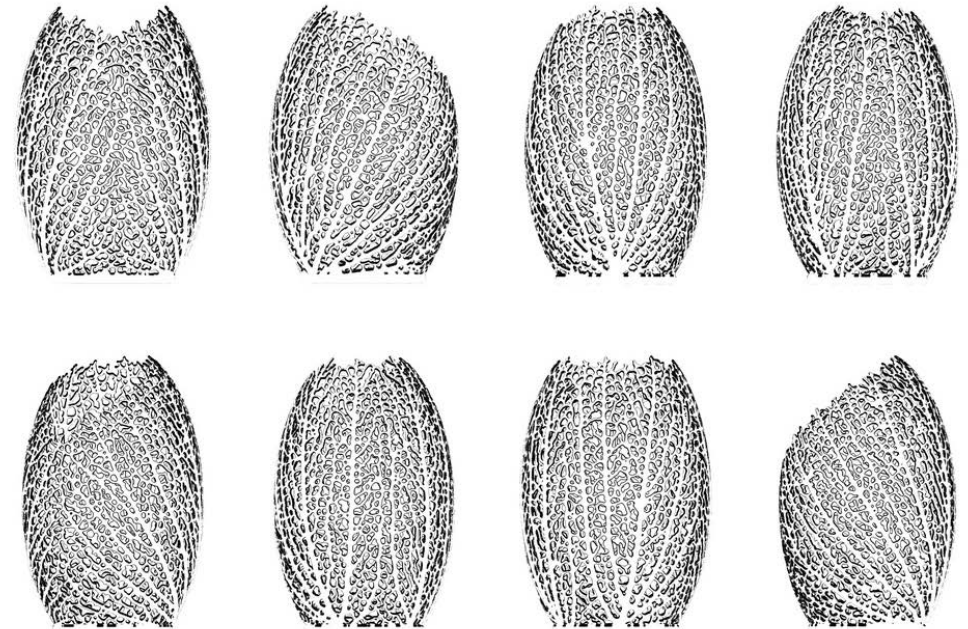
Die Hyphae lamp ist eine Serie von Tischleuchten. Inspiriert von der Äderung von Blättern, welche Flüssigkeiten durch einen pflanzlichen Organismus transportieren, wurde ein Algorithmus der diese physikalischen Wachstumsprinzipien imitiert erstellt.



Nervous System: Hyphae Lamp

Ausgehend von einem ersten Keim wird ein hierarchisches Netzwerk aufgebaut, in dem Agenten sich fortlaufend verzweigen und verschmelzen. Diese dicht verbundene Struktur ist zugleich filigran und durchlässig als auch äußerst stabil. Jede Leuchte ist durch diesen generativen Prozess erstellt worden. Da bei jedem Durchgang andere Verzweigungen und Verästelungen generiert werden, gleicht keine der anderen. Die durch den Algorithmus generierten 3D-Informationen werden direkt ausgelesen und an einen 3D-Drucker übergeben. (36)

(36) Nervous System (o.J.): „Hyphae Lamp“. <<http://n-e-r-v-o-u-s.com/shop/generativeProduct.php?code=99>> Stand: 2012. Zugriff 10.06.2012, 19:58 MESZ]



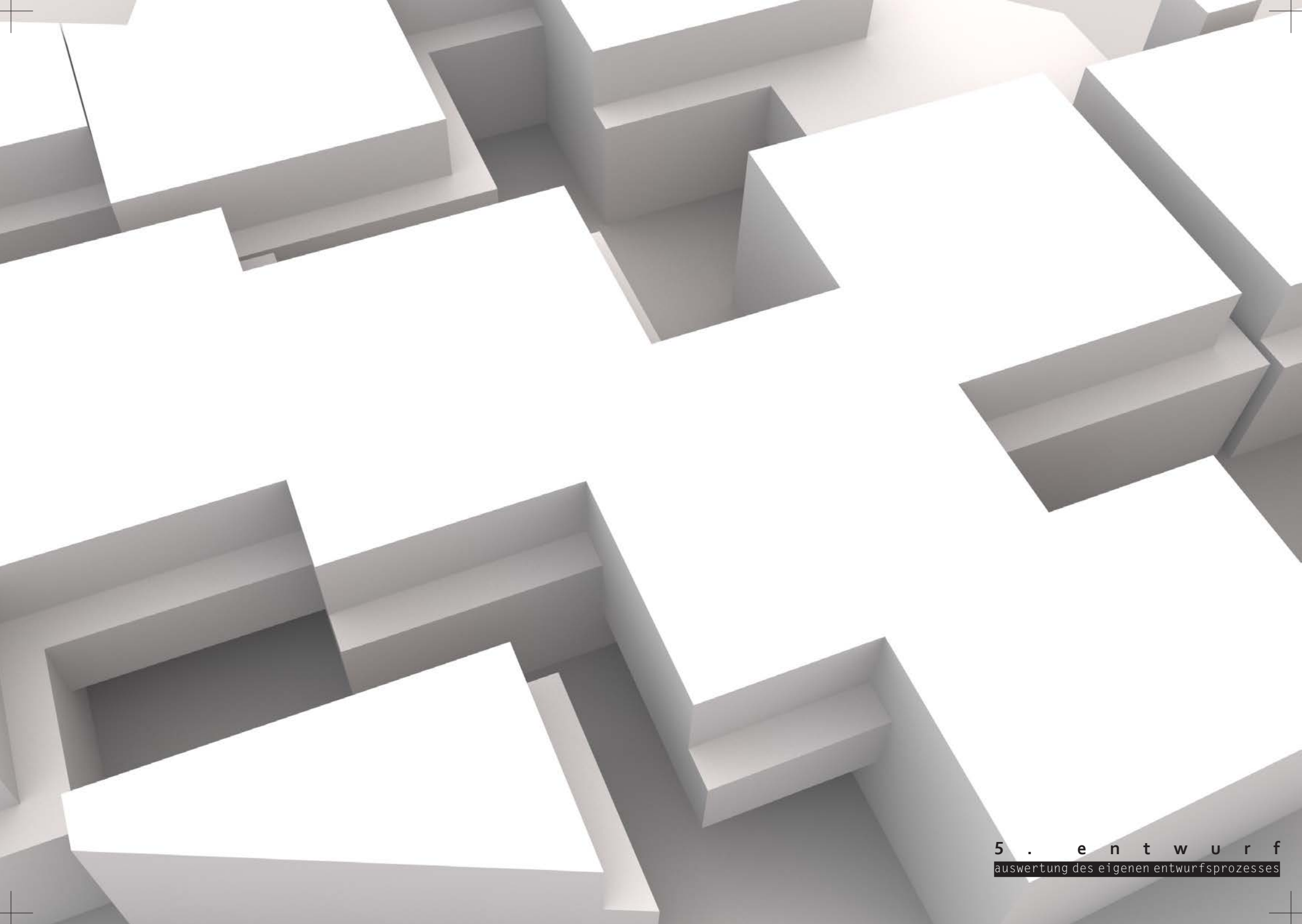
Nervous System: Hyphae Lamp



Nervous System: Hyphae Lamp



Nervous System: Hyphae Lamp



5 . e n t w u r f
auswertung des eigenen entwurfsprozesses

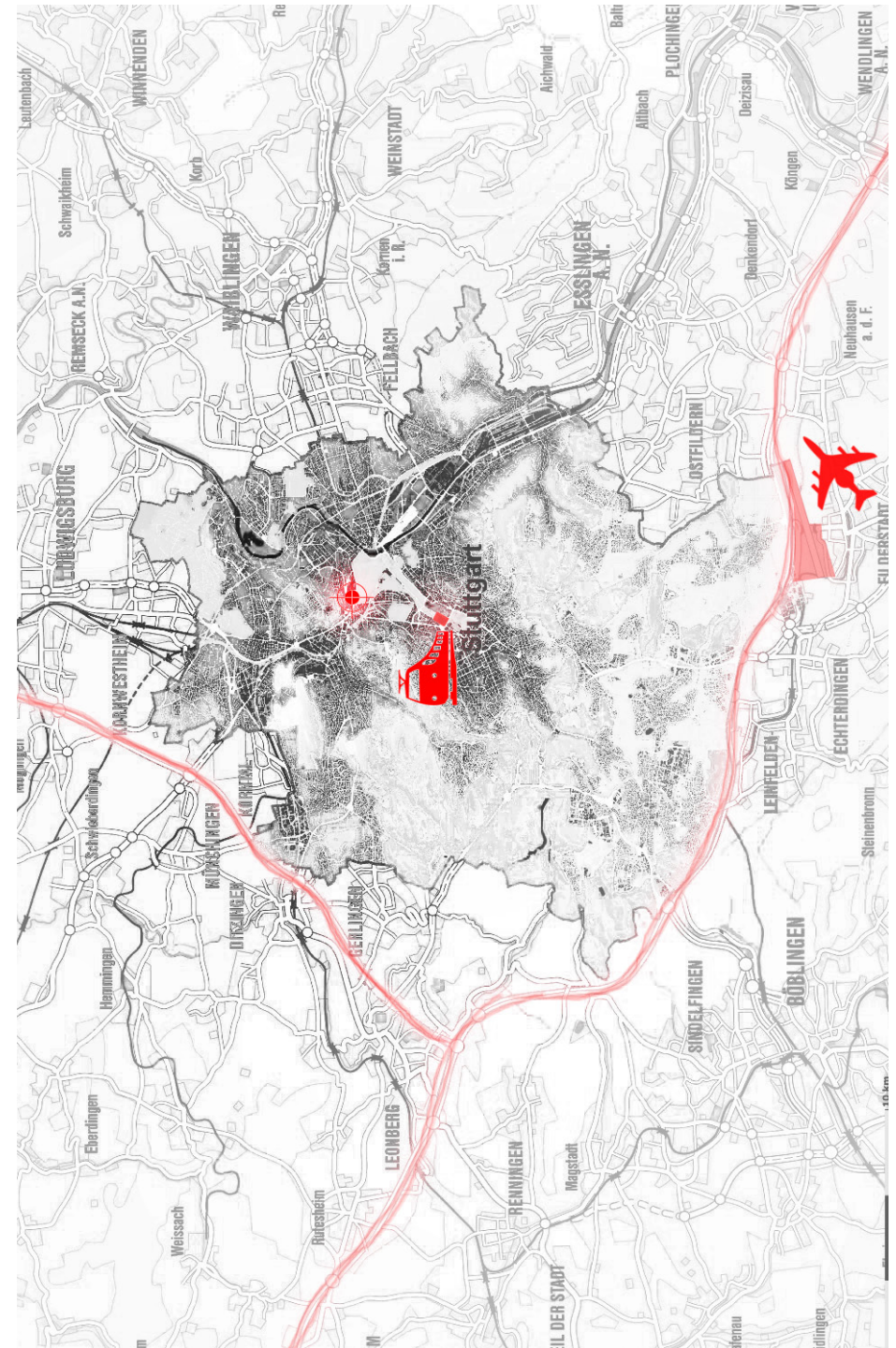
makroebene - stadt Stuttgart ist die Landeshauptstadt Baden-Württembergs und mit über 600.000 Einwohnern dessen größte Stadt. Sie ist die sechstgrößte Stadt Deutschlands und bildet das Zentrum der Europäischen Metropolregion Stuttgart mit etwa 5,3 Mio. Einwohner.

Das Stuttgarter Stadtbild wird durch viele Anhöhen (z. T. Weinberge), Täler (insbesondere das Neckartal) und Grünanlagen (unter anderem Rosensteinpark, Schlossgarten) geprägt. Stuttgart zählt zu den einkommensstärksten und wirtschaftlich bedeutendsten Städten Deutschlands und Europas. Die Region Stuttgart ist mit ca. 1500 ansässigen kleinen und mittelgroßen Unternehmen eines der Zentren des deutschen Mittelstandes. Dabei handelt es sich in erster Linie um Zulieferer für die großen, global agierenden Automobil- und Maschinenbau-Firmen. In der Stadt und ihrer Umgebung haben sich unter anderem viele Hightech-Unternehmen angesiedelt, darunter Daimler, Porsche, Siemens oder Bosch, die zum Teil hier ihr weltweites Hauptquartier haben.

Das Grundstück »Rosenstein-Tor« liegt im Norden von Stuttgart, in 4 km Entfernung vom Stadtkern, zwischen den Stadtteilen Stuttgart-Nord, Bad Canstatt und Burgholzhof.

Entfernungen

- . HBF ca. 3 km
- . Innenstadt ca. 4 km
- . A81 ca. 8 km
- . Flughafen Stuttgart ca. 17 km

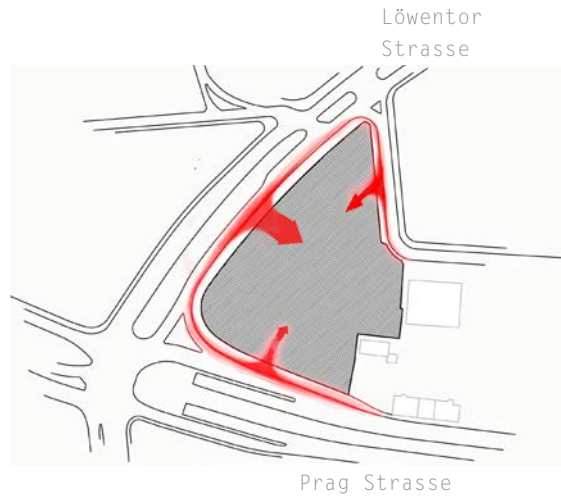


mikroebene städtebauliche situation

Das Projektgrundstück liegt an der Ecke von Pragstraße und Löwentorstraße.

Hierbei handelt es sich um ein unbebautes Grundstück, das von drei Seiten erschlossen werden kann.

Der Standort befindet sich direkt unterhalb Stuttgarts größtem Verkehrsknotenpunkt Pragsattel. Der Pragsattel ist ein bedeutendes städtebauliches Entwicklungsgebiet. Zahlreiche neue Projekte befinden sich derzeit in Planung oder im Bau.



Grundstücksgröße .	10.320 qm	
GRZ = 0.6 .	6.192 qm	Oberbaubare Fläche
GFZ := 2.4 .	24.800 qm	
	26.000 qm	ca BGF Fläche



Panorama Pragstraße

Das Projektgrundstück liegt zum einen am Fuße des nördlich gelegenen Pragsattel-Weinbergs. Zum anderen gegenüber des südlich gelegenen Löwentors, welches den Auftakt zu dem etwa 65 Hektar umfassenden Rosensteinparkes bildet, und befindet sich damit in unmittelbarer Nähe des sogenannten Stuttgarter »grünen Us«, ein durch den Schlossgarten, den Rosensteinpark, sowie den Killesbergpark definierter grüner Gürtel. Durch den alten Baumbestand und die großflächigen Wiesen gilt der Rosensteinpark als größter englischer Landschaftspark im Südwesten Deutschlands.



Rosensteinpark
Blick vom Löwentor

Die umgebende Bebauung ist von Verwaltungsbauten, sowie Industrie- und Gewerbeanlagen geprägt. Nordwestlich schließt sich ein Wohngebiet an. Die städtebauliche Situation stellt sich äußerst heterogen dar und wird stark durch die hochfrequentierte Pragstraße geprägt.

verkehrsanbindung Quer durch Stuttgart verlaufen die B 10 (Pforzheim–Stuttgart–Ulm), die B 14 (Schwäbisch Hall–Stuttgart–Herrenberg), die B 27 (Heilbronn–Stuttgart–Tübingen) und die B 295 (Calw–Leonberg–Stuttgart).

Bis auf die B 14 treffen alle auf dem Pragsattel zusammen, dem größten Verkehrsknoten der Stuttgarter Innenstadt.

Das Projektgrundstück liegt in direkter Nebenlage und wird durch den 2006 eröffneten Pragtunnel, der das erhöhte Verkehrsaufkommen in den Rush-hour-Zeiten bewältigt, optimal erschlossen.

Diese Lage an den Haupteinfallstrassen hat große Vorteile, wie die überregionale gute Erreichbarkeit und die gute Wahrnehmung in der Öffentlichkeit. Es bieten sich daher Nutzungen an, die einer hohen öffentlichen Aufmerksamkeit bedürfen und gut erschlossen sein müssen.

Die gute Verkehrsanbindung bringt aber auch den Nachteil eines erhöhten Lärm- und Abgasaufkommens mit sich. Die Architektur muss auf diese Situation reagieren. Nutzungen, die viel Ruhe erfordern, scheinen zunächst nicht geeignet.

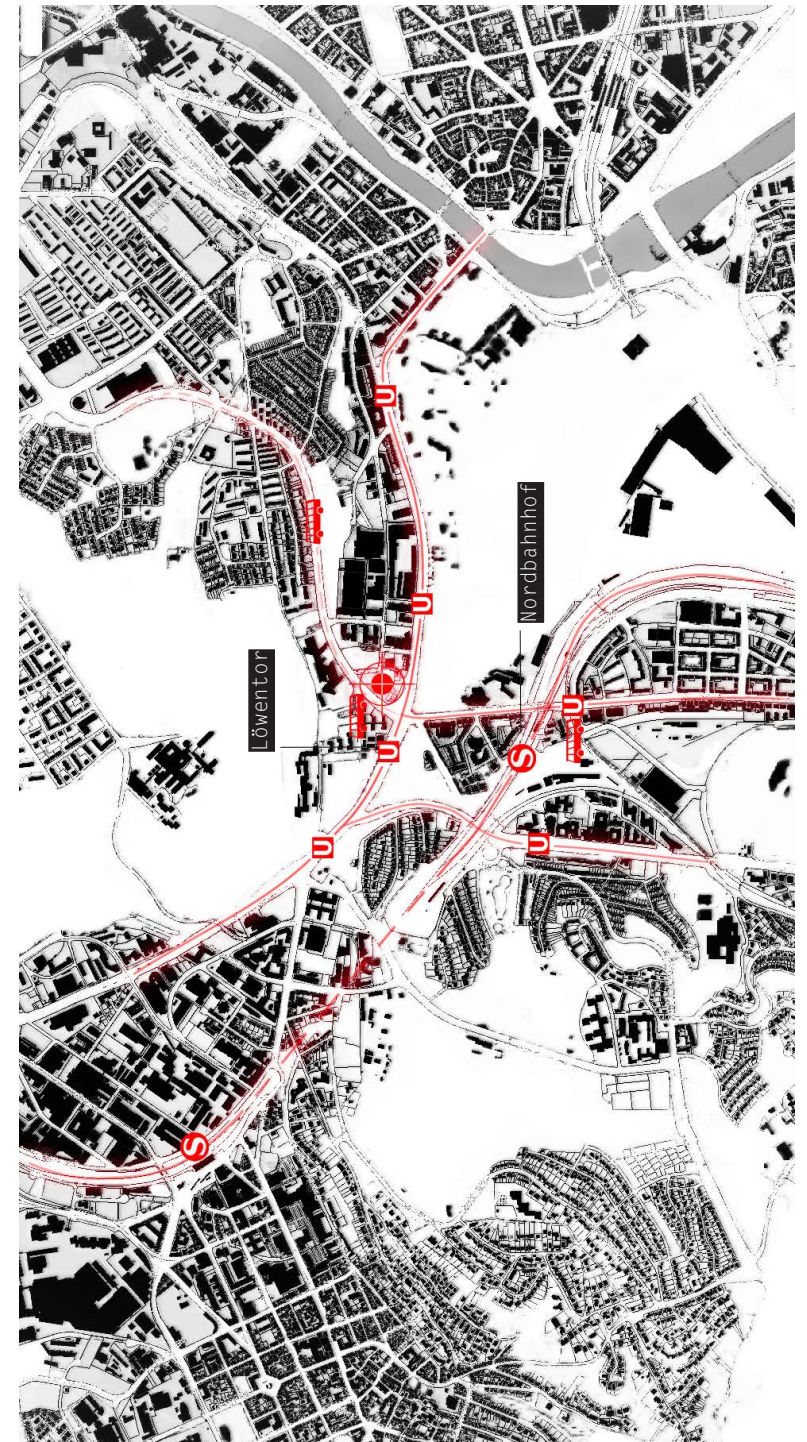


öpnv Die ÖPNV-Anbindung ist optimal. Die Haltestelle Löwentor mit Anbindung an die Stadtbahnlinien U13 und U15 ist ca. 50 m entfernt. Außerdem sind die Linien U6 und U7 an der ca. 300 m entfernten Station Pragsattel zu erreichen.

Die S-Bahn Haltestelle Nordbahnhof mit Anbindung an die Linie S4, S5 und S6 ist ca. 500 m entfernt. In direkter Nachbarschaft liegt auch die Bushaltestelle 55 und 56.

Außerdem wird derzeit die Stadtbahnlinie U12 erweitert und in diesem Rahmen über die Löwentorstrasse bis nach Remseck geführt.

Es bietet sich demnach auch hier eine Nutzung an, die von dieser sehr guten ÖPNV-Anbindung profitiert.



makroebene nutzungsschwerpunkte



Einzelhandel

Aus nebenstehender Grafik wird deutlich, dass in der näheren Umgebung keine Häufung von Lebensmittelmärkten auftritt. Der nächste Supermarkt mit Vollsortiment liegt in ca. 700 m Entfernung.

Die Nahversorgung des Gebietes um den Pragsattel ist nicht ausreichend gewährleistet und hat deutliche Potentiale. -> hierbei ist die Anzahl der Arbeitsplätze und Wohnungen in unmittelbarer Nähe entscheidend



Hotels

Außerdem wird deutlich, dass sich in der näheren Umgebung trotz guter Anbindung keine Hotelstandorte befinden. Diese konzentrieren sich hauptsächlich auf den Innenstadtbereich Stuttgarts.



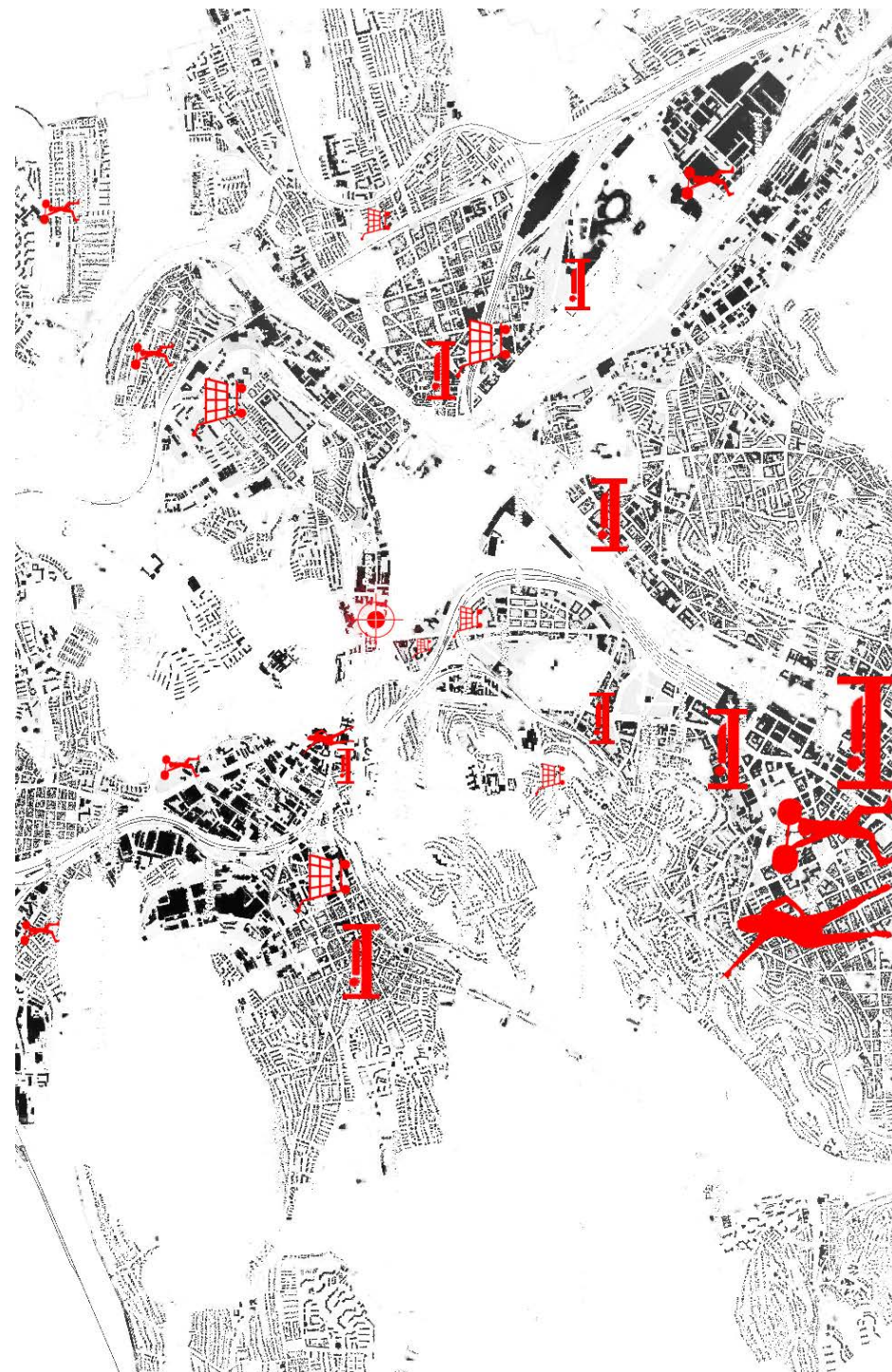
Fitness-Studios

Einen Schwerpunkt von Fitness-Studios ist eindeutig im Stadtzentrum von Stuttgart zu erkennen. Weitere Studios liegen gut verteilt in den jeweiligen Randquartieren. Im Bereich Fitness-Studio ist ein großes Potential gegeben, da im näheren Umfeld keinerlei Standorte vorhanden sind.



Discotheken

Die Discotheiken reihen sich nahezu alle entlang der B27 in der Innenstadt Stuttgarts aneinander. Lediglich ein Ausreißer liegt auf dem Pragsattel. Auch das Zentrum der Kinolandschaft befindet sich hier (nähe Hauptbahnhof). Auch ist ein für diese Nutzungen größeres Potential am Projektgrundstück zu sehen.



mikroebene umfeldnutzung

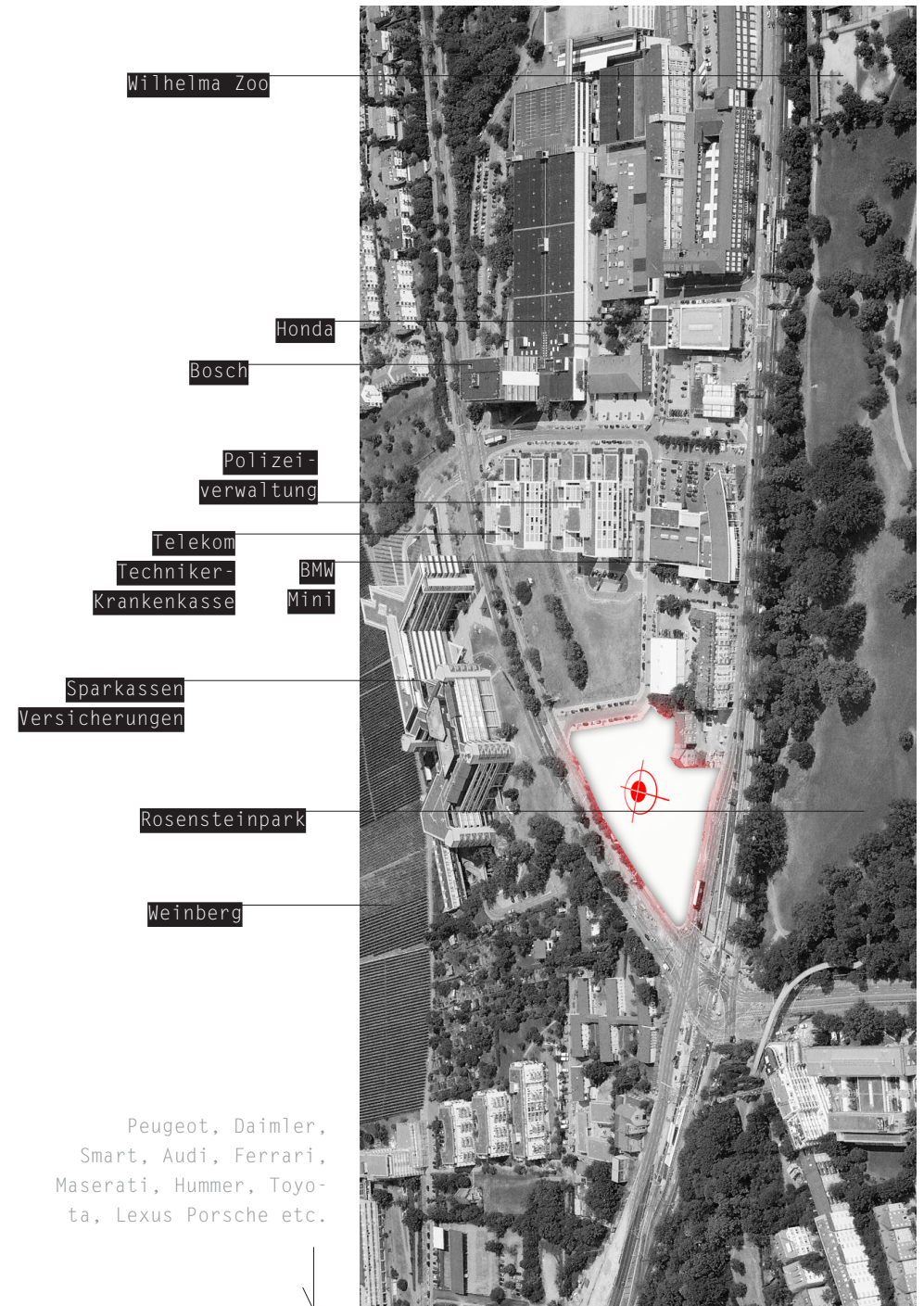
Neben seiner optimalen Verkehrsanbindung zeichnet sich der Standort durch die in seiner Umgebung ansässigen Nutzungsschwerpunkte aus.

In unmittelbarer Nachbarschaft und entlang der B10 befinden sich die Verkaufsstandorte namhafter Automobilhersteller, wie BMW mit Mini, Peugeot, Daimler mit Smart, Audi, Ferrari, Maserati, Hummer, Toyota mit Lexus und Porsche etc.

Daneben gibt es weitere Servicezentren und Dienstleistungen rund um das Automobil, wie Autovermietungen, Waschanlagen oder Werkstätten.

In direkter Nachbarschaft befindet sich die Konzernzentrale der SV Sparkassenversicherung, welches mit ca. 76.000 qm Fläche und rund 1500 Arbeitsplätzen das zweitgrößte Gebäude Stuttgarts ist. Weitere Nutzungen in unmittelbarer Nähe sind z.B. die Techniker Krankenkasse, Telekom, Bosch und die Polizeiverwaltung.

Außerdem bietet die Parkanlage Rosensteinpark/Schlosspark über der Strasse Verweilmöglichkeiten. Darüberhinaus sind auch ein kulturelles Angebot und Freizeitmöglichkeiten durch diverse Museen und die Wilhelma (Zoo) in unmittelbarer Nähe gegeben. Für Wohnen ist der Standort nicht geeignet, da er als GE ausgewiesen ist. Es ist auch keine Entwicklung denkbar, da durch den Verkehrsknotenpunkt Lärm- und Geruchsimissionen vorhanden sind.



auswertung
nutzungskonzept

Büro

Zwar liegt der Standort zur Zeit am Rande einer gefragten Bürolage, gilt aber eher als 2B-Bürostandort. Konkurrenz ist vor allem durch Stuttgart 21 gegeben. Dennoch weist er durch seine Sichtbarkeit und Verkehrsanbindung großes Potential auf. Auch durch den bereits etablierten Standort der Sparkassen Versicherung können sich Synergien ergeben. Diese ständig wachsende Struktur bedarf viel Flächen, welche in die direkte Nachbarschaft ausgelagert werden könnten.

Vorgeschlagen wird ein Mix aus Büro-, Hotel- und Gewerbeflächen

Flexibel nutzbar als Erweiterungsflächen für die Sparkassen Versicherung, als vermietbare Bürofläche oder aber als Therapiefläche...

Shopping

Keine klassische Einzelhandelslage wie in der Innenstadt, jedoch ist Potential durch den Nutzungsschwerpunkt im direkten Umfeld gegeben. Außerdem herrscht in diesem Gebiet eine prinzipielle Nahversorgungslücke. Darüber hinaus bietet sich die sehr gute Verkehrsanbindung für Handel mit hohem PKW-Verkehr an.

Nahversorgung, SB-Warenhaus, Bekleidungsgeschäfte, kleinere Boutiquen, ...

Hotel

Kein gesetzter Hotelstandort, jedoch Potential, da gute Verkehrsanbindung und am Rande eines etablierten Büroquartiers mit Entwicklungspotential.

Business, Kongress, ...

Lifestyle

Das Zentrum für Lifestyle ist die Innenstadt Stuttgarts, hier lassen sich die Schwerpunkte der Nutzungen Fitness Studio, Kino und Discothek finden. Ein Lifestyle-Center in der Peripherie mit guter Anbindung weist daher sehr gute Potentiale auf.

Gastronomie, Bars, Fitness Studio, Discothek, Bowling, ...

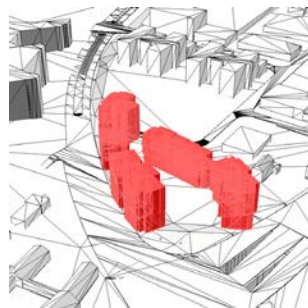
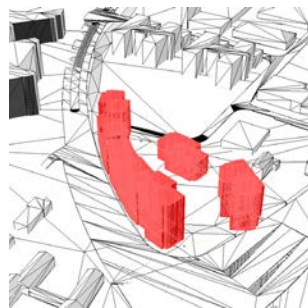
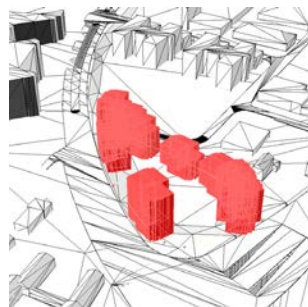
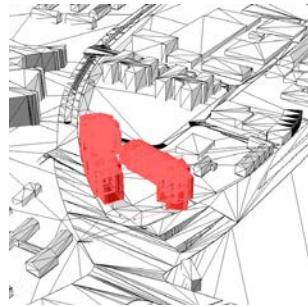
phase 1
analyse

Gestartet wurde mit einer klassischen Analyse: Es wurden städtebauliche Randbedingungen, die Verkehrsanbindung und schließlich Nutzungen untersucht. Auf dieser Grundlage sind erste Ideen und Konzepte entwickelt worden.

In den Gesprächen im Vorfeld mit den Vertretern des Grundstückseigentümers (Sparkassen Versicherungen) wurde deutlich gemacht, dass sich das Bauprojekt erst ab einer gewissen Größenordnung rentiert und man in der derzeitigen wirtschaftlichen Lage ein solches Risiko nur bei Großprojekten eingehen wird. Das bedeutet, dass die durch die GFZ vorgegebene Fläche von 24.800 qm (10.320 qm x 2.4) möglichst genau erreicht werden muss.

Auf dieser Basis wurden erste Voruntersuchungen zu GRZ- und GFZ-Flächen, sowie der damit verbundenen Gebäudehöhe angestellt.

Eigens dafür ist ein Algorithmus entwickelt worden, der Gebäudevolumen mit einer Gebäudetiefe von 16m zufällig entlang eines definierten Pfades verteilt. Mittels Attraktoren kann dabei die Gebäudehöhe und damit auch die Dichte der Baukörper gesteuert werden.



Aus den generierten Varianten konnte die Erkenntnis gezogen werden, dass mit einer Bebauung entlang den Grundstücksgrenzen die Gebäudehöhe auf ein städtebaulich nicht verträgliches Maß (ca. 7 Vollgeschosse) ansteigt.

Lineare oder punktförmige Bautypologien, die zwar für eine Büronutzung angemessen sind, aber bei ausreichender Abstandsfläche und Erschließung zu wenig GFZ-Fläche bieten, scheidet daher zunächst aus. Hingegen würden mit einer flächigen Bebauung, die sich durchaus für Einzelhandel eignet, die erlaubten 24.800 qm bereits schon mit 4 Vollgeschossen erreicht werden.

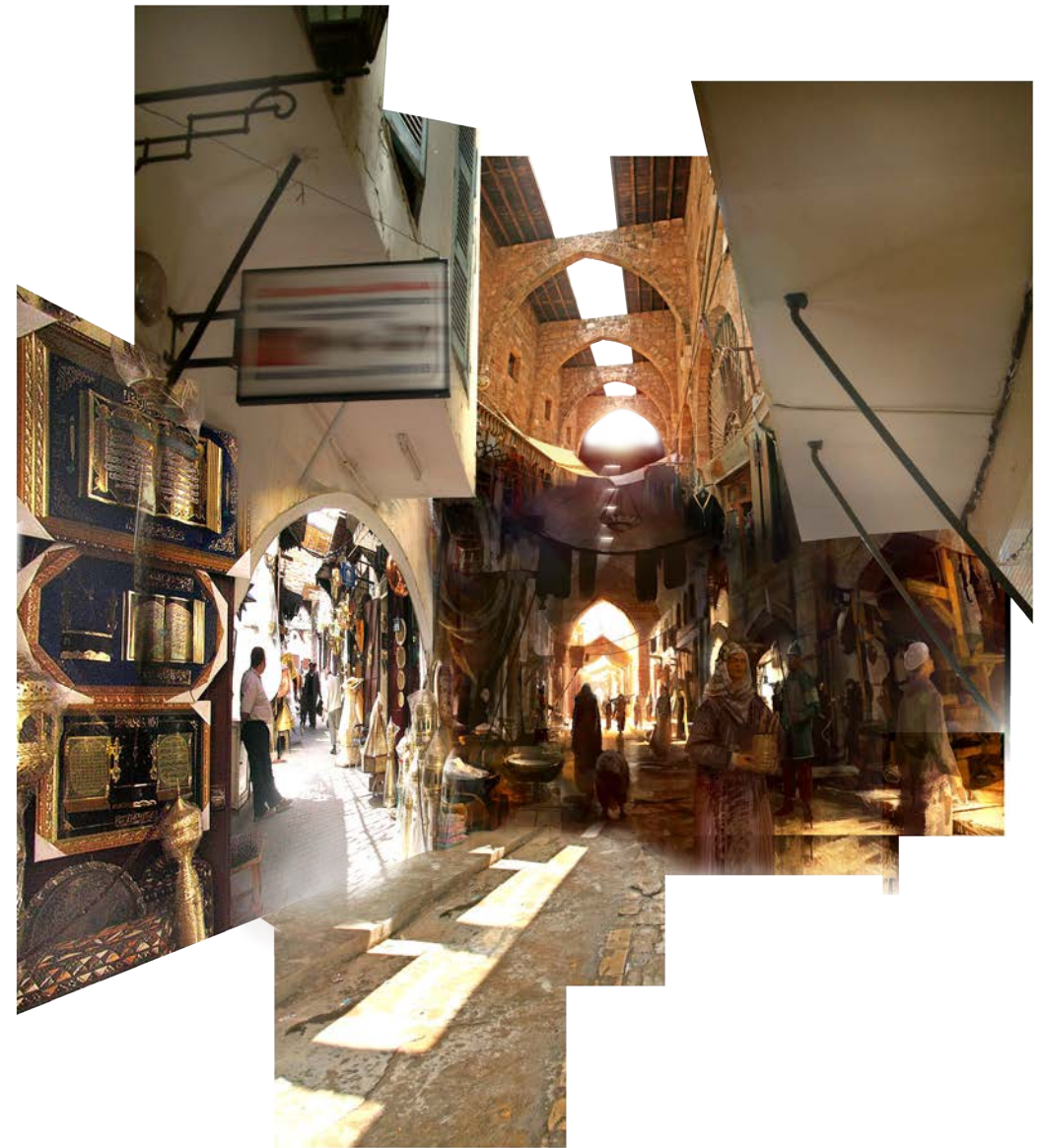
Es gilt also eine hohe Dichte zu generieren und gleichzeitig für Belichtung der Büro- und Hotelnutzung zu sorgen.

phase 2 konzeptidee

Aus den divergierenden Anforderungen der unterschiedlichen Nutzungen wurde in der nächsten Entwurfsphase das Konzept der Teppichbebauung entwickelt. Dieses zeichnet sich durch eine Ausrichtung der Innenräume zu einem Innenhof aus. Die Orientierung der Räume nach Innen gestattet sehr verdichtete Strukturen, die geschossweise variiert werden können und somit sehr flexibel einsetzbar sind. Dieser Typus, obwohl er als ein Produkt der modernen Stadt gesehen wird, ist antiken Ursprungs und typisch für Asien, Afrika und europäische Altstädte.

Die Introvertiertheit dieses Typus, bei dem der Kontakt nach Aussen eingeschränkt werden kann, macht ihn gerade an dieser hochfrequentierten Stelle, die stark durch Lärm- und Geruchsemissionen belastet ist, interessant.

Öffnungen nach Aussen können sparsam und gezielt gesetzt werden, um im Inneren eine eigene Landschaft aus Gassen, Plätzen, Strassen, Wegen und Grünflächen erlebbar zu machen.



Leitbild

phase 3 algorithmus

„Und was Ist Zufall anders
als der rohe Stein. Der Leben
annimmt unter Bildners Hand?
Den Zufall gibt die Vorsehung
- Zum Zwecke Muss ihn der
Mensch gestalten.“

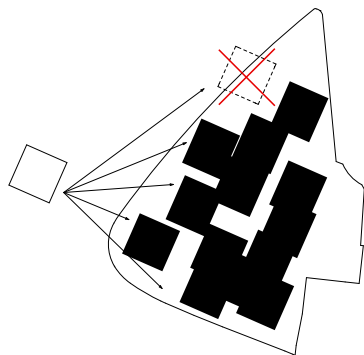
Friedrich Schiller, "Don Carlos"

In der klassischen Vorgehensweise würde in dieser Phase durch Skizzen und Modellstudien eine Struktur entwickelt werden. Es würde mit verschiedenen Ansätzen experimentiert und Varianten erstellt werden.

Der generative Entwurfsprozess verlangt jedoch zunächst nach einer Abstrahierung der Konzeptidee. Es musste daher zuerst genau überlegt werden wie eine solche Teppichbebauung umgesetzt werden kann:

Dazu war die erste Intention, einen Algorithmus zu erstellen, welcher zufällig rechteckige Gebäudevolumina innerhalb des Grundstückes anordnet. Durch Steuerung der Dichte, sowie der Abmessung dieser Boxen kann in den einzelnen Geschossen eine ähnliche Struktur erreicht werden, die jedoch auf die jeweilige Nutzungsanforderung reagiert. Im Erdgeschoss soll eine Durchwegung mit großzügigen Freibereichen und großen Gebäudetiefen entstehen, im obersten Geschoss hingegen eine geringere Gebäudetiefe und eine gleichmäßige, wenn auch nicht so hohe Dichte.

Durch Zufallszahlen werden unterschiedliche Koordinaten für die einzelnen Boxen errechnet. Jede Koordinate wird daraufhin auf Position innerhalb oder außerhalb der Grundstücksgrenzen überprüft. Liegt sie außerhalb, wird sie gelöscht und eine neue Koordinate wird erstellt.

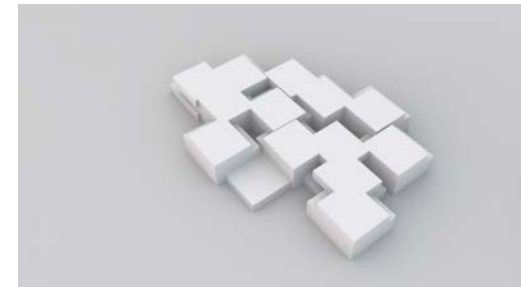


Durch generieren neuer Zufallszahlen können nun theoretisch unendlich viele Varianten einer Verteilung der Boxen innerhalb der Grundstücksgrenzen erzeugt werden.

phase 4 auswertung

Die ersten nach diesem Prinzip generierten Versionen wiesen eine sehr heterogene Struktur mit spannenden Raumqualitäten im Inneren auf. Interessant war, dass sich unterschiedliche Räume wie enge und weite Bereiche ergaben, welche sich zum Teil formierten und als Durchwegung interpretiert werden konnten.

Durch Analysieren dieser Varianten konnte im Weiteren festgestellt werden, dass für den Strassenraum entlang der Prag- und Löwentor Strasse eine geschlossene und klare Gebäudekante äußerst wichtig ist.

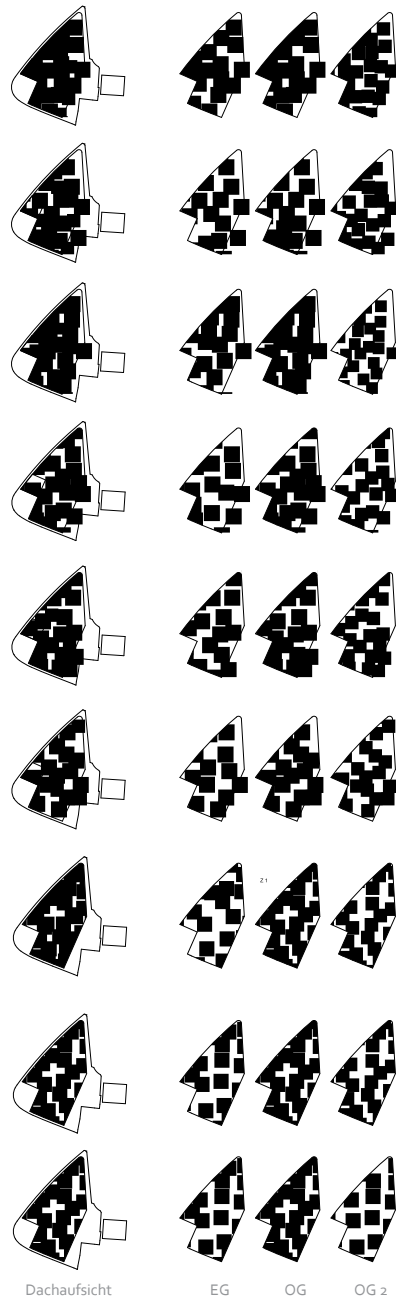


phase 5 optimierung

In der nächsten Entwurfsphase wurde der Algorithmus angepasst, optimiert und erweitert. Die Erkenntnisse, die durch die erste Generation von Varianten gewonnen werden konnten flossen dabei direkt ein:

Die Verteilung der Gebäudevolumina wurde jetzt nicht mehr ausschließlich innerhalb der Grundstücksgrenzen zugelassen, sondern im Bereich der Prag- und Löwentor Strasse auch außerhalb. Dabei werden jedoch die außerhalb einer definierten Kante befindlichen Volumina weggeschnitten, sodass eine klare Gebäudekante entsteht. Aufgrund der unklaren Gebäudegeometrie (entlang Prag- und Löwentor Strasse homogene, entlang Löwentorbogen heterogene Kante) wurde der Algorithmus weiter angepasst, sodass letztendlich eine eindeutige Gebäudefigur entsteht.

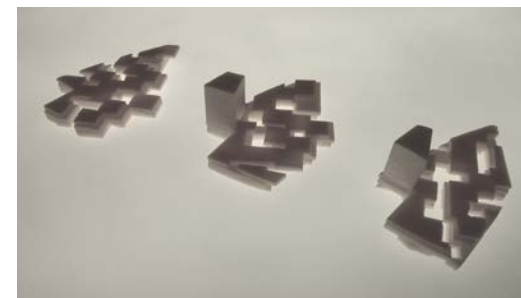
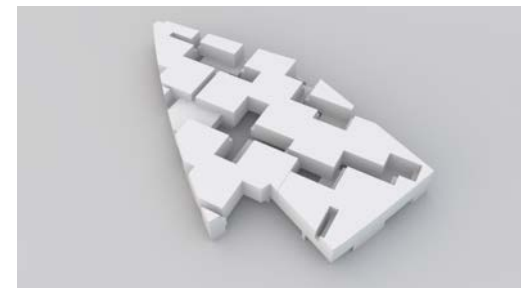
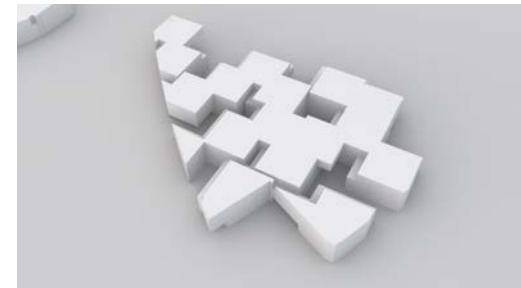
Mit Hilfe dieses Algorithmus konnte nun eine Vielzahl von Entwurfsvarianten erzeugt werden. Dabei war es einfach und schnell möglich mit unterschiedlichen Abmessungen, Dichten und Geometrien der Volumina zu experimentieren.



Auswahl der generierten Entwurfsvarianten

phase 6 bewertung

Die generierten Varianten wurden nun nach den Kriterien Durchwegung, Raumqualität, Belichtung, Organisation und GFZ-Fläche bewertet. Die besten drei Varianten wurden dann »manuell« angepasst, das heißt zu enge Bereiche und unsaubere Kanten entfernt bzw. verändert und schließlich als Arbeitsmodell im Maßstab 1:1000 erstellt.



Dieser Schritt vom digitalen, 3-dimensionalen Modell zum materiellen, erlebbaren Modell ist äußerst wichtig. Der Idealfall wäre, dass jede Variante an einen 3D-Drucker übergeben und im Rapid-Prototyping-Verfahren als Werkstück erstellt wird.

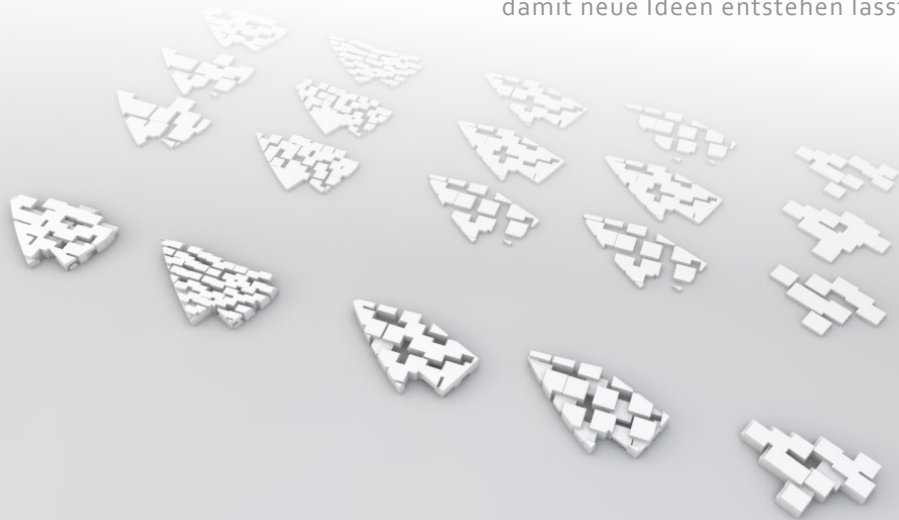
phase 7 Auf der Grundlage der besten Entwurfsvariante wurde klassisch weitergearbeitet, das heißt Tragkonstruktion, Erschließung, sanitäre Einrichtungen etc. geplant.

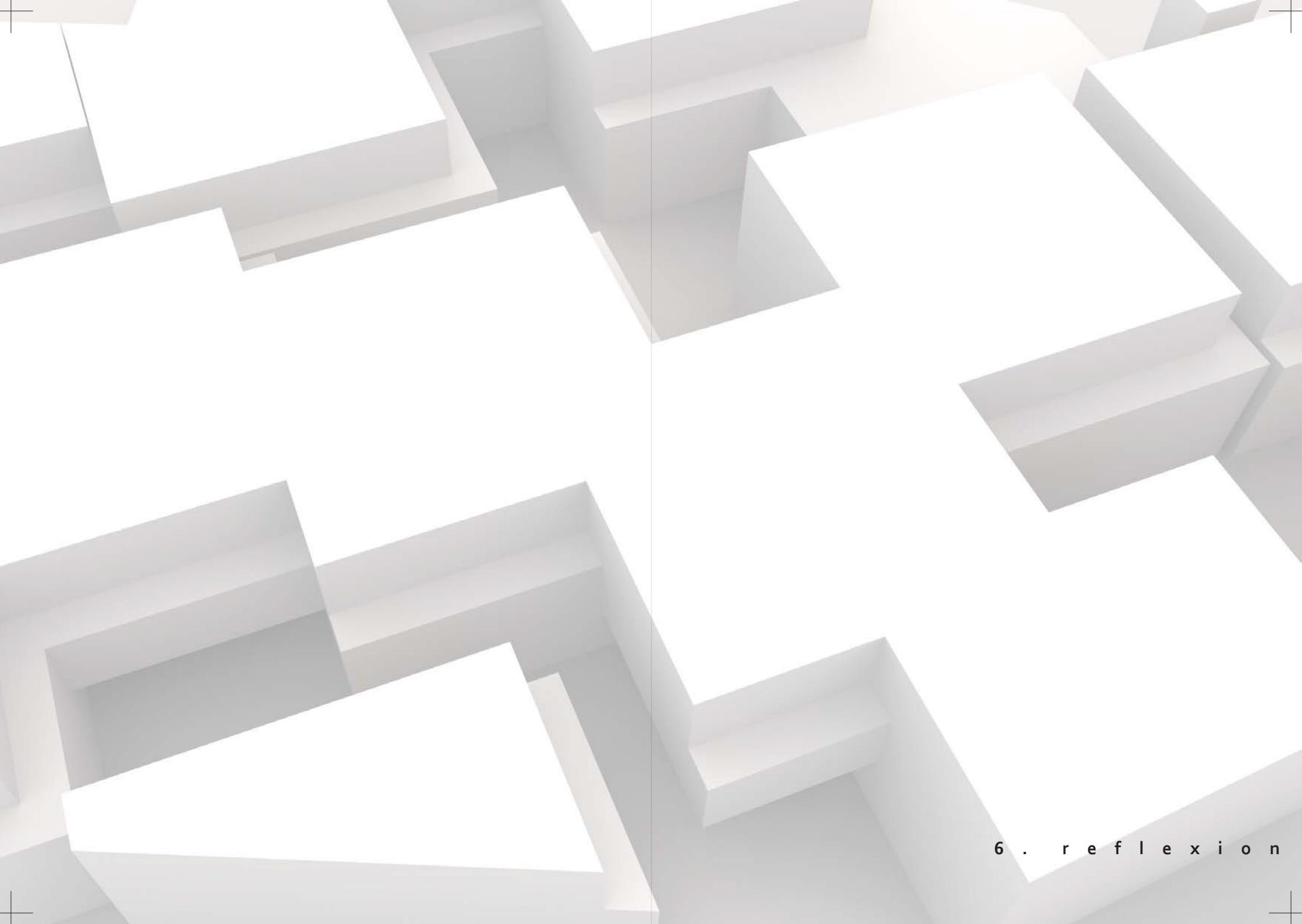
entwurf

Die vorhergehenden Schritte waren wichtige Bestandteile des allgemeinen Entwurfsprozesses. Das Arbeiten in diesem Variantenreichtum half enorm bei der Entwicklung des Entwurfskonzeptes, sowie der Organisation und der Formgenese.

Wenngleich dieser Entwurfsansatz nur in Teilen eines integralen generativen Entwurfsprozesses entspricht, war er in jedem Fall dienlich und lohnenswert und brachte ganz eigene Qualitäten, die auf konventionelle Weise vermutlich so nicht entstanden wären.

Jede Variante kann auch als Inspirationsquelle betrachtet werden, die spezielle Eigenschaften besitzt und damit neue Ideen entstehen lässt.





6 . r e f l e x i o n

chancen für die architektur

Üblicherweise wird das architektonische Lösungsfeld zum einen durch das Raumprogramm und die örtlichen Gegebenheiten, zum anderen durch die Intention, Erfahrungswerte, typologisches Denken und persönliche Präferenzen des Entwerfers eingeschränkt. Traditionell können aufgrund des unverhältnismäßigen Aufwandes nur eine begrenzte Anzahl von Varianten erstellt werden. Gerade der generative Entwurfsprozess zeichnet sich jedoch durch einen enormen Variantenreichtum aus. Durch Veränderung einiger Parameter oder Prozesse entstehen völlig neue Alternativen. Diese Charakteristik bietet eine große Chance für die Architektur, da das Arbeiten in Varianten für den Gestaltungsprozess unabdingbar ist.

Dabei können neue Lösungswege und Ideen entstehen, die man im Vorhinein nicht in Betracht zog. Insbesondere da Algorithmen unvoreingenommen und frei von typologischem Denken sind, wird das Lösungsfeld grundlegend erweitert:

So besteht die Chance der Entdeckung neuer Genotypen und deren Einbindung in den Entwurfsprozess. Außerdem führt die rationale Auseinandersetzung mit Algorithmen zu einem tieferen Verständnis des Entwurfsprozesses und der konzeptionellen Idee selbst.

Die gefundenen Lösungen könnten sich daher durch eine hohe Integrität auszeichnen, da sie direkt aus den projektspezifischen Parametern und eng nach einer Konzeptidee entwickelt wurden. Erstmals in der Tradition der Architektur könnte die Suche nach »tiefer liegenden« Typologien, Archetypen und Systemen und die Transzendierung dieser in bisher unbekannte Formen möglich werden.

fazit

„We go our own way. We do architecture solely with structure. A day will come when others who have something important to give will no longer do what we are now doing. Architecture must grow out of its own time, just as the old architecture did. Every epoch achieves as much as it ventures to achieve.“

Mies van der Rohe

Jeder Entwurfsprozess ist charakterisiert durch ständiges Oszillieren zwischen der Erstellung von Ideen und derer augenblicklichen Simulation, Bewertung und Prüfung durch den Entwerfer.

Im generativen Entwurfsprozess geschieht das nicht anders, jedoch meist schneller und zielgerichteter. Entwurfsgedanken können am effektivsten bewertet werden, indem Alternativen und Variationen untersucht werden. Je mehr dabei zur Verfügung stehen, desto höher liegt auch die Chance den besten Lösungsansatz herauszufiltern und ein breites Spektrum des Lösungsfeldes abzudecken.

Durch Abstrahieren und Zerlegen der Entwurfsidee in einen Computeralgorithmus, ist es zum einen möglich eine fast unendliche Anzahl von Varianten zu generieren und dabei eine Vielzahl von Kriterien gleichzeitig zu beachten. Zum anderen muss die Entwurfsidee explizit ausgedrückt werden und ist demnach durch alle Maßstäbe gegenwärtig.

Außerdem ist es möglich, wie das Beispiel des Olympiastadion Projektes in Peking zeigt, die Algorithmen

weiterzuentwickeln, sodass sie die Auswahl der besten Lösung selbst übernehmen. Der Einsatz generativer Prozesse lässt hier neue Möglichkeiten entstehen, die manuell nicht beherrschbar wären.

Die unter Architekten weitverbreitete Skepsis und das daraus entstehende Vorurteil, der Computer sollte oder könnte den Entwerfer ersetzen hat sich hingegen als absurd erwiesen:

Die steuernde und konzeptionelle Kompetenz bleibt nach wie vor beim Architekten. Die Synthese aus dem deterministischen, logischen Vorgehen eines Algorithmus und dem intuitiven menschlichen kann jedoch auf völlig neue Lösungen führen.

Dazu ist aber eine tiefe Auseinandersetzung mit der Softwareumgebung, insbesondere Programmierung nötig. Zwar gibt es bereits eigens für Gestalter entwickelte Tools, die das Programmieren erleichtern, für einen integralen generativen Entwurfsansatz ist jedoch fundiertes Hintergrundwissen und weitreichende Programmiererfahrung notwendig, welche als Architekt derzeit in der Regel noch schwer erreichbar ist.

Dennoch ist dieser Ansatz bestimmt keine Modeerscheinung, sondern wird vor dem Hintergrund zunehmender Komplexität der Gesellschaft, der gebauten Anthroposphäre und der Vernetzung der Welt von Bedeutung sein.

ausblick

Bereits während meiner Studienzeit zeichnete sich ab, dass der Umgang mit generativer Gestaltung sehr schnell ein selbstverständlicher sein wird. Generative Gestaltungsprozesse sind eine solide Grundlage für den Umgang mit steigender Komplexität und Geschwindigkeit von heutigen Entwurfsaufgaben.

Die Erweiterung der technischen Möglichkeiten führt zu ständigen Impulsen auf diesem Feld. So war es beispielsweise noch vor wenigen Jahren fast unmöglich komplexe 3-dimensionale Objekte zu erstellen, heute kann dies nahezu jeder bereits auf einem Laptop. Dieses technologische Potential, sowie das Selbstverständnis des Computereinsatzes, mit dem die heutige Generation heranwächst, wird generative Gestaltung auch weiterhin antreiben.

Ich schließe mich der verbreiteten These an, dass Programmieren und damit auch generative Gestaltungsprozesse, zu einer universellen Kulturtechnik und neuer »Fremdsprache« werden wird, die sich, ähnlich wie Film oder Fotografie im letzten Jahrhundert, schnell etabliert.

- a n h a n g

quellenverzeichnis

Bücher

Hartmut Bohnacker, Benedikt Groß, Julia Laub, Claudius Lazzeroni (Hrsg.) (2009): Generative Gestaltung: Entwerfen. Programmieren. Visualisieren. Mainz: Hermann Schmidt Verlag

Asterios Agkathidis (Hrsg.), Markus Hudert (Hrsg.), Gabi Schillig (Hrsg.) (2009): form defining strategies. experimental architectural design. 2. Aufl. überarbeitet. Tübingen: Wasmuth Ernst Verlag

Terzidis, Kostas. Algorithmic architecture (2006). Oxford: Architectural Press

Svenia Schneider: -digitale Form im architektonischen Entwurfs- und Arbeitsprozess zu Beginn des 21. Jahrhunderts in: Caroline Y. Robertson von Trotha, Robert Hauser (Hg.) (2011): Neues Erbe. Aspekte, Perspektiven und Konsequenzen der digitalen Überlieferung. Karlsruhe: KIT Scientific Publishing S.247-257

Dörte Kuhlmann, Heimo Schimek (Hrsg.) (2001): Cybertecture. Die 4. Dimension in der Architektur. Wien: Löcker

John Frazer (1995): An Evolutionary Architecture. London: Architectural Association Publications

John Maeda (2004): Creative Code. Ästhetik und Programmierung am MIT Media Lab. Boston: Birkhäuser Verlag

Zeitschriften

Nikolaus Kuhnert, Anh-Linh Ngo: Entwerfen im digitalen Zeitalter, in Editorial ARCH+ Nr.189 „Entwurfsmuster“, Oktober 2008, S.6-9

Tobias Wallisser: Vom Blop zur algorithmisch generierten Form, in ARCH+ Nr.189 „Entwurfsmuster“, Oktober 2008, S.120-123

Achim Menges: Architektonische Form- und Materialwerdung am Übergang von Computer Aided zu Computational Design, in Detail „Analog und Digital“, 50. Serie 2010, 5, S.420-422

Patrick Schumacher : Parametrismus. Der neue internationale Style, in: ARCH+ Nr.195 „Istanbul wird grün“, November 2009

Oliver Fritz: Programmirtes Entwerfen, in ARCH+ Nr.189 „Entwurfsmuster“, Oktober 2008, S.60-61

Martin Schroth: Digitale Formgenerierung, in ARCH+ Nr.189 „Entwurfsmuster“, Oktober 2008, S.124-125

Patrik Schumacher: Autopoietic Elegance in: MRGD Morphe, RIEA (Research Institute of Experimental Architecture). 2008

Websites

Kein Autor (o.J.): „Generative Gestaltung“. <http://de.wikipedia.org/wiki/Generative_Gestaltung> [Stand:22.01.2012. Zugriff 20.04.2012, 18:41 MESZ]

Kein Autor (o.J.): „Generative Kunst“. <http://de.wikipedia.org/wiki/Generative_Kunst> [Stand:12.09.2010. Zugriff 20.04.2012, 18:43 MESZ]

Schmidt-Friderichs GmbH (o.J.): „Generative Gestaltung“. <<http://www.generative-gestaltung.de/>> [Stand:21.04.2012. Zugriff 21.04.2012, 10:12 MESZ]

Reinhard König (2012): „Entwicklung des parametrischen und algorithmischen Entwerfens“. <<http://entwurforschung.de/entwicklung-des-parametrischen-und-algorithmischen-entwerfens/>> [Stand: 09.04.2012. Zugriff 22.04.2012, 11:00 MESZ]

Philip Belesky (2010): „Ghost in the Machines: Parametric architecture and the philosophy of giles deleuze“ <<http://www.manifoldblog.com/?p=116>> [Stand: 14.02.2010. Zugriff 10.04.2012, 16:45 MESZ]

Kein Autor (o.J.): „Building Information Modeling“. <http://de.wikipedia.org/wiki/Building_Information_Modeling> [Stand: 21.03.2012. Zugriff 22.04.2012, 10:31 MESZ]

Bibliographisches Institut GmbH (o.J.): „Bedeutung; Rechtschreibung: Programm“. <<http://www.duden.de/rechtschreibung/Programm>> [Stand: 2012. Zugriff 25.04.2012, 10:55 MESZ]

Kein Autor (o.J.): „Programmierung“. <<http://de.wikipedia.org/wiki/Programmierung>> [Stand: 14.04.2012. Zugriff 25.04.2012, 11:11 MESZ]

Microsoft (2007): „Grundlagen: Wie Programmierung funktioniert?“ <[http://msdn.microsoft.com/de-de/library/ms172579\(v=vs.90\).aspx](http://msdn.microsoft.com/de-de/library/ms172579(v=vs.90).aspx)> [Stand: 2012. Zugriff 25.04.2012, 11:12 MESZ]

Kein Autor (o.J.): „Algorithmus“. <<http://de.wikipedia.org/wiki/Algorithmus>> [Stand: 24.04.2012. Zugriff 27.04.2012, 13:05 MESZ]

Kein Autor (o.J.): „Menger-Schwamm“. <<http://www.mathetreff-online.de/mathelexikon/m/menger-schwamm>> [Stand: 08.09.2011. Zugriff 28.04.2012, 10:40 MESZ]

Kein Autor (o.J.): „Evolutionärer Algorithmus“. <http://de.wikipedia.org/wiki/Evolutionärer_Algorithmus> [Stand: 21.04.2012. Zugriff 28.04.2012, 18:10 MESZ]

Scott Davidson (2012): „About Grasshopper...“. <<http://www.grasshopper3d.com/>> [Stand: 2012. Zugriff 01.05.2012, 16:06 MESZ]

Kein Autor (o.J.): „Open-Source“. <<http://de.wikipedia.org/wiki/Open-Source>> [Stand: 11.04.2012. Zugriff 01.05.2012, 15:20 MESZ]

Autodesk, Inc. (2012): „Design Visualization“. <<http://usa.autodesk.com/adsk/servlet/item?siteID=123112&id=13591323>> [Stand: 2012. Zugriff 02.05.2012, 08:23 MESZ]

Kein Autor (o.J.): „Maya“. <[http://de.wikipedia.org/wiki/Maya_\(Software\)](http://de.wikipedia.org/wiki/Maya_(Software))> [Stand: 17.02.2012. Zugriff 02.05.2012, 09:20 MESZ]

Kein Autor (o.J.): „Processing“. <<http://www.processing.org/>> [Stand: 2012. Zugriff 02.05.2012, 12:44 MESZ]

Markus Braach (2010): „National Beijing Olympic Stadium „. <http://www.kaisersrot.ch/kaisersrot-02/2003_National_Beijing_Olympic_Stadium_%28CN%29.html> [Stand: 21.03.2010. Zugriff 04.06.2012, 12:55 MESZ]

SLIK-Architekten (o.J.): „Wohn- und Gewerbesiedlung Kalkbreite“ <<http://www.slik.ch/?pid=12>> [Stand: 2012. Zugriff 06.06.2012, 14:53 MESZ]

Marc Fornes (o.J.): „09 Ap. Symmetries“. <<http://theverymany.com/constructs/09-aperiodic-symmetries/>> Stand: 2012. Zugriff 06.06.2012, 16:58 MESZ]

Nervous System (o.J.): „Hyphae Lamp“. <<http://n-e-r-v-o-u-s.com/shop/generativeProduct.php?code=99>> Stand: 2012. Zugriff 10.06.2012, 19:58 MESZ]

Andere

Reinhard König, Christian Bauriedel: „Computergenerierte Stadtstrukturen“. Diplomarbeit an der Techn. Universität Kaiserslautern, 2004

Hans-Georg Beckmann: „Zelluläre Automaten“. Virtuelle Lehrerfortbildung im Fach Informatik in Niedersachsen. 2003

Steffen Harbich: „Einführung genetischer Algorithmen mit Anwendungsbeispiel“. E-Paper der Universität Magdeburg. 2007

De Landa, Manuel. "Deleuze and the Use of the Genetic Algorithm in Architecture," Between Bladerunner and Mickey Mouse: New Architecture in Los Angeles Exhibition, Madrid, 22.04.2001.
Aus <<http://www.egs.edu/faculty/manuel-de-landa/articles/>>

Zubin Khabazi: „Generative Algorithms using Grasshopper“ E-Paper von <<http://www.grasshopper3d.com/page/tutorials-1>>

Skripte

Ricky Kamfolt: MEL-Skript für Menger-Schwamm <<http://migugi.net/mel/fractalSolids/>> [Stand: 2012. Zugriff 28.04.2012, 22:55 MESZ]

abbildungsverzeichnis

Grafik

Kaisersrot: Strukturvarianten Olympiastadion. <http://www.kaisersrot.ch/kaisersrot-02/2003_National_Beijing_Olympic_Stadium_%28CN%29.html> [Stand: 21.03.2010. Zugriff 04.06.2012, 12:55 MESZ]

Thomas Wallmeyer: Olympiastadion. <<http://www.fotocommunity.de/pc/pc/cat/388724/display/12519235>> [Stand: 14.04.2008. Zugriff 04.06.2012, 12:59 MESZ]

Kaisersrot: Kalkbreite. <http://www.kaisersrot.ch/kaisersrot-02/2008__Kalkbreite.html> [Stand: 21.03.2010. Zugriff 04.06.2012, 12:56 MESZ]

Marc Fornes: og Ap. Symmetries. <<http://theverymany.com/constructs/og-aperiodic-symmetries/>> Stand: 2012. Zugriff 06.06.2012, 16:58 MESZ]

Nervous System: Hyphae Lamp. <<http://n-e-r-v-o-u-s.com/shop/generativeProduct.php?code=99>> Stand: 2012. Zugriff 10.06.2012, 19:58 MESZ]

vielen dank an





